Group-based Discovery in Low-duty-cycle Mobile Sensor Networks

Liangyin Chen*, Yu Gu†, Shuo Guo§, Tian He§, Yuanchao Shu‡†, Fan Zhang‡ and Jiming Chen‡
*College of Computer Science, Sichuan University
† Singapore University of Technology and Design
§Department of Computer Science and Engineering, University of Minnesota
‡ Department of Control, Zhejiang University

***Abstract*—Wireless Sensor Networks have been used in many mobile applications such as wildlife tracking and participatory urban sensing. Because of the combination of high mobility and low-duty-cycle operations, it is a challenging issue to reduce discovery delay among mobile nodes, so that mobile nodes can establish connection quickly once they are within each other's vicinity. Existing discovery designs are essentially pair-wise based, in which discovery is passively achieved when two nodes are pre-scheduled to wake-up at the same time. In contrast, for the first time, this work reduces discovery delay significantly by proactively referring wake-up schedules among a group of nodes. Because proactive references incur additional overhead, we introduce a novel selective reference mechanism based on spatiotemporal properties of neighborhood and the mobility of the nodes. Our quantitative analysis indicates that the discovery delay of our group-based mechanism is significantly smaller than that of the pair-wise one. Our testbed experiments using 40 sensor nodes confirm our theoretical analysis, showing one order of magnitude reduction in discovery delay compared with traditional pair-wise methods with only 0.5%∼8.8% increase in energy consumption.**

I. Introduction

Wireless Sensor Networks have been proposed for use in many challenging applications, such as military surveillance, scientific exploration and structural monitoring. Sustainable deployment of these systems calls for energy-efficient designs. Extensive research has indicated that energy in low-power sensors is consumed mostly by being ready for potential incoming packets, a problem commonly referred to as *idle listening*. For example, the widely used ChipCon CC2420 radio [1] draws 19.7mA when receiving or *idle listening*, which is actually larger than the 17.4mA used when transmitting. More importantly, packet transmission time is usually very small (e.g., about 1 millisecond to transmit a TinyOS packet using a CC2420 radio), while the duration of *idle listening* for reception can be orders of magnitude longer. For example, most environmental applications, such as Great Duck Island [15], sample the environment at relatively low rates (on the order of minutes between samples). With a comparable current draw and a 3∼4 orders of magnitude longer duration waiting for reception, *idle listening* is a major energy drain that, if not optimized, accounts for most energy in communication.

Therefore, the most effective energy conservation technique is to reduce duty-cycle by listening briefly and shutting down radios most of the time (e.g., 99% or more). Such a simple low-duty-cycle operation, however, leads to a challenging issue: how nodes within physical vicinity can discovery each other if they listen to the channel in an asynchronous manner. This issue becomes even more challenging when low-duty-cycle operation is combined with the mobility of sensor nodes in many applications such as in ZerbraNet [7], opportunistic data collection (SNIP) [19] and urban sensing [5]. Because mobility invalidates many assumptions implicit in low power static designs [4], such a combination imposes a time constraint on how fast nodes shall finish discovery before they are physically disconnected.

Node discovery in low-duty-cycle network has attracted research attention in recent years. Previous works mainly focus on how to ensure a pair of nodes can wake up simultaneously through a certain type of scheduling algorithms. Notable discovery designs include: stochastic-based protocols [2], [13], quorum-based protocols [11], [12], [16], [20], Disco [3] and U-Connect [8]. These designs successfully ensure that a pair of nodes finish discovery within a bounded delay. Although literature is encouraging, we believe there are rooms to improve. Specifically, we notice all existing designs essentially are pair-wise based. Discovery is achieved only when two nodes are pre-scheduled to wake-up simultaneously. However, if nodes can share their known schedules with each other, discovery can be achieved in a more proactive and fast manner with small overhead.

This paper presents a *Group-based Discovery* method as a performance add-on to existing pair-wise mobile discovery designs. It essentially builds a schedule reference mechanism among nodes to expedite the discovery process. The operation of schedule reference is straightforward. For example, after node A discovers node B using a traditional pair-wise method, node B can proactively refer (push) the wakeup-schedules of its known neighbors (such as node C) to node A. Consequently, node A can quickly discover node C indirectly via node B. Clearly, excessive reference operations would introduce high overhead in communication. The challenging issue of our work is to design a *selective* reference mechanism to trade off between reducing discovery delay and overhead to achieve that.

In summary, our contributions are as follows:

- To the best of our knowledge, all previous work focuses on scheduling designs for pair-wise discovery. We investigate how group-based discovery can reduce discovery delay with small overhead. We are the first to provide theoretical analysis of group-based discovery delay for mobile low-duty-cycle networks and compared with pair-wise one.
- Utilizing spatiotemporal properties of neighborhood, we propose a selective reference mechanism that can avoid unnecessary references while still speeding up overall discovery process.
- We implement and evaluate our design in a physical testbed consisting of 40 nodes, indicating that our design is suitable for resource constrained sensor nodes. One order of magnitude reduction in discovery delay indicates our design is very effective for mobile sensor network with high density.

The rest of the paper is organized as follows: Section II discusses the related work. Section III introduces the network model and assumptions. Section IV introduces the basic design and makes a theoretical analysis of discovery delay. Section V introduces an advanced selective reference design. Section VI presents the experiment results using a 40-node testbed. Section VII concludes the work.

II. Related Work

Node discovery is nothing new and has a rich literature in both ad hoc and wireless sensor networks. Discovery in always-awake networks mainly focuses on network models with directional antennas [17], [6], [18], while solutions for discovery in duty-cycled networks are highly diverse, especially in mobile environments which impose time constraints on how fast discovery should be finished. Notable ones includes: stochastic-based protocols [13], [2], quorum-based protocols [16], [20], [12], [11], Disco [3], U-Connect [8], multi-channel discovery [9] and collision-aware discovery [10]. In birthday protocol [13], nodes listen, transmit or sleep in a probabilistic round-robin fashion, which statistically trades off between discovery energy with discovery latency. Due to the stochastic nature of its operation, there is no guarantee on the worst-case discovery latency. Quorum-based protocols [16], [20], [12], [11] address this limitation by ensuring existence of overlapped wake-up durations between pair-wise nodes within a bounded time. For instances, in [16], Tseng construct a $m \times m$ grid matrix within contiguous slots. A node arbitrarily picks one column and one row of entries from the matrix to transmit and receive, respectively. Since m is a global parameter, all nodes are required to operate in a symmetric duty cycle setting (i.e., all nodes consume same amount of energy for discovery purpose). To support asymmetric duty-cycle setting, Zheng et al. [20] apply optimal block designs using difference sets to detect neighboring nodes in finite time without requiring slot alignment. Based on their method, the discovery problem in asymmetric duty-cycle setting reduces to an NP-complete minimum vertex cover problem requiring a centralized solution. To provide a distributed solution in an asymmetric design, Disco [3] introduces a neighbor discovery method based on the Chinese Reminder Theorem [14], in which each node selects a pair of primes as period independently based on the requirement of its duty cycle. For example, if node i select T_{i0} and T_{i1} as its working periods, after node i start to work, once its time counter can divide by T_{i0} or T_{i1}, it will wake up, or else be in sleep. In addition to Disco [3], CQS-pair [12] and GQS-pair [11] present wakeup scheduling schemes for heterogeneous quorum-based system. Using heterogeneous quorums, nodes can have different cycle lengths and hence different duty-cycle settings. Recently, U-connect [8] proposes a unified neighbor discovery protocol for symmetric and asymmetric duty cycle settings. Specifically authors show that U-Connect is an 1.5-approximation algorithm for the symmetric asynchronous discovery scenario, and the existing protocols such as Quorum and Disco are 2-approximation algorithms.

Although neighbor discovery techniques are diverse, all of them focus on pair-wise discovery. None of aforementioned works investigate how to increase discovery probability and decrease discovery delay by sharing schedule information among a group of nodes during the discovery process. In this work, we introduce a generic reference mechanism on top of current discovery methods. It serves as a performance add-on to existing discovery methods, therefore is complementary to the state-of-the-art.

III. System Models and Assumptions

In this section, we define the network model and assumptions related to group-based discovery design for mobile networks.

A. Network Model

We assume a network with n mobile nodes running under a low-duty-cycle mode, i.e., a node remains dormant most of time and becomes active only briefly (e.g., less than 5%) to sense and communicate. When a node is in the active state, it can receive packets transmitted from neighboring nodes. When a node is in the dormant state, it turns off all function modules except a timer for the purpose of waking itself up. In other words, a node can wake up to transmit a packet at any time, but can receive packets only when it is in its active state.

The working schedule of a mobile node denotes the active-dormant behaviors of the mobile node over its lifetime. It consists of a set of *active instances*, during which a node can receive packets. Each active instance m at node i can be represented by a tuple (t_m^i, d_m^i), where t_m^i denotes the starting time of the active instance and d_m^i denotes the corresponding duration of the active instance m. Since many sensor node working schedules are periodic [3], it is often sufficient to represent an infinite sequence of active instances, using repeated subsequences with a period time T_i.

Let Γ_i be the working schedule of node i and the number of active instances within a period be M, we can have

$\Gamma_i = \{(t_1^i, d_1^i), (t_2^i, d_2^i), ..., (t_M^i, d_M^i)\}$. According to its working schedule, a node continuously transits its state between active and dormant state. Therefore, the duty-cycle of node i is $\frac{\sum_{m=1}^{M} d_m^i}{T}$.

To simplify our description, in the rest of the paper we assume all active instances have the same durations (τ). When a node is said to be active at time t, it has an active instance that starts at time t with duration of τ. We note that this definition of working schedule can actually accommodate active instances with varying durations. Essentially, if we let τ be the finest granularity of time durations, we can represent any node schedule with the fixed τ.

B. Neighbor Discovery Model

For two neighboring mobile nodes i and j to discover each other, they need to be within each other's communication range and their active instances shall at least partially overlap in time (note strict alignment of time instances is not required). Formally, let the duration that two nodes i and j are within each other's communication range be $[t, t + \Delta t]$, and the working schedule of node i and node j during this Δt time be $\Gamma_i^{\Delta t}$ and $\Gamma_j^{\Delta t}$ respectively, then these two nodes can discover each other if $\Gamma_i^{\Delta t} \cap \Gamma_j^{\Delta t} \neq \emptyset$. The discovery times are the elements in the set $\Gamma_i^{\Delta t} \cap \Gamma_j^{\Delta t}$. For example, if node i and node j are within each other's communication range during time [100, 200], and the active instances of node i and node j during this time interval are $\{135, 178\}$ and $\{116, 178\}$, respectively. Then node i and node j will be able to discover each other during this encounter at time 178. We note similar to legacy pairwise methods (e.g., Quorum [16], [20], [11], [12], Disco [3] and U-Connect [8]), by sending two discovery messages at both beginning and end of an active instance, we can ensure discovery in the presence of clock skew/drift without the assumptions of time synchronization and aligned time instances.

IV. Basic Group-based Discovery Design

In this section, we introduce the basic design of group-based discovery and quantitatively compare our group-based discovery design with legacy pairwise node discovery approaches. Since legacy pairwise designs have effectively handled mobility in the network, in this section, we focus on explaining how our group-based discovery can reduce discovery delay in the network.

A. The Design for Group-based Discovery

In traditional pairwise discovery methods for low-power wireless mobile devices, a node is able to discover a neighboring node if and only if it wakes up at the same time as its neighboring node (such as Disco). Different from pairwise discovery, in our group-based discovery design, we let individual nodes actively share their existing neighbors' working schedules with the new node that they have just discovered. In this way, the new node can quickly become aware of the wake-up times of surrounding nodes and actively verify whether it can communicate with those nodes at their wake-up times.

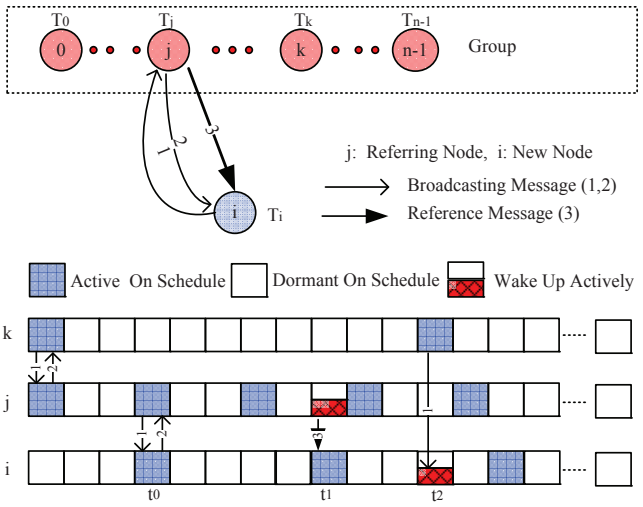


Fig. 1. Design of Group-based Discovery

Figure 1 shows the process of group-based discovery. Without loss of generality, we assume in Figure 1 except node i, all other nodes have discovered each other and formed a group. We note if the initial size of the group is one, group-based discovery behaves the same as the pairwise discovery, because no schedule reference is needed. With more than two nodes in a group, our group-based discovery follows the following steps:

1) For an individual node in the network, according to its working schedule, it will periodically become active during its wakeup time instance and broadcast its existence with its own working schedule. In Figure 1, this is denoted by broadcasting message 1 and 2 from node i and node j, respectively (as a reminder, two messages are needed to accommodate time drift). At time t_0, two nodes i and j wake up with partial overlap and are within each other's communication range, upon successful reception of each other's broadcasting message, these two nodes discover each other and become aware of each other's working schedule.
2) As node j has already discovered its group and is aware of the working schedules of other nodes in the group, it will wake up at the next active instance of node i (time t_1) and send the working schedules of all others in the group to node i. Here we call this neighbor sharing message as the *reference message*, which is shown as message 3 in Figure 1. The node j, which sends out this reference message, is called *referring node*. And for those nodes included in the reference message, we call them *referred nodes*.
3) Upon the reception of the reference message, node i starts to verify one by one whether the referred nodes from node j are indeed its own neighbors. We note *verification can be done silently without additional messages*. Figure 1 shows how verification is conducted: if node k is the node that wakes up first after node i has received the reference message, node i wakeup at

the next active instance of node k (time t_2), trying to receive broadcasting messages from the node k. Upon reception, node i confirms that node k is indeed within its communication range and adds node k to its neighbor table. This verification step continues until node i has finished verifying all referred nodes from node j.

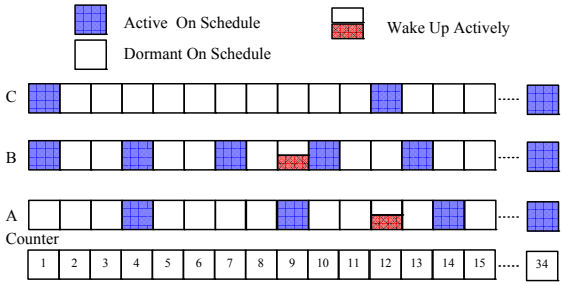


Fig. 2. An Example of Group-based Discovery

Case Study: Figure 2 shows a case study of group-based discovery process for nodes A, B and C, which are all physically within each other's communication range. Firstly, as both B and C wake up at time 1, they discover each other at time 1 and form a group. Then at time 4, node A and node B wake up simultaneously and become aware of each other. After node B discover both node A and node C, it proactively wakes up at time 9 when node A is scheduled to be active, and notifies the working schedule of node C to node A. Finally at time 12, node A wakes up and silently verifies whether node C is its neighbor or not. After time 12, node A has discovered both node B and C. In contrast, if we adopt the pairwise discovery approaches [3], node A can discover node B and C only after time 34.

B. Qualitative Comparison: Group-based vs. Pairwise

This section proves qualitatively the performance gain of the group-based discovery method over the traditional pairwise discovery in terms of discovery delay.

For a pair of neighboring node i and j, assume the duration that they are within each others communication range is T. Let Q be a superset of non-empty subsets under $\mathbb{Z}+_n$. Each element Γ_i in Q is the wake up schedule of node i running a certain pairwise discovery algorithm. Let C be the set of possible discovery times between node i and j, we have $C = \Gamma_i \cap \Gamma_j$. Accordingly, the first discovery delay is $\min C$. Suppose the group-based reference mechanism adds additional wake-up instances to node i, so that it can proactively discover other nodes by augmenting the original schedule Γ_i to $\Gamma_i^{'}$. The new set of possible discovery times between node i and node j therefore is $C^{'} = \Gamma_i^{'} \cap \Gamma_j$. Since $\Gamma_i \subseteq \Gamma_i^{'}$, clearly $C \subseteq C^{'}$. Therefore the pairwise discovery delay $\min C$ is larger than or at least equal to the group-based discovery delay $\min C^{'}$.

C. Analytic Comparison: Group-based vs. Pairwise

In previous section, we qualitatively prove that the group-based discovery design always has smaller or equal discovery delay than that of the pairwise discovery solutions (note the chance of equal discovery delay is extremely small). In this section, we quantitatively compare the difference of discovery delay between pairwise and group-based discovery methods. Without loss of generality, we choose Disco [3] as the underlying pairwise discovery design. Similar derivation can be applied to the existing discovery design as well [13], [2], [16], [20], [12], [11], [8].

For pairwise discovery methods such as Disco [3] and U-Connect [8], they ensure that a pair of neighboring nodes i and j can discover each other within a bounded time $T_{i,j}$. Since pairwise discovery methods guarantee a pair of nodes can discover each other within a bounded time $T_{i,j}$, if they are within each others communication range and $t \geq T_{i,j}$, node i and node j can discover each other with 100% probability. Before time $T_{i,j}$, the probability that node i and node j discovers each other can be represented by $f(i,j,t)$, no matter it is uniformly distributed or not. Consequently, to represent the probability that a new node i discovers a node j in the group before time t, we can use the following equation:

$$P_{ij}(t) = \begin{cases} f(i,j,t), t \in [0, T_{i,j}) \\ 1, t \geq T_{i,j} \end{cases} \tag{1}$$

Then for pairwise discovery methods, the probability distribution function for node i discovering all n nodes in the group before time t can be represented as $P_p(t) = \prod_{j=0}^{n-1} P_{ij}(t)$. The corresponding probability density function is:

$$p_p(t) = \sum_{j=0}^{n-1} [P_{ij}(t)' \prod_{k=0, k\neq j}^{n-1} P_{ik}(t)] \tag{2}$$

To calculate the expected time for node i discovering all n nodes in the group, we have:

$$\begin{aligned} t_p = E_p(t) &= \int_0^{T_{\max,i}} t p_p(t) dt \\ &= \int_0^{T_{\max,i}} t \sum_{j=0}^{n-1} [P_{ij}(t)' \prod_{k=0, k\neq j}^{n-1} P_{ik}(t)] dt \end{aligned} \tag{3}$$

where, $T_{max,i} = Max(T_{i,0}, T_{i,1}, ..., T_{i,n-1})$.

For the group-based discovery method, the probability that a new node i discovers one of the node in the group before time t is $P_g(t) = 1 - \prod_{j=0}^{n-1}(1 - P_{ij}(t))$. The corresponding probability density function therefore can be represented as:

$$p_g(t) = \sum_{j=0}^{n-1} [P_{ij}(t)' \prod_{k=0, k\neq j}^{n-1} (1 - P_{ik}(t))] \tag{4}$$

The expected time for node i discovering at least one node in the group is:

$$\begin{aligned} E_g(t) &= \int_0^{T_{\min,i}} t p_g(t) dt \\ &= \int_0^{T_{\min,i}} t \sum_{j=0}^{n-1} [P_{ij}(t)' \prod_{k=0, k\neq j}^{n-1} (1 - P_{ik}(t))] dt \end{aligned} \tag{5}$$

Where $T_{min,i} = Min(T_{i,0}, T_{i,1}, ..., T_{i,n-1})$. After node i discovers a node, say node j in the group, according to our group-based discovery design, node j would share working schedules of all nodes in its group with node i. Then node i proactively wakes up at the active instances of non-discovered nodes in the group. Consequently, as long as all those non-discovered nodes in the group wake up at least once after node i and node j having discovered each other and node i knowing schedules of nodes in the group, node i would have discovered all nodes in the group. The maximal total time for node i discovering all n nodes in the group therefore can be expressed by the following formula:

$$t_g \leq E_g(t) + 2Max(T_{gap}) \leq E_g(t) + 2T \tag{6}$$

Where T_{gap} is the time gap between two consecutive wakeups of a node pair, which is smaller than a period T. After the first discovery, in the worst case, a node takes another two $Max(T_{gap})$ delay to finish reference and verification.

D. Numeric Comparison: Group-based vs. Pairwise

Based on Equations 3 and 6, we can now numerically show the performance difference in discovery delay between the group-based design and Disco. In Disco, each node i has two independent prime periods (T_{i0}, T_{i1}). The discovery probability of node i (T_{i0}, T_{i1}) and j (T_{j0}, T_{j1}) can be viewed as uniformly distributed within $T_{i,j}$. And $T_{i,j} = \frac{T_{i0}T_{j0}T_{i1}T_{j1}}{T_{i0}T_{j0}+T_{i0}T_{j1}+T_{i1}T_{j0}+T_{i1}T_{j1}}$, which is the expected bounded time that node i and j discover each other. Then, we can represent discovery probability as Equation 7.

$$P_{ij}(t) = \begin{cases} \frac{t}{T_{i,j}}, t \in [0, T_{i,j}) \\ 1, t \geq T_{i,j} \end{cases} \tag{7}$$

We set a $2-20$ nodes network with duty cycle changing from 1.4% to 0.7%, and they all have already known each other's schedules. Then, there is another node, whose duty cycle is 1%, moving close to the group to discover those nodes. Figure 3 shows the discovery delay for Disco and group-based discovery designs under different numbers of nodes. From Figure 3, we can see under all numbers of nodes, the discovery delay of our group-based discovery design is much smaller than that of the Disco discovery design's. For example, when the number of nodes is more than 15, the delay of pairwise discovery is more than 10 times longer than our group-based method. More interestingly, we observe that as number of nodes increases, the discovery delay for our group-based discovery method actually decreases while the discovery delay for pairwise discovery design increases almost linearly with the increasing number of nodes. This is a clear indication that group-based discovery scales well when a network becomes very dense.

V. Advanced Group-based Discovery Design

In Section IV, we introduce the basic concept of group-based discovery. In Basic design, we have a node j announces

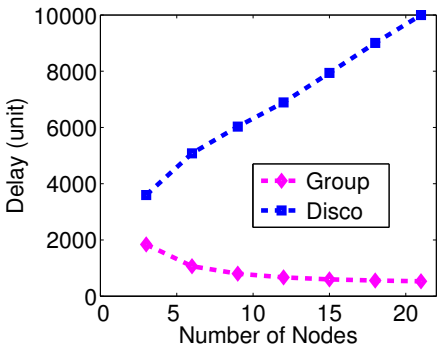


Fig. 3. Delay Comparison of Disco and Group Discovery

all its neighbor information to node i that it has just discovered. This simple solution ensures that the node i newly added into the group would have a complete picture of nodes in the surrounding area, however this would also waste energy unnecessarily, because not all known nodes of node j are neighboring nodes of node i.

In this section, we introduce a selective reference mechanism exploiting spatiotemporal properties of mobile nodes in the network. It is based on a simple rule: node B should avoid referring the schedule of its neighbor node (say C) to node A, if node C is *not* a neighbor of node A. This is because that node A cannot communicate with a non-neighboring node C physically, even after node A knows its wake-up schedule. Following this rule, we can reduce the reference overhead of the referring node B, while still expediting the neighbor discovery process of A.

A. Spatial Selection

In this section, we provide theoretical foundation for referring neighboring nodes based on spatial properties. The main idea of this spatial selection design is to estimate the *closeness (or proximity)* of two neighboring nodes based on the number of common neighbors they share. It is noted that spatial selection does not require calculating the exact distance between two nodes, the estimated distance between two nodes (using neighbor information) merely serves as an indicator of the closeness of two nodes. In the rest of this section, we use the terms *distance* and *closeness* interchangeably.

According to this estimated closeness among different mobile nodes, we then are able to selectively choose the most appropriate neighbors to refer to the newly discovered mobile node and reduce energy consumption for our group-based discovery design.

1) Theoretical Foundation: Intuitively, when two mobile nodes are closer to each other, it is more likely they would share more common neighboring nodes. For the purpose of theoretical analysis, here we assume (i) uniform node distribution in a node's neighborhood, and (ii) unit disk communication model. We note that relaxation of these assumptions only degrade the performance of the protocol, but not the correctness of the design. For example, we can use a conservative radius in the unit disk communication model to increase the possibility of neighborhood at the cost of fewer opportunities for reference.

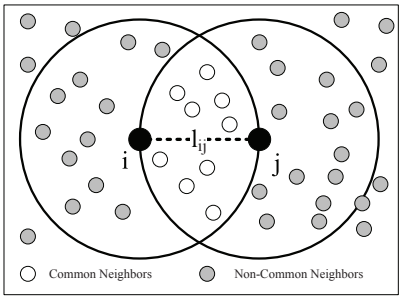


Fig. 4. Overlapping Communication Region Example

As shows in Figure 4, under the assumption of local uniform distribution, the number of common neighbors is proportional to the size of the overlapping region between node i and node j with a distance of l_{ij}. Let $N_{ij}(l)$ denote the number of common neighbors, then we can have the following formula:

$$N_{ij}(l_{ij}) = \frac{\lambda}{\pi R^2}(2R^2 \arccos(\frac{l_{ij}}{2R}) - l_{ij}\sqrt{R^2 - (\frac{l_{ij}}{2})^2}) \quad (8)$$

where λ and R is the average node density and the communication range of the mobile device, respectively.

According to Equation 8, the closeness between two mobile nodes monotonically decides the number of common neighbors they have (assuming local uniform node density). By comparing the neighbor table information of two mobile nodes, we can easily find their common neighbors. Let M_{ij} be the number of common neighbors between node i and j, we estimate the closeness between those two nodes by following formula:

$$l_{ij} = N_{ij}^{-1}(M_{ij}) \quad (9)$$

N_{ij}^{-1} is the inverse function of the function 8.

Using Equation 9, we can estimate the closeness l_{jk} between referring node j and referred node k, as well as the closeness l_{ij} between referring node j and the newly discovered node i

To calculate the probability that node j's neighboring nodes i and k are within each other's communication range, let us look at the illustration shown in Figure 5. From Figure 5, it is clear that if node k falls within the overlapping communication region between node i and node j, node k is a common neighbor for node i and node j. Obviously, if we fix l_{jk} and l_ij, then node k can only be situated on the dashed circle. If $l_{jk} + l_{ij} > R$, node k is the common neighbor of node i and node j, only if node k is located at the dashed circle segment inside the circle i. According to the law of cosines, the angle α in Figure 5 can be represented as $\alpha = \arccos(\frac{l_{jk}^2+l_{ij}^2-R^2}{2l_{jk}l_{ij}})$. Then the probability that the referred node k by referring node j is also the neighbor of node i can be expressed as: $\frac{2\alpha}{2\pi} = \frac{1}{\pi}\arccos(\frac{l_{jk}^2+l_{ij}^2-R^2}{2l_{jk}l_{ij}})$. When $l_{jk} + l_{ij} \leq R$, then it is clear from Figure 5 that node k is always falling into the overlapped region between node i and node j. Therefore, the probability that node k is a common neighbor of node i and j is 100% in this scenario.

By combining above two cases, we can have the following equation to represent the probability that node j's neighboring nodes i and node k are also within each others communication range as:

$$P_{j,ik}(l_{jk}, l_{ij}) = \begin{cases} \frac{1}{\pi}\arccos(\frac{l_{jk}^2+l_{ij}^2-R^2}{2l_{jk}l_{ij}}) & l_{jk} + l_{ij} > R \\ 1 & l_{jk} + l_{ij} \leq R \end{cases} \quad (10)$$

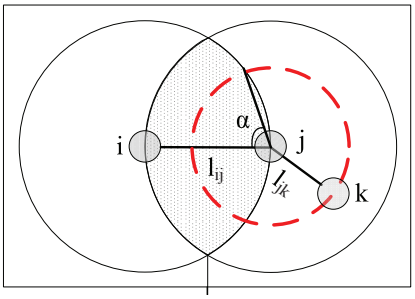


Fig. 5. Neighbor Probability of One Node's Two Neighbors

Clearly, by setting different threshold values for $P_{j,ik}(l_{jk}, l_{ij})$, a node can selectively reference its neighboring nodes to a newly discovered node, therefore tradeoff among energy consumption, discovery delay and discovery probability. For example, if system wants to reduce energy consumption, we should set a high threshold value and have the nodes in the network reference less neighbors for discoveries. On the other hand, when system demands low discovery delay, we should set a low threshold value.

2) Analysis of the Worst and Average Cases: Based on the analysis above, in this section we study the worst and average case that group-based reference would be helpful. By taking the derivative of Equation 8, we have $N_{ij}(l)' = \frac{\lambda(-2+\frac{5}{4}(\frac{l}{R})^2)}{\pi R\sqrt{1-(\frac{l}{2R})^2}} < 0$. Consequently, Equation 8 is a monotonically decreasing function with respect to l. For example, when $l = 0$, which means node i and node j are identically located, we have $N_{ij}(0) = \lambda$. In contrast, when $l = R$, we have $N(R) \approx 0.391\lambda$. This indicates even when node i and node j are at the edge of their communication range (the worst case), they still share about 39.1% of common neighbors.

In addition, when mobile nodes are uniformly distributed in the network, the probability density function for the distance between two neighboring nodes can be represented as:

$$p(l) = \frac{2l}{R^2} \quad (11)$$

Then the expected closeness between two neighboring nodes is:

$$E(l) = \int_0^R p(l) l\, dl = \frac{2}{3}R \quad (12)$$

Given this expected closeness of $\frac{2}{3}R$ between two neighboring nodes, according to Equation 8, in the average case, the number of common neighbors of two neighboring nodes

is approximately 0.5836λ . In other words, if node i and node j are neighboring nodes, another node k which is the neighbor of node i has about 58.36% chance to be the neighbor of node j, a sufficiently large chance to motivate us to use the group-based discovery method.

B. Temporal Selection

Due to the mobility, the neighborhood of a mobile node i also changes dynamically. Therefore we need to have a low-cost and systematic method to discard the stale neighbor information for mobile nodes in the network. For group-based discovery design, this discard of stale neighbor information is particular important as it directly affects the energy consumption for node discoveries. If a node is sending out those stale neighbor nodes, it wastes its own energy for data transmission, as well as the energy for the reception node to verify those stale neighbors. In this section, we discuss theoretical foundations for setting Time-to-live (TTL) values for neighbor information in a mobile network.

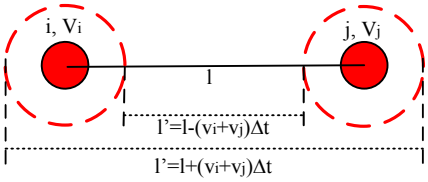


Fig. 6. Temporal Selection Example

Taking scenario in Figure 6 as an example, after time Δt, node i and node j are still within the range of the dashed circles. Assuming the velocity of node i and node j is v_i and v_j respectively, we can calculate the closeness between node i and node j after time Δt is within the range of $[l-(v_i+v_j)\Delta t, l+(v_i+v_j)\Delta t]$.

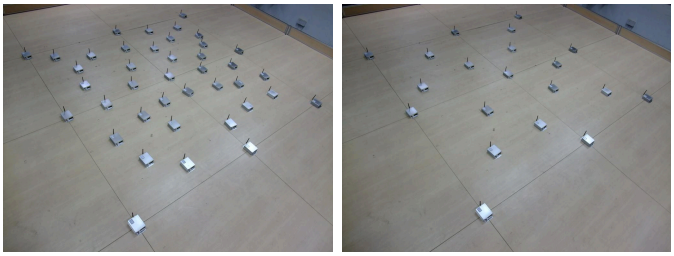
Let the max node velocity in the network be v_{max}, which is an application-specific parameter set by users. As the closeness l between two neighboring nodes i and j can be estimated by Equation 9, the minimal duration for those two nodes moving out each other's communication range is $\frac{R-l}{2v_{max}}$. So as long as we set $TTL_{ij} \leq \frac{R-l}{2v_{max}}$, there is a high probability that a neighboring node j of node i is still within its communication range after they first discovered each other. Similar to the threshold value for spatial selection, a smaller TTL value leads to less energy consumption but also smaller discovery probability. While a larger TTL value costs more energy, but with larger discovery probability for mobile nodes.

VI. Experiment

In order to validate our group-based discovery design in practice, we have fully implemented both basic and advanced group-based discovery designs on the TinyOS/Mote platform in nesC. To compare the performance of our group-based discovery designs, we also implement Disco [3] on our platform.

A. Experimental Setup

During the experiment, we place up to 40 sensor nodes on a 2.6m×2.6m square field. Figure 7 shows a picture of the testbed for the experiment. During the experiment, we do not use any synchronization mechanism to synchronize the clock among nodes in the network. Similar to Disco [3], each node in the network randomly generates its working schedule based on a designated duty cycle, and periodically wakes up according to its working schedule. In our experiment, we set the duration of one time instance to 200ms. The default duty cycle for the network is 3%. And each cases run 3 times.



(a) One-Hop Scenario (b) Two-Hop Scenario

Fig. 7. The Testbed for the Experiment

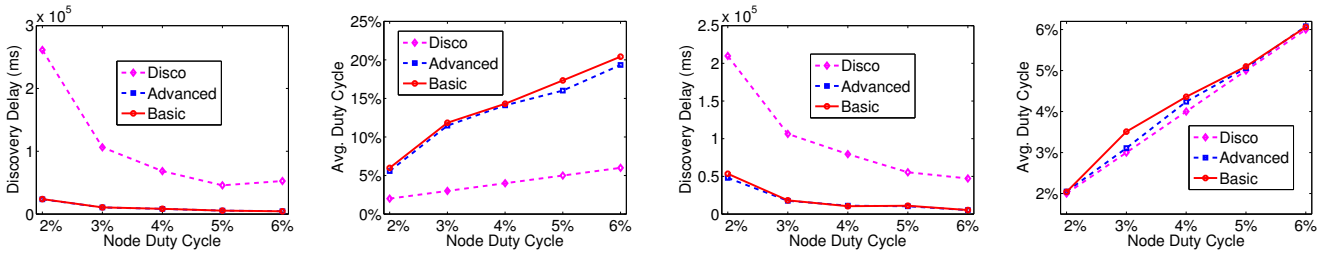
B. Performance Experiments

One-Hop Experiments: By using the maximum transmission power for each node, all deployed nodes are within a one-hop neighborhood. For one-hop experiments, we collect the data after each node having discovered all its neighbors, i.e., the percentage neighbor discovered is 100%.

Two-Hop Experiments: We reduce the communication range of each node to 2m by decreasing its transmission power and only 20 nodes are deployed. Each experiment lasts for ten minutes, which is slightly longer than the maximum designed bounded time for discovery.

1) Impact of Node Duty Cycles: The first experiment tries to investigate the impact of duty cycle on system performances. Figure 8 shows for both single-hop and multi-hop experiments, discovery delay decreases for all three designs when the duty cycle in the network increases. However, under all duty cycles, the discovery delay for Disco is significantly longer than the group-based discovery designs. For example, for single-hop experiments, when the duty cycle is 2%, the discovery delay for Disco is 261,320ms, which is over one order of magnitude longer than that of the advanced design (23,564ms). This result matches our theoretical analysis in Section V quite well. As for energy, both Basic and Advanced designs consume more than that by Disco due to additional reference operations. For example, in one-hop scenario (Figure 8(b)), it is 3% in Disco, 11.8% in Basic, and 11.4% in Advanced, and in two-hop scenario (Figure 8(d)), it is 3% in Disco, 3.5% in Basic, and 3.1% in Advanced. We note in one-hop scenario, we have a higher node density, therefore more nodes need to wake up proactively for reference and verification.

2) Impact of Packet Loss: The loss of packets would increase the delay of reference and verification, consequently increasing the discovery delay. In this experiment, we set the



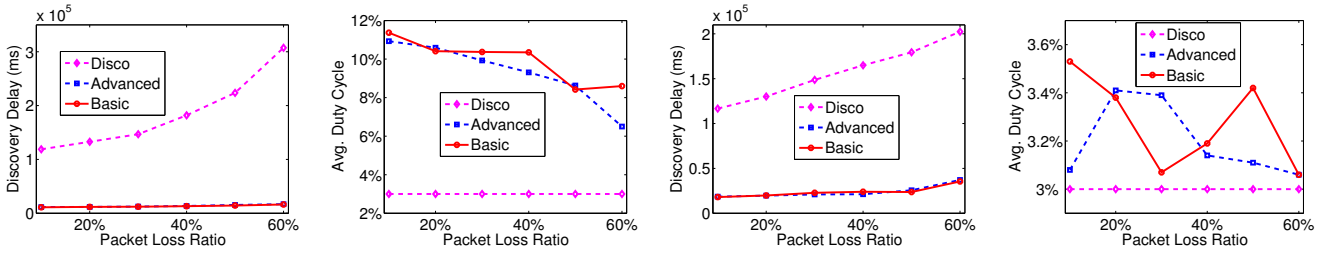
(a) Single-Hop Discovery Delay vs. Duty Cycle

(b) Single-Hop Discovery Energy vs. Duty Cycle

(c) Multi-Hop Discovery Delay vs. Duty Cycle

(d) Multi-Hop Energy vs. Duty Cycle

Fig. 8. Impact of Duty Cycle



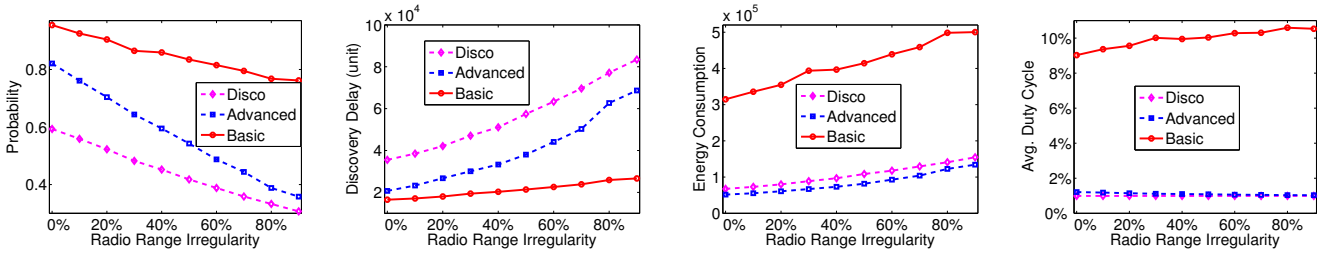
(a) Single-Hop Discovery Delay vs. Packet Loss Ratio

(b) Single-Hop Energy vs. Packet Loss Ratio

(c) Multi-Hop Discovery Delay vs. Packet Loss Ratio

(d) Multi-Hop Energy vs. Packet Loss Ratio

Fig. 9. Impact of Packet Loss Ratio



(a) Discovery Probability vs. Radio Range Irregularity

(b) Discovery Delay vs. Radio Range Irregularity

(c) Total Energy Consumption vs. Radio Range Irregularity

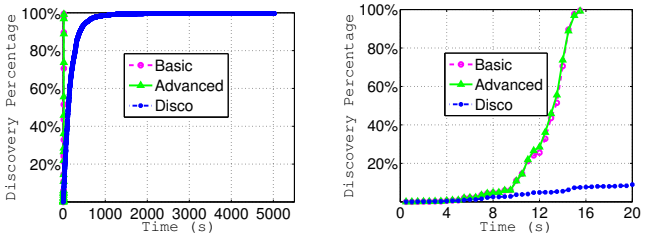
(d) Actual Duty Cycle vs. Radio Range Irregularity

Fig. 10. Impact of Radio Range Irregularity

packet loss ratio be 10%-60% by random dropping packets intentionally in the testbed. Figure 9 shows the impact of packet loss on discovery delay and energy consumption. Discovery delay in Advanced and Basic is far less than that in Disco, especially in the cases with higher packet loss ratio, such as in 60% packet loss ratio case in one-hop scenario, the delay of Disco is 307,420ms, nearly 20 times more than 16,348ms of Basic and 16,888ms of Advanced. And the delay in the Basic and Advanced designs keeps to be stable relatively, but increases significantly in Disco. As far as the energy consumption, packet loss does not affect the pair-wise method, because broadcasting messages are not retransmitted. However, packet loss reduces energy consumption in Advanced and Basic methods, because few reference messages will be propagated further within a neighborhood.

3) Impact of Radio Irregularity: In our theoretical analysis, we assume the unit disk communication model. However, previous works have shown that the radio communication range is highly irregular [21]. Therefore it is critical to understand the impact of radio Irregularity. Unfortunately, in a physical test-bed, the degree of radio irregularity is difficult to control and quantify. In this part of evaluation, we *simulate* the system performance under different degrees of radio irregularity. We set communication range to 100 meters and make the radio irregularity change randomly from 10% to 100% [21]. As degree of radio irregularity increases, we can see the discovery probability for all three schemes decreases. For the discovery delay, both pairwise and advanced discovery methods increase as the degree of radio irregularity increases. However even at about 80% degree of radio irregularity, compared with the pairwise design, our advanced group design has about 5.6% higher discovery probability, 18.7% lower discovery delay and with 13.0% less energy consumption.

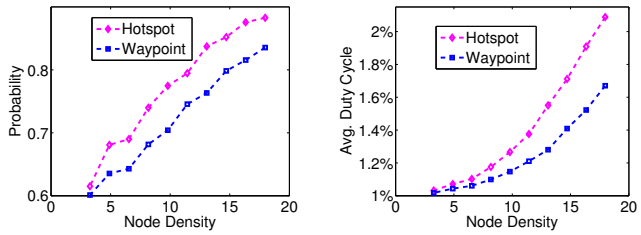
4) Discovery Percentage over Time: In this experiments, Figure 11 plots the node discovery percentage over time in one-hop scenario. Figure 11(a) shows that over 98% node pairs have discovered each other before 1000s in the Disco method, but it takes more than 4000s for the remaining 2%



(a) Long-Tail Delay (b) Discovery Percentage

Fig. 11. The Percentage of Discovery over Time

nodes pairs discovering each other. In contrast, Figure 11(a) illustrates that there are nearly no long-tail for the discovery time in both Basic and Advanced methods, a clear indication that group methods are much better than the Disco method in the worst discovery delay, thanks to our reference mechanism. Figure 11(b) takes a close look at the first 16 seconds. It shows that all node pairs in group designs have discovered each other before time 16s, i.e. 100% discovery percentage, which is far more than that in Disco (below 10%).



(a) Probability vs. Density (b) Energy vs. Density

Fig. 12. Impact of Mobility Patterns

5) Impact of Mobility Patterns: In analysis, we assume the mobile nodes in the network are locally uniformly distributed. In order to reveal the impact of different node distributions, Figure 12 shows the performance using advanced group discovery design under both random waypoint mobility model(uniform node distribution) and hotspot mobility model(nonuniform node distribution). Figure 12(a) shows that by increasing node density, the discovery probabilities for both mobility models increase. However, the discovery delay is shorter in the hotspot mobility model. This is because under the hotspot mobility model, nodes are clustered in a few locations in a network, allowing more group discovery among neighboring nodes. And due to increased reference operations under the hotspot mobility model, we also observe higher energy consumption under the hotspot model over the random waypoint model.

VII. Conclusion

This paper presents a Group-based Discovery method as a performance add-on to existing pair-wise discovery designs. It essentially builds a schedule reference mechanism among nodes to expedite the discovery process of pair-wise discovery designs. Our work is the first to provide theoretical analysis of group-based discovery delay. We introduce the basic operation, followed by an advanced group-based discovery design, which selectively choose neighboring nodes for energy-efficient reference. We evaluate our designs in a physical test-bed. Compared with the state-of-the-art pairwise solutions, our designs show one order of magnitude reduction in discovery delay with only maximum 8.8% increase in energy.

Acknowledgment

This research was supported in part by the US National Science Foundation (NSF) grants CNS-0845994, CNS-0917097, IBM OCR Fund, grant SUTD SRG ISTD 2010 002 and SUTD-ZJU/RES/03/2011.

References

[1] *2.4 GHz IEEE 802.15.4 / ZigBee-Ready RF Transceiver (Rev. B)*. Available at http://focus.ti.com/docs/prod/folders/print/cc2420.html.

[2] S. A. Borbash, A. Ephremides, and M. J. McGlynn. An asynchronous neighbor discovery algorithm for wireless sensor networks. *Ad Hoc Networks*, 2007.

[3] P. Dutta and D. Culler. Practical asynchronous neighbor discovery and rendezvous for mobile sensing applications. In *SenSys'08*, 2008.

[4] P. Dutta and D. Culler. Mobility changes everything in low-power wireless sensornets. In *HotOS-XII'09*, 2009.

[5] S. B. Eisenman, E. Miluzzo, N. D. Lane, R. A. Peterson, G.-S. Ahn, and A. T. Campbell. The bikenet mobile sensing system for cyclist experience mapping. In *SenSys'07*, 2007.

[6] G. Jakllari, W. Luo, and S. V. Krishnamurthy. An integrated neighbor discovery and mac protocol for ad hoc networks using directional antennas. *Trans. Wireless Commun.*, 2007.

[7] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. Peh, and D. Rubenstein. Energy-Efficient Computing for Wildlife Tracking: Design Tradeoffs and Early Experiences with ZebraNet. In *Proc. of ASPLOS-X*, October 2002.

[8] A. Kandhalu, K. Lakshmanan, and R. R. Rajkumar. U-connect: a low-latency energy-efficient asynchronous neighbor discovery protocol. In *IPSN'10*, 2010.

[9] N. Karowski, A. Viana, and A. Wolisz. Optimized asynchronous multi-channel neighbor discovery. In *Infocom'11*, 2011.

[10] R. Khalili, D. Goeckel, D. Towsley, and A. Swami. Neighbor discovery with reception status feedback to transmitters. In *Infocom'10*, 2010.

[11] S. Lai, B. Ravindran, and H. Cho. Heterogenous quorum-based wake-up scheduling in wireless sensor networks. *Computers, IEEE Transactions*, 2010.

[12] S. Lai, B. Zhang, B. Ravindran, and H. Cho. Cqs-pair: Cyclic quorum system pair for wakeup scheduling in wireless sensor networks. *Principles of Distributed Systems, LNCS*, 2008.

[13] M. J. McGlynn and S. A. Borbash. Birthday protocols for low energy deployment and flexible neighbor discovery in ad hoc wireless networks. In *MobiHoc'01*, 2001.

[14] I. Niven, H. S. Zuckerman, and H. L. Montgomery. An introduction to the theory of number. In *John Wiley and Sons*, 1991.

[15] R. Szewczyk, A. Mainwaring, J. Polastre, J. Anderson, and D. Culler. An analysis of a large scale habit monitoring application. In *SenSys'04*, 2004.

[16] Y.-C. Tseng, C.-S. Hsu, and T.-Y. Hsieh. Power-saving protocols for ieee 802.11-based multi-hop ad hoc networks. In *INFOCOM'02*, 2002.

[17] S. Vasudevan, J. Kurose, and D. Towsley. On neighbor discovery in wireless networks with directional antennas. In *INFOCOM'05*, 2005.

[18] S. Vasudevan, D. Towsley, and D. Goeckel. Neighbor discovery in wireless networks and the coupon collectors problem. In *Mobicom'09*, 2009.

[19] X. Wu, K. Brown, and C. Sreenan. Snip: A sensor node-initiated probing mechanism for opportunistic data collection in sparse wireless sensor networks. In *INFOCOM'11*, 2011.

[20] R. Zheng, J. C. Hou, and L. Sha. Asynchronous wakeup for ad hoc networks. In *MobiHoc'03*, 2003.

[21] G. Zhou, T. He, and J. A. Stankovic. Impact of Radio Irregularity on Wireless Sensor Networks. In *MobiSys'04*, 2004.