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

***Abstract*—**

I. Introduction

II. System Background and Related Work

Encounternet [1]

III. System Model

A. Communication Model

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.

Suppose time is divided into slots of equal length t_0 , which 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).

Duty cycle mechanism. In the wildlife tracing system *ATLAS*, each agent (animal) is equipped with energy-restricted tags. An agent has 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. In each time slot, an agent u_i 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:

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

B. Interference Model

We consider graph model as the interference model in our protocol. Graph model is a popular one that enables the development of efficient algorithms for crucial networking problems. Some other models, such as the signal to interference plus noise ratio (*SINR*) model, are more complicated and lack good algorithmic features. In addition, it is shown that SINR can be transformed to the graph model by particular means.

In the graph model, we assume that the communication range of each agent is D meters and an agent u_i can discover its nearby peer u_j in time slot t if and only if u_j is the only peer in u_i's range that transmits and u_i is listening in the slot.

Suppose the distance between two nodes u_i and u_j is $\Delta_{i,j}$. The connectivity of the edge $e_{i,j}$ in the graph model is formulated as:

$$\omega_e = \begin{cases} 0 & D \le \Delta_{i,j} \\ 1 & \Delta_{i,j} \le D \end{cases}$$

An agent u_i can discover its peer u_j in time slot t if and only if u_j is the only peer transmitting in the range of u_i ($\Delta_{i,j} \le D$) and u_i is listening in the slot. An agent suffers from collisions when it receives simultaneous messages.

Collision detection mechanism. A listening agent can distinguish whether collisions occur or no nearby peer is transmitting, apart from successful discovery.

C. Problem formulation

In this paper, we study the encounter problem in a wildlife tracing system. We first give the definition of the *Encounter process*.

Definition 1: Encounter is defined as the process that an agent detects and monitors a period of close proximity a wildlife tracking system to another peer.

The problem in this paper is formulated as follows:

Problem 1: We define a D-encounter problem as a least time interval of T_0 during which agent u_i and u_j satisfying $\Delta_{i,j}(t) \le D$.

The aim is to design a protocol to guarantee any D-encounter can be detected and recorded within T time slots with high probability (T is the upper bound of the protocol, e.g. $T = O(n \ln n)$).

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 T is relatively short (compared to the kinematic velocity of agents) in the real world (e.g., less than 1 second), we assume the communication connectivity of the agents is constant in T time slots.

#Clique description (TBD) ...

IV. Adaptive Wildlife Encounter protocol

In this section, we introduce 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 the encounter process 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 and $Listen$ 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) and record the encounter process to its log. After the encounter process is finished, the agent turns back to the detecting stage.

Remark 1: In the AWE protocol, there is no need to synchronize the stage between agents. For example, two agents encounter at first and both turn to the connecting stage, then a peer come nearby them starting from the detecting stage. AWE still works in this case. The proof of correctness will be presented in section V.

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

A. Detecting stage

In the detecting stage, every T_1 time slots is a complete round. 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_1 time slots in every period of T_1. 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, we first introduce the Relax Difference Set (RDS). We use the RDS technique to guarantee that every encounter pair of agents turn on the radio in the same time slot at least once in each round T_1.

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, as depicted in Figure **??** (TBD). Suppose the duty cycle is set as 0.4, i.e., there are four active slots in every 10 time slots. it is easy to show that $R = \{1, 2, 3, 6\}$ is a RDS under Z_{10}:

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

In every round of 10 time 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 round; otherwise it turns to the $Sleep$ state. Consider any two agents in the detecting stage: if the time drift between their time rounds is δ, there exists at least one ordered pair (a_i, a_j) in R such that $a_i - a_j \equiv \delta \pmod{10}$. That is, in each period T_1, any two agents at least once turn on the radio in the same time slot.

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 $|R| \geq \sqrt{T}$ [2]. We present a linear algorithm to construct a RDS with cardinality $\lceil \frac{3\sqrt{T}}{2} \rceil$ under Z_T in Alg. 1.

We show the correctness of the construction formally.

Lemma 1: Set $R = \{r_0, r_1, ..., r_{\lambda+\mu-1}\}$ constructed in Alg. 1 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 within 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. ■

Algorithm 2 Detecting Algorithm

1: $T := \lceil \frac{9}{4\theta^2} \rceil$; $\omega := \frac{1}{2}$; $t := 0$;
2: Invoke Alg. 1 to construct $R = \{r_0, r_1, ..., r_{\lceil \frac{3\sqrt{T}}{2} \rceil}\}$ under Z_T;
3: **while** $True$ **do**
4: $\quad$ **if** $(t + 1) \in R$ **then**
5: $\quad\quad$ Generate a random float $\rho \in (0, 1)$;
6: $\quad\quad$ **if** $\rho < \omega$ **then**
7: $\quad\quad\quad$ Transmit a beacon;
8: $\quad\quad$ **else**
9: $\quad\quad\quad$ Listening;
10: $\quad\quad\quad$ **if** Detects energy (a beacon or a collision by multiple beacons) **then**
11: $\quad\quad\quad\quad$ Turn to **Connecting Stage**;
12: $\quad\quad\quad$ **end if**
13: $\quad\quad$ **end if**
14: $\quad$ **else**
15: $\quad\quad$ $Sleep$;
16: $\quad$ **end if**
17: $\quad$ $t := (t + 1)\%T$;
18: **end while**

Based on the RDS, we present the operations in the detecting stage as Alg. 2. An agent turns on its radio

according to the RDS sequence and in each active slot it transmits a beacon with probability ω and listen with probability $1-\omega$. 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 collisions activate the agent to the connecting stage. Here we fix the transmitting probability as $\omega = \frac{1}{2}$ since it is the optimal probability for two-agent encounter, the most common case in the Encounternet system (??Not sure).

Remark 2: Though $\omega = \frac{1}{2}$ is not the optimal probability in multi-agent encounter cases, the probability that an agent detects a peer in a time slot grows as the number of agents increases. This is because a new agent will not interrupt but help other agents to detect peers if it is in $Transmit$ state in a time slot.

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 only one peer is transmitting.

Algorithm 3 Connecting Algorithm

```
1: t := 0; ω := 1/2; ϵ;
2: while True do
3:    In the first sub-slot:
4:    Transmit a message containing ID with probability ω
      and listen with probability 1 − ω;
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:       else if channel is idle then
10:         ω := min{(1 + ϵ)ω, ζ};
11:      end if
12:   else
13:      if detect energy then
14:         Keep listening in all the rest time slots of this
            round;
15:      else
16:         ω := ω/(1+ϵ);
17:      end if
18:   end if
19:   t := (t + 1);
20:   if t == T then
21:      if no peer is found in this round then
22:         Turn to Detecting Stage;
23:      else
24:         t := 0; ω := 1/2;
25:      end if
26:   end if
27: end while
```

The collision detection (CD) mechanism is incorporated in this stage to increase of efficiency. This mechanism enables the listening agent to notify its transmitting peers of the transmission outcomes, and hence they take measures to reduce the collisions.

Each time slot is divided into two sub-slots and every T_2 complete time slots consists of a round. Agents execute transmission or reception in the first sub-slot, and maintain the EncounterList and reply a beacon in the second sub-slot (if receive a message successfully in the first sub-slot). The algorithm in detecting stage is formulated as Alg. 3.

As discussed in section IV, T_2 is relatively short in real world, thus the communication connectivity stays stable in a round. However, due to the 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.

Since the number of the nearby peers is unknown to each agent, the transmitting probability is initially set as $\frac{1}{2}$. In the first sub-slot of each time slot, an agent transmit a message containing ID with probability ω and listen with probability $1-\omega$.

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 upper bound ζ).
   - if the agent detects collisions, it does nothing.
2) The agent is in $Transmit$ state in the first sub-slot:
   - if the agent detects energy (beacon or collisions) 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 time slots of this round.
   - if the agent detects nothing in this sub-slot, it means ts previous message failed to propagate due to simultaneous transmitting. Thus it divides its transmitting probability by a factor $(1+\epsilon)$.

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

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] 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.