AWE: A Mac-Layer Encounter Protocol for a Wildlife Tracking System

***Abstract*—**

I. Introduction

II. Background and Related Work

Encounternet [1]

III. System Model

In this paper, we study the encounter problem in a wildlife tracing system. We call individual animals as *agents*, and *peers* are referred as other agents that distinguish from a specific one. The definition of the *encounter process* is formulated as follows.

Definition 1: Encounter is defined as the process that an agent detects and records other peer(s) if they keep a period of close proximity $\Delta \leq D$ in the wildlife tracking system.

In the following, we describe the system model for theoretical analysis in this paper.

A. Communication Model

In the wildlife tracing system *Encounternet*, the encounter behavior is a common biological phenomenon and happens when more than one agents gather closely, constituting a single clique of size k ($k \geq 2$). Note that, k is not known to each agent and the whole clique composes a sing-hop network for communication due to the proximity.

Each agent is equipped with a radio tag. An agent that has its radio on can choose to be in the $transmit$ state or the $listen$ state:

- **Transmit state:** an agent transmits (broadcasts) a message containing its ID on the channel;
- **Listen state:** an agent listens on the channel to receive messages from peers.

We also call an agent keeps in the *listen* state for a period of consecutive slots as *quiet* state.

Suppose time is divided into synchronized slots of equal length $2\hat{t_0}$ [2], [3] , where $\hat{t_0}$ is assumed to be sufficient to finish a complete communication process (one agent transmits a message including its ID and a peer receives the message).

An agent transmits successfully in a time slot if and only if it is the only one transmitting and all the other peer(s) will receive its message and record its ID in this single-hop network. Otherwise the channel is detected as *idle* if there is no transmission and *busy* if there are simultaneous messages incurring collisions on the channel.

In the wildlife tracing system *Encounternet*, on the one hand, each agent is equipped with an energy-restricted tag; on the other hand, encounter process happens occasionally, and thus it is a waste of battery energy if an agent turns on the radio while it does not encounter with any peer(s) at the moment. Therefore, in order to keep a balance between the energy consumption and the efficiency of the encounter process, we introduce the duty cycle mechanism [4].

Duty cycle mechanism. An agent has the capability to turn off the radio to save energy for most of the time, and may only be active (transmitting or receiving) during a fraction θ of the time.

Incorporating the duty cycle mechanism into the Mac layer of the radio tag, in each time slot an agent u_i is able to adopt an action as:

$$s_i^t = \begin{cases} Sleep & sleep\ with\ probability\ (1-\theta_i) \\ Transmit & transmit\ with\ probability\ \theta_i p \\ Listen & listen\ with\ probability\ \theta_i(1-p) \end{cases}$$

Duty cycle is defined as the fraction of time an agent turns its radio on, which is formulated as:

$$\theta_i = \frac{|\{t : 0 \leq t < t_0, s_i(t) \in \{Transimit, Listen\}\}|}{t_0}.$$

Next, we introduce another efficient technique called collision detection mechanism. This technique is carried out by the physical carrier sensing [5], which is part of the 802.11 standard, and provided by a Clear Channel Assessment (CCS) circuit.

Collision detection mechanism. A listening agent can distinguish whether the channel is *idle* or *busy*, apart from successfully receiving a message.

B. Problem formulation

We formulate the problem in this paper as follows.

Problem 1: Consider $\hat{T}$ slots which is a small enough period in reality. We define an encounter problem as to design a protocol to guarantee all the agents in the clique can receive message from each other at least once if they encounter for at least $\hat{T}$ time slots and record the encounter process.

We look into the problem and find the key challenge is the uncertainty of dynamic movements of agents. Despite the dynamicity in this real system, when $\hat{T}$ is short enough relative to the time required for an agent to move a step (a short distance) in reality (e.g., less than 1 second), we can make a reasonable assumption that the communication connectivity of the agents is stable during each $\hat{T}$ time slots.

IV. ADAPTIVE WILDLIFE ENCOUNTER PROTOCOL

In this section, we present our Adaptive Wildlife Encounter (AWE) protocol. The pseudo-code of the protocol is given in Algorithm 2 and Algorithm 3.

AWE consists of two stages: detecting stage and connecting stage.

- **Stage 1: detecting stage.** In this stage, an agent attempts to detect whether there are nearby peers, regardless of who they are.
- **Stage 2: connecting stage** In this stage, an agent attempts to identify the nearby peer(s) and record their IDs to its log.

Initially, each agent starts from the detecting stage. In the detecting stage, an agent turns its radio to the $sleep$ state most of the time, and switches to $transmit$ state or $listen$ state at intervals. In the connecting stage, agents only switch between $transmit$ state and $listen$ state.

The key idea of AWE is that, any single agent keeps in detecting stage to reduce ineffective energy consumption. When encounter happens, it detects the existence of nearby peers and turns to the connecting stage to identify those peers (or a peer) as fast as possible and record the encounter process to its log. when the encounter process is determined to be finished in the connecting stage, the agent turns back to the detecting stage.

Remark 1: In the AWE protocol, there is no need to synchronize the stage between agents and AWE still works when encounter peers are in different stage, e.g., an agent in detecting stage bursts into a stable clique in connecting stage. The proof of correctness will be presented in section V.

In the following, we describe the operations of these two stages in detail.

A. Detecting stage

In the detecting stage, energy efficiency is achieved by the duty cycle mechanism, e.g., denote the predefined duty cycle for all the agents is θ, the tag radio of each agent will work θT_0 slots in every period of T_0 slots.

However, it is very ineffective when two agents encounter and one is in $sleep$ state while the other is transmitting or listening. To technically achieve synchronizing the time that agents turn on the radio without extra cost, we introduce the technique of Relax Difference Set (RDS) [6]. We use the RDS technique to guarantee that every encounter pair of agents turn on the radio in the same slot at least once in each round T_0.

RDS is an efficient tool to construct cyclic quorum systems. The definition is:

Definition 2: A set $R = \{a_1, a_2, ..., a_k\} \subseteq Z_T$ (the set of all non-negative integers less than T) is called a RDS if for every $d \neq 0 \pmod{T}$, there exists at least one ordered pair (a_i, a_j) such that $a_i - a_j \equiv d \pmod{T}$, where $a_i, a_j \in D$.

We now give an example to explain how RDS works to help synchronization. Suppose the duty cycle is set as 0.4, i.e., there are 4 active slots in every 10 time slots. It is easy to show that $R = \{1, 2, 3, 6\}$ is a RDS under Z_{10}:

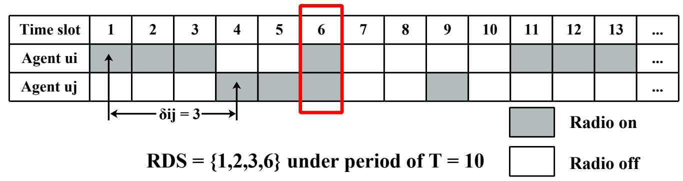

Fig. 1. A example of how RDS works to help synchronization. Consider a period of ten slots and the time drift between two agents u_i and u_j is 3. There exists an ordered pair $(6, 3)$ in the constructed RDS such that $6 - 3 \equiv 3 \pmod{10}$. Thus they will determinately turn on the radio at the same slot in every period T, which is the 6^{th} slot in every period of u_i and the 3^{th} slot in that of u_j respectively.

$$2-1=1, \quad 3-1=2, \quad 6-3=3, \quad 6-2=4,$$
$$6-1=5, \quad 2-6=6\ (mod\ 10), \quad \ldots, \quad \ldots,$$

In every period of ten slots, for any $i = \{0, 1, \ldots, 9\}$, if $i \in R$, then the agent turns on its radio in the i^{th} slot in this period; otherwise it turns off the radio to the $sleep$ state. An example is depicted in Figure 1.

Algorithm 1 RDS Construction Algorithm

1: $R := \emptyset$; $\lambda := \lceil \sqrt{T} \rceil$, $\mu := \lceil \frac{\lceil \sqrt{T} \rceil}{2} \rceil$;
2: **for** $i = 1 : \lambda$ **do**
3: $\quad R := R \cup i$;
4: **end for**
5: **for** $j = 1 : \mu$ **do**
6: $\quad R := R \cup (1 + j * \lambda)$;
7: **end for**

It has been proved that any RDS must have cardinality $|R| \geq \sqrt{T}$ [6]. We present a linear algorithm to construct a RDS with cardinality $\lceil \frac{3\sqrt{T}}{2} \rceil$ under Z_T in Alg. 2.

We show the correctness of the construction formally.

Lemma 1: Set $R = \{r_0, r_1, ..., r_{\lambda+\mu-1}\}$ constructed in Alg. 2 is a RDS, where $|R| = \lambda + \mu = \lceil \sqrt{N} \rceil + \lceil \frac{\lceil \sqrt{N} \rceil}{2} \rceil \approx \lceil \frac{3\sqrt{N}}{2} \rceil$.

Proof: Obviously, if there exists one ordered pair (a_i, a_j) satisfying $a_i - a_j \equiv d \pmod{N}$, an opposing pair (a_j, a_i) exists such that $a_j - a_i \equiv (N - d) \pmod{N}$. Thus we only need to find at least one ordered pair (a_i, a_j) for each $d \in [1, \lfloor N/2 \rfloor]$.

In the construction, λ in Line 1 is the smallest integer satisfying $\lambda^2 \geq N$. Every d in range $[1, \lfloor N/2 \rfloor]$ can be represented as: $d = 1 + j \times \lambda - i$, where $1 \leq j \leq \mu, 1 \leq i \leq \lambda$. Thus, there exists $a_j = 1 + j \times \lambda$ from Line. 3 and $a_i = i$ from Line. 6 satisfying $a_j - a_i \equiv d$. Then, the lemma can be derived. ■

Based on the RDS, we present the operations in the detecting stage as Alg. 2. Each time slot is divided into two sub-slots. When an agent turns on its radio according to the RDS sequence, in the first sub-slot it transmits a beacon with probability ω_0 and listens with probability $(1 - \omega_0)$. In the second sub-slot:

Algorithm 2 Detecting Algorithm

$T_0 := \lceil \frac{9}{4\theta^2} \rceil$; $\omega_0 := \frac{1}{2}$; $t := 0$;
Invoke Alg. 2 to construct $R = \{r_0, r_1, ..., r_{\lceil \frac{3\sqrt{T}}{2} \rceil}\}$ under Z_T;
while $True$ **do**
  if $(t+1) \in R$ **then**
    In the first sub-slot:
    Transmit a beacon with probability ω_0 and listen with probability $1 - \omega_0$;
    In the second sub-slot:
    if the agent is in $listen$ state in the first sub-slot **then**
      if detects energy (a beacon or a collision by multiple beacons) in the first sub-slot **then**
        Transmit a beacon and turn to the **connecting stage**;
      end if
    else if detects energy (a beacon or a collision by multiple beacons) in this sub-slot **then**
      Turn to the **connecting stage**;
    end if
  else
    $Sleep$ in the whole slot;
  end if
  $t := (t+1)\%T_0$;
end while

1) The agent is in $listen$ state in the first sub-slot:
   - if the agent detects a beacon (or beacons) in the first sub-slot, it transmits a beacon (a bit is OK) as an acknowledgement on the channel in the second sub-slot and turn to the connecting stage; otherwise it does nothing.
2) The agent is in $transmit$ state in the first sub-slot:
   - if the agent detects a beacon (or beacons) in this sub-slot, it turns to the connecting stage; otherwise it does nothing.

As discussed before, the aim of this stage is to detect nearby peer(s) as fast as possible (if exists), and either successful transmission or detecting busy on the channel activates the agent to switch to the connecting stage. Hence we fix the transmitting probability as $\omega_0 = \frac{1}{2}$.

B. Connecting stage

In the connecting stage, agents attempt to identify the nearby peers and record the encounter to its log. A successful identification happens only if the agent is listening and exactly one peer is transmitting.

The collision detection (CD) mechanism is incorporated in this stage to increase of efficiency. This mechanism enables the listening agent to notify the transmitting peers of the transmission outcomes, and thus they take measures to reduce the collisions if not successful.

In this stage, every $\hat{T}$ slots consists of a round. Each agent repeats the operations in Alg. 3 round by round and it turns to the connecting stage when it cannot find any peer in a complete round, as the operation in Line 25.

Algorithm 3 Connecting Algorithm

1: $t := 0$; $\omega_t := \zeta$;
2: **while** $True$ **do**
3:   ***In the first sub-slot:***
4:   Transmit a message containing ID with probability ω_t and listen with probability $1 - \omega_t$;
5:   ***In the second sub-slot:***
6:   **if** the agent is in $listen$ state in the first sub-slot **then**
7:     **if** reveive a message successfully **then**
8:       Record the source ID and transmit a beacon;
9:       Set $\omega_{t+1} := \frac{\omega_t}{(1+\epsilon)}$;
10:     **else if** channel is idle **then**
11:       Set $\omega_{t+1} := min\{(1+\epsilon) \cdot \omega_t, \zeta\}$;
12:     **else**
13:       Set $\omega_{t+1} := \frac{\omega_t}{(1+\epsilon)}$; \\ the channel is busy
14:     **end if**
15:   **else**
16:     **if** detect beacons in this sub-slot **then**
17:       $\omega_{t+1} := 0$;
18:     **else**
19:       Set $\omega_{t+1} := \frac{\omega_t}{(1+\epsilon)}$;
20:     **end if**
21:   **end if**
22:   $t := (t+1)$;
23:   **if** $t == \hat{T}$ **then**
24:     **if** no peer is found in this round **then**
25:       Turn to the **detecting stage**;
26:     **else**
27:       $t := 0$; $\omega_t := \zeta$;
28:     **end if**
29:   **end if**
30: **end while**

As discussed in section IV, $\hat{T}$ is relatively short in real world, thus the communication connectivity stays stable in a round. However, due to the dynamic movements of agents, the communication connectivity may change from round to round, so all the parameters will be initialized at the beginning of each round and adaptively adjusted later according to the transmission outcome.

Each time slot is divided into two sub-slots. Agents execute transmission or reception in the first sub-slot, and in the second sub-slot take actions responding to the outcome of the previous sub-slot (success/fail to transmit/receive a message).

Since the number of the nearby peers is unknown to each agent, the transmitting probability is initially set as ζ, which is a pre-defined constant.

In the first sub-slot of each slot t, an agent transmits a message containing ID with probability ω_t and listen with probability $(1 - \omega_t)$. In the second sub-slot:

1) The agent is in $listen$ state in the first sub-slot:
   - if the agent receives a message successfully, it decodes and records the source ID in the message,

and transmits a beacon (a bit is OK) as an acknowledgement on the channel in the second sub-slot.
  - if the channel is idle, this means there is a chance to transmit successfully and it multiplies its transmitting probability by a factor $(1+\epsilon)$ (no larger than the pre-defined constant ζ).
  - if the agent detects collisions, it divides its transmission probability by a factor $(1+\epsilon)$.
2) The agent is in $transmit$ state in the first sub-slot:
  - if the agent detects beacons in this sub-slot, this means its previous message has been successfully received by its nearby peers, and it keeps listening in all the rest first sub-slots of this round, which is called *quiet* state.
  - if the agent detects nothing in this sub-slot, it means its previous message failed to propagate due to simultaneous transmissions. Thus it divides its transmission probability by a factor $(1+\epsilon)$.

Factor $(1+\epsilon)$ is a pre-defined constant to adjust the transmission adaptively. For simplify, we set $(1+\epsilon) := 2$ for analysis in the next Section. In the end of a complete round, if there is no peer detected in this whole round, which indicates the encounter process is finished, the agent turns to the detecting stage .

V. Analysis of the AWE Protocol

In this section, we prove that with a clique of k agents, AWE protocol guarantees each agent can record all the peers in $O(k)$ slots with high probability. Note that, k is not known in the execution of the protocol but just for analysis in this section. Formally, this conclusion is derived from the following theorem.

Theorem 1: Consider an encounter process of k agents (k is a integer $k \geq 2$) :

(1) An agent in detecting stage will switch to the connecting stage in $O(\theta^{-2})$ slots with high probability. Recall that θ is the pre-defined duty cycle and can be seen as a constant.

(2) When all the agents are in the connecting stage, all of them will successfully transmit once and turn to the *quiet* state in $O(k)$ slots with high probability.

We prove these two conclusions in Theorem 1 in Section V-B, and Section V-A respectively. Then, We explain how the AWE protocol deals with the dynamic movements of agents and records a duration encounter process in Section V-C.

A. Time bound for detecting stage

When an encounter happens, an agent in detecting stage will switch to connecting stage very soon. We derive this conclusion from the following two lemmas.

Lemma 2: Consider any two agents u_i and u_j in detecting stage. In each period T_0, they will turn on the radio in the same slot at least once.

Proof: Assume the time drift between u_i and u_j is δ_{ij} (mod T_0). In the RDS constructed under T_0, there exists at least one ordered pair (a_i, a_j) such that $a_i - a_j \equiv \delta_{ij}$ (mod T_0). Thus the a_i^{th} slot in a period of u_i is exactly the a_j^{th} slot in a period of a_j^{th} and both of them turn on the radio in this slot according to Alg. 2 Line 4, which completes the proof. ■

Lemma 3: When an encounter happens, an agent in detecting stage will switch to the connecting stage in $\frac{\ln \eta}{\ln(\frac{1}{2}+\frac{1}{2^k})}\lceil \frac{9}{4\theta^2} \rceil$ slots with high probability.

Proof: First, consider k agents in the detecting stage at the beginning. By Lemma 2, any two peers can turn on their radios in the same slot at least once during every period of T_0. For a specific agent, the probability it detects a peer in a period of T_0 is at least $Pr \geq 1-(\frac{1}{2}+\frac{1}{2^k})$. Hence given a small enough constant η, it holds with high probability that an agent detects peer(s) in $\frac{\ln \eta}{\ln(\frac{1}{2}+\frac{1}{2^k})}$ periods, which is $\frac{\ln \eta}{\ln(\frac{1}{2}+\frac{1}{2^k})}\lceil \frac{9}{4\theta^2} \rceil$ slots in total.

Consider two agents encounter at a time and one turn to the connecting stage first. Then the other will switch to the connecting stage very soon since the prior one in connecting stage increases its transmission probability (because there is no duty cycle in the connecting stage) to let other peers detect it.

■

B. Time bound for connecting stage

Consider the connecting stage. We analyze the upper bound of slots for all the agents in the clique to transmit successfully. Note that, an agent transmitting successfully will be recorded by all the other peers in the clique, and then it switches to the *quiet* state. Denote S_t as the set of agents which have not switched to the *quiet* state in time slot t, and $|S_t|$ is the cardinality. Thus the upper bound of the protocol is the maximum time slots for a clique to turn to $|S_t| = 0$.

In the following, we first show the upper bound of slots it takes from the beginning to $|S_t| \leq \log k$ in Lemma 4. Then we show the upper bound it takes to $|S_t| = 1$ in Lemma 14. Finally we present that it takes for all agents to successfully transmit (i.e, $|S_t| = 0$.) in $O(k)$ in Lemma 15.

Lemma 4: At time $T_2 = T_1 + \gamma_1 \cdot k = O(k)$ it holds with high probability that $|S_{T_2}| \leq \log k$.

To prove this lemma, we need to introduce and prove some small lemmas at first. We review two useful lemmas as follows.

Lemma 5: Consider a set of l agents, $u_1, u_2, \ldots, u_l$. For an agent u_i, it transmits with probability $0 < \omega(u_i) < \frac{1}{2}$. Let p_0 denote the probability that the channel is idle in a time slot; and p_1 denote the probability that there is exactly one transmission in a time slot. Then $p_0 \cdot \sum_{i=1}^{l}\omega(u_i) \leq p_1 \leq 2 \cdot p_0 \cdot \sum_{i=1}^{l}\omega(u_i)$

Lemma 6: With $a_i \in [0, \frac{1}{2}]$ for $i = 0, 1, \ldots,$ it holds that

$$4^{-\sum_i a_i} \leq \prod_i (1-a_i) \leq e^{-\sum_i a_i}. \qquad (1)$$

The proof of Lemma 5 can be referred to [7] and the proof of Lemma 6 can be referred to [8]. Based on these two lemmas, we get the following conclusion.

Lemma 7: For a time slot $T > 0$ with $|S_t| \geq \log k$, if there exist constants $\alpha_1, \alpha_2 \geq 1$ such that $\alpha_1 \leq \sum_{u \in S_t} \omega_t(u) \leq \alpha_2$, then with constant probability there is one active node switching to the *quiet* state in each time slot.

Proof: In each time slot, the channel is idle with probability at least $4^{-2\alpha}$, and there is exactly one transmission on the channel with probability at least $\alpha \cdot 4^{-2\alpha}$. By the Chernoff bound, it holds that with constant probability (given α_1 and α_2), there are one agent switch to the quiet state in each slot. ■

Next we show that after all the agents turn to the connecting stage, the summation of the total agents' transmission probabilities will go between α_1 and α_2 soon, where α_1 and α_2 are constants defined in Lemma 7.

Lemma 8: For a time slot with $\sum_{u \in S_t} \omega_t(u) = \alpha$, it holds that $Pr[\sum_{u \in S_t} \omega_{t+1}(u) \leq \alpha \cdot \frac{3}{4}] \geq \frac{7}{8}$ for large enough α.

Proof: The probability that there are more than one agents transmit in time slot t is at least $1 - exp\{-\alpha\}$ according to Equation (1). All the agents will halve their transmission probabilities if the channek is not idle in slot t. Denote X as the random variable that indicates the value of $\sum_{u \in S_{t+1}} \omega_{t+1}(u)$. We get,

$$Pr[X = \sum_{u \in S_t} \frac{\omega_t(u)}{2}] \geq 1 - exp\{-\alpha\}$$

which is at least $7/8$ when α is large enough. Hence,

$$X \leq \frac{7}{8} \cdot \frac{1}{2} \cdot \alpha + \frac{1}{8} \cdot 2\alpha < \frac{3}{4} \cdot \alpha.$$

Therefore, it holds with high probability that for large α, $Pr[\sum_{u \in S_t} \omega_{t+1}(u) \leq \alpha \cdot \frac{3}{4}] \geq \frac{7}{8}$. Note that, we did not consider the effect when an agent turns to the *quiet* state, which only makes the summation decrease and hence is not harmful. ■

Lemma 9: There exists a constant $\hat{\alpha_2} > 1$, such that among $\gamma \log k$ slots (not necessarily consecutive) with $\sum_{u \in S_t} \omega_t(u) \geq \hat{\alpha_2}$ and sufficiently large $\gamma > 0$, there are at least $\frac{3}{4}\gamma \log k$ slots with $\sum_{u \in S_{t+1}} \omega_{t+1}(u) < \frac{3}{4} \sum_{u \in S_t} \omega_t(u)$, with probability $1 - O(k^{-1})$.

Proof: Let $T := \gamma \log k$, and X_t be the random variable that indicates the value of $\sum_{u \in S_{t+1}} \omega_{t+1}(u) / \sum_{u \in S_t} \omega_t(u)$. Then by Lemma 8, it holds that $Pr[X_t \leq \frac{3}{4}] \geq \frac{7}{8}$. Let Y_t be the binary random variable that takes value 1 if $X_t \leq \frac{3}{4}$. Note that given $\sum_{u \in S_t} \omega_t(u) \geq \hat{\alpha_2}$, $E[Y_t] \geq \frac{7}{8}$ always hold. Hence, $E[\sum_{t=1}^{T} Y_t] \geq T \cdot \frac{7}{8}$, and it holds that $Pr[\sum_{t=1}^{T} Y_t \leq T \cdot \frac{3}{4}]$ by the Chernoff bound. That is with probability $1 - O(k^{-1})$, there are at least $T \cdot \frac{3}{4}$ slots t with $\sum_{u \in S_{t+1}} \omega_{t+1}(u) / \sum_{u \in S_t} \omega_t(u) \leq 3/4$, which completes the proof. ■

Lemma 10: There exists a constant $\hat{\alpha_2} > 1$, such that during any period of $\gamma \log k$ slots with sufficiently large $\gamma > 0$, the probability that within the considered period there is a slot t with $\sum_{u \in S_t} \omega_t(u) \leq \hat{\alpha_2}$ is $1 - O(k^{-1})$.

Proof: Denote $T := \gamma \log k$ and the period of T starts from slot t_0. By Lemma 9, with probability at least $1 - On^{-1}$, it holds that

$$\begin{aligned}\sum_{u \in S_{t_0+T}} \omega_{t_0+T}(u) &\geq \sum_{u \in S_{t_0}} \omega_{t_0}(u) \cdot (\frac{3}{4} \cdot \frac{3}{4} \cdot \frac{3}{4} \cdot 2)^{\frac{T}{4}} \\ &= \sum_{u \in S_{t_0}} \omega_{t_0}(u) \cdot (\frac{27}{32})^{\frac{T}{4}}.\end{aligned}$$

Since $\sum_{u \in S_{t_0+T}} \omega_{t_0+T}(u) < k$ and $T = \gamma \log k$, we know that $\sum_{u \in S_{t_0+T}} \omega_{t_0+T}(u)$ is at most $\hat{\alpha_2}$ for large enough γ. ■

Lemma 11: There exists a constant $\hat{\alpha} \geq 0.01$ such that for any time t with $\sum_{u \in S_t} \omega_t(u) = \hat{\alpha}$, it holds that

$$Pr[\sum_{u \in S_{t+1}} \omega_{t+1}(u) \geq \hat{\alpha} \cdot \frac{4}{3}] \geq \frac{7}{8}. \quad (2)$$

Proof: Denote X as the random variable that indicates the value of $\sum_{u \in S_{t+1}} \omega_{t+1}(u)$. The probability that there are no transmissions in time slot t is at least $4^{-\hat{\alpha}}$. Hence,

$$Pr[X = \sum_{u \in S_t} 2 \cdot \omega_t(u)] \geq 1 - 4^{-\hat{\alpha}}$$

which is at least $7/8$ when $\hat{\alpha}$ is close to 0.01. Hence,

$$X \geq \frac{7}{8} \cdot 2 \cdot \hat{\alpha} + \frac{1}{8} \cdot \frac{1}{2} \hat{\alpha} > \frac{4}{3} \cdot \hat{\alpha}.$$

Therefore, it holds with high probability that for small $\hat{\alpha}$ close to 0.01, $Pr[\sum_{u \in S_t} \omega_{t+1}(u) \leq \hat{\alpha} \cdot \frac{4}{3}] \geq \frac{7}{8}$. ■

Lemma 12: There exists a constant $\hat{\alpha_1} > 0$, such that among $\gamma \log k$ slots (not necessarily consecutive) with $\sum_{u \in S_t} \omega_t(u) \leq \hat{\alpha_1}$ and sufficiently large $\gamma > 0$, there are at least $\frac{3}{4}\gamma \log k$ slots with $\sum_{u \in S_{t+1}} \omega_{t+1}(u) \geq \frac{4}{3} \sum_{u \in S_t} \omega_t(u)$, with probability $1 - O(k^{-1})$.

Proof: Denote $T := \gamma \log k$, X_t as the random variable that indicates the value of $\sum_{u \in S_{t+1}} \omega_{t+1}(u) / \sum_{u \in S_t} \omega_t(u)$, and Y_t to be the binary random variable that takes value 1 if $X_t \leq \frac{4}{3}$. Note that given $\sum_{u \in S_t} \omega_t(u) \geq \hat{\alpha_1}$, $E[Y_t] \geq \frac{7}{8}$ always hold. Hence, $E[\sum_{t=1}^{T} Y_t] \geq T \cdot \frac{7}{8}$, and it holds that $Pr[\sum_{t=1}^{T} Y_t \leq T \cdot \frac{3}{4}]$ by the Chernoff bound. That is with probability $1 - O(k^{-1})$, there are at least $T \cdot \frac{4}{3}$ slots t with ■

Lemma 13: Let t_0 be the first time slot in which $\sum_{u \in S_t} \omega_t(u)$ drops below $\hat{\alpha_2}$. in the subsequent $T := \tau \cdot \log k$ slots where $\tau > 0$ and k is large enough, the following hold:

(1) there are at least $\frac{3}{4} \cdot T$ slots t with $\sum_{u \in S_t} \omega_t(u) \leq \alpha_2$, where $\alpha_2 > \hat{\alpha_2}$ is a constant.

(2) there are at least $\frac{3}{4} \cdot T$ slots t with $\sum_{u \in S_t} \omega_t(u) \geq \alpha_1 \cdot k$, where $\alpha_1 < \hat{\alpha_1}$ is a constant.

With Lemma 7, 8, 9, 10, 11, 12 and 13, we now prove Lemma 4.

Then we show that when the number of agents not switching to the *quiet* state is less than $\log k$, the process accelerates to only one agent not in *quiet* state with high probability. This is formulated as the following lemma.

Lemma 14: At time $T_3 := T_2 + \gamma_2 \cdot \log k = O(k)$ it holds with high probability that $|S_{T_3}| = 1$.

Proof: After T_2 slots, it holds with high probability that $|S_{T_2}| \leq \log k$. Then it takes at most $\gamma_2 \cdot \log k$ slots to keep $\beta_1 \leq \sum_{u \in S_t} \omega_t^u \leq \beta_2$, where β_1 and β_2 are two constants. Afterwards, there is a time slot $T_3 := T_2 + \gamma_2 \cdot \log k$ such that $|S_{T_3}| = 1$, where $\gamma_2 > 0$ is a large enough constant. Otherwise during the period from T_2 to T_3, with high probability there are more than $\log k$ agents switching to the *quiet* state. ■

Finally we need to prove the following lemma.

Lemma 15: For time T_3 when there is only one agent not switching to *quiet* state, at time $T_4 := 2 \cdot T_3 + \log k = O(k)$ it holds with high probability that this agent successes to transmit.

Proof: Since at time T_3 there is only one agent still attempting to transmit, the transmission probability will get to ζ at time $2 \cdot T_3$ if it keeps listening. Note that if agent u transmit with probability ζ for $\log k$ slots, then with high probability there exist one time slot in which agent u successes to transmit. ■

According to the Lemma 4, 14 and 15, we get the upper bound $O(k)$ of the AWE protocol with respect to the clique size k. Note that, since the round of each agent may not be synchronized, AWE requires an overlap of at least T_4 time slots in a complete round between each agent. This can be easily solved as we set $\hat{T} := 2 \cdot T_4$ in Algorithm 3.

C. Dynamicity

VI. Evaluation

VII. Conclusions

References

[1] I. I. Levin, D. M. Zonana, J. M. Burt, and R. J. Safran, "Performance of encounternet tags: Field tests of miniaturized proximity loggers for use on small birds," *Plos One*, vol. 10, no. 9, p. e0137242, 2015.

[2] M. Xu, M. Zhao, and S. Li, "Lightweight and energy efficient time synchronization for sensor network," in *International Conference on Wireless Communications, NETWORKING and Mobile Computing, 2005. Proceedings*, 2005, pp. 947–950.

[3] F. Sivrikaya and B. Yener, "Time synchronization in sensor networks: a survey," *Network IEEE*, vol. 18, no. 4, pp. 45–50, 2004.

[4] W. Zhang, Y. Zhou, M. A. Suresh, and R. Stoleru, "Performance analysis and tuning of coexisting duty cycling wifi and wireless sensor networks," in *IEEE International Conference on Sensing, Communication, and NETWORKING*, 2017, pp. 1–9.

[5] X. Yang and N. Vaidya, "On physical carrier sensing in wireless ad hoc networks," in *INFOCOM 2005. Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, 2005, pp. 2525–2535 vol. 4.

[6] W.-S. Luk and T.-T. Wong, "Two new quorum based algorithms for distributed mutual exclusion," in *Distributed Computing Systems, 1997., Proceedings of the 17th International Conference on.* IEEE.

[7] A. Richa, C. Scheideler, S. Schmid, and J. Zhang, "A jamming-resistant mac protocol for multi-hop wireless networks," in *Twenty-Seventh ACM Symposium on Principles of Distributed Computing*, 2010, pp. 45–54.

[8] S. Daum, M. Ghaffari, S. Gilbert, F. Kuhn, and C. Newport, "Maximal independent sets in multichannel radio networks," in *ACM Symposium on Principles of Distributed Computing*, 2013, pp. 335–344.