History-Aware Adaptive Backoff for Neighbor Discovery in Wireless Networks

Zimu Yuan, Lizhao You, Wei Li, Biao Chen, Zhiwei Xu

1 Institute of Computing Technology, Chinese Academy of Sciences, China
2 State Key Laboratory for Novel Software Technology, Nanjing University, China
3 Department of Computer and Information Science, University of Macau, Macao

Received 10 February 2012; In final form 10 February 2012

The ability of discovering neighboring nodes, namely neighbor discovery, is essential for the self-organization of wireless ad hoc networks. In this paper, we first propose a history-aware adaptive backoff algorithm for neighbor discovery using collision detection and feedback mechanisms. Given successful discovery feedbacks, undiscovered nodes can adjust their contention windows. With collision feedbacks and historical information, only transmission nodes enter the re-contention process, and decrease the size of their contention windows to accelerate the neighbor discovery process. Then, we give a theoretical analysis of our algorithm on discovery time and energy consumption, and derive the optimal size of contention windows. Finally, we validate our theoretical analysis by simulations, and show the performance improvement over existing algorithms.

Key words: wireless ad hoc networks; neighbor discovery; MAC protocols; history-aware; adaptive backoff;
1 INTRODUCTION

Neighbor discovery is a fundamental step for the initialization of wireless ad-hoc networks, and the knowledge of neighbors is essential for further operations. For example, Johnson et al. [1] assume that nodes know the information about their one-hop neighbors in order to perform routing in multi-hop networks. Rhee et al. [2] use a two-hop neighbor list to develop TDMA MAC protocols.

For a given node, the problem of neighbor discovery is to find all nodes in its transmission range by receiving packets from its neighbors. We want to find all neighbors using minimal costs, e.g., minimal discovery time and energy consumption. However, this is not easy because all nodes will share a channel to transmit packets. If two or more nodes transmit at the same time, a collision happens at the reception nodes. Therefore, the key to successful neighbor discovery is to solve the collision caused by simultaneous transmissions. Many algorithms [3, 4, 5, 6] have been proposed to handle collisions, especially probabilistic discovery algorithms [7, 8, 9]. For example, [9] proposes several ALOHA-like algorithms, and transforms the neighbor discovery time analysis to the Coupon Collectors’ Problem to deal with collisions.

However, many of the above algorithms [5, 6, 7] did not consider to use collision detection and feedback mechanisms to solve neighbor discovery problem. Vasudevan et al. [8] first show the huge improvement of discovery time when collision detection and feedback mechanisms are used. In [12], the authors propose a reliable energy detection mechanism in the physical layer that allows receivers to detect collisions, and feedback the reception status. They also show that discovery time of this method is significantly smaller than the case where the reception status is not available. Therefore, our main idea focuses on the feedback model based on energy detection and a more realistic backoff model.

In this paper, we propose a history-aware adaptive backoff neighbor discovery algorithm. The algorithm performs in rounds. In each round, a node randomly backoffs some time slots, and then transmits determinedly. After discovering a new node, the contention window is different. In other words, our algorithm uses an adaptive backoff mechanism. Furthermore, in each round, if a collision happens due to the same slot is selected, we utilize historical information (transmission or reception in the collision slot) to divide nodes into different states, and re-contention only happens among those transmission nodes. We also decrease the size of the contention window after the first collision to accelerate the process of neighbor discovery. To the best
of our knowledge, the concepts of adaptive backoff and history-aware for probabilistic neighbor discovery have not been studied before under feedback scenarios. The main contribution of this paper can be summarized as follows:

• We propose a history-aware adaptive backoff neighbor discovery algorithm, and establish a theoretical model for performance analysis.

• We show through simulations that our algorithm allows nodes to discover their neighbors much faster than ALOHA-like algorithms, while consuming less energy.

The rest of the paper is organized as follows. Section 2 introduces the model and assumptions. Section 3 describes our algorithm. Section 4 gives a detailed analysis. We show simulation results in Section 5. The related work is given in Section 6. Finally, we conclude and give the future work in Section 7.

2 NETWORK MODEL AND ASSUMPTIONS

In wireless networks, nodes are distinguished by unique identifiers, such as the MAC address, the location of the node, and etc. Nodes exchange the unique identifiers for neighbor discovery by sending control packets. The problem of neighbor discover for a node is to find its neighbors’ identifiers with minimal costs, e.g., minimal discovery time or energy consumption. Each node is equipped with a transceiver that allows a node either transmit or receive packets, but not simultaneously. Two nodes are neighbors if they are in the communication range of each other. For the sake of simplicity, we assume errors caused by fading are negligible. In other words, if packets are transmitted without collisions, they must be received correctly. We also assume time is slotted with equal-length, and all nodes are synchronized on the slot boundary. Both of the above assumptions are adopted by [7, 11, 12].

In addition, nodes have collision detection capability, i.e., the receiver can tell collisions from successful receptions. Also, there is a feedback mechanism for transmitters, which is proposed by [11, 12]: each time slot is further divided into two sub-slots (namely transmission sub-slot and feedback sub-slot). If receivers cannot decode the packets transmitted in the first sub-slot, they reply a small packet in the feedback slot to notify transmitters. If the transmission node in the first sub-slot detects energy in the second sub-slot, it assumes that there is a collision. Otherwise, it assumes that the transmission is successful. Note that Khalili et al. [12] give a detailed physical layer design about collision detection and feedback mechanisms based on the energy
detection technique. Furthermore, Sen et al. [20] utilize a cross-correlation technique to do collision detection, and show that this technique can detect other users even if the signal from others (named interference) is 15dB less than the target signal.

3 HISTORY-AWARE ADAPTIVE BACKOFF ALGORITHM

In this section, we first give the basic idea of our algorithm, and then describe the algorithm. To show the key idea, we first consider algorithms in a clique with \( n \) known nodes beforehand.

Before describing our algorithm, we give two definitions: round and phase. A round starts when all active nodes participate in the backoff, which is the basic unit of neighbor discovery for nodes. We call a round finished if there is a collision or a successful transmission. Note that the length of a round must be smaller or equal to the length of the contention window, since the channel can not be idle within a contention window. The time to discover a new node is called phase, which may be composed of many rounds. If there is a successful transmission, we call a round finished, and a phase also ends.

3.1 Basic Idea

Our algorithm includes two key components: adaptive backoff and history-aware collision resolution. Furthermore, the length of the contention window of a round is fixed. However, the contention window may be different in different rounds, since some rounds are for contentions, and some rounds are for re-contentions. For different phases, the size of the contention window is different since our algorithm is adaptive.

Let the \( j \)-th phase be the process of discovering the \( j \)-th node, \( W_j \) be the initial contention window of the \( j \)-th phase, and \( W'_j \) be the re-contention window of the \( j \)-th phase, where \( j \in [1, n] \). Suppose we are in the \( j \)-th phase, all nodes first randomly choose a time slot \( S \) from \([1, W_j]\) to transmit. Before their transmissions, they just listen to the channel. At each time slot, based on the feedback information, the algorithm behaves as follows:

- If there is a successful transmission feedback, then the successful node is discovered by all other nodes. It turns to keep silent, and quits the neighbor discovery transmission, just listening. All remaining nodes start a new round with a new contention window \( W_{j+1} \). This is called adaptive backoff.

- If there is a collision feedback, then all transmission nodes in the collision slot enter a re-contention round with a new contention window \( W'_j \)
while other reception nodes quit the re-contention. The process goes on until a node is discovered. This is called history-aware collision resolution that is based on the historical information.

Note that the optimal value of $W_j$ and $W'_j$ can be obtained by theoretical analysis to make the best trade-off.

We illustrate our key idea by Figure 1 to make our algorithm easy to understand. Taking nodes $(A, B, C, D)$ for example, all of them are in a clique, which means that they are in the transmission range of each other. For the sake of simplicity, we assume $W_1 = W_2 = W'_1 = 3$. At the beginning, all nodes choose $S \in [1, W_1]$ to backoff. Here both nodes $A$ and $B$ choose slot 0 to transmit, $C$ chooses slot 1, and $D$ chooses slot 2. At the first sub-slot of the first slot, obviously a collision occurs; nodes $(C, D)$ cannot decode the error packets and feedback an error message at the second sub-slot. Then, nodes $(A, B)$ know that there is a collision, and $(C, D)$ cancel the scheduled transmission. $(A, B)$ start a new round in slot 1. In the new round, only $A$ and $B$ participate in the contention, choosing a backoff in $[1, W'_1]$. Both of them choose slot 2 to transmit, and a collision happens again. A new round starts in slot 3, in which $A$ chooses slot 3, and $B$ chooses slot 4. Then $A$ transmits successfully. $B$ cancels the scheduled transmission, and begins a new round with nodes $(C, D)$ with the contention window $[1, W_2]$. The process goes on until all nodes are discovered.

### 3.2 Formal Description

We use a state transform diagram including states and events to describe our algorithm. Given the feedback model with two sub-slots in one time slot, we classify three types of events according to the time they happened: before a time slot, at the first sub-slot and at the second sub-slot. Supposing there is $j$ nodes undiscovered now. The meaning of these events are explained as
follows:

**Section 1: Before a time slot**

At this time, following events may happen to a node:

- **$E_{0r}$**: re-deciding the random back-off slots $W$ from $[0, W_j - 1]$;
- **$E_{0s}$**: re-deciding the random back-off slots $W$ from $[0, W'_j - 1]$;
- **$E_{0d}$**: counting down the value of $W$ for transmissions;
- **$E_{0e}$**: increasing the value of $W$ to delay transmissions;
- **$E_{idle}$**: doing nothing.

**Section 2: At the first sub-slot**

Here, reception nodes will get packets from transmission nodes. Packets may be correctly decoded or not. Thus, three types of events may happen to a reception node:

- **$E_{1s}$**: receiving packets successfully;
- **$E_{1f}$**: failing to receive packets;
- **$E_{1i}$**: the channel is idle.

For a transmission node, following events may happen:

- **$E_{1t}$**: transmitting at the first sub-slot.

**Section 3: At the second sub-slot**

At the second sub-slot, following events may happen to a node who senses the channel:

- **$E_{2y}$**: receiving collision feedbacks;
- **$E_{2n}$**: receiving collision feedbacks;
- **$E_{2t}$**: transmitting a collision feedback;
- **$E_{2a}$**: choosing a new backoff slot.

Next we will present the states of nodes at different times, which is critical for understanding our algorithms. There are five states in each section. At the beginning of neighbor discovery, all nodes are in

- **$S_A$**: node $i$ needs to decide (or re-decide) the random backoff slot $W$ from $[0, W_j - 1]$.

If a node transmits successfully in the current slot, it changes state to

- **$S_F$**: node $i$ has been discovered by all its neighbors. In this state, node $i$ keeps in the silent mode for the rest of time, which means node $i$ just senses the channel to discover new neighbors.

Other nodes change to state $S_A$ to begin the new time slot. If there is a collision, based on the feedback information, nodes can classify themselves into transmission nodes and reception nodes. For those transmission nodes who transmitted a packet at the last first sub-slot, their states change to
FIGURE 2
State Transition Diagram in Our Algorithm
• $S_B$: node $i$ needs to re-decide the random back-off slot $W$ from $[0, W'_j - 1]$, the same behaviour as the random back-off algorithm depicted in the above part.

For those reception nodes who received a collision packet at the last first sub-slot, their states change to

• $S_C$: node $i$ chooses to delay $W'_j$ slots to avoid potential collisions.

Note that in a network, there exists some nodes who are two-hop away from transmitters. These nodes can detect feedback signals at the last second sub-slot. In order to prevent the potential transmissions, we also make these nodes in state $S_C$.

Then if the current slot is idle, for those backoff nodes, they change to state

• $S_D$: it is a special state for those backoff nodes in a new round. If $W = 1$, it means that node $i$ intends to transmit at this slot. Otherwise, $W = W - 1$.

Compared with backoff nodes, silent nodes have different actions. In state $S_C$, they change the backoff period $W$ to $-W'_j$. Therefore, they need to increase $W$ to realize the delay action. So the new state is

• $S_E$: it is a special state for those silent nodes in a new round. If $W = -1$, it means that the silent period of $W'_j$ will end in the next slot. Otherwise, $W = W + 1$.

For nodes with different states, different events will bring them to different states. However, some events only happen in certain states, e.g., $S_A$ with $E_{0r}$. The valid combinations and transitions are shown in Figure 2. Based on this state transition diagram, we present our history-based random backoff algorithm in Algorithm 1.

**Distributed Implementation:** Our algorithm can be easily applied to a network environment with the unknown number of neighbors. It only needs to remove the while loop of line 5 and line 28 and add a time waiting process. For example, when a node is set to state $S_F$, it keeps on the reception status for a period. When a node finds the channel just remaining idle in a period, e.g., a contention window $W$, it terminates the neighbor discovery process cause all its neighbors finish discovery processes with state set to be $S_F$ in this case. When all nodes finish their discovery processes, the neighbor discovery for the whole network also ends.
Algorithm 1: History-Aware Adaptive Back-off Algorithm

**Input:** a clique of $n$ nodes; phase $j$, contention window $W_j$, re-contention window $W'_j$

**Output:** time slots for node $i$ to discover all its neighbors

1: define $k = n$ as the number of neighbors undiscovered

2: define $T_i = 0$ as the current time slot

3: define $S_i = S_A$ as the current state

4: define $W = 0$ as the transmission slots for each round

5: while $k > 0$ do

6: $T_i = T_i + 1$

7: if $S_i = S_A$ then $W \in [1, W_{n-k}]; S_i = S_D$

8: elif $S_i = S_B$ then $W \in [1, W_{n-k}]; S_i = S_D$

9: elif $S_i = S_C$ then $W = -W_{n-k}'; S_i = S_E$

10: elif $S_i = S_D$ then $W = W - 1$

11: elif $S_i = S_E$ then $W = W + 1$

12: end if

13: Node $i$ transmits or receives at the first sub-slot

14: if $W \neq 1$ and fail to receive packets then $S_i = S_C$

15: elif $W \neq 1$ and packets are successfully received then

16: $k = k - 1$

17: if $S_i \neq S_F$ then $S_i = S_A$ end if

18: end if

19: if node $i$ receives error packets at the first sub-slot, node $i$

20: if $W \neq 1$ and detects energy at the second sub-slot then

21: $S_i = S_C$

22: elif $W = 1$ and detects energy at the second sub-slot then

23: $S_i = S_B$

24: elif $W = 1$ and does not detect energy at the second sub-slot then

25: $S_i = S_F$

26: elif $W = 0$ and $S_i = S_E$ then $S_i = S_A$

27: end if

28: end while

29: return $T_i$
4 THEORETICAL ANALYSIS

This section gives a theoretical analysis of our algorithm. The metric is discovery time and energy consumption, two important concerns of neighbor discovery. In order to show the key idea, we give the analysis under a single clique of \( n \) nodes, and verify the theoretical results under the network scenario through simulations. Moreover, we will first establish the model for contention window \( W \) to achieve adaptive backoff. We further conduct an analysis on re-contention window \( W' \) to utilize historical information to reduce collisions. Note that the optimal value depends on phases, i.e., for phase \( j \) we have \( W_j \) and \( W'_j \). The important notations used in this section are summarized in Table 1.

4.1 Analysis on Adaptive Backoff Approach

**Discovery Time**

Let \( S_j \) be the state when there are \( j \) nodes undiscovered. In each round, there are two states: a node transmits successfully, or there is a collision. Then, the former probability can be denoted by \( P_{sc} \), and the latter \( P_{fc} \). Let \( X_j \) be the time slots needed to discover all \( j \) nodes. According to the conditional expectation formula, we have

\[
E[X_j] = E[E[X_j|Y]] = P_{sc}E[X_j|Y_{sc}] + P_{fc}E[X_j|Y_{fc}] \quad (1)
\]

where \( P_{sc} \) denotes the probability that the current round is successful, and \( P_{fc} \) denotes the collision probability. Actually, \( P_{sc} = P_{j-1} \) and \( P_{fc} = P_j \).

For a successful round, we can divide \( X_j \) into \( X_{j-1} \) and \( T'_j \), where \( T'_j \) denotes the time slots spent in the current round. For the collision round, \( X_j \) can be divided into \( X'_j \) and \( T'_j \), where \( X'_j \) denotes the time slots spent on discovering \( j \) nodes after the current collision round. Therefore, we have

\[
E[X_j|Y_{sc}] = E[(X_{j-1} + T'_j)|Y_{sc}] = E[X_{j-1}|Y_{sc}] + E[T'_j|Y_{sc}] \quad (2)
\]

\[
E[X_j|Y_{fc}] = E[(X'_j + T'_j)|Y_{fc}] = E[X'_j|Y_{fc}] + E[T'_j|Y_{fc}] \quad (3)
\]

Notice that the situation of current round is independent to previous rounds, so \( E[X_{j-1}|Y_{sc}] = E[X_{j-1}] \) and \( E[X'_j|Y_{fc}] = E[X_j] \). Then, we have

\[
E[X_j|Y_{sc}] = E[X_{j-1}] + E[T'_j|Y_{sc}] \quad (4)
\]

\[
E[X_j|Y_{fc}] = E[X_j] + E[T'_j|Y_{fc}] \quad (5)
\]
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n$</td>
<td>The number of nodes in a clique.</td>
</tr>
<tr>
<td>$X_j$</td>
<td>Time slots needed to discover $j$ nodes.</td>
</tr>
<tr>
<td>$S_j$</td>
<td>The state that there are $j$ nodes undiscovered in the current slot.</td>
</tr>
<tr>
<td>$W_j$</td>
<td>The initial contention window for $S_j$.</td>
</tr>
<tr>
<td>$W'_j$</td>
<td>The contention window after first collision for $S_j$.</td>
</tr>
<tr>
<td>$Y_{sc}$</td>
<td>The event that a node transmits successfully in the current round.</td>
</tr>
<tr>
<td>$Y_{fc}$</td>
<td>The event that there is a collision in the current round.</td>
</tr>
<tr>
<td>$P_{j,j}$</td>
<td>The probability that there is a collision in the current round for $S_j$, $Pr[Y_{fc}] = P_{j,j}$.</td>
</tr>
<tr>
<td>$P_{j,j-1}$</td>
<td>The probability that a node transmits successfully in the current round for $S_j$, $Pr[Y_{sc}] = P_{j,j-1}$.</td>
</tr>
<tr>
<td>$P_j$</td>
<td>The probability that nodes choose a slot to transmit when there are $j$ nodes undiscovered.</td>
</tr>
<tr>
<td>$T_j$</td>
<td>The expected time slots from $S_j$ to $S_{j-1}$.</td>
</tr>
<tr>
<td>$T'_j$</td>
<td>Time slots spent on the current round.</td>
</tr>
<tr>
<td>$N_j$</td>
<td>Energy consumption from $S_j$ to $S_0$.</td>
</tr>
<tr>
<td>$N'_j$</td>
<td>Energy cost from $S_j$ to $S_{j-1}$.</td>
</tr>
<tr>
<td>$Tx$</td>
<td>Transmission energy consumption in the first sub-slot.</td>
</tr>
<tr>
<td>$Rx$</td>
<td>Reception energy consumption in the first sub-slot.</td>
</tr>
<tr>
<td>$Ax$</td>
<td>Feedback transmission energy consumption in the second sub-slot.</td>
</tr>
<tr>
<td>$Bx$</td>
<td>Feedback reception energy consumption in the second sub-slot.</td>
</tr>
<tr>
<td>$Sh_m$</td>
<td>In the previous round, $m$ nodes transmit, esp. $Sh_{j+1} = S_j$.</td>
</tr>
<tr>
<td>$Ph_{m,k}$</td>
<td>The transfer probability that from state $Sh_m$ to state $Sh_k$. In other words, $m$ nodes participate in the round, while only $k$ nodes transmit.</td>
</tr>
<tr>
<td>$Th_m$</td>
<td>The expected time cost from $Sh_m$ to the other states.</td>
</tr>
<tr>
<td>$Eh_m$</td>
<td>The expected time cost needed for a node to finish the current phase from state $Sh_m$, esp. $Eh_{j+1} = T_j$.</td>
</tr>
<tr>
<td>$Nh_m$</td>
<td>The expected energy cost needed for a node to finish the current phase from state $Sh_m$.</td>
</tr>
</tbody>
</table>
With equation (4) and (5), we can rewrite equation (1) as

\[
E[X_j] = E[X_{j-1}] + \frac{P_{sc}E[T'_j|Y_{sc}] + P_{fc}E[T'_j|Y_{fc}]}{P_{sc}}
\]  

(6)

For \(S_j\), \(P_{sc} = P_{j,j-1}\), \(P_{fc} = P_{j,j}\), and

\[P_{j,j-1} + P_{j,j} = 1\]  

(7)

we can define

\[T_j = \frac{P_{j,j-1}E[T'_j|Y_{j,j-1}] + P_{j,j}E[T'_j|Y_{j,j}]}{P_{j,j-1}}\]  

(8)

Obviously \(T_j\) is the expectation time cost from state \(S_j\) to \(S_{j-1}\), i.e., the expectation time of phase \(j\). Note that \(E[X_0] = 0\), so the expectation of overall discovery time is

\[E[X_n] = \sum_{j=1}^{n} T_j\]  

(9)

If we can minimize \(T_j\) for each \(j\), we can minimize \(E[X_n]\). So we try to derive a formula about \(T_j\).

Note that the current contention window is \(W_j\) for phase \(S_j\). In each time slot the clique has only three states: idle, collision, and successful transmission. Moreover, if there is a collision or a successful transmission in slot \(i\), it means that the channel is idle in the past \(i-1\) slots, since the collision or successful transmission leads to the end of the current phase.

For a successful transmission state, it can happen in slot \(i\) with probability \(j(1-i/W_j)^{j-1}/W_j\), where \((1-i/W_j)^{j-1}\) means \(j-1\) nodes never transmit in all \(i\) slots. So we have

\[P_{j,j-1} = \sum_{i=1}^{W_j} \frac{j}{W_j} (1 - \frac{i}{W_j})^{j-1}\]  

(10)

Then we have

\[P_{j,j-1}E[T_j|Y_{j,j-1}] = P_{j,j-1} \sum_{i} iP(T_j = i|Y_{j,j-1})\]

\[= \sum_{i} iP(T_j = i, Y_{j,j-1}) = \sum_{i=1}^{W_j} \frac{ij}{W_j} (1 - \frac{i}{W_j})^{j-1}\]  

(11)
Similarly, for a collision slot, we have

\[
P_{j,j}E[T_j|Y_{j,j}] = \sum_{i=1}^{W_j} i \left( (1 - \frac{(i-1)}{W_j})^j - (1 - \frac{i}{W_j})^j - \frac{j}{W_j} (1 - \frac{i}{W_j})^{j-1} \right)
\]  

(12)

where \((1 - (i - 1)/W_j)^j\) denotes the probability that the channel is idle in the past \(i - 1\) slots, \((1 - i/W_j)^j\) means that the channel is idle for all \(i\) slots, and \(j(1 - i/W_j)^{j-1}/W_j\) means only one node transmits at slot \(i\).

Combining equation (8), (9), (11) and (12), we have

\[
E[X_n] = \sum_{j=1}^{n} T_j = \sum_{j=1}^{n} \sum_{i=1}^{W_j} i \frac{N_{j-1}}{j} W_j^{i-1} 
\]

(13)

Note that it is not easy to derive a close-form formula for \(E[X_n]\) if we do not make any approximations. However, approximations may lose some accuracies. Instead, we keep this form and use numerical calculation to derive the optimal \(W_j\) for each \(T_j\) by simulations.

Energy Consumption

The analysis of energy consumption is similar to that of discovery time. Let \(T_x\) be the energy consumption for transmissions, \(R_x\) for receptions, \(A_x\) for feedback transmissions, and \(B_x\) for feedback receptions. According to the conditional expectation formula, energy consumption of the clique of \(S_j\) is

\[
E[N_j] = E[E[N_j|Y]] = P_{j,j-1}E[N_j|Y_{j,j-1}] + P_{j,j}E[N_j|Y_{j,j}] 
\]

(14)

\[
E[N_j] = \sum_{j=1}^{n} N_j' 
\]

(15)

where

\[
N_j' = \frac{P_{j,j-1}E[N_j|Y_{j,j-1}] + P_{j,j}E[N_j|Y_{j,j}]}{P_{j,j-1}} 
\]

(16)

\[
P_{j,j-1}E[N_j|Y_{j,j-1}] = \sum_{i=1}^{W_j} \left( T_x + ((iN - 1)R_x + iNB_x) \frac{j}{W_j} (1 - \frac{i}{W_j})^{j-1} \right) 
\]

(17)
\[ P_{j,i}E[N_j|Y_{j,i}] = \sum_{i=1}^{W_j} \sum_{k=2}^{j} (kT_x + (N - k)A_x + (iN - k)R_x + ((i - 1)N + k)B_x) \times \left( \frac{j}{k} \right) \frac{1}{W_j} (1 - \frac{i}{W_j})^{j-k} \]  

(18)

Using equation (7), (15), (16), (17) and (18), we can get a formula for \( E[N_j] \).

### 4.2 Analysis on History-Aware Approach

In order to accelerate the process in a phase, we use the history-aware approach to handle collisions. Thus, we need to optimize the new contention window \( W'_j \) for \( S_j \). Our algorithm improves the process of backoff in a single phase. Therefore, we only need to consider the situation from \( S_j \) to \( S_{j-1} \).

In order to present the procedure clearly, a state transition diagram is given in Figure 3.

**Discovery Time**

Let us consider \( S_j \). Let \( Sh_j \) be the state that \( j \) nodes transmit in the previous round, and \( Sh_{j+1} \) is the starting state of \( S_j \). Note that \( Sh_1 \) means only one node transmits, and the phase terminates definitely. Let \( Eh_m \) be the expected time slots needed for a node to finish the current phase from state \( Sh_m \) to state \( Sh_1 \). Note that \( Eh_{j+1} \) is \( T_j \). We need to minimize \( Eh_{j+1} \) to get the optimal \( W'_j \).

At the beginning of state \( Sh_{j+1} \), we use the contention window \( W_j \) with transmission probability in the single slot \( P_j = 1/W_j \) obtained from equation (13). When a collision happens, \( Sh_{j+1} \) exactly transfers to state \( Sh_2 - Sh_j \). After the first collision, we need to decrease the number of potential transmission nodes and reduce the length of the contention window from \( W_j \) to \( W'_j \). Now the transmission probability in the single slot is \( P'_j = 1/W'_j \). After a successful transmission, the state transfers to state \( Sh_1 \), and there are \( j - 1 \) nodes undiscovered in the clique. The current phase ends. According to the conditional expectation formula, we have

\[
Eh_m = \begin{cases} 
  \sum_{k=1}^{j+1} Ph_{j+1,k}Eh_k + Th_{j+1} & \text{if } m = j + 1 \\
  \sum_{k=1}^{2} Ph_{m,k}Eh_k + Th_m & \text{else } 2 \leq m \leq j 
\end{cases}
\]  

(19)

where \( Ph_{m,k} \) denotes the transfer probability from \( Sh_m \) to \( Sh_k \), \( Th_{j+1} \) and \( Th_m \) denotes the expected time cost from \( Sh_m \) to \( Sh_k \).
For the case that \( k \) nodes transmit simultaneously at slot \( i \) with \( m \) nodes intending to transmit during this round, the probability can be expressed as\
\[
\binom{m}{k} P_j^k (1 - iP_j)^{m-k}.
\]
Therefore, the transfer probability \( P_{hm,k} \) from state \( S_h_m \) to state \( S_h_k \) can be represented as

\[
P_{hm,k} = \begin{cases} 
W_j \sum_{i=1}^{j} \binom{j}{k} P_j^k (1 - iP_j)^{j-k} & \text{if } m = j + 1 \\
W_j' \sum_{i=1}^{m} \binom{m}{k} P_j'^k (1 - iP_j')^{m-k} & \text{if } 2 \leq m \leq j \\
0 & \text{otherwise}
\end{cases}
\]

(20)

The time cost \( T_h_m \) can be represented as

\[
T_h_m = \begin{cases} 
\sum_{k=1}^{j} \sum_{i=1}^{j} \binom{j}{k} P_j^k (1 - iP_j)^{j-k} & \text{if } m = j + 1 \\
\sum_{k=1}^{m} \sum_{i=1}^{m} \binom{m}{k} P_j'^k (1 - iP_j')^{m-k} & \text{if } 2 \leq m \leq j \\
0 & \text{otherwise}
\end{cases}
\]

(21)
We can rewrite equation (19) as \( PX = 0 \), where
\[
P = \begin{bmatrix}
Ph_{1,1} - 1 & 0 & \ldots & 0 & 0 & Th_1 \\
Ph_{2,1} & Ph_{2,2} - 1 & \ldots & 0 & 0 & Th_2 \\
Ph_{3,1} & Ph_{3,2} & \ldots & 0 & 0 & Th_3 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
Ph_{j,1} & Ph_{j,2} & \ldots & Ph_{j,j-1} - 1 & 0 & Th_j \\
Ph_{j+1,1} & Ph_{j+1,2} & \ldots & Ph_{j+1,j} - 1 & -1 & Th_{j+1}
\end{bmatrix}
\] (22)

\[
X = \begin{bmatrix}
Eh_1 \\
Eh_2 \\
\vdots \\
Eh_{j+1}
\end{bmatrix}
\] (23)

Note that \(-1\) in (22) is got from the right side of equation (19). Because \( Eh_1 = 0 \), by solving the linear system \( PX = 0 \), we can get the expression of \( Eh_{j+1} \) (exactly \( T_j \) defined in Table 1), and derive the optimal backoff window \( W_j' \) after collisions happened. We minimize \( Eh_{j+1} \) (or \( T_j \)) for each case that \( j \) nodes undiscovered to get an optimal \( W_j' \). Then, we calculate the minimum \( E[X_n] \) as
\[
E[X_n] = \sum_{j=1}^{n} T_j = \sum_{j=1}^{n} Eh_{j+1}
\] (24)

Note that \( Sh_m, Ph_{m,k}, \text{ and } Eh_m \) are defined in each phase, which are separated across different \( S_j \). Due to the same reason, we use numerical calculation and find that \( W_j' \) usually has a small value by simulations.

**Energy Consumption**

The analysis of energy consumption is similar to the analysis of discovery time. The energy cost \( Nh_m \) with \( m \) nodes undiscovered can be represented as
\[
Nh_m = \begin{cases} 
N_1 + \sum_{k=2}^{j} \sum_{i=1}^{W_j} C_1 P_1 & \text{if } m = j + 1 \\
N_3 + \sum_{k=2}^{j} \sum_{i=1}^{W_j'} C_1 P_2 & \text{else } 2 \leq m \leq j
\end{cases}
\] (25)

where
\[
N_1 = \sum_{i=1}^{W_j} j P_j (1 - i P_j)^{j-1} (T x + (in - 1) R x + in B x)
\] (26)
\[ C_1 = kTx + (n - k)Ax + (in - k)Rx + ((i - 1)n + k)Bx \]  
\[ P_1 = \binom{j}{k} P^k_j (1 - iP_j)^{j-k} \]  
\[ P_2 = \binom{m}{k} P'^k_j (1 - iP'_j)^{j-k} \]  
\[ N_3 = \sum_{i=1}^{W_j'} (Tx + (in - 1)Rx + inBx)mP'^k_j (1 - iP'_j)^{j-1} \]  

Then, we substitute \( Th_m \) to \( Nh_m \) for (19), and calculate the total energy cost with the same steps as the calculation of total discovery time.

**Hybrid Optimization**

Discovery time and energy consumption can be combined to set up a hybrid optimization function, and this function is in general specified by the target application. That is, we have

\[ f(E[X_n], E[N_n]) = \alpha E[X_n] + \beta E[N, n] \]  

The weights \((\alpha, \beta)\) can be assigned with certain values to meet the demand of a specific application. For example, if the demand is in time of emergency, a large \( \alpha \) with a small \( \beta \) will ensure quick discovering, though possibly with higher energy consumption. On the other hand, for the energy saving demand, a small \( \alpha \) with a large \( \beta \) will save the energy cost.

## 5 PERFORMANCE EVALUATION

### 5.1 Simulation Setup

Our simulation environment includes two types of setting: the clique and the network. In a clique setting, nodes are neighbors mutually. In a network setting, we generate a 3km×3km region, and nodes are randomly distributed with the transmission range of 150m. The average number of neighbors per node is set from 1 to 30 so as to see different performance of algorithms. The number of nodes in a network is determined by the average of neighbors, and we run 30 times for each average of neighbors from 1 to 30 to validate the results for different node placements.
We compare our algorithm with the ALOHA-with-feedback \[9\] \[12\] algorithm. In \[3\] \[12\], the advantage of ALOHA-with-feedback algorithm is obvious when it is compared with other algorithms without feedback. So, in this paper, we will not compare our algorithm with ALOHA-like algorithms.

5.2 Simulation Results

Evaluation In Cliques

It is hard to give the theoretical comparison of the neighbor discovery time between our algorithm and the ALOHA-with-feedback algorithm. Thus, for different clique sizes, we calculate the expectation of neighbor discovery time by our algorithm (denoted by \(E_0(x)\)) based on the theoretical analysis in Section 4, and compare it with the ALOHA-with-feedback algorithm \[9\] \[12\] (denoted by \(E_1(x)\)). Figure 4 depicts the result of \(E_1(x) - E_0(x)\) as the clique size changes. The value of \(E_1(x) - E_0(x)\) is always greater than 0, that is, our algorithm needs shorter time to accomplish neighbor discovery than the ALOHA-with-feedback algorithm. As we can see in Figure 4, the value of \(E_1(x) - E_0(x)\) becomes larger when the clique size increases from 2 to 200. In fact, the clique size of 200 is large enough when considering the realistic network. The theoretical result shows our algorithm gets better performance than the ALOHA-with-feedback algorithm.

We implement both algorithms, and run each algorithm 30 times for each clique size with the range from 2 to 200. The average neighbor discovery time by our algorithm (denoted by \(A_0(x)\)) has almost the same value with the
theoretical result $E_0(x)$. Also, the average discovery time of the ALOHA-
with-feedback algorithm (denoted by $A_1(x)$) is almost the same as $E_1(x)$.
As Figure 5 shows, the result of $A_1(x) - A_0(x)$ is similar to that of $E_1(x) -
E_0(x)$ shown in Figure 4 when the clique size’s range is from 2 to 200. The
simulation result confirms our theoretical analysis. Furthermore, Figure 6 and
Figure 7 show that our analysis is close to simulation results for our algorithm,
both on discovery time and energy consumption.

The analysis results of the clique can also be used to give the average
neighbor discovery time and average energy cost for nodes in a network. As
we simulate networks with different densities, the result is close to our anal-
ysis. It means that the clique analysis can be applied to a network, which has
also been verified by [9].

Evaluation In Networks
We also evaluate the performance of the history-aware adaptive backoff algo-

rithm and ALOHA-with-feedback algorithm under the network environment.
In the network, nodes have no idea of their neighbors immediately after they
are deployed. Therefore, they can’t decide the optimal transmitting proba-
bility. In fact, we use the average number of neighbors (denoted by $N$) of
the whole network for the initial transmitting probability setting. For the
ALOHA-with-feedback algorithm, we set the initial transmitting probability
(denoted by $P_1$) to $1/(N + 1)$, the optimal probability in the clique anal-
ysis [9]. When collisions happen, we decrease the transmitting probability
from $P_1$ to $P_1/(1 + P_1)$; when the channel is idle, we increase the transmitting probability from $P_1$ to $P_1/(1 - P_1)$. This setting helps the ALOHA-with-feedback algorithm get better performance. For the history-aware adaptive backoff algorithm, we set the initial contention window for a node as $W = N + 1$, that is, the transmitting probability $P_2 = 1/W = 1/(N + 1)$; we set reduced contention window for a node as $W' = 3$ when the node was interfered by collisions, namely set $P_2' = 1/3$. Note that $W = N + 1$ is the optimal backoff window through our analysis. $W'$ usually has very small values (stated in Section 4), so we simply set $W' = 3$ here. The values of $P_2$ and $P_2'$ are constant, no matter if a collision happens, or the channel is idle. This setting is disadvantageous to the performance of the history-aware adaptive backoff algorithm. However, our backoff algorithm still gets much better performance than the ALOHA-with-feedback algorithm despiting this disadvantageous settings.

Figure 8 depicts the neighbor discovery time needed in a network setting. Figure 9 shows the energy consumption of the neighbor discovery phase for networks. For the energy consumption, we simply set $Ax: Bx: Tx: Rx = 1: 1: 1: 1$ ($Ax, Bx, Tx, Rx$ are defined in Table II), which can be adjusted by a more accurate power assumption model. We run 30 times for each $N$ (namely density). As we can observe in Figure 8, neighbor discovery with ALOHA-with-feedback endures severe performance degradation when $N$ increases, while neighbor discovery using history-aware adaptive backoff algorithm maintains
a stable performance with less degradation. Different from the clique setting, in a network, collisions happen not just between neighbor nodes. In fact, interferences also exist between nodes with common neighbors, as we mentioned earlier. Therefore, more intensive collisions happen in the network environment. The ALOHA-with-feedback algorithm gets worse performance in this situation, while the history-aware adaptive backoff algorithm utilizes historical information of collisions to divide nodes into different state, thus avoiding the future collision. In Figure 9, we can get an intuitive view. For example, when \( N \geq 10 \), neighbor discovery with the history-aware adaptive backoff algorithm saves more than 40% energy compared to the ALOHA-with-feedback algorithm for each \( N \). When \( N \geq 20 \), it can save more than 50% energy.

6 RELATED WORK

Our work is related to the research on (1) neighbor discovery algorithms; (2) contention resolution algorithms.

**Neighbor discovery algorithms:** In general, neighbor discovery algorithms can be divided into three kinds: (1) deterministic [3]; (2) multi-user detection based [4–6]; (3) randomized [7–14]. Keshavarzian et al. [3] present a deterministic protocol, which depends on additional orthogonal codes design. Deterministic algorithms often requires unrealistic assumptions such as synchronization and priori knowledge of the number of neighbors [9]. On the other hand, multi-user detection based protocols [4–6] focus on the signal processing technical design in the physical layer. Hence, here we focus on probabilistic algorithms due to easy implementation and better performance.

McGlynn et al. [2] design an efficient probabilistic algorithm for neighbor discovery for the first time. Vasudevan et al. [8] propose neighbor discovery algorithms in wireless networks with directional antennas. Recent work [9–11, 13] focuses on the analysis of neighbor discovery running time. Vasudevan et al. [9] prove the running time order of ALOHA-like algorithms without feedback is \( O(n \log n) \) in single-hop networks, where \( n \) is the number of nodes in cliques. You et al. [11] extend the protocol to the scenario of duty-cycle wireless sensor networks, and prove that the discovery time order is \( O(\log \log n) \). Zeng et al. [11] and You et al. [13] consider this problem in wireless networks with multi-packet reception capability, and prove the time order is \( O(n/k \log n) \) in cliques, where \( k \) is the number of allowed simultaneous transmission packets. Mittal et al. [13] study the problem in multi-channel cognitive radio networks. Nevertheless, most work [7, 9–11]
considers algorithms without feedback information. Vasudevan et al. [5] first discuss the improvement of feedback mechanism. They prove that, if nodes have collision detection capability, the neighbor discovery time order can be reduced to $O(n)$ with feedback mechanism. Khalili et al. [12] further discuss the feedback mechanism from the view of the physical layer, and present an improved ALOHA-with-feedback algorithm. Our work can be viewed as a further exploration and an improvement based on the feedback model. We utilize the collision feedback information and historical behavior (transmission or reception in the collision slot) to handle collisions, and thus improve the neighbor discovery efficiency, i.e., using less time and less energy.

**Contention resolution algorithms:** Our approach uses adaptive backoff mechanism, which is similar to the contention window adjustment in contention resolution algorithms [15–19] for MAC designs. However, our objective is different. For MAC designs, their objective is to maximize the channel utilization, i.e., throughput while maintaining time fairness. In other words, contention resolution protocols often focus on finding algorithms to achieve throughput-optimal or delay-optimal, given certain arrival rate and queue model. However, neighbor discovery only wants to send one successful packet for each node and to optimize the overall discovery time or energy consumption. Hence, some nodes can quit the contention in our algorithms, and never transmit after they were discovered. Moreover, we use a history-aware approach to resolve collisions, i.e., only transmission nodes re-contents, which is not fully utilized by contention resolution algorithms.

7 CONCLUSIONS AND FUTURE WORK

In this paper, we propose a history-aware adaptive backoff neighbor discovery algorithm under collision detection and a feedback model. We also give a theoretically analysis of our algorithm on neighbor discovery time and energy consumption, and evaluate our algorithm through simulations comparing to existing algorithms. In the future, we want to complete our work considering unreliable channel models, node crash, and asynchronous scenarios.

8 ACKNOWLEDGMENTS

This work is supported in part by the National Basic Research (973) Program of China (Grant No.2011CB302502, 2011CB302803), the major national science and technology special projects (Grant No. 2010ZX03004-003-03), and China NSF (Grant No. 60825205, 61073152).
REFERENCES


