JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 37, 859-883 (2021)
DOI: 10.6688/JISE.202107_37(4).0008

A Cooperative Rotational Sweep Scheme to Bypass Network Holes in Wireless Geographic Routing*

JUNG-TSUNG TSAI[1] AND YUNGHSIANG S. HAN[2]
[1]*Department of Computer Science and Information Engineering*
National Taiwan Normal University
Taipei, 106 Taiwan
[2]*School of Electrical Engineering and Intelligentization*
Dongguan University of Technology
Dongguan, 523808 P.R. China
E-mail: jutsai@csie.ntnu.edu.tw; yunghsiangh@gmail.com

Geographic routing in wireless ad hoc networks is characterized by routing decisions made from locally available position information, which entails network scalability. However, it requires an effective recovery approach to sending a packet bypassing network holes whenever the simple greedy forwarding fails. Among well-known approaches, rotational sweep routing algorithms based on a circular arc are able to achieve packet delivery guarantee as well as low routing path stretch under the impractical assumption that a wireless link exists between two nodes if and only if their distance is less than one unit. Instead, we propose a cooperative rotational sweep algorithm taking into account practically imperfect wireless connections. The algorithm involves a regular rotational sweep procedure and a cooperative one both making use of iterative sweeps with circular arcs of decreasing size subjective to a minimum size constraint. Essentially, the cooperative rotational sweep procedure resolves hidden node issues through exploiting packet header overheads for memory of the latest routing path while iterative sweeps reduce the possibility of missing pivotal relays. Simulation results demonstrate that the proposed scheme presents the benefit of using packet header overheads to support high end-to-end routing success probabilities without sacrificing the inherent feature of localized routing.

Keywords: geographic routing, network hole, rotational sweep algorithm, circular arc, edge intersection

1. INTRODUCTION

Geographic routing supports the scalability of wireless ad hoc or sensor networks through routing decisions made from local state information, on neighborhood node positions and the destination of a packet. To obtain such information, it suffices that each node knows its own position via GPS devices or location services and exchanges it with neighbors through, for example, IEEE 802.11 beacon messages and that the source of a

Received July 18, 2019; revised September 29, 2019; accepted November 15, 2019.
Communicated by Jiann-Liang Chen.
* The preliminary form of this paper was presented in IEEE International Conference on Communications, Kuala Lumpur, Malaysia, May 2016.

packet is aware of its destination [1, 2, 3]. The overheads of routing table maintenance or performing route discovery procedures in conventional address-based routing are thus unnecessary.

The most popular and simple decision approach of geographic routing is greedy forwarding [4, 5] by which a node selects as the next relay a neighbor closer to the destination of a packet than the node itself. Greedy forwarding is loop-free. However, it will fail and stop at a node when such a neighbor does not exist, causing the packet *stuck* at the node. This is known as a *local minimum* issue that generally arises at a node on the boundary of a network void or hole [6]. The issue is inevitable in wireless networks, considering that some voids may be created due to topography obstructors, node mobility, or defunct devices from power loss. To succeed in delivering a packet, it is thus imperative to have an auxiliary routing approach to bypassing a routing hole so as to recover from greedy forwarding failure.

Several recovery methods have been developed, strongly relying on the unit disk assumption (UDA) that a wireless link exists between two nodes if and only if their distance is less than one unit. They can be divided into two categories, planar graph based face routing [2, 3, 7, 8, 9, 10] and rotational sweep algorithms tracking nodes on a void boundary [11, 12, 13]. Each of them can be combined with greedy forwarding to achieve packet delivery guarantee. In particular, the rotational sweep algorithm based on curved stick [13] or twisting triangle (TT) [12] uses a sweep curve (SC), which is a circular arc of the transmission radius or one side of a Reuleaux triangle [12] (see Fig. 1), to sweep the coverage area of a wireless node and locate the first hit node as the next relay. By this recovery method, low routing path stretch, in terms of hop counts, and delivery guarantee can be both achieved. Similarly, routing decisions in [14] further take into account information on the representative angle of a neighbor node, the largest angle between consecutive edges incident at the node, for a partial vision of next two hop topology, which significantly cuts down routing path stretch.

However, UDA is fundamentally inappropriate for realistic wireless channels because of variable transmission ranges due to signal path loss, interference and fading. Those recovery approaches based on UDA are thus impractical. In this work, we consider that the existence of a bidirectional link between two nodes is completely determined by the received signal power level above some threshold at both ends. For such a non-UDA network setting, we propose a recovery method adapted from the rotational sweep algorithm [13] (referred to as TT [12] here).

Under non-UDA, a number of issues arise from performing TT to locate the next relay besides the well-known edge intersection [11, 13, 15] or crossing link problem [3, 12, 16]. The first is what size of an SC should be used since there is no magic distance of one unit fit for all scenarios. The second is the hidden node issue that some nearby one should have been hit and selected as the next relay but has no direct link to the node performing TT. The third is the problem of a neglected neighbor which was not selected due to the size of SC either too small or too large. The fourth is how to detect a routing loop since the general way proposed under UDA no longer applies [17].

This work is focused on a cooperative rotational sweep scheme (α-CO$_m$TT) to effectively bypass a routing hole and recover from greedy forwarding failure under non-UDA. The scheme particularly preserves the feature of localized routing and thus network scalability. Specifically, the main contributions can be stated in four parts:

1. Propose an adaptive rotational sweep algorithm (α-TT), one key component of α-CO$_m$TT, for a progressive search. It resolves the SC size and neglected node issues by utilizing iterative sweeps first with SCs of diminishing sizes determined from the link distance to each connected neighbor and then with SCs shrinking continuously with sweeping, called extended sweep (eSW). The sweep stops when the constraint of a minimum SC size η_L is reached.

2. Develop an adaptive cooperative rotational sweep algorithm (α-CO$_m$), the other key component of α-CO$_m$TT, for a retracted search. It resolves the hidden node issue or some edge intersection problem through semi-replicating previous α-TT procedures. This is enabled by using an extra packet header field to carry a list of some visited node positions, of at most size m.

3. Find out conditions for determining a routing loop based on the carried information in the overhead of a packet and prove that if the overhead of a packet header is unconstrained, a routing loop can be detected surely and as early as possible.

4. Embed in α-CO$_m$TT a distance threshold to packet destination (d_t) within which operation mode tends to aggressively change to greedy one. The threshold d_t is another factor affecting routing performance, besides m and η_L.

1.1 Related Work

To resolve the local minimum issue, a number of hole boundary traversal schemes have been described notionally like rotating a curve of some geometry. In the BoundHole algorithm [15] a straight line of one unit, the transmission radius under UDA, is utilized to sweep a hole area according to the so-called righthand rule. The first node hit by the line is chosen as the next relay. Used alone, the method is unable to resolve crossing-link issues. Instead, greedy anti-void routing (GAR) [11] was proposed to solve the boundary finding problem and determine the next relay by implementing a Rolling ball with a radius of *half* the radio transmission range. Unfortunately, it guarantees delivery at the expense of increased mean hop counts.

One of the rotational sweep algorithms in [12] rotates a twisting triangle, which is a Reuleaux triangle formed by the intersection of three circles placed on the corners of an equilateral triangle with each side of one transmission radius. The rotation angle required for the circular arc of a twisting triangle to hit a node is used to determine MAC contention latency. It is a beaconless or contention-based routing [16, 18, 19] involving both MAC contention deference and routing decisions, which requires the support of RTS-CTS messaging. Instead, localized routing [13] utilizes as a sweep curve the Curved Stick (CS) which is a circular arc with chord length and radius of one transmission range. Since a CS is exactly one side of Reuleaux triangle, this scheme is similar to the TT algorithm except the start-sweep point. Both use an SC with a smaller curvature than the rolling ball and a chord length of the transmission radius to achieve delivery guarantee and lower mean path hop counts under UDA.

Cooperative relaying [20] provided a MAC-network cross-layer protocol for forwarder selection as well as a MAC-PHY cross-layer one for relay selection, which improved per-hop routing reliability under realistic wireless communication channels. Our preliminary study [17] considered an incomplete non-UDA in that no wireless link exists

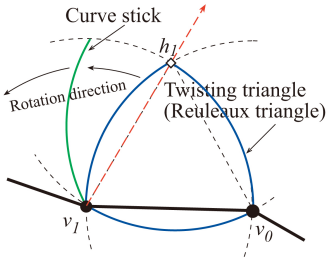


Fig. 1. A curve stick [13] and a twisting triangle [12] hinged at v_1 and a start sweep point h_1.

between two nodes absolutely if their distance is larger than one unit, that a link exists between them if their distance is less than a given threshold $r_c \in [0,1]$, a constant modeling *connection regularity*, and that a link may exist between them depending on received signal power level if their distance is in the range $(r_c, 1]$. It was found that applied directly, those schemes [13, 14] developed under UDA are hardly to achieve a sensible routing success probability. A cooperative rotational sweep method, based on TT, was then proposed to jointly operate with original TT as a recovery scheme. The cooperative approach relies on duplicated sweeps but performed by the node holding a packet on behalf of previously visited ones whose positions are carried with the packet. With the approach, a high routing success probability can be achieved at the cost of packet header overheads.

1.2 Outline

The rest of the paper is organized as follows. In Section 2 we describe system models and present basic rotational sweep algorithm α-TT involving eSW. In Section 3 we present cooperative algorithm α-CO$_m$ and then combine both into recovery scheme α-CO$_m$TT. In Section 4, we investigate routing-loop detection issues, introduce distance threshold d_t into α-CO$_m$TT for control of operation mode change and show the proposed scheme by an example. Section 5 presents simulation results with discussions. We conclude in Section 6 with a brief remark.

2. SYSTEM MODEL AND ROTATIONAL SWEEP ALGORITHMS

Let $G(V,E)$ represent an ad hoc network graph where V is the set of all nodes and E is the set of all bidirectional links. For nodes $v_i, v_j \in V$, edge $v_iv_j \in E$ if and only if v_i and v_j can communicate with each other directly. The set of neighbors of node v_i is represented by $N_i = \{v_j \in V; v_iv_j \in E, j \neq i\}$. Let $\overline{N}_i(\eta) = \{v_j \in V; |v_iv_j| \leq \eta, j \neq i\} - N_i$ denote a set of nodes which are within the distance η but have no bidirectional link to node v_i, *i.e.* the set of nodes in a nominal circular coverage area of radius η, denoted by $C_i(\eta)$, that are hidden to node v_i.

Let $A_f(v_i) = \{v_j \in N_i; |v_jD| < |v_iD|\}$, a set consisting of the neighbors of node v_i which are closer to D than node v_i is.

Consider that node v_i creates or receives a packet destined to D in greedy mode. If $A_f(v_i) \neq \emptyset$, the packet will advance to a selected node in $A_f(v_i)$. Otherwise the event of local minimum occurs at stuck node v_i, denoted by v_A, due to $A_f(v_i) = \emptyset$. To lead

the packet away from stuck node v_A, a recovery process is then invoked. Essentially, the routing process alternates *greedy* and *recovery* mode operations in a full course of delivering a packet. It keeps operating in greedy mode whenever possible, switches to recovery mode when the greedy one fails, and returns to greedy mode when some given condition is satisfied.

Before developing our recovery methods, we first recall key procedures and features of rotational sweep algorithm CS[13], *i.e.* TT [12], used in recovery mode to select the next relay under UDA. Notionally an SC is a circular arc of radius $\eta = 1$ with chord length $\eta = 1$ as shown in Fig. 1 where it is hinged at node v_1 and swept counterclockwise with the curvature center trailing behind according to the righthand rule to locate and select the first hit node in the coverage of v_1. At the beginning of sweep, the SC is positioned with the other end at start-sweep point h_1, the righthand-side intercept of unit circles C_0 centered at v_0 and C_1 centered at v_1 if v_0 is the previous SC hinged node choosing v_1 or the intercept of line v_1D and circle C_1 if the recovery mode is initialized at stuck node v_1. Forwarded by this approach under UDA, the packet surely moves along a network void boundary [13] over which a node with the least distance to D must exist. Method TT keeps looking for switching to greedy mode operation at the earliest moment when a void boundary node is found closer to D than the stuck node is. When the packet returns to stuck node, say v_1, and the SC is hinged at v_1 and swept to cross $h_1 = v_1D \cap C_1$ without hitting a node, this implies that stuck node v_1 is the one on the network void boundary closest to D, that the network fragment containing v_1 and that containing D are disconnected, and that a routing loop occurs.

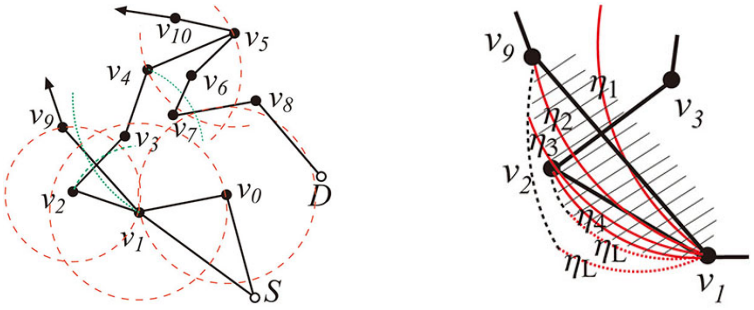
In realistic wireless coverage, the benefit and importance of a single value $\eta = 1$ of SC chord length no more exists. To cope with the imperfection, we first consider iterative rotational sweeps based on SCs, circular arcs, of decreasing size in place of the rotational sweep algorithm [12, 13, 14, 17] in recovery mode operation. However, we assume that there is a constraint on the minimum size of SC, with chord length η_L and hence curvature $1/\eta_L$. It is considered protocol-wise available at each node.

Additionally we assume that a packet in recovery mode includes a field capable of carrying a list T_r of position information of visited nodes for at most m entries. With T_r available, we explore the cooperative rotational scheme [17] for a retracted search to modify a routing trajectory and thus resolve hidden node issues in realistic wireless coverage.

We simply consider the scheme of Greedy-Forwarding (GF) [4] for greedy mode operation. By GF, node v_i selects the next relay in $A_f(v_i)$ with the least distance to D. Note that some hidden node in $\overline{N_i}$ may not exist in $A_f(v_i)$. We design recovery scheme α-CO$_m$TT, involving algorithms α-TT and α-CO$_m$. Like TT, it attempts to forward a packet along a network void boundary. We focus on the method of α-TT here but leave α-CO$_m$TT involving α-CO$_m$ in Section 3.

2.1 Adaptive Rotational Sweep Algorithm Based on TT (α-TT)

Consider that node v_i is to select a connected neighbor in N_i as the next relay for a packet with routing information $\{D, v_A, v_{i-1}\}$, where v_A is the stuck node and v_{i-1} is hypothetically the upstream node of v_i on a network hole boundary. If nodes v_i and v_{i-1} are not connected directly, we view v_iv_{i-1} as a *virtual* link, to be revealed in Section 3.



(a) (b)

Fig. 2. (a) Recovery mode starts at v_0 for a packet from S to D. Node v_1 performs eSW to locate node v_2, resolving the issue of crossing edges v_1v_9 and v_2v_3. Node v_5 performs α-CO to locate next relay v_6 that should have been selected by but was hidden from node v_4 previously; (b) Enhanced sweep (eSW) searches to locate v_2 are highlighted. The eSW has been invoked twice here.

2.1.1 Initial SC size

To perform iterative rotational sweeps, what is the initial size η of SC hinged at v_i? Naturally, it is set at $\max\{\max_{v_\ell \in N_i} |v_iv_\ell|, \eta_L\}$ so that at least one node is surely hit in case that v_{i-1} is the only neighbor of v_i and v_iv_{i-1} is very long.

2.1.2 Start-sweep line

The second issue is where to start sweeping. Instead of the start-sweep point adopted in [13] under UDA, we use a start-sweep line $v_ih(v_{i-1}, v_i)$ on which the chord of any size of SC hinged at v_i is aligned at the beginning of sweep and on the righthand of which the SC curvature center is located. If v_i is exactly the stuck node v_A that initiates current recovery mode operation and starts on the trajectory of tracing a network hole boundary, then h is the intercept of circle $C_i(\eta_L)$ and $\overrightarrow{v_iD}$ or simply $h = D$. With a little abuse of notation, define point $h(D, v_i) = D$. Otherwise, node v_i is the downstream node of v_{i-1} on a traced network hole boundary. In this case, $h(v_{i-1}, v_i)$ is the righthand intercept of circles $C_{i-1}(\max\{\eta_L, |v_iv_{i-1}|\})$ and $C_i(\max\{\eta_L, |v_iv_{i-1}|\})$ from v_{i-1} to v_i.

Specifically, the start-sweep line is $v_ih(v_{i-1}, v_i)$, where $h(v_{i-1}, v_i)$ is the coordinate of D if v_i is v_a and v_{i-1} is found to be D and otherwise the coordinate of $C_{i-1}(\eta) \cap C_i(\eta)$ on the righthand of v_{i-1} to v_i where $\eta = \max\{\eta_L, |v_iv_{i-1}|\}$.

2.1.3 Iterative sweeps and where to stop sweeping

With the initial SC size η and start-sweep line $v_ih(v_{i-1}, v_i)$ resolved, iterative sweeps of SC always hinged at v_i are applied. Upon performing each SC sweep counterclockwise, the first hit node, say $v_{i+1} \in N_i$, is chosen as the latest candidate of relay and the distance $\max\{|v_iv_{i+1}|, \eta_L\}$ is used as the new SC size for iteration. In case multiple nodes are hit simultaneously, the one farthest to v_i is the latest candidate. If such a diminished SC reaches to minimum chord size η_L, the iteration stops after sweeping with this SC of minimum size once and the last hit one is selected as the next relay. On the other hand, if the diminished SC with chord size larger than η_L is swept to first hit the same node determining the current SC size, an extended sweep, eSW, is invoked to further resolve

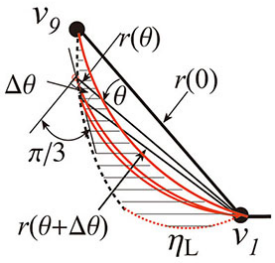


Fig. 3. Enhanced sweep (eSW) search with sweep curves, hinged at v_1, decreasing in size continuously from $|v_1v_9|$ to η_L.

the next relay selection issue, introduced next.

To describe how eSW works, we consider example network graphs in Figs. 2 (a) and (b) where $|v_1S| > |v_1v_9| > |v_1v_0| > |v_1v_2| > \eta_L$. Suppose that node v_1 is to select the next relay from $N_1 = \{v_0, v_2, v_9, S\}$ for a packet received from v_0 and destined to D in recovery mode. Initially the SC of size $\eta_1 = |v_1S|$ hinged at v_1 is swept from $v_1h(v_0, v_1)$ to hit and select the latest candidate node v_9. Since $|v_1v_9| = \max\{|v_1v_9|, \eta_L\}$, the SC is reduced to size $\eta_2 = |v_1v_9|$ and swept again to hit and select node v_9 the second time. Note that there is a cross-edge issue caused by links v_2v_3 and v_1v_9 but unresolved by the SC of curvature $1/\eta_2$. To solve this problem, our approach is to sweep with an SC of continuously decreasing size, less than $|v_1v_9|$, so that node v_2 will be hit and selected, shown by the SC of size η_3 in Fig. 2 (b). This constitutes the first use of eSW. With this latest hit node v_2 and $|v_1v_2| > \eta_L$, the SC of size $\eta_4 = |v_1v_2|$ is swept and first hits v_2 again. The iteration hasn't stopped yet since the SC size $\eta_4 > \eta_L$. The procedure of eSW is applied again until the SC size is reduced from $|v_1v_2|$ down to η_L. The last hit node v_2 is then selected as the next relay. Note that the condition for sweep stop delimits a search area not too deep and away from network hole boundary, determined by the minimum value η_L, in case that node v_2 does not exist.

By eSW, the SC continues shrinking in size with one end fixed at v_1 and the other end moved inwards along the SC curve, as shown in Fig. 3. Mathematically, we can view v_1 and line v_1v_9 as the pole O and polar axis of a polar coordinate system (r, θ). Then, the moving end of SC is initially located at $(|v_1v_9|, 0)$. Suppose that the moving end of SC is now at $(r(\theta), \theta)$ with $r(\theta) > \eta_L$. The angle between the tangent line of the SC at point $(r(\theta), \theta)$ and the line from the point to the pole is $\pi/6$. After the SC is rotated counterclockwise by $\Delta\theta$ and reduced in size simultaneously, we have the approximation $r(\theta + \Delta\theta) \approx r(\theta) - r(\theta)\Delta\theta \tan(\frac{\pi}{2} - \frac{\pi}{6})$ and hence $\frac{r(\theta+\Delta\theta)-r(\theta)}{\Delta\theta} \approx -\sqrt{3}r(\theta)$ after rearrangement. As $\Delta\theta \to 0$, we have $\frac{dr}{d\theta} = -\sqrt{3}r(\theta)$ and solution $r(\theta) = Ce^{-\sqrt{3}\theta}$ for some constant C. Furthermore, $C = r(0) = |v_1v_9|$ and $r(\theta) = \eta_L$ for $\theta = \frac{\ln|v_1v_9| - \ln\eta_L}{\sqrt{3}}$. Thus we have proved the following.

Proposition 1 *For an SC of size $\eta_0 > \eta_L$ with one end hinged at the pole O of a polar coordinate system, the other end initially placed at $(r(0) = \eta_0, \theta = 0$ radian$)$, and its curvature center on the righthand of its chord, the SC is rotated counterclockwise and decreases in size with θ simultaneously by moving the other end inwards along the SC curve until $r(\theta) = \eta_L$ for some θ. Then, the area swept by such an SC of continuously de-*

creasing size is enclosed by the initial SC of size $r(0) = \eta_0$, the boundary curve described by

$$r(\theta) = \eta_o e^{-\sqrt{3}\theta} \tag{1}$$

for $\theta \in [0, \frac{\ln \eta_0 - \ln \eta_L}{\sqrt{3}}]$ and the SC of chord $r(\theta) = \eta_L$ placed at $\theta = \frac{\ln \eta_0 - \ln \eta_L}{\sqrt{3}}$.

2.2 Procedures α-TT and eSW

With all required function elements presented previously, both eSW and α-TT are summarized in the following procedures.

Procedure $eSW(N_i, v_i, v_{i-1}, v_k)$ that returns v_{i+1}.

Let the initial SC have chord length $\eta = r(0) = |v_i v_k|$. Sweep the SC of size $r(\theta) = r(0)e^{-\sqrt{3}\theta}$ decreasing with θ from start-sweep line $v_i v_k$ until

(a) a node $v_{i+1} \in N_i$ is first hit and selected or

(b) $r(\theta)$ equals η_L when the extended search stops. In this case, let $v_{i+1} \leftarrow v_k$.

Procedure $\alpha - TT(N_i, v_i, v_{i-1})$ that returns next relay v_n.

1. Node v_i with neighbor set N_i sets initial SC size $\eta \leftarrow \max\{\max_{v_\ell \in N_i}\{|v_i v_\ell|\}, \eta_L\}$ and remember the node $v_k \leftarrow \arg\max_{v_\ell \in N_i}\{|v_i v_\ell|\}$.
2. Determines $h(v_{i-1}, v_i)$ and start-sweep line $v_i h(v_{i-1}, v_i)$.
3. Sweep the SC of size η from start-sweep line $v_i h(v_{i-1}, v_i)$ to get the first hit node $v_{i+1} \in N_i$ and proceed to do one of next three actions accordingly.
   - η is η_L:
     Node $v_n \leftarrow v_{i+1}$ is selected as the next relay.
   - $\eta > \eta_L$ and $v_{i+1} \neq v_k$, *i.e.* the latest hit node v_{i+1} is not the previous hit one v_k:
     Let $\eta = \max\{|v_i v_{i+1}|, \eta_L\}$, remember $v_k \leftarrow v_{i+1}$, and go to step 3.
   - $\eta > \eta_L$ and v_{i+1} is exactly node v_k:
     Perform Procedure $eSW(N_i, v_i, v_{i-1}, v_k)$ to obtain the next candidate $v_{i+1} \in N_i$ and go to one of next two actions accordingly.
     (a) $v_{i+1} \neq v_k$, the latest hit node v_{i+1} by eSW is not the previous one v_k:
     Let $\eta = \max\{|v_i v_{i+1}|, \eta_L\}$, remember $v_k \leftarrow v_{i+1}$, and go to step 3.
     (b) v_{i+1} is exactly node v_k (for no further candidate node is located) :
     Node $v_n \leftarrow v_{i+1}$ is selected as the next relay.

3. COOPERATIVE ROTATIONAL SWEEP, α-CO$_m$TT

The spirit of cooperative rotational sweep was introduced in [17] to cope with hidden node issues. This can be seen from the network graph in Fig. 2 (a). Consider that a packet

from S to D has arrived at node v_4 in recovery mode. Although v_6, v_7 and v_{10} are close to v_4, they have no direct connection to node v_4 and are thus hidden to it. If scheme α-TT is used alone in recovery mode, the packet is first routed to node v_5 since node v_4 with only neighbors $N_4 = \{v_3, v_5\}$ available for selection is unable to locate v_6. This packet is then routed to node v_{10} by node v_5, going in the wrong direction, because the SC swept from $v_5h(v_4, v_5)$ will miss node v_6 but hit and select node v_{10}.

The basic approach to resolving the above issue is as follows. Since node v_4 is unsure that the selected relay v_5 under available information N_4 is exactly a network hole boundary node, in case that hidden nodes exist, node v_4 transfers the sweep information $v_4h(v_3, v_4)$ encapsulated in the packet to node v_5 and asks it to perform the same sweep but over the set $N_5 \cup \{v_5\} = \{v_4, v_6, v_5, v_{10}\}$ known to v_5 for a double check. Upon receiving the packet from v_4, node v_5 duplicates the previous sweep initialized with SC of size $\max\{\eta_L, |v_4v_5|\}$ from start-sweep line $v_4h(v_3, v_4)$, on behalf of node v_4, to locate a node in the set $N_5 \cup \{v_5\}$. Through this extra operation of cooperative sweep, node v_5 finds that node v_4 should have selected node v_6 instead of v_5 itself as the next relay. For correct routing, node v_5 sends to node v_6 the packet without including any information of node v_5 so that node v_6 seems to receive the packet directly from node v_4, bypassing node v_5. Thereby, a virtual link between node v_4 and v_6 appears to exist now.

The above suggests that for correct routing, a node should perform cooperative sweeps first when receiving a packet in recovery mode.

3.1 α-CO, One Cooperative Occasion

Let $\alpha\text{-}CO(N_i, v_i, v_{j+1}, v_j, v_{j-1})$ denote the cooperative procedure performed by node v_i attempting to replicate the α-TT that was performed previously at node v_j from start-sweep line $v_jh(v_{j-1}, v_j)$ to select its downstream node v_{j+1}. In case that node v_i has just been selected by the previous SC sweep from $v_jh(v_{j-1}, v_j)$, the downstream node is v_i itself and we let $v_{j+1} = \emptyset$. Technically, the set of candidate nodes for selection in cooperative search is now augmented to $N_i \cup \{v_i\} \cup \{v_{j+1}\}$. With limited available information, the initial SC size is set to $\eta = \max\{|v_jv_{j+1}|, \eta_L\}$ or $\eta = \max\{|v_jv_i|, \eta_L\}$ if v_{j+1} is $\emptyset$. Except these initial settings different from that of α-TT, the procedure then follows that of iterative sweeps with deceasing SC sizes by α-TT. The α-CO is used to check whether v_{j+1} (or v_i if v_{j+1} is $\emptyset$) is selected again but from the augmented set $N_i \cup \{v_i\} \cup \{v_{j+1}\}$. Specifically, we have the following.

Procedure α***-CO***$(N_i, v_i, v_{j+1}, v_j, v_{j-1})$ that returns next relay v_n.

1. Node v_i with neighbor set N_i:
   Augment the set $N_i \leftarrow N_i \cup \{v_i\} \cup \{v_{j+1}\}$ for backtracking;
   Set initial SC size $\eta = \max\{|v_jv_{j+1}|, \eta_L\}$ and remember the node $v_k \leftarrow v_{j+1}$ if $v_{j+1} \neq \emptyset$; Otherwise, $\eta = \max\{|v_jv_i|, \eta_L\}$ and $v_k \leftarrow v_i$.

2. Let $v_i \leftarrow v_j$, $v_{i-1} \leftarrow v_{j-1}$, and start-sweep line $v_ih(v_{i-1}, v_i) \leftarrow v_jh(v_{j-1}, v_j)$.

3. Follow all step 3 of Procedure $\alpha - TT(N_i, v_i, v_{i-1})$ to obtain next relay v_n.

In the previous example (see Fig. 2 (a)), we have only described an instance where node v_5 performs α-CO with $v_{j-1} = v_3, v_j = v_4, v_{j+1} = \emptyset$ and $v_i = v_5$ to locate next relay v_6

from the set $\{v_4, v_6, v_{10}, v_5\}$ after receiving a packet from v_4 in recovery mode. Obviously, the earlier instance is that node v_5 performs α-CO with $v_{j-1} = v_2, v_j = v_3, v_{j+1} = v_4$ and $v_i = v_5$ to locate next relay v_4 from $\{v_4, v_6, v_{10}, v_5\}$.

3.2 α-CO$_m$ for Unconstrained List Size m

Let the size m of position list $T_r(i),\ i = 0, 1, 2, \cdots, m-1$ in a packet be theoretically unconstrained, referred to by $m = \infty$ thereafter. From the fact that the CS/TT rotational sweep algorithm [13] keeps selecting the next relay on a network hole boundary under UDA, we attempt to achieve the same purpose through utilizing the recorded history information in T_r to perform a sequence of α-CO accordingly, check and update the list T_r if necessary. The method is described next.

When a packet is stuck at node v_A by GF, no enough information supports an instance of performing α-CO. Node v_A simply performs α-TT to select the next relay and initializes the list T_r to have the only entry $T_r(0) = v_A$ as the head node on the trajectory of routing along a network hole boundary.

When node v_i receives a packet with list T_r in recovery mode, it first checks whether the node sequence in T_r is orderly situated along a network hole boundary and v_i itself is truly the downstream node of the last one in T_r, by performing α-CO sequentially on behalf of those nodes $T_r(j), j = 0, 1, \cdots, |T_r| - 1$ over the node sets $N_i \cup \{v_i\}$ and T_r available. In other words, such a sequence of performing α-CO is used to check whether there is a node $v_{i+1} \in N_i \cup \{v_i\}$ should have been selected earlier but hidden to previously visited node $T_r(j)$ for some j and $v_{i+1} \neq T_r(j+1)$. Specifically, node v_i performs a sequence of α-CO until one of the following conditions is met:

(i) The sequence in T_r has all been orderly followed and the final selected node v_{i+1} is exactly v_i. Then, node i updates T_r by pushing v_i into T_r and determines the selected relay v_{i+1} being v_i itself for no node hidden to visited ones in T_r is found.

(ii) The sequence in T_r has all been orderly followed and the final selected node v_{i+1} is not v_i. Then, node v_i determines as the next relay the node v_{i+1} hidden to the last recorded one in T_r without updating T_r.

(iii) The sequence in T_r has been orderly followed until $T_r(j)$ where the selected node v_{i+1} is not $T_r(j+1)$ but exactly v_i. Then, node v_i updates T_r by the replacement $T_r(j+1) \leftarrow v_i$ and removing away all entries after $T_r(j+1)$ and determines the selected relay v_{i+1} being v_i itself.

(iv) The sequence in T_r has been orderly followed until $T_r(j)$ where the selected node $v_{i+1} \neq T_r(j+1)$ and $v_{i+1} \neq v_i$. Then, node v_i updates T_r by removing away all entries after $T_r(j)$ and determines next relay v_{i+1}.

Thus, α-CO$_\infty$ performed at node i for a retracted search is described as follows:
Procedure α-CO$_\infty$ (N_i, v_i, T_r) that updates T_r and returns next relay v_n.

1. Begin with $j = 0$.
2. Apply procedure $\alpha - CO(N_i, v_i, T_r(j+1), T_r(j), T_r(j-1))$ if $j > 0$, or $\alpha - CO(N_i, v_i, T_r(j+1), T_r(j), D)$ if $j = 0$, to obtain the selected node v_{i+1}. Then, take one of next four actions accordingly.
   - $v_{i+1} = v_i$ and $T_r(j+1) = \emptyset$:
     Store v_i in $T_r(j+1)$;

Return node $v_n \leftarrow v_{i+1}$.

- $v_{i+1} \neq v_i$ and $T_r(j+1) = \emptyset$:
  Returns the next relay $v_n \leftarrow v_{i+1}$
- $v_{i+1} \neq T_r(j+1)$ and $T_r(j+1) \neq \emptyset$:
  Remove all recorded nodes after $T_r(j)$ by setting $T_r(\ell) = \emptyset$ for all $\ell \geq j+1$;
  Store v_i in $T_r(j+1)$ if $v_{i+1} = v_i$;
  Return node $v_n \leftarrow v_{i+1}$.
- $v_{i+1} = T_r(j+1)$ and $T_r(j+1) \neq \emptyset$:
  Increment $j \leftarrow j+1$ and then go back to Step 2.

3.3 Recovery Scheme α-CO$_\infty$TT With Unconstrained List Size m

As mentioned previously, a node first performs procedure α-CO$_\infty$ to resolve hidden node issues when receiving a packet in recovery mode. This thus possibly induces successive α-CO$_\infty$ procedures for routing path correction. Finally, progressive procedure α-TT is called. However, if the node initiates recovery mode operation upon GF failure, it only performs α-TT because of no history information T_r. Specifically, recovery scheme α-CO$_\infty$TT is as follows:

1. If node v_i initializes recovery mode operation to forward a packet, let $T_r(0) \leftarrow v_i$ and apply α-TT(N_i, v_i, D) to obtain next relay v_n.

2. Otherwise, node v_i receives a packet in recovery mode with information T_r, D and $v_A = T_r(0)$. It first performs α-CO$_\infty(N_i, v_i, T_r)$ to obtain the selected node v_{i+1} and update history list T_r if necessary. Let $j = \min\{\ell : T_r(\ell+1) = \emptyset\}$ so that $T_r(j)$ is the last nonempty entry in T_r.

   - If $v_{i+1} = v_i$, apply α-TT$(N_i, v_i, T_r(j))$ to obtain next relay v_n;
   - Otherwise, the next relay is $v_n \leftarrow v_{i+1}$.

3.4 Recovery Scheme α-CO$_m$TT With Constrained List Size m

The size of T_r carried in a packet in recovery mode is practically limited up to some finite number $m \geq 2$, *i.e.* $|T_r| \leq m$. Under the constraint, the first entry $T_r(0)$ is popped out when the position of a node is pushed from the tail into a full list of T_r. To differentiate and indicate whether such an event of overflowing recorded data from T_r ever happens, we assume that a bit $B_I \in \{\text{TRUE, FALSE}\}$ for the purpose is also carried in a packet in recovery mode. Therefore, a sequence of procedure α-CO starts from $T_r(0)$ if overflow status B_I is FALSE, since the available information is sufficient to determine start-sweep line $T_r[0]h(D, T_r[0])$; Otherwise, it starts with the SC hinged at $T_r[1]$, swept from line $T_r[1]h(T_r[0], T_r[1])$. The latter for B_I =TRUE is because there is no way to find the start-sweep line and replicate the previous sweep of SC hinged at $T_r[0]$ for a better utilization of available information, even if $T_r[0]$ is exactly position v_A due to recurrent visits [13].

Thus, node v_i has information $\{T_r, B_I, v_A, D\}$ from a packet in recovery mode besides local state $N_i \cup \{v_i\}$. The procedure of α-CO$_m$TT for some finite $|T_r| \leq m$ all follows that of α-CO$_\infty$TT except the following modifications:

1. "Initialize indicator $B_I = \text{FALSE}$" is added into Step 1 of α-CO$_\infty$TT in Section 3.3 when recovery mode begins.

2. In procedure α-CO$_\infty$ in Section 3.2:

   - Step 1 is replaced by "Begin with $j = 0$ if $B_I = \text{FALSE}$ for no T_r overflow event; Otherwise, begin with $j = 1$."
   - In the procedure calls of Step 2, let $T_r(j+1) = \emptyset$ if $j = m-1$ because of constraint $|T_r| \leq m$.
   - In action items 1 and 3 of Step 2,"Store v_i in $T_r(j+1)$" is replaced by "Store v_i in $T_r(j+1)$ if $j+1 < m$; Otherwise, pop out $T_r(0)$, push v_i into T_r from the tail, and let $B_I = \text{TRUE}$".

4. SYSTEM OPERATION IN RECOVERY MODE

Recovery scheme α-CO$_m$TT with either constrained or unconstrained m involves nodes from list T_r in a retracted search. However, two adjacent nodes in the traced trajectory T_r may only have a virtual link between them. Will the scheme forward a packet through a virtual link? We answer this question by the following result.

Proposition 2 *α-CO$_m$TT always selects the next relay on a true link from the current node.*

Proof: The next relay is selected either by α-TT or α-CO$_m$. For α-TT, node v_i selects next relay v_{i+1} from N_i. Thus $v_{i+1} \in N_i$ and $v_i v_{i+1}$ is a true link. For α-CO$_m$, node v_i performs α-CO to trace and search $T_r(j)$ with increasing j sequentially for a node $v_{i+1} \neq T_r(j)$ and $v_{i+1} \in N_i \cup v_i \cup T_r(j)$ at the earliest logged node $T_r(j)$. Therefore, the selected node $v_{i+1} \in N_i$ holds if $v_{i+1} \neq v_i$ and $v_i v_{i+1}$ is a true link.

Given a network graph G of finite size, a packet to D is stuck at some non-isolated node v_A where recovery scheme α-CO$_m$TT, with constrained or unconstrained m, was initialized to forward the packet. With scheme α-CO$_m$TT applied alone, the packet will ultimately keep traveling in a routing loop. The issue is then how to detect such an event as early as possible with certainty from the list of nodes in T_r.

The following proposition for α-CO$_\infty$TT says that it is the time and place where twice the same cycle of nodes sequentially listed in T_r is found and the first node of the cycle is to be pushed into T_r again.

Proposition 3 *Suppose that recovery scheme α-CO$_\infty$TT is invoked at stuck node $v_A \in G$ with $N_{v_A} \neq \emptyset$ to forward a packet. Then the packet is trapped into a routing loop with certainty the moment when performing α-CO$_\infty$, some node v_{i+1} selected by the node itself is to push its position into $T_r(i+1)$ while such conditions $T_r(i-j) = T_r(i-k-j)$, $j = 0, 1, \cdots, k-1$, for some $k \geq 2$ and $v_{i+1} = T_r(i-k+1) = T_r(i-2k+1)$ are first satisfied.*

Proof: It implies that $T_r(i) = T_r(i-k) \neq T_r(i-2k)$. Otherwise, the above conditions have been satisfied earlier at node $T_r(i)$.

We first show that information available at earlier node $v_i \leftarrow T_r(i)$ is not sufficient to support that the packet has been definitely locked in a given loop forever. Let us consider the case that $T_r(i-k)T_r(i-k+1)$ is a virtual link across a "bay" in a network hole area, created when some nodes on the boundary of the bay were previously visited after node $T_r(i-k)$. Possibly some of their positions might have been sequentially pushed into and then all removed from the list T_r at node $T_r(i-k+1)$ (by the third action of step 2 of procedure α-CO$_\infty$ in Section 3.2). Thus, those visited nodes in the bay did not check whether they had neighbors hidden to node $T_r(i-k+1)$ by the sweep process of α-CO from the newly created start-sweep line $T_r(i-k+1)h(T_r(i-k), T_r(i-k+1))$. Now, starting from node v_i which is $T_r(i) = T_r(i-k)$, the packet will eventually visit those nodes in the bay again. In this case, some visited node v_i^- in the bay may have a neighbor v_i^+ hidden to node $T_r(i-k+1)$ and selected by performing α-CO from $T_r(i-k+1)h(T_r(i-k), T_r(i-k+1))$. When this turns out to be true, all entries in T_r after $T_r(i-k+1)$ are purged off, *i.e.* $T_r(\ell) = \emptyset$ for all $\ell > i-k+1$, according to the third action of step 2 of procedure α-CO$_\infty$. The packet is then sent to node v_i^+, leaving the given loop.

We then claim that after the position of $v_{i+1} = T_r(i-k+1) = T_r(i-2k+1)$ is pushed into $T_r(i+1)$, the packet will arrive at node $v_{i+2} \leftarrow T_r(i-k+2) = T_r(i-2k+2)$ directly or indirectly where the position of v_{i+2}, which is exactly $T_r(i-k+2) = T_r(i-2k+2)$, is pushed into $T_r(i+2)$. By induction, the sequence of nodes in the given cycle will then keep repeating in T_r.

To validate the claim, let us first divide the ordered nodes in T_r into three sublists S_l, S_p and S_h on the latest cycle, previous cycle and further history path, respectively; That is,

$$S_l = \{T_r(i+1), T_r(i), \cdots, T_r(i-k+2)\}, \quad (2)$$

$$S_p = \{T_r(i-k+1), T_r(i-k), \cdots, T_r(i-2k+2)\}, \quad (3)$$

$$S_h = \{T_r(i-2k+1), T_r(i-2k), \cdots, T_r(0)\}. \quad (4)$$

Then, $S_l = S_p$ and all have the same head node $T_r(i+1) = T_r(i-k+1) = T_r(i-2k+1)$. Obviously, the result from performing α-CO$_\infty$ with the list $T_r = \{S_l, S_p, S_h\}$ is the same as that from performing α-CO$_\infty$ with the list $\{S_p, S_h\}$ since $S_l = S_p$ is simply duplicated in T_r. Consequently, the development of routing from node $T_r(i+1)$ to some node $T_r(i+2)$ will be the same as that of routing from node $T_r(i-k+1)$ to $T_r(i-k+2)$. Thus $T_r(i+2) = T_r(i-k+2)$. This proves the above claim.

Note that a complete double cycle of the same sequence of k nodes in T_r does not imply that an actual routing loop consists of exactly k nodes. Some selected relays on hole bay boundary mentioned in the above proof that may or may not be ever stored into T_r are also in the actual loop. Those k nodes in T_r can be viewed as the boundary nodes of a network hole built by α-CO$_\infty$TT in a given finite network graph G and stuck node v_A with $N_{v_A} \neq \emptyset$.

For α-CO$_m$TT with constrained $|T_r| \leq m$ applied from stuck node v_A in the same network graph G, the number k of nodes on a constructed hole boundary may take a different value depending on the constrained size of m. Obviously, the smaller m the

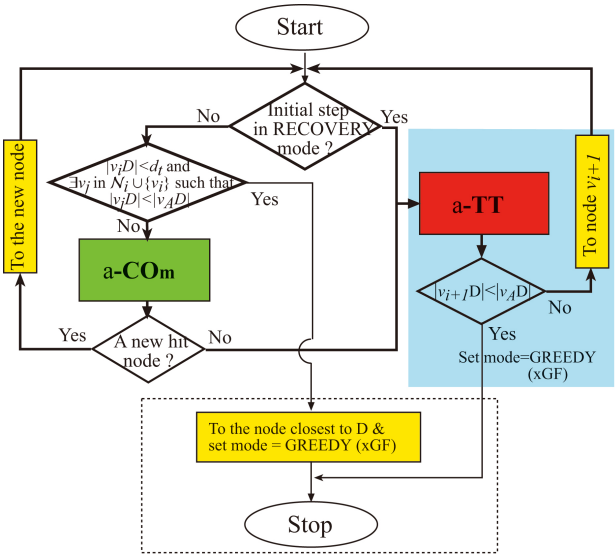


Fig. 4. The flow chart of recovery scheme involving α-CO$_m$TT and operation mode change to GF. The way to terminate routing is not shown.

larger k. If $2k > m$, the list size $|T_r| \leq m$ is too small to carry a double cycle of k nodes and hence the routing loop is undetectable by the two conditions in Proposition 3. On the other hand, T_r occasionally holds all the $2k$ positions of a double-cycle for some sufficient large maximum list size $m \geq 2k$.

In summary, it is not guaranteed to detect the occurrence or existence of any routing loop based on only a finite constrained history list T_r available for decisions. However, we have the following result.

Corollary 1 *For α-CO$_m$TT with constrained $|T_r| \leq m$ started from stuck node $v_A \in G$ with $N_{v_A} \neq \emptyset$, the recovery mode has been locked in a routing loop if the list of nodes in T_r satisfies the two conditions in Proposition 3.*

4.1 Operation Mode Change From Recovery to Greedy

It is necessary to switch from α-CO$_m$TT to GF somewhere in order to keep relaying a packet definitely toward destination D and also to avoid facing more potential routing problems in recovery mode. This demands to change to GF as early as possible. However, routing by α-CO$_m$TT has the benefit of memory from T_r in a packet for tracking a network hole boundary. Routing by GF does not have. This demands not to change to GF too hasty.

Taking into account the above conflicts of effect, the proposed recovery scheme is depicted in the flowchart of Fig. 4. Besides α-TT and α-CO$_m$ presented previously, it includes two exits to GF separately controlled by:

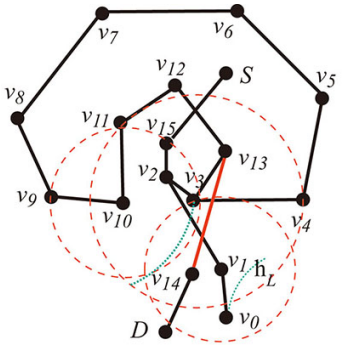


Fig. 5. A network graph of 18 nodes and edges.

4.1.1 Condition $|v_{i+1}D| < |v_AD|$

Immediately after procedure α-TT, it checks whether next relay v_{i+1} selected by α-TT at node v_i is closer to D than stuck node v_A. This is a more constrained and thus conservative condition to return to GF because only node v_{i+1} selected by α-TT is involved.

We overlay this typical condition for operation mode change with a more aggressive one that there is some $v_j \in N_i \cup \{v_i\}$ such that $|v_jD| < |v_AD|$. This condition involves node v_i itself and all of its neighbors for test but is limited to apply in some network areas, as described next.

4.1.2 Distance d_t

It is a threshold of distance to destination, within which recovery mode operation tends to be switched to GF in a more aggressive manner. The threshold d_t is viewed as a design parameter. Specifically, when receiving a packet in recovery mode, node v_i checks conditions

$$\begin{aligned} &|v_iD| < d_t \text{ and there is some node} \\ &v_j \in N_i \cup \{v_i\} \text{ such that } |v_jD| < |v_AD|. \end{aligned} \tag{5}$$

If so, node v_i changes operation mode to GF and sends the packet directly to the neighbor $v_j \in N_i$ closest to D if $v_j \neq v_i$.

As shown in Fig. 4, the recovery mode is initialized to perform α-TT on the righthand branch of the flow chart whenever routing operation at a node, say node v_i, encounters a GF failure. Thereafter a node, v_i, receiving a packet in recovery mode starts from the lefthand branch, first checking whether condition (5) is true. If so, it changes operation mode as stated previously. If not, it proceeds to perform α-CO$_m$.

4.2 An Example of α-CO$_m$TT Operation

To see how scheme α-CO$_m$TT works to recover from GF failure, we consider the network graph in Fig. 5 where a packet destined to D has been forwarded from its source S through nodes v_{15}, v_2, v_1 and stuck at node v_0 by GF. Suppose that α-CO$_m$TT with unconstrained m is employed in recovery mode. Then, the sequence of processing is rowwise listed in Table 1. The node where the packet reaches is listed in column 1.

The method in operation is in column 2. Each entry in column 3 only gives the chord length of an SC employed to sweep from the start-sweep line listed in the same row of column 4 and determine the selected node in the corresponding row entry of column 5, although iterative sweeps and/or eSW are performed. The history state T_r after α-CO$_m$TT operation at a node is in column 6.

Table 1. Recovery scheme α-CO$_\infty$TT begins at node v_0 in the network of Fig. 5 and continues in this mode from node(row) v_0 to v_{14} where operation mode changes to GF.

Pkt@	Proc.	SC η	from	hit	list T_r
v_0	α-TT	η_L	$v_0h(D,v_0)$	v_1	v_0
v_1	α-CO	η_L	$v_0h(D,v_0)$	v_1	v_0,v_1
	α-TT	$\lvert v_1v_2\rvert$	$v_1h(v_0,v_1)$	v_2	
v_2	α-CO	$\lvert v_1v_3\rvert$	$v_1h(v_0,v_1)$	v_3	v_0,v_1
v_3	α-CO	$\lvert v_1v_3\rvert$	$v_1h(v_0,v_1)$	v_3	v_0,v_1,v_3
	α-TT	$\lvert v_3v_4\rvert$	$v_3h(v_1,v_3)$	v_4	
v_4	α-CO	$\lvert v_3v_4\rvert$	$v_3h(v_1,v_3)$	v_4	v_0,v_1,v_3,v_4
	α-TT	$\lvert v_4v_5\rvert$	$v_4h(v_3,v_4)$	v_5	
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
v_{10}	α-CO	η_L	$v_9h(v_8,v_9)$	v_{10}	$v_0,v_1,v_3,v_4,\cdots,$
	α-TT	$\lvert v_{10}v_{11}\rvert$	$v_{10}h(v_9,v_{10})$	v_{11}	v_{10}
v_{11}	α-CO	$\lvert v_{10}v_{11}\rvert$	$v_{10}h(v_9,v_{10})$	v_{11}	$v_0,v_1,v_3,v_4,\cdots,$
	α-TT	η_L	$v_{11}h(v_{10},v_{11})$	v_{12}	v_{10},v_{11}
v_{12}	α-CO	η_L	$v_{11}h(v_{10},v_{11})$	v_{12}	$v_0,v_1,v_3,v_4,\cdots,$
	α-TT	$\lvert v_{12}v_{13}\rvert$	$v_{12}h(v_{11},v_{12})$	v_{13}	v_{10},v_{11},v_{12}
v_{13}	α-CO	η_L	$v_{10}h(v_9,v_{10})$	v_3	$v_0,v_1,v_3,v_4,\cdots,$ v_{10}
v_3	α-CO	η_L	$v_{10}h(v_9,v_{10})$	v_3	$v_0,v_1,v_3,v_4,\cdots,$
	α-TT	$\lvert v_3v_4\rvert$	$v_3h(v_{10},v_3)$	v_4	v_{10},v_3
v_4	α-CO	$\lvert v_3v_4\rvert$	$v_3h(v_{10},v_3)$	v_4	$v_0,v_1,v_3,v_4,\cdots,$
	α-TT	$\lvert v_4v_5\rvert$	$v_4h(v_3,v_4)$	v_5	v_{10},v_3,v_4
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
v_{10}	α-CO	η_L	$v_9h(v_8,v_9)$	v_{10}	$v_0,v_1,v_3,v_4,\cdots,$
	α-TT	$\lvert v_{10}v_{11}\rvert$	$v_{10}h(v_9,v_{10})$	v_{11}	$v_{10},v_3,v_4,\cdots,v_{10}$
v_{11}	α-CO	$\lvert v_{10}v_{11}\rvert$	$v_{10}h(v_9,v_{10})$	v_{11}	$v_0,v_1,v_3,v_4,\cdots,$
	α-TT	η_L	$v_{11}h(v_{10},v_{11})$	v_{12}	$v_{10},v_3,v_4,\cdots,v_{11}$
v_{12}	α-CO	η_L	$v_{11}h(v_{10},v_{11})$	v_{12}	$v_0,v_1,v_3,v_4,\cdots,$
	α-TT	$\lvert v_{12}v_{13}\rvert$	$v_{12}h(v_{11},v_{12})$	v_{13}	$v_{10},v_3,v_4,\cdots,v_{12}$
v_{13}	α-CO	η_L	$v_3h(v_{10},v_3)$	v_{14}	$v_0,v_1,v_3,v_4,\cdots,$ v_{10},v_3
v_{14}	GF			D	

We see from the table that when the packet reaches node v_{13} the first time, procedure α-CO$_m$ performed there finds that its connected neighbor v_3 is hit by the SC of length $\eta_L = \max\{\eta_L, |v_{10}v_3|\} < |v_{10}v_{11}|$ swept from start-sweep line $v_{10}h(v_9, v_{10})$ now but hidden to the hole boundary node v_{10} earlier. This entails a modification of recorded hole boundary by purging nodes v_{11} and v_{12} from list T_r. The packet is then sent to the selected relay v_3. At node v_3 the second time, position v_3 is pushed into the tail of T_r after α-CO$_m$ operation. This notionally creates a virtual link $v_{10}v_3$ to close up the bay area formed by nodes $v_{10}, v_{11}, v_{12}, v_{13}$ and v_3 according to the development of listed nodes in T_r.

We again see that when the packet reaches node v_{13} the second time, next to the bottom row, α-CO$_m$ operation there finds that its connected neighbor v_{14} is hit by SCs of length $|v_3v_4|$ and of length $\eta_L = \max\{\eta_L, |v_3v_{14}|\}$ swept from the start-sweep line $v_3h(v_{10}, v_3)$, due to the virtual link $v_{10}v_3$ created previously, and finally selected as the next relay. The latest sequence of recorded nodes $v_4, v_5, \cdots, v_{12}$ is then purged from T_r. At node v_{14}, if procedure α-CO$_m$ were performed first, node v_{14} should be inserted into T_r, creating another virtual link v_3v_{14}. Instead, operation mode is changed to GF there. On the other hand, suppose that link $v_{13}v_{14}$ does not exist. Then the row of v_{13} in the table after α-CO$_m$ procedure would have $T_r = \{v_0, v_1, v_3, v_4, \cdots, v_{10}, v_3, v_4, \cdots, v_{10}\}$ in the last column after v_{11} and v_{12} are removed from the tail of T_r. Thereafter, the packet is sent from v_{13} to node v_3 where position v_3 is to be inserted into T_r after α-CO$_m$ operation, not shown in the table. This exactly satisfies the conditions in Proposition 3 to conclude the occurrence of routing in a loop.

5. SIMULATIONS AND RESULTS

5.1 Network Graphs for Simulations

We consider two types of network topology of the same size 40×40 in the first quadrant of a two-dimensional Cartesian system. One without artificial voids is uniformly distributed with 4800, 4200, and 3600 nodes for three different node density levels, respectively. The other with two artificial voids shown in Fig. 6 (Refer to [14, 17] for the geometry of the two holes) is uniformly distributed with 4320, 3840, and 3360 nodes over areas other than the two voids. In each network, the node closest to position $(5,5)$ or $(35,35)$ is designated as the source S and destination D of a packet, respectively.

In a network topology, any two nodes, say v_i and v_j, have a bidirectional link between them if and only if node v_i receives from v_j signals with power level larger than some given threshold p_t and vice versa. Otherwise, they have no direct communication link.

We consider Rayleigh fading, fix the mean received signal power level to be 0 dB at distance 1, the reference distance [21], and use two path loss exponents 3 and 4 separately to create networks of different wireless connection regularity. Furthermore, we set the minimum power level $p_t = \ln(10)/2$. Thus, for a signal transmitted by node v_i and path loss exponent 4, node v_j receives mean signal power $1/|v_iv_j|^4$. The probability of the received signal power larger than p_t is then $e^{-p_t|v_iv_j|^4}$ [20, 22, 23]. By our assumption of bidirectional communications, nodes v_i and v_j with $|v_iv_j| = 1$ have a link between them with probability $e^{-\ln(10)/2} \times e^{-\ln(10)/2} = \frac{1}{10}$. Note that under the constraint of the same received signal power level at reference distance, a network created based on path loss

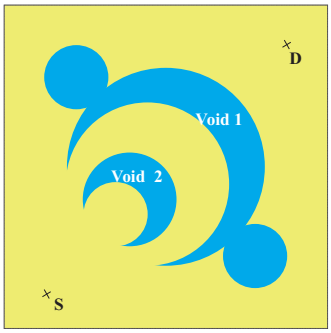


Fig. 6. A network area of 40×40 in the first quadrant has two artificial voids with geometry defined in [14, 17].

Table 2. Totally 4×10^4 random network graphs are created for each sample of network setting in node density and wireless path loss exponent. Each entry in column 2 or 4 shows the number of them having a path between the node closest to $(5,5)$ and that closest to $(35,35)$ for a given number of nodes, listed in column 1, uniformly distributed. Correspondingly the mean shortest path hop count is listed in column 3 or 5.

Path Loss E.	4		3	
Nodes	Connected	Avg. hops	Connected	Avg. hops
Networks without artificial void				
4800	38155	76.25	36670	79.57
4200	33072	87.11	27455	94.18
3600	9095	108.00	2141	112.55
Networks with two artificial voids				
4320	38105	101.84	36080	105.14
3840	29588	111.44	21405	116.94
3360	7374	124.95	2214	130.37

exponent 4 supports a higher degree of connection regularity than that based on path loss exponent 3.

A network graph is created once and remains the same in a full session of routing a packet from S to D. To send another packet, another random network graph is created in advance. Totally, there are 4×10^4 network graphs created and hence packets sent, which represents the size of a simulation sample for a given network setting in node density and path loss exponent. In each sample, the number of network graphs in which a path exists between S and D is given in columns 2 and 4 of Table 2 for path loss exponents 4 and 3, respectively. Those numbers are obtained by applying the Dijkstra's shortest path algorithm [24] with global state information of each network graph.

They are used in computing routing success probabilities (RT SP). Their mean *shortest* path hop counts are then listed in columns 3 and 5 of the table accordingly. We can see that for a given number of nodes deployed, each sample of random networks created with path loss exponent 4 provides stronger connectivity between S and D (or shorter mean path hop counts) than that with path loss exponent 3, due to the setting of distance-path loss model mentioned previously.

Table 3. Maximum routing success probability (RT SP) under unlimited history list size m and optimal η_L and d_t for scheme GF combined with α-CO$_\infty$TT.

Path Loss E.	4		3	
Nodes	RT SP	η_L ; d_t	RT SP	η_L ; d_t
Networks without artificial void				
4800	.9344	1.26; 2	.8069	1.28; 4
4200	.7283	1.18; 4,5	.4531	1.26; 5
3600	.424	1.16; 3,6	.264	1.21; 17,19
Networks with two artificial voids				
4320	.9066	1.20; 3	.7838	1.28; 9
3840	.7514	1.18; 8	.5245	1.23; 10
3360	.544	1.19; 5,9,12	.357	1.25; 64

5.2 Performance

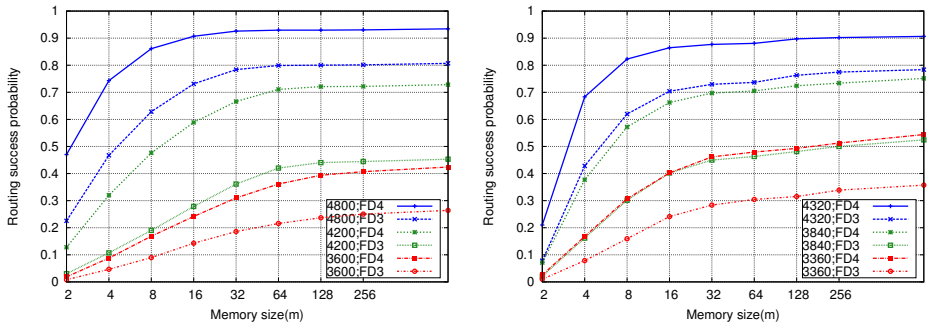
5.2.1 The effect of memory size m

In simulation, we set $m = 25600$ as a hypothetically unlimited memory size on T_r and only apply Proposition 3 to stop routing in a loop after recovery scheme α-CO$_\infty$TT is invoked. Results on the proportion of packets successfully delivered from S to D are listed in columns 2 and 4 of Table 3, under the term RT SP, for path loss exponents 4 and 3, respectively. They are the maximum achievable performance of RT SP for GF working with α-CO$_\infty$TT under optimal minimum SC size η_L and distance threshold d_t listed in columns 3 and 5.

For α-CO$_m$TT with constrained size m, we apply Proposition 3 as well as a maximum hop count limit, 3200 or 6400, in recovery mode to stop routing. The impact of finite list sizes $m = 2^1, 2^2, \cdots, 2^8$ in scheme α-CO$_m$TT on the performance of successful delivering a packet from S to D is shown in Figs. 7 (a) and (b) for networks without and with two artificial voids, respectively. In each figure, those for path loss exponents 4 and 3 are separately denoted by FD4 and FD3. The performance for unlimited memory size $m = \infty$ is plotted on the rightmost, over x-coordinate 1024. All are optimal performance for each given network sample, total number of nodes, and m. Correspondingly, Figs. 8 (a) and (b) illustrate their routing path stretch, the ratio of mean hop count by GF$\oplus\alpha$-CO$_m$TT to that by the shortest path routing.

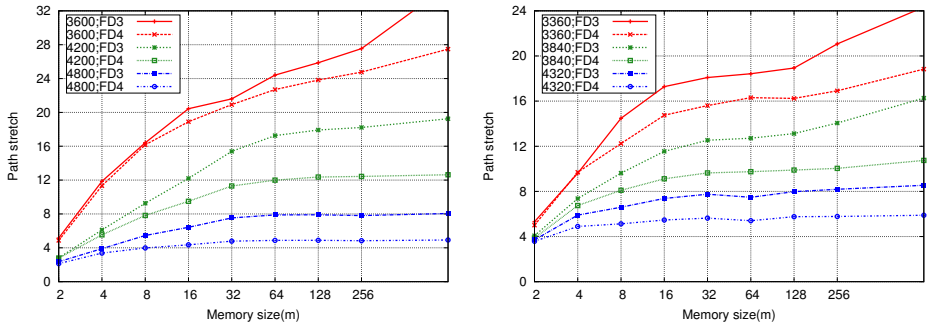
We can see from Figs. 7 (a) and (b) that routing success probabilities obviously increase with m for all cases of network setting and are then leveling off at some moderate size m. These reveal a worthy tradeoff of exploiting extended packet headers for cooperative sweep operation to bypass routing holes. We also see that the performance for a given network setting with FD4 is better than that with FD3 since the network with path loss exponent 4 provides a better network connectivity seen from Table 2 and thus results in shorter routing path stretches shown in Figs. 8 (a) and (b). Certainly the higher the network node density, the higher the routing success probability seen from Fig. 7 and thus the lower the routing path stretch seen from Fig. 8, due to less or smaller network voids. Obviously, the cost of larger m for performance is routing path stretch, which is significant particularly for low density networks with 3600 or 3360 nodes.

We limit the following presentation for two representative network settings only.



(a) (b)

Fig. 7. The probability of successful routing from S to D versus memory length (m) for networks (a) without artificial voids and (b) with two artificial voids.

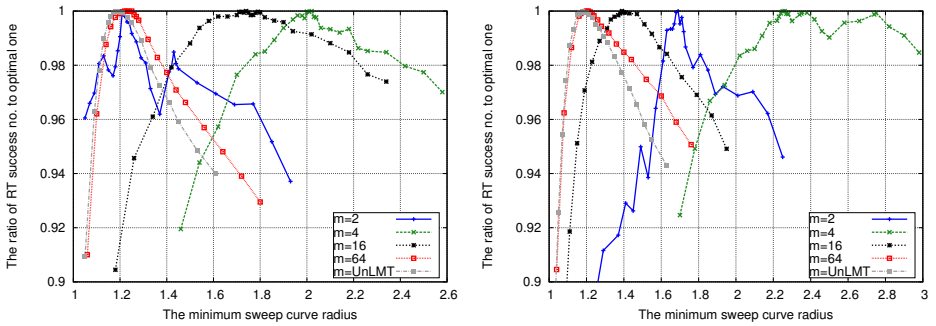


(a) (b)

Fig. 8. Average SD-path hop count stretch versus memory length (m) for networks (a) without artificial voids and (b) with two artificial voids.

5.2.2 The effect of minimum sweep curve size, η_L

Figs. 9 (a) and (b) show the ratio of optimal routing success probability for a given minimum SC size η_L to the global optimal one versus η_L, for networks with no artificial voids and 4200 nodes and with two artificial voids and 3840 nodes, respectively. For simplicity, only results for list size $m = 2, 4, 16, 64, \infty$(UnLMT) and path loss exponent 4 are plotted. We see that for each given list size m, there exists a global optimal SC size, denoted as $\eta_L^*(m)$, for maximum routing success performance and that except $m = 2$, $\eta_L^*(m)$ generally decreases with $m \geq 4$ and approaches to the corresponding entry listed in column 3 of Table 3 for unlimited m. We also see that the loss of routing success performance is more sensitive for assigned minimum SC size $\eta_L(m) < \eta_L^*(m)$. The reason is mainly that if $\eta_L(m)$ is set too small, the selected relay tends to lie further out of a network void boundary, which potentially violates the criteria of performing rotational sweep to select the next relay on a void boundary and leads to looping beyond the network void encountered. On the other hand, if $\eta_L(m)$ is too large, the SC may miss hitting some edge-intersection node [13] that may have an edge on a path to D, causing performance loss.

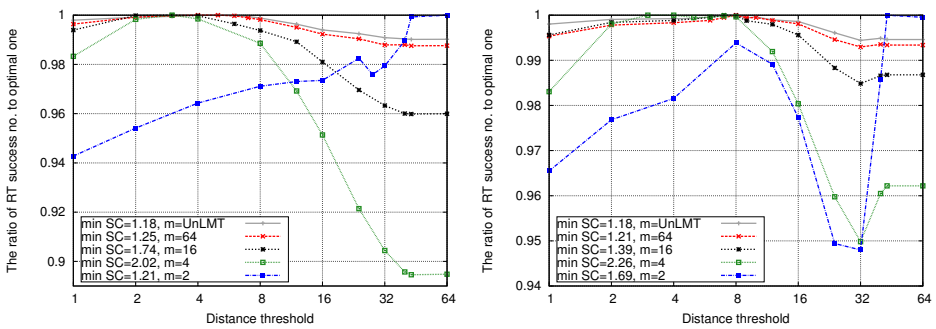


(a) (b)

Fig. 9. The ratio of optimal routing success performance under a given minimum SC size η_L to the global optimal one versus η_L, for path loss exponent 4 and networks (a) with no artificial void and 4200 nodes and (b) with two artificial voids and 3840 nodes. Results are optimized with regard to d_t. UnLMT indicates ∞.

5.2.3 The effect of distance threshold, d_t

For each $m = 2, 4, 16, 64, \infty$ and optimal minimum SC size $\eta_L^*(m)$ assigned, the ratio of routing success numbers to the optimal one versus d_t is shown in Figs. 10(a) and 10(b) for the previous samples of networks. Here the setting $d_t = 64$ on the x-coordinate implies that the routing scheme always seeks to operate GF since $64 > 40\sqrt{2} > |SD| \approx 30\sqrt{2}$. We see that for each m there exists a range of optimal d_t^* for maximum routing success performance. Except $m = 2$, d_t^* is at about 4 for the sample of networks without artificial voids and about 8 for that with two artificial voids. This agrees with the design objective of using d_t described in Section 4.1.2. Recovery scheme α-CO$_m$TT with insufficient memory size $m = 2$ is unable to resolve most local minimum issues through taking more detour hops, which may then lead to facing further routing problems. This thus supports the strategy of switching to GF as early as possible, by setting $d_t^* \geq |SD| \approx 30\sqrt{2}$.



(a) (b)

Fig. 10. The ratio of routing success performance to optimal achievable one versus distance threshold d_t for path loss exponent 4 and networks (a) with no artificial void and 4200 nodes and (b) with two artificial voids and 3840 nodes. Optimal minimum SC size η_L for each m is employed. UnLMT indicates ∞.

5.2.4 Discussions and further issues

It should be noted that localized rotational sweep routing algorithms [12, 13] developed under UDA are useless in those network topologies in simulation while protocols based on distributed Bellman-Ford algorithm [24], like AODV [25, 26], can easily reach routing success probability 1 and short path stretches at the cost of network state message propagation and routing table updates to counter the effect of changes in network topology. Exploiting packet header overheads, our scheme nicely preserves the feature of localized routing still at the cost of path stretches but lacks the guarantee of packet delivery. To be feasible, the size of packet header overheads should be limited while routing success probability should be high. This implies that the distance of SD by a routing request should be limited in some range, unacceptable for large or low node-density networks. This issue can be resolved by having a large network structured logically into a hierarchy of two layers where some nodes actively function as key relay stations, like cluster heads, whose positions are available through location services. In this context, sending a packet to a far destination can rely on a sequence of shorter and more reliable routes between relay stations [27, 28]. We leave this subject for further study.

6. CONCLUDING REMARKS

We have presented a cooperative rotational sweep scheme α-CO$_m$TT to bypass network voids and recover from greedy forwarding failure in geographic routing under realistic wireless coverage. The scheme is designed to identify and select a hidden node as the next relay through retracted search algorithm α-CO$_m$ and to locate the next relay through progressive search algorithm α-TT. Both algorithms involve iterative rotational sweeps with circular arcs of decreasing size subjective to a minimum size constraint. The proposed scheme retains the feature of localized routing and thus scalability. Simulation results have shown that a reasonable routing success probability can be achieved at the cost of packet overheads carrying a few visited node positions in recent history of recovery mode operation. Furthermore, the routing success performance is affected by the minimum sweep curve size and by the distance threshold to packet destination imposed for aggressively switching from recovery mode to greedy one. Future research challenges include an analytic way to resolve the optimal minimum size of circular arc employed in α-CO$_m$TT and a practical way to determine its value.

ACKNOWLEDGMENT

We would like to thank the anonymous reviewers for their detailed and helpful comments. This work was supported by the National Science Council of Taiwan under Grants MOST 104-2221-E-003-012, MOST 105-2221-E-003-003 and MOST 106-2221-E-003-005. Han's work was supported by the National Natural Science Foundation of China (Grant No. 61671007), Start Fund of Dongguan University of Technology (KCYXM2017025).

REFERENCES

1. H. Takagi and L. Kleinrock, "Optimal transmission ranges for randomly distributed packet radio terminals," *IEEE Transactions on Communications*, Vol. 32, 1984, pp. 246-257.
2. E. Kranakis, H. Singh, and J. Urrutia, "Compass routing on geometric networks," in *Proceedings of the 11th Canadian Conference on Computational Geometry*, 1999, pp. 51-54.
3. B. Karp and H. T. Kung, "GPSR: Greedy perimeter stateless routing for wireless networks," in *Proceedings of IEEE/ACM Mobicom*, 2000, pp. 243-254.
4. G. G. Finn, "Routing and addressing problems in large metropolitan-scale internetworks," Technical Report ISI/RR-87-180, Information Sciences Institute, USC, 1987.
5. J.-T. Tsai and Y.-H. Han, "Alternative forwarding strategies for geographic routing in wireless networks," *International Journal of Ad Hoc and Ubiquitous Computing*, Vol. 27, 2018, pp. 295-307.
6. D. Chen and P. K. Varshney, "A survey of void handling techniques for geographic routing in wireless networks," *IEEE Communications Surveys and Tutorials*, Vol. 9, 2007, pp. 50-67.
7. P. Bose, P. Morin, I. Stojmenovic, and J. Urrutia,"Routing with guaranteed delivery in ad hoc wireless networks," in *Proceedings of the 3rd International Workshop Discrete Algorithms and Methods for Mobile Computing and Communications*, 1999, pp. 48-55.
8. F. Kuhn, R. Wattenhofer, Y. Zhang, and A. Zollinger, "Geometric routing: Of theory and practice," in *Proceedings of the 22nd ACM Symposium on Principles of Distributed Computing*, 2003, pp. 63-72.
9. F. Kuhn, R. Wattenhofer, and A. Zollinger, "An algorithmic approach to geographic routing in ad hoc and sensor networks," *IEEE/ACM Transactions on Networking*, Vol. 16, 2008, pp. 51-62.
10. B. Karp, "Geographic routing for wireless networks," Ph.D. Thesis, Division of Engingeering and Applied Sciences, Harvard University, 2000.
11. W.-J. Liu and K.-T. Feng, "Greedy routing with anti-void traversal for wireless sensor networks," *IEEE Transactions on Mobile Computing*, Vol. 8, 2009, pp. 910-922.
12. S. Rüehrup and I. Stojmenović, "Optimizing communication overhead while reducing path length in beaconless georouting with guaranteed delivery for wireless sensor net- works", *IEEE Tranactions on Computers*, Vol. 62, 2013, pp. 2440-2453.
13. A. Mostefaoui, M. Melkemi, and A. Boukerche, "Localized routing approach to bypass holes in wireless sensor networks," *IEEE Transactions on Computers*, Vol. 63, 2014, pp. 3053-3065.
14. J.-T. Tsai and Y.-H. Han, "Geographic routing with enhanced local information for wireless networks," *Journal of Information Science and Engineering*, Vol. 32, 2016, pp. 1261-1288.
15. Q. Fang, J. Gao, and L. Guibas, "Locating and bypassing routing holes in sensor networks," in *Proceedings of IEEE INFOCOM*, 2004, pp. 2458-2468.
16. S. Rührup, H. Kalosha, A. Nayak, and I. Stojmenović, "Message-efficient beaconless georouting with guaranteed delivery in wireless sensor, ad hoc, and actuator networks," *IEEE/ACM Transactions on Networking*, Vol. 18, 2010, pp. 95-108.

17. J.-T. Tsai and Y.-H. Han "Cooperative rotational sweep schemes for geographic routing", in *Proceedings of IEEE International Conference on Communications*, 2016, pp. 3878-3884.
18. M. Zorzi and R. R. Rao, "Geographic random forwarding (GeRaF) for ad hoc and sensor networks: multihop performance," *IEEE Transactions on Mobile Computing,* Vol. 2, 2003, pp. 337-348.
19. J. A. Sanchez, P. M. Ruiz, and R. Marin-Perez,"Beacon-less geographic routing made practical: challenges, design guidelines, and protocols," *IEEE Communication Magazine*, Vol. 47, 2009, pp. 85-91.
20. T. Aguilar, S.-J. Syue, V. Gauthier, H. Afifi, and C.-L. Wang, "CoopGeo: A beaconless geographic cross-layer protocol for cooperative wireless ad hoc networks," *IEEE Transactions on Wireless Communications*, Vol. 10, 2011, pp. 2554-2565.
21. T. S. Rappaport, *Wireless Communications: Principles and Practice*, Prentice-Hall, NJ, 1996.
22. J.-T. Tsai, "Transmission rate scheduling and stopping time for time-sensitive multicast stream traffic in cellular networks," *IEEE Transactions on Wireless Communications*, Vol. 13, 2014, pp. 1754-1765.
23. S. Ross, *Stochastic Processes*, 2nd ed., John Wiley & Sons, NY, 1996.
24. D. Bertsekas and R. Gallager, *Data Networks*, 2nd ed., Englewood Cliffs, Prentice-Hall, NJ, 1992.
25. C. E. Perkins and E. M. Royer, "Ad hoc on-demand distance vector routing," in *Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications*, 1999, pp. 90-100.
26. A. S. Tanenbaum and D. J. Wetherall, *Computer Networks*, 5th ed., Pearson International, 2011.
27. Y. B. Ko and N. H. Vaidya, "Location-aided routing (LAR) in mobile ad hoc networks," in *Proceedings of ACM/IEEE MobiCom*, 1998, pp. 66-75.
28. Q. Fang, J. Gao, L. Guibas, V. de Silva, and L. Zhang, "GLIDER: Gradient landmark-based distributed routing for sensor networks," in *Proceedings of IEEE INFOCOM*, 2005, pp. 339-350.
29. Y-J Kim, R. Govindan, B. Karp, and S. Shenker, "On the pitfalls of geographic face routing," in *Proceedings of DIALM-POMC*, 2005, pp. 34-43.

Jung-Tsung Tsai received his BS, MS, and Ph.D. degrees from the National Tsing-Hua University, the National Taiwan University, and the University of California, San Diego, in 1985, 1987, and 1995, respectively, all in Electrical Engineering. He served in Taiwan Marine Corps from 1987 to 1989. He was a Lecturer at Kuang-Wu Institute of Technology, Taipei, from 1989 to 1991. He was a Postdoctoral Researcher from 1995 to 1996 in the Department of Electrical and Computer Engineering at UCSD. He worked in the Industrial Technology Research Institute, Hsinchu, Taiwan, developing web-based network management tools from 1996 to 1997. He joined the faculty of National Taiwan Normal University in 1997, where he is currently a Professor in the Department of Computer Science and Information Engineering. His research interests include stochastic scheduling, routing, mobility management, and rate adaptation.

Yunghsiang S. Han received his Ph.D. degree from Syracuse University in 1993. He was, from 1993 to 1997, with HuaFan University, with National ChiNan University from 1997 to 2004, and with National Taipei University from 2004 to 2010. From August 2010 to January 2017, he was with the Department of Electrical Engineering at National Taiwan University of Science and Technology as Chair Professor. Now he is with School of Electrical Engineering & Intelligentization at Dongguan University of Technology, China. He is also a Chair Professor at National Taipei University from February 2015. His research interests are in error-control coding, wireless networks, and security. He has published several highly cited works and serves as the editors of several international journals. Dr. Han was the winner of the Syracuse University Doctoral Prize and a Fellow of IEEE. One of his papers won the prestigious 2013 ACM CCS Test-of-Time Award in cybersecurity.