GPS-Free, Boundary-Recognition-Free, and Reliable Double-Ruling-Based Information Brokerage Scheme in Wireless Sensor Networks

Chia-Hung Lin, Jian-Jhih Kuo, Bing-Hong Liu, and Ming-Jer Tsai
Department of Computer Science
National Tsing Hua University
Hsinchu, Taiwan, ROC
E-mail: mjtsai@cs.nthu.edu.tw

Abstract—We study the information brokerage schemes in wireless sensor networks, which allow consumers to obtain data from producers by replicating and retrieving data in a certain set of sensors, and propose a novel information brokerage scheme, termed RDRIB. Unlike existing information brokerage schemes, RDRIB guarantees successful data retrieval without using any boundary detection algorithm and the geographic location information acquired by the global positioning system (GPS). In RDRIB, the double-ruling technique is used to replicate and retrieve the data within a constructed virtual boundary, and simulations show that RDRIB has good performance in terms of the replication memory overhead, the replication message overhead, the retrieval message overhead, the retrieval latency, and the construction message overhead.

Index Terms—Information brokerage scheme, GPS-free, boundary-recognition-free, wireless sensor network.

1 INTRODUCTION

A wireless sensor network consists of several sensors, each of which has the ability to collect, process, and store environmental information as well as to communicate with others via inter-sensor communication. These characteristics allow wireless sensor networks to be used in a wide range of applications, including environmental monitoring, battlefield surveillance, health care, and so on. In these applications, the data sensed by sensors are processed and interpreted within the network such that information exchange between sensors is necessary. In addition, many routing protocols, which forward data from a source node to a destination node, are proposed for wireless sensor networks [1], [2], [3], [4], [5], [6]; each of them requires a reliable information exchange scheme that can provide the source node with the location information of the destination node. In this paper, we undertake the development of the methods of information exchange in wireless sensor networks.

Consider the data that is collected or generated by a node (a producer) and that, later, a node (a consumer) is interested in. To exchange the data between the producer and the consumer, since the consumer does not know which node has the data-of-interest, one trivial scheme is to flood a query throughout the network and retrieve the data-of-interest by receiving the response generated from the producer. By contrast, another trivial scheme is to replicate the data of the producer to all nodes in the network such that each consumer can retrieve the data-of-interest immediately. However, each trivial scheme demands a great deal of message or memory overhead to replicate or retrieve the data.

For efficient information exchange, many information brokerage schemes, such as GLS [7], GHT [8], FMMS [9], XYLS [10], iMesh [11], Double Rulings [12], 3DLS [13], HD [14], LBIB [15], and Hop/SHU [16], are proposed. However, these schemes each have at least one of the following problems:

(1) GPS is needed. In GLS, GHT, FMMS, XYLS, iMesh, and Double Rulings, each sensor node is required to possess the geographic location information, which is obtained with difficulty in wireless sensor networks because the global positioning system (GPS) consumes a large amount of power and does not work indoors.

(2) Distance-sensitivity is not provided. GLS, GHT, and 3DLS are not distance-sensitive because the path of the data retrieval for a consumer may not be proportional to the distance between the producer and the consumer. By contrast, FMMS, XYLS, iMesh, Double Rulings, and Hop/SHU use the double-ruling technique to approach this problem.

(3) Successful data retrieval is not guaranteed. In FMMS, XYLS, 3DLS, LBIB, and Hop/SHU, it is not guaranteed that a consumer can always retrieve the data-of-interest.

(4) Global topology information is required. In HD and LBIB, each node is required to obtain rather complex global topology information, which incurs a great deal of message and memory overhead.

(5) Network boundary detection is necessitated. In
Hop/SHU, the network boundary is required to be identified perfectly. However, existing boundary detection algorithms demand a great deal of message communication overhead and can work only in rather dense networks [17], [18], [19].

(6) Large-sized message for data replication is demanded. In 3DLS, the message for data replication must be extremely large. However, the sizes of the packets in wireless sensor networks are limited.

To the best of our knowledge, no GPS-free, boundary-recognition-free, and reliable double-ruling-based information brokerage schemes exist to date, thereby providing motivation for this paper. The remainder of this paper is organized as follows. Research related to the GPS-free information brokerage scheme is presented in Section 2. A boundary-recognition-free and GPS-free double-ruling-based information brokerage scheme (DRIB) that does not require global topology information to replicate and retrieve the data is proposed in Section 3. A reliable boundary-recognition-free and GPS-free double-ruling-based information brokerage scheme (RDRIB) is presented in Section 4. We analyze RDRIB in Section 5, and evaluate, by simulations, the performance of RDRIB in Section 6. Finally, we conclude this paper with a discussion of future research in Section 7.

2 RELATED RESEARCH

3DLS [13], HD [14], LBIB [15], and Hop/SHU [16] are GPS-free information brokerage schemes. In 3DLS, each node is assigned a virtual height, reflecting the network density in the surrounding area, and a node having the highest virtual height in the 1-hop neighborhood is a hilltop. The data are replicated (or retrieved) in the hilltops encountered by the message for data replication (or retrieval). 3DLS is not distance-sensitive and has no theoretical analysis on the successful rate of data retrieval. In addition, the message for data replication is extremely large because it must contain the identifiers (IDs) of all nodes in the 1-hop neighborhood of the nodes encountered, which makes 3DLS very hard to implement in wireless sensor networks.

In HD, the nodes in the network are classified into a hierarchy of clusters in which each node belongs to one cluster at each level. The smallest cluster (at the lowest level) consists of only one node, and a cluster at level $k$ is a neighboring cluster of a node if the cluster contains one node whose hop distance from the node is not greater than $\alpha \cdot 2^{k+1}$ for a given number $\alpha$. The data are replicated in the hashed nodes in all neighboring clusters of the producer at all levels; the data are retrieved by querying the hashed nodes in the clusters in which the consumer resides in an increasing order of levels until a hashed node with the data-of-interest is reached. HD guarantees successful data retrieval; however, it demands a great deal of message and memory overhead to replicate the data.

GLIDER [20] is a virtual-coordinate-based routing protocol, in which all nodes are divided into tiles. In GLIDER, the connections between neighboring tiles are represented by combinatorial Delaunay triangulation (CDT) on landmarks, and a packet is routed via the shortest path from the source tile to the destination tile in CDT. In LBIB, the data are replicated and retrieved along the path toward a hashed tile using inter-tile routing of GLIDER. Within each tile, in addition to the path toward the hashed tile, the data are replicated and retrieved along two paths each toward a guide until one boundary node in the current tile is contacted, in which the guide is the landmark of the tile sharing the boundaries of the current and next tiles, if it exists. Otherwise, the guide is a boundary node shared by the current and next tiles. The data are forwarded using inter-tile and intra-tile routing of GLIDER if the guide is a landmark and a boundary node, respectively. In the scheme, if landmarks are not well chosen, many boundary nodes are shared by neighboring tiles. Therefore, some data are ultimately replicated (or retrieved) in the boundary nodes shared by the hashed tile and its previous tile selected by CDT of the shortest path from the producer (or consumer) tile to the hashed tile, resulting in unsuccessful data retrieval. Because no theoretical methods are capable of selecting a good set of landmarks, LBIB cannot guarantee successful data retrieval. In addition, similar to HD, LBIB requires each node to obtain considerably complex global topology information.

In Hop/SHU, the network boundary is first identified using the boundary detection algorithm [17], and then partitioned into four continuous pieces. The data are replicated along the paths toward the first and third pieces, and retrieved along the paths toward the second and fourth pieces. In this scheme, consumers may fail to retrieve the data of producers if the network boundary is not identified perfectly. However, existing boundary detection algorithms demand a great deal of message communication overhead and can work only in rather dense networks [17], [18], [19], making Hop/SHU hard to be used in wireless sensor networks in actual practice.

3 THE DRIB

The fundamental concept of DRIB is to construct a cycle of four axes as the virtual boundary, replicate and retrieve the data in the exterior of the virtual boundary along the path toward the node on the virtual boundary, and replicate and retrieve the data in the interior of the virtual boundary along the path using the double-ruling technique, as illustrated in Fig. 1. DRIB establishes a guide both to assist a node in the exterior of the virtual boundary in replicating and retrieving the data along the path toward the node on the virtual boundary, and to assist a node in the interior of the virtual boundary in replicating (or retrieving) the data along the paths toward axes $Z \leftrightarrow X$ and $Z' \leftrightarrow Y$ (or $X \leftrightarrow Z'$ and $Y \leftrightarrow Z$), respectively. In the establishment of a guide, each node, $u$, is assigned four entries: $u.ZX$, $u.XZ'$, $u.Z'Y$, and $u.YZ$, which denote the hop distances, using
3.1 The Guide

Definition 1. On a path \((u_1, u_2, \ldots, u_n)\), \(u_{i+1}\) (or \(u_{i-1}\)) for \(1 \leq i \leq n - 1\) is called the succeeding (or preceding) node of \(u_i\) (or \(u_{i+1}\)) for \(1 \leq i \leq n - 1\). A path is an outer-succeed-outer path if the succeeding node of an outer node is an outer node.

3.1 The Guide

The establishment of the replication and retrieval guide is to identify the nodes in the interior or exterior of the virtual boundary, and evaluate the lengths of the outer-succeed-outer paths from each of the four axes to each node. In the first phase, a virtual boundary is constructed, as described in subsection A.1. In the second phase, nodes are classified into four types: inner, outer, boundary, and near-boundary, as described in subsection A.2. Finally, each node evaluates the lengths of the outer-succeed-outer paths from each of the four axes in the third phase, as described in subsection A.3. A snapshot of the construction of the virtual boundary and the classification of nodes are illustrated in Fig. 2.

A.1 Construction of the Virtual Boundary

The virtual boundary consists of four anchors \(X, Y, Z,\) and \(Z'\), and four axes \(Z \leftrightarrow X, X \leftrightarrow Z', Z' \leftrightarrow Y,\) and \(Y \leftrightarrow Z\), which are the shortest paths from \(Z\) to \(X\), from \(X\) to \(Z'\), from \(Z'\) to \(Y\), and from \(Y\) to \(Z\), respectively; \(X\) (or \(Y\)) is the node with the maximum hop distance from \(W\) (or \(X\)), or, in the case of a tie, that has the maximum ID, among all nodes whose hop distances from \(X\) and \(Y\) differ by 1, where \(W\) denotes the preprogrammed node. Anchors are elected and axes are constructed in the same manner as those for anchors and axes in ABVCap_{Uni} [21] and ABVCap [22], described as follows.

Each node, \(u\), first evaluates the hop distance from \(W\), \(Hop(u, W)\), implemented as follows. \(W\) generates and broadcasts a \(W_{SET}\) message containing a hop counter that is initially set to 1 and advanced in increments by the forwarding nodes. Once a node, \(u\), receives a \(W_{SET}\) message that contains the smallest hop counter among all \(W_{SET}\) messages received, \(u\) broadcasts the \(W_{SET}\) message, and sets \(Hop(u, W)\) to the hop counter contained in the \(W_{SET}\) message.

Subsequently, anchor \(X\) is selected, and each node, \(u\), evaluates the hop distance from \(X\), \(Hop(u, X)\), implemented as follows. Each node, \(u\), having a greater hop distance from \(W\), or, in the case of a tie, that has a greater ID, than each neighbor generates and broadcasts an \(X_{SET}\) message containing \(u\)'s ID, \(Hop(u, W)\), and a hop counter that is initially set to 1 and advanced in increments by the forwarding nodes. Once a node, \(u\), receives an \(X_{SET}\) message generated by the node, \(v\), that has the maximum hop distance from \(W\), or, in the case of a tie, that has the maximum ID, among all \(X_{SET}\)
messages received, \( u \) then broadcasts the \( X \_SET \) message, sets \( \text{Hop}(u, X) \) to the hop counter contained in the \( X \_SET \) message, and identifies \( v \) as anchor \( X \), if the \( X \_SET \) message contains the smallest hop counter among all received \( X \_SET \) messages generated by \( v \). After anchor \( X \) is elected, anchors \( Y, Z, \) and \( Z' \) are each elected in a manner analogous to that for anchor \( X \) one by one.

Axis \( Z \leftrightarrow X \) is constructed, implemented as follows. \( Z \) generates a \( Z \leftrightarrow X \_SET \) message. Once a node, \( u \), generates or receives a \( Z \leftrightarrow X \_SET \) message, \( u \) forwards the \( Z \leftrightarrow X \_SET \) message to a neighbor, \( v \), with \( \text{Hop}(v, X) = \text{Hop}(u, X) - 1 \). Each node receiving the \( Z \leftrightarrow X \_SET \) message is located on axis \( Z \leftrightarrow X \). Axes \( X \leftrightarrow Z', Z' \leftrightarrow Y \), and \( Y \leftrightarrow Z \) are each established in a manner analogous to that for axis \( Z \leftrightarrow X \).

A.2 Classification of Nodes

Algorithm 1

1. For any node on an axis:
   a) Mark itself as a boundary node.
2. For any node that does not lie on an axis and has a neighbor on an axis:
   a) Mark itself as a near-boundary node.
3. For anchor \( Z' \):
   a) Generate an unmarked Inner_SET message and forward the unmarked message to a neighbor whose hop distance from \( Z \) is smaller by 1.
4. For any node receiving an unmarked Inner_SET message:
   a) Forward the unmarked message to a neighbor whose hop distance from \( Z \) is smaller by 1, if the node lies on an axis or has a neighbor on an axis.
   b) Change the unmarked message to a marked message, and broadcast the marked message, if the node does not lie on an axis and does not have a neighbor on an axis.
5. For any node receiving a marked Inner_SET message:
   a) Broadcast the marked message, if the node does not lie on an axis and does not have a neighbor on an axis.
6. For any node that does not lie on an axis and does not have a neighbor on an axis:
   a) Mark itself as an inner node, if it receives a marked/unmarked Inner_SET message.
   b) Mark itself as an outer node, if it does not receive a marked/unmarked Inner_SET message for \( t \) periods of time.

In DRIB, a node is a boundary node if the node lies on an axis; otherwise, if the node is a neighbor of a boundary node, it is a near-boundary node; otherwise, if the node is the selected node or there is a path, not passing through a boundary node nor a near-boundary node, from the selected node, it is an inner node; otherwise, it is an outer node. Nodes are marked as inner or outer nodes, implemented as follows. Anchor \( Z' \) generates an Inner_SET message, which is forwarded to a node whose hop distance from \( Z \) is smaller by 1 until anchor \( Z \) or a node that is not a boundary node nor a near-boundary node (the selected node) is reached. Once a node that is not a boundary node nor a near-boundary node receives the Inner_SET message, the node is marked as an inner node and broadcasts the message. Each node that is not a boundary node nor a near-boundary node is marked as an outer node if it does not receive the Inner_SET message. Algorithm 1 describes how to classify nodes in a distributed manner.

A.3 Establishment of the Guide

Algorithm 2

1. For any node, \( u \), on axis \( Z \leftrightarrow X \):
   a) Assign \( uZX \) to 0.
   b) Generate and broadcast a \( ZX \_SET \) message containing a hop counter set to 1.
2. For any node, \( v \), receiving a \( ZX \_SET \) message from a node, \( w \):
   a) Assign \( vZX \) to the hop counter contained in the message, and broadcast the message with the hop counter increased by 1, if the following conditions hold:
   1) \( w \) is not an outer node or \( v \) is an outer node, and
   2) the message contains the smallest hop counter among all \( ZX \_SET \) messages generated or received.

Each node, \( u \), evaluates the lengths of the outer-succeed-outer paths from axes \( Z \leftrightarrow X \), \( X \leftrightarrow Z' \), \( Z' \leftrightarrow Y \), and \( Y \leftrightarrow Z \), denoted by \( uZX, uXZ', uZ'Y \), and \( uYZ \), respectively, implemented as follows. Each boundary node, \( u \), on axis \( Z \leftrightarrow X \) sets \( uZX \) to 0, and generates and broadcasts a \( ZX \_SET \) message containing a hop counter initially set to 1. Once a node, \( v \), receives a \( ZX \_SET \) message from a node \( w \), \( v \) sets \( vZX \) to the hop counter contained in the message, increases the hop counter by 1, and broadcasts the updated message if \( w \) is not an outer node or \( v \) is an outer node. If \( v \) receives more than one \( ZX \_SET \) message, \( v \) broadcasts the message containing the smallest hop counter among all \( ZX \_SET \) messages received and sets \( vZX \) according to the message. In addition, \( uXZ', uZ'Y \), and \( uYZ \) are evaluated in a manner analogous to that for \( uZX \). Algorithm 2 describes how a node \( u \) evaluates \( uZX \) in a distributed manner.

3.2 The Scheme

DRIB is a double-ruling-based scheme. The data are replicated (or retrieved) along two paths toward axes \( Z \leftrightarrow X \) and \( Z' \leftrightarrow Y \) (or \( X \leftrightarrow Z' \) and \( Y \leftrightarrow Z \)), respectively, if the producer (or consumer) is a boundary, near-boundary, or inner node (a non-outer node); otherwise, the data are first replicated (or retrieved) along the path toward the closest axis until a non-outer node is reached, and then along two paths toward axes \( Z \leftrightarrow X \) and \( Z' \leftrightarrow Y \) (or \( X \leftrightarrow Z' \) and \( Y \leftrightarrow Z \)).


**Algorithm 3**

1. For any producer, $u$:
   a) $u$ generates ZX and Z’Y messages containing the data if $u$ is a non-outner node.
   b) $u$ generates an Axis\_APPROACH message containing the data if $u$ is an outer node.

2. For any non-outer node, $u$, receiving an Axis\_APPROACH message:
   a) $u$ generates ZX and Z’Y messages containing the data.

3. For any outer node, $u$, generating or receiving an Axis\_APPROACH message:
   a) $u$ forwards the message to a neighbor $v$ with $\min(v.ZX,v.XZ',v.Z'Y,v.YZ) < \min(u.ZX,u.XZ',u.Z'Y,u.YZ)$.

4. For any non-outer node, $u$, generating or receiving a ZX (or Z’Y) message:
   a) $u$ forwards the ZX (or Z’Y) message to a non-outer neighbor $v$ with $v.ZX = u.ZX - 1$ (or $v.Z'Y = u.Z'Y - 1$).

respectively, as implemented in the following. A producer, $u$, generates ZX and Z’Y messages containing the data and forwards the messages to two non-outer neighbors $v$ and $w$ with $v.ZX = u.ZX - 1$ and $w.Z'Y = u.Z'Y - 1$, respectively, if $u$ is a non-outer node; otherwise, $u$ generates an Axis\_APPROACH message containing the data and forwards the message to a neighbor $v$ with $\min(v.ZX,v.XZ',v.Z'Y,v.YZ) < \min(u.ZX,u.XZ',u.Z'Y,u.YZ)$, where $\min(a,b,c,d)$ equals the smallest value of $a$, $b$, $c$, and $d$. Once a node, $u$, receives an Axis\_APPROACH message, $u$ forwards the message to a neighbor $v$ with $\min(v.ZX,v.XZ',v.Z'Y,v.YZ) < \min(u.ZX,u.XZ',u.Z'Y,u.YZ)$ if $u$ is an outer node; otherwise, $u$ generates ZX and Z’Y messages containing the data and forwards the messages to two non-outer neighbors $v$ and $w$ with $v.ZX = u.ZX - 1$ and $w.Z'Y = u.Z'Y - 1$, respectively. In addition, if a node, $u$, receives a ZX (or Z’Y) message, $u$ forwards the message to a non-outer neighbor $v$ with $v.ZX = u.ZX - 1$ (or $v.Z'Y = u.Z'Y - 1$). Algorithm 3 describes how a node replicates the data using DRIB in a distributed manner. Data retrieval is implemented in a manner analogous to that for data replication.

### 3.3 An DRIB Example

An example of DRIB is presented in Fig. 3, where node 32 is the preprogrammed node $W$. In the construction of the guide, nodes 33, 39, 36, and 35 are selected as anchors $X$, $Y$, $Z$, and $Z'$, respectively, and the shortest paths from node 36 to node 33, from node 33 to node 35, from node 35 to node 39, and from node 39 to node 36 are established as the axes $Z \leftrightarrow X$, $X \leftrightarrow Z'$, $Z' \leftrightarrow Y$, and $Y \leftrightarrow Z$, respectively, in the first phase. In the second phase, node 18 is a boundary node because node 18 lies on an axis, and node 32 is a near-boundary node because node 32 is a neighbor of the boundary node 18. Node 35 generates an Inner\_SET message. The message is first forwarded to node 22 because node 22 has a smaller hop distance from $Z$ than node 35. Subsequently, the message is forwarded to node 17. Because node 17 is not a boundary node nor a near-boundary node, node 17 is marked as an inner node and broadcasts the Inner\_SET message. Node 3 is marked as an inner node after it receives the Inner\_SET message from node 17. Nodes 9, 38, and 30 are marked as outer nodes because each of them is not a boundary node nor a near-boundary node and does not receive the Inner\_SET message. In the third phase, because node 19 lies on axis $Z \leftrightarrow X$, $19.ZX = 0$. Node 19 generates and broadcasts a ZX\_SET message containing a hop counter equal to 1. After node 31 receives the ZX\_SET message from node 19, node 31 sets $31.ZX$ to the hop counter contained in the message, increases the hop counter by 1, and broadcasts the updated message. It is noted that $40.Z'Y = 5$ although node 40 has a neighbor, node 38, with $38.Z'Y = 3$, because nodes 38 and 40 are outer and non-outer nodes, respectively. Consider the ZX and Z’Y messages generated by node 2 (a producer). The ZX message containing the data of node 2 is first forwarded to non-outner node 17 because $17.ZX = 2.ZX - 1$. Subsequently, the ZX message is forwarded to node 31, and then forwarded to node 19. The Z’Y message is forwarded to non-outner node 12 because $12.Z'Y = 2.Z'Y - 1$. In addition, node 4 (a consumer) generates the ZX’ and YZ messages containing the type of the data-of-interest (the data of node 2). The ZX’ message is forwarded to node 10 along the path 4, 21, 27, 6, 10; the YZ message is forwarded to node 15. During the transmission of the ZX’ message from node 21 to node 27, node 31 receives the ZX’ message, and sends the data of node 2 to node 21. The
4 The RDRIB

In DRIB, the paths of data replication and retrieval may lie in the exterior of the virtual boundary, resulting in unsuccessful data retrieval, as illustrated in Fig. 4. The primary concept of RDRIB is to prohibit data from being replicated and retrieved along the paths in the exterior of the virtual boundary. The RDRIB guide, the RDRIB scheme, and an RDRIB example are given in Sections 4.1, 4.2, and 4.3, respectively.

4.1 The Guide

To prohibit the data from being replicated and retrieved along the paths in the exterior of the virtual boundary, two successive near-boundary nodes or two successive boundary nodes on different axes are not allowed on the paths of data replication and retrieval in RDRIB. Meanwhile, a near-boundary node is prohibited from being the preceding node of a non-outter node on the paths of data replication and retrieval in RDRIB, if it possesses at least two neighboring boundary nodes or all of its neighboring boundary node and the neighboring boundary nodes of its neighboring boundary node do not lie on the same axis. In addition, to complete the construction of the guide, the inner node is changed to an outer node if no outer-succeed-outer path exists from the axes to an inner node; subsequently, the node re-evaluates the lengths of the outer-succeed-outer paths from the axes.

The RDRIB guide is constructed in three phases: Construction of the Virtual Boundary, Classification of Nodes, and Establishment of the Guide, analogously to those for the DRIB guide. The following explanation is pertinent only to the construction of the RDRIB guide, which is different from that of the DRIB guide. In the phase of Classification of Nodes, after each node has been classified, a near-boundary node is re-marked as an outer node if the node has at least two neighboring boundary nodes or all of its neighboring boundary node and the neighboring boundary nodes of its neighboring boundary node do not lie on the same axis. In the phase of Establishment of the Guide, a near-boundary node ignores the ZX_SET message received from another near-boundary node, and a boundary node ignores the ZX_SET message received from another boundary node on a different axis. If an inner node, u, does not receive the ZX_SET message from an inner, near-boundary, or boundary node, u is re-marked as an outer node, and evaluates u.ZX according to the ZX_SET message received from an outer node. u.ZX’, u.ZY’, and u.YZ are evaluated in a manner analogous to that for u.ZX. The algorithm for classifying nodes in the phase of Classification of Nodes in RDRIB (Algorithm 4) can be obtained by adding Line 7 to Algorithm 1, described as follows.

7. For any node marked as a near-boundary node:
   a) Re-mark itself as an outer node, if the node has at least two neighboring boundary nodes or all of its neighboring boundary node and the neighboring boundary nodes of its neighboring boundary node do not lie on the same axis.

The algorithm for a node, u, to evaluate u.ZX in the phase of Establishment of the Guide in RDRIB (Algorithm 5) can be obtained by modifying Line 2 and adding Line 3 to Algorithm 2, described as follows.

2. For any node, v, receiving a ZX_SET message from a node, w:
   a) Assign v.ZX to the hop counter contained in the message, and broadcast the message with the hop counter increased by 1, if the following conditions hold:
      1) v or w is not a near-boundary node,
      2) v and w are not boundary nodes on different axes,
      3) w is not an outer node or v is an outer node, and
      4) the message contains the smallest hop counter among all ZX_SET messages generated or received.

3. For any node marked as an inner node:
   a) Re-mark itself as an outer node, if the node does not receive the ZX_SET message for t’ periods of time.

4.2 The Scheme

The data are replicated and retrieved in RDRIB in a manner analogous to that for data replication and retrieval in DRIB. The difference between DRIB and RDRIB is in the transmission of the ZX, ZX’, ZY, and YZ messages used to replicate or retrieve the data. In the following, we adhere to the transmission of the ZX message in RDRIB and omit the transmission of
boundary nodes of the neighboring boundary node, do not lie on the same axis. \( Z'Y = 5 \) although node 27 has a neighbor, node 31, with \( Z'Y = 3 \). The observation results from the fact that node 27 ignores the \( Z'Y \) message received from node 31 because nodes 27 and 31 are both near-boundary nodes. Nodes 10 and 29 are neighbors with \( 10.Z'Y = 3 \) and \( 29.Z'Y = 5 \), respectively. Nodes 10 and 29 are boundary nodes on different axes; therefore, node 29 ignores the \( Z'Y \) message received from node 10. Similarly to DRIB, the \( ZX \) message generated by node 2 (a producer) is forwarded to node 19 along the path 2, 17, 31, 19, and the \( ZY \) message generated by node 2 is forwarded to node 12. Consider the \( XZ' \) and \( YZ \) messages generated by node 34 (a consumer). The \( XZ' \) message is forwarded to node 5 along the path 34, 17, 3, 11, 5 and the \( YZ \) message is forwarded to node 15. After node 17 receives the \( XZ' \) message, it sends the data-of-interest (the data of node 2) to node 34. In addition, consider the \( XZ' \) and \( YZ \) messages generated by node 16 (a consumer). The \( YZ \) message is first forwarded to node 8 because node 16 is a near-boundary node and node 8 with \( 8.YZ = 16.YZ - 1 \) is neither an outer node nor a near-boundary node. Subsequently, node 8 forwards the \( YZ \) message to node 36 because node 8 is a boundary node and node 36 with \( 36.YZ = 8.YZ - 1 \) is neither an outer node nor a boundary node on a different axis. Similarly, the \( XZ' \) message is forwarded to node 33 along the path 16, 8, 19, 29, 33. After node 19 receives the \( XZ' \) message, it sends the data-of-interest (the data of node 2) to node 16 by backtracking the path traversed by the \( XZ' \) message. By contrast, the \( XZ' \) and \( YZ \) messages generated by node 16 may be forwarded along the path 16, 28, 33 and the path 16, 13, 36, respectively, in DRIB, in which case node 16 cannot retrieve the data of node 2.

5 Analysis and Discussion of RDRIB

In Section 5.1, we first show that each node can always forward the data using the RDRIB guide in Theorem 1. Subsequently, we show that RDRIB guarantees successful data retrieval if the virtual boundary is a simple cycle (a cycle with no repeated nodes or edges aside from the necessary repetition of the first and last node) without intersecting edges in Theorem 2. Finally, we show that RDRIB is distance-sensitive in a continuous domain with a convex shape if both the consumer and the producer are non-outside nodes in Theorem 3. In Section 5.2, successful data retrieval is discussed.

5.1 Analysis of RDRIB

In RDRIB, all sensors are assumed to be static and have the same transmission range. In the following proofs, without loss of generality, we assume that anchors \( Z, X, Z', \) and \( Y \) lie in the counterclockwise direction on the virtual boundary, and that the first node that is marked as an inner node lies in the interior of the virtual boundary. We also assume that the virtual boundary is
a simple cycle without intersecting edges. The following notations are necessary for the analysis of RDRIB.

**Definition 2.** A reduced-replication (or reduced-retrieval) path for a producer (or consumer), denoted by \( P_{rp} \) (or \( P_{rt} \)), is the path \((w'_1, \ldots, w'_2, w'_1, w_2, w_1, w_2, \ldots, w_n)\), where \( w_0 \) is the first non-outer node on the path of the data replication (or retrieval) for the producer (or consumer), \( w'_1 \) and \( w_n \) are nodes on axes \( Z \leftrightarrow X \) (or \( X \leftrightarrow Z' \)) and \( Z' \leftrightarrow Y \) (or \( Y \leftrightarrow Z \)), respectively, and \((w'_1, w'_2, \ldots, w'_m)\) and \((w_1, w_2, \ldots, w_m)\) are the parts of the path of the data replication (or retrieval) for the producer (or consumer) from \( w'_1 \) to \( w'_m \) and from \( w_1 \) to \( w_n \), respectively.

**Definition 3.** An edge is an inner edge if any point on the edge does not lie in the exterior of the virtual boundary. A path \((u_1, u_2, \ldots, u_n)\) is a non-outer-replication path (or non-outer-retrieval path) if \( u_1 \) and \( u_n \) are nodes on axes \( Z \leftrightarrow X \) (or \( X \leftrightarrow Z' \)) and \( Z' \leftrightarrow Y \) (or \( Y \leftrightarrow Z \)), respectively, and any edge on the path is an inner edge.

**Theorem 1.** In a connected network, RDRIB assigns each node a guide.

**Proof:** Let \( u \) be a node in the network. We need to show \( u \) evaluates \( u.ZX, u.XZ', u.Z'Y \), and \( u.YZ \). We adhere to the evaluation of \( u.ZX \) and omit that of the others due to their similarities. If \( u \) is a boundary node on axis \( Z \leftrightarrow X \), \( u.ZX = 0 \). Otherwise, it suffices to show there is an outer-succeed-outer path from a boundary node on axis \( Z \leftrightarrow X \) to \( u \). If \( u \) is a boundary node not on axis \( Z \leftrightarrow X \), there is an outer-succeed-outer path on the virtual boundary from a boundary node on axis \( Z \leftrightarrow X \). If \( u \) is a near-boundary node, \( u \) has a neighboring boundary node \( v \); therefore, the path on the virtual boundary from a boundary node on axis \( Z \leftrightarrow X \) to \( v \), followed by edge \((v, u)\), is the desired outer-succeed-outer path. If \( u \) is an inner node, \( u \) receives a \( ZX.Z \) message; otherwise, \( u \) is changed to an outer node. Clearly, the path traversed by the \( ZX.Z \) message received is the desired outer-succeed-outer path. If \( u \) is an outer node, let \( v \) be the non-outer node with minimum hop distance from \( u \). If \( v \) is a boundary node on axis \( Z \leftrightarrow X \), the shortest path from \( v \) to \( u \) is the desired outer-succeed-outer path. Otherwise, the outer-succeed-outer path from a boundary node on axis \( Z \leftrightarrow X \) to \( v \), followed by the shortest path from \( v \) to \( u \), is the desired outer-succeed-outer path. □

**Lemma 1.** If \( P_{rp} \) and \( P_{rt} \) intersects, consumer \( c \) can retrieve the data of producer \( p \).

**Proof:** This is clearly true if there is a node that lies on both \( P_{rp} \) and \( P_{rt} \). We only need to consider the case that edges \((u_1, u_2)\) and \((v_1, v_2)\) which lie on \( P_{rp} \) and \( P_{rt} \), respectively, intersects at a point. Consider the quadrangle \( u_1 v_1 v_2 v_3 u_1 \). Because there exists at least one interior angle with radian not smaller than \( \pi/2 \) in quadrangle \( u_1 v_1 v_2 v_3 \), at least one of \( |u_1, v_1|, |v_1, u_2|, |u_2, v_2|, \) and \( |v_2, u_1| \) is not greater than \( \pi \), where \( \pi \) denotes the transmission range of a node and \( |u, v| \) denotes the Euclidean distance between nodes \( u \) and \( v \). Therefore, at least one of \( v_1 \) and \( v_2 \) can always send back the data of \( p \) received from \( u_1 \) or \( u_2 \) to \( c \).

**Lemma 2.** For any non-exterior-replication path \( P'_{rp} \) and any non-exterior-retrieval path \( P'_{rt} \), \( P'_{rp} \) and \( P'_{rt} \) intersect.

**Proof:** Let \( P'_{rp} = (u_0, u_1, \ldots, u_n) \) and \( P'_{rt} = (v_0, v_1, \ldots, v_m) \), where \( u_0, u_n, v_0, v_m \) lie on axes \( Z \leftrightarrow X, Z' \leftrightarrow Y, X \leftrightarrow Z' \), and \( Y \leftrightarrow Z \), respectively. Because the virtual boundary is a simple cycle without intersecting edges, \( u_0, v_0, u_n, v_m \) lie in order on the virtual boundary. Because any point on the edge on \( P'_{rp} \) or \( P'_{rt} \) does not lie in the exterior of the virtual boundary, \( P'_{rp} \) and \( P'_{rt} \) intersect. □

**Lemma 3.** For any reduced-replication path (or reduced-retrieval path), there exists a non-exterior-replication path (or non-exterior-retrieval path) which consists of the parts of the reduced-replication path (or reduced-retrieval path).

**Proof:** By symmetry, we only need to show this lemma holds for any reduced-replication path \( P_{rp} = (u_0, u_1, \ldots, u_n) \). We first show two claims: C1) if \((v_1, v_3) \leq R \) and \((v_2, v_4) \leq R \) in a convex quadrangle \( v_1 v_2 v_3 v_4 \), then \(|v_i, v_{i+1}| \leq R \) and \(|v_i, v_{i+1}| \leq R \) for some \( 1 \leq i \leq 4 \), C2) if \((u_i, u_j)\) intersects with an edge \((a_j, a_{j+1})\) on an axis, then one of \( u_{i-1} \) and \( u_i \) is a near-boundary node, and the other is \( a_j \) or \( a_{j+1} \). C1 is clearly true because there exists at least one interior angle with radian not smaller than \( \pi/2 \) in quadrangle \( v_1 v_2 v_3 v_4 \). For C2, by C1, at least one of \( u_{i-1} \) and \( u_i \) is a neighbor of a boundary node \( a_j \) or \( a_{j+1} \). Because an inner node cannot be a neighbor of a boundary node, at least one of \( u_{i-1} \) and \( u_i \) is not an inner node. In addition, at least one of \( u_{i-1} \) and \( u_i \) is not a boundary node; otherwise, two intersecting edges \((u_{i-1}, u_i)\) and \((a_j, a_{j+1})\) both lie on the virtual boundary. This implies that at least one of \( u_{i-1} \) and \( u_i \) is a near-boundary node. Let, without loss of generality, \( u_{i-1} \) be a near-boundary node. Because a near-boundary node has only one neighboring boundary node in RDRIB, at least one of \( a_j \) and \( a_{j+1} \) is a neighbor of \( u_i \), implying that \( u_i \) is not an inner node. Therefore, \( u_i \) is a boundary node. Because \( u_i \) is a neighboring boundary node of \( u_{i-1} \), boundary nodes \( a_j \) and \( a_{j+1} \) cannot be neighbors of \( u_{i-1} \). This implies that \( a_j \) and \( a_{j+1} \) are neighbors of \( u_i \) by C1. In RDRIB, for a near-boundary node, all of its neighboring boundary node and the neighboring boundary nodes of its neighboring boundary node lie on the same axis. Therefore, \( a_j, a_{j+1}, \) and \( u_i \) lie on the same axis. This implies \( u_i = a_j \) or \( u_i = a_{j+1} \) because an axis is the shortest path between two anchors.

Subsequently, we prove that there exists a non-exterior-replication path which consists of the parts of \( P_{rp} \). It suffices to show C3) a path passing through only near-boundary nodes in the interior of the virtual boundary, inner nodes, and boundary nodes, denoted by
is a circle, the path of the data retrieval for a consumer is specifically, if the convex shape of the continuous domain does not lie in the exterior of the virtual boundary. By C2, it suffices to show all inner nodes on \( P_i \) lie in the interior of the virtual boundary. For any inner node, there is a path passing through only inner nodes, denoted by \( P_o \), from the first node that is marked as an inner node. We need to show any edge on \( P_2 \) does not intersect with an edge on the virtual boundary. Suppose that there is an edge \((v_{k-1}, v_k)\) on \( P_2 \) intersects with an edge on the virtual boundary. By C2, one of \( v_{k-1} \) and \( v_k \) are inner nodes. For C4, if \( P_{rp} \) does not pass through any near-boundary node in the exterior of the virtual boundary, \( P_{rp} \) is the desired path. Let \( u_{i-1} \) be a near-boundary node on \( P_{rp} \) that lies in the exterior of the virtual boundary. It suffices to show \( u_{i-1} \) is the first non-outer node on the path of the data replication for the producer and \( u_{i-2} = u_i \), in which case the path, \((u_0, \cdots, u_{i-3}, u_{i-2} = u_i, u_{i+1}, \cdots, u_n)\), consisting of the part of \( P_{rp} \), \((u_0, \cdots, u_{i-3}, u_{i-2})\), and the part of \( P_{rp} \), \((u_i, u_{i+1}, \cdots, u_n)\), is the desired path. Because two successive near-boundary nodes are not allowed in \( P_{rp} \), \( u_i \) is a boundary node or an inner node. This implies that edge \((u_{i-1}, u_i)\) must intersect with an edge on the virtual boundary because \( u_{i-1} \) and \( u_i \) lie in the exterior and interior of the virtual boundary, respectively. By C2, \( u_i \) is a boundary node. Similarly, \( u_{i-2} \) is a boundary node. Because \( u_{i-1} \) can have only one neighboring boundary node, \( u_{i-2} = u_i \). This implies that \( u_{i-1} \) is the first non-outer node on the path of the data replication for the producer.

**Theorem 2.** If the virtual boundary is a simple cycle without intersecting edges, then a consumer, \( c \), can always retrieve the data of a producer, \( p \).

**Proof:** Let \( P_{rp} \) and \( P_{rt} \) denote the path of the data replication for a producer and the path of the data retrieval for a consumer, respectively. According to Lemma 3, there exist a non-exterior-replication path, \( P'_{rp} \), consisting of the parts of \( P_{rp} \) and a non-exterior-retrieval path, \( P'_{rt} \), consisting of the parts of \( P_{rt} \). According to Lemma 2, \( P_{rp} \) and \( P_{rt} \) intersects. We, therefore, conclude that consumer \( c \) can retrieve the data of producer \( p \) according to Lemma 1.

**Theorem 3.** For a continuous domain with a convex shape, the path of the data retrieval for a consumer is proportional to the distance between the producer and the consumer if both the consumer and the producer lie in the interior of the virtual boundary that is a simple cycle without intersecting edges. Specifically, if the convex shape of the continuous domain is a circle, the path of the data retrieval for a consumer is not greater than the distance between the producer and the consumer in the interior of the virtual boundary that is a simple cycle without intersecting edges.

**Proof:** In a continuous domain, we use the Euclidean distance between two nodes as the hop distance between two nodes in RDRIB (as used in [20]). Because the continuous domain has a convex shape, the shortest path between two nodes \( u \) and \( v \) is the line segment between \( u \) and \( v \), denoted by \( uv \). Thus, the virtual boundary constructed by RDRIB is a quadrangle \( ZXZ'Y \). In addition, because anchors \( X, Y, Z, \) and \( Z' \) established by RDRIB must lie in the boundary of the continuous domain, the radians of the interior angles \( \angle Z, \angle X, \angle Z', \) and \( \angle Y \) in quadrangle \( ZXZ'Y \) are each not greater than \( \pi \).

Let \( \overline{cc} \) and \( \overline{cp} \) be the shortest path from the consumer, \( c \), to axes \( X \leftrightarrow Z \) and \( Y \leftrightarrow Z \), respectively, and let \( \overline{pp} \) and \( \overline{pp'} \) be the shortest path from the producer, \( p \), to axes \( Z \leftrightarrow X \) and \( Z' \leftrightarrow Y \), respectively. Because at least one of \( \overline{cc} \) and \( \overline{cp} \) intersects with at least one of \( \overline{pp} \) and \( \overline{pp'} \), we assume, without loss of generality, that \( \overline{cc} \) intersects with \( \overline{pp} \) at a node, \( i \). It suffices to show \( \frac{\overline{pp}}{\theta} \leq \frac{1}{\sin(\pi - \theta)} \) for C1) \( \overline{cc} > \overline{cp} \) and C2) \( \overline{cc} \leq \overline{cp} \), where \( \theta \) denotes the radian of the maximum interior angle of quadrangle \( ZXZ'Y \). For C1, \( \angle lip < \frac{\pi}{2} \). Let \( \overline{cj} \) be the shortest path from \( c \) to the line containing \( p \) and \( i \). Clearly, \( \frac{\overline{cp}}{\theta} \leq \frac{\overline{cj}}{\theta} = \sqrt{\overline{cj}} \sin(\pi - \theta) \). Because \( \angle lip = \pi - \angle X \geq \pi - \theta \), \( \frac{\overline{cp}}{\theta} \leq \frac{1}{\sin(\pi - \theta)} \).

Consider a continuous domain with a circle shape. The distance between anchors \( X \) and \( Y \) equals the diameter of the circle because in RDRIB, \( Y \) is the node with the maximum distance from \( X \). Also, the distance between anchors \( Z \) and \( Z' \) equals the diameter of the circle. This implies that quadrangle \( ZXZ'Y \) is a rectangle, and thus, \( \theta = \frac{\pi}{2} \). Therefore, \( \frac{\overline{pp}}{\theta} \leq \frac{1}{\sin(\pi - \theta)} = 1 \).

**5.2 Extension of RDRIB**

If the virtual boundary is not a simple cycle without intersecting edges, \( P_{rp} \) and \( P_{rt} \) may not intersect, resulting in unsuccessful data retrieval, as illustrated in Fig. 6. To avoid this situation, an extended version of RDRIB, denoted by RDRIB+, is discussed here. In RDRIB+, if the first non-outer node, \( u \), on the path of the data retrieval for a consumer fails to retrieve the data-of-interest of the consumer using RDRIB, \( u \) generates a \( ZX \) RETRIEVE message and a \( Z'Y \) RETRIEVE message if \( u.ZX \leq u.Z'Y \) and \( u.ZX > u.Z'Y \), respectively. The ZX RETRIEVE message and the ZY RETRIEVE message are forwarded.
toward axes $Z \leftrightarrow X$ and $Z' \leftrightarrow Y$, respectively, in a manner analogous to that for the ZX message. Once the ZX.RETRIEVE (or $Z'Y$.RETRIEVE) message reaches a node on axis $Z \leftrightarrow X$ (or $Z' \leftrightarrow Y$), two copies of the message are generated and forwarded by nodes on axis $Z \leftrightarrow X$ (or $Z' \leftrightarrow Y$) toward anchors $Z$ and $X$ (or anchors $Z'$ and $Y'$), respectively.

**Theorem 4.** In RDRIB+, the extension version of RDRIB, a consumer, $c$, can always retrieve the data of a producer, $p$.

**Proof:** Let $u$ be the first non-outer node on the path of the data retrieval for consumer $c$. By symmetry, we assume $u.ZX \leq u.Z'Y$. Clearly, there is at least one node, $v$, on axis $Z \leftrightarrow X$ on the path of the data replication for producer $p$. It is easy to verify that $v$ must receive the ZX.RETRIEVE message generated by $u$. This implies that the data-of-interest of consumer $c$ must be forwarded from $v$ to $u$ by backtracking the path traversed by the ZX.RETRIEVE message generated by $u$, and then be forwarded from $u$ to consumer $c$ by backtracking the path of the data retrieval for consumer $c$. □

### 6 Performance Evaluation

Simulations using a packet-level simulator and the network simulator NS-2 (version 2.34) were used to evaluate the performance of the proposed scheme. In our simulations, 100 connected networks were generated by randomly deploying nodes in square regions. The simulation settings are summarized in Table 1; the network density denotes the average number of neighbors per node; the transmission range of nodes is a circle of radius 1 in the unit disk graph (UDG) communication model; two nodes are linked if their distance is not greater than 0.5, are not linked if their distance is greater than 1, and are linked with a probability ranging from 0 to 1 otherwise in the quasi unit disk graph (QUDG) communication model. RDRIB (the non-extended version) was compared with HD [14], LBIB [15], Hop/SHU [16], and GHT [8] in terms of the retrieval rate, the replication memory overhead, the retrieval message overhead, the retrieval latency, and the construction message overhead. GHT, a GPS-assisted information brokerage scheme, was compared in our simulation because it is the foremost and best-known one based on a distributed hash table. In GHT, a geographic hash table is used to hash a data type to a geographic location. The data are replicated and retrieved in the node closest to the hashed geographic location. The retrieval latency denotes the average ratio of the successful to total data retrieval for consumers. The replication memory overhead denotes the average number of nodes that store the data during the data replication for a producer. The replication (or retrieval) message overhead denotes the average total number of hops traversed by the messages transmitted during the data replication (or retrieval) for a producer (or consumer). The retrieval latency denotes the average minimum number of hops (or the average minimum time) to retrieve the data-of-interest for a consumer in the simulation using the packet-level simulator (or NS-2). The construction message overhead denotes the average total number of hops traversed by the messages transmitted in the construction of a information brokerage scheme. In the implementation of each scheme, a node stops forwarding the message for data retrieval if at least one neighbor of the node has the data-of-interest. In the implementation of GHT, the Gabriel graph [23] planarization algorithm and the CLDP [24] planarization algorithm are employed to construct the faces in the UDG and QUDG models, respectively. In the implementation of HD, LBIB, or Hop/SHU, the settings of parameters are the same as that in the original paper. Empirical data were obtained by averaging data of 200 producer-consumer pairs randomly selected from 100 networks.

Table 2 summarizes NS-2 settings, where the maximum delay for forwarding (or broadcasting) a message equals 0.005 (or 0.25). Because the packet transmission time for 32-byte data in IEEE 802.11 equals 0.0015 [25], the maximum time for forwarding (or broadcasting) a message per hop equals 0.0065 (or 0.2515). Consider RDRIB implemented in NS-2. Because an inner node should wait for a ZX.SET message broadcast from a node on axis $Z \leftrightarrow X$ before remarking itself as an outer node, the waiting time for marking an inner node as an outer node ($t'$) in Algorithm 5 is set to $0.2515 \times 2 \times \text{Hop}(W, X)$, where $2 \times \text{Hop}(W, X)$ is the upper bound on the network diameter (the maximum hop distance between any pair of nodes in the network). Note that in RDRIB, each node receives $\text{Hop}(W, X)$.
which is the hop distance between the preprogrammed node \( W \) and anchor \( X \), in the phase of Construction of the Virtual Boundary. In addition, the waiting time for marking a node as an outer node \((t)\) in Algorithm 4 is set to \((0.0065 + 0.2515) \times 2 \times Hop(W, X)\) because before a node marks itself as an outer node, the node should first wait for the unmarked Inner_SET message forwarded from anchor \( Z' \) to the first inner node marked (which may take up to \(0.0065 \times 2 \times Hop(W, X)\)), and then wait for a marked Inner_SET message broadcast from the first marked inner node to itself (which may take up to \(0.2515 \times 2 \times Hop(W, X)\)). Sections 6.1–6.6 describe the first simulation results illustrated in Fig. 7, and Sections 6.7–6.9 describe the second simulation results illustrated in Fig. 8.

6.1 Retrieval Rate
As expected, RDRIB and HD each guarantee successful data retrieval in all cases. In Hop/SHU, the retrieval rate is nearly 100% in a network with a high density. As the network density decreases from 31 to 16, the retrieval rate decreases because it is more difficult to obtain a perfect network boundary in a network with a lower density. In a network with a density equal to 16, a long path is often identified as the network boundary. Thus, the paths of the data retrieval and the data replication intersect with difficulty, resulting in a considerably low retrieval rate. By contrast, a node or a short path is identified as the network boundary in a network with a density equal to 6 or 11, in which case the data are replicated and retrieved toward a node or a short path, resulting in a high retrieval rate. LBIB cannot guarantee successful data retrieval, which is reasonable because a good set of landmarks is difficult to obtain. The higher the network density, the lower the retrieval rate because more boundary nodes are shared by neighboring tiles in a network with a higher density.

6.2 Replication Memory Overhead
The difference between RDRIB, LBIB, and Hop/SHU is negligible. In Hop/SHU, the replication memory overhead in a network with a density equal to 6, 11, or 16 is astonishingly low, which occurs because two replication messages toward the first and third pieces of the network boundary encounter many nodes in common due to the fact that a node, a short path, or a long path is identified as the network boundary. In HD, the data are replicated in the hashed nodes in all neighboring clusters of the producer at all levels. Because a node has a total of approximately 110–220 neighboring clusters at all levels in networks with densities ranging from 6 to 31, HD has a greater replication memory overhead, compared with RDRIB, LBIB, and Hop/SHU. In addition, the higher the network density, the smaller the replication memory overhead in RDRIB, LBIB, and Hop/SHU. This is because the progress distance between two nodes is greater in a network with a higher density. In HD, as the network density increases, the number of neighboring clusters of a node dramatically increases, leading to a greater replication memory overhead in a network with a higher density.

6.3 Replication Message Overhead
For each scheme, the replication message overhead is greater than the replication memory overhead. This re-
result occurs because a node that stores the data of a producer could exist on multiple replication paths for the producer, in which case the node transmits multiple replication messages generated by the producer. The difference between RDRIB, LBIB, and Hop/SHU is negligible except in a network with a density equal to 6. In a network with a density equal to 6, a node is identified as the network boundary in Hop/SHU, in which case all data of producers are replicated toward the node identified as the network boundary, resulting in a greater replication message overhead than for RDRIB and LBIB. The replication message overhead of HD is approximately 40 times of that of RDRIB, LBIB, or Hop/SHU because in HD a node has a total of approximately 110–220 neighboring clusters at all levels in networks with densities ranging from 6 to 31. As the network density increases from 6 to 16, the replication message overhead of HD decreases due to a dramatic increase in the progress distance between two nodes. As the network density increases from 16 to 31, the progress distance between two nodes slightly increases while the number of neighboring clusters of a node dramatically increases, leading to an increase in the replication message overhead. In addition, as the network density increases, the number of neighboring clusters of a node at low levels increases far more quickly than that at high levels. Therefore, the replication message overhead of HD increases far more slowly than its replication memory overhead increases as the network density increases from 16 to 31.

6.4 Retrieval Message Overhead

In RDRIB, LBIB, and Hop/SHU, the results on the retrieval message overhead and the replication message overhead are similar because the data are retrieved in a manner analogous to that for data replication. In each of RDRIB, LBIB, and Hop/SHU, a node stops forwarding the message for data retrieval if the node has one neighbor with the data-of-interest, resulting in a smaller retrieval message overhead than the replication message overhead. HD has the smallest retrieval message overhead. This observation results from the fact that the number of nodes that store the data-of-interest in HD (the replication memory overhead) is greater than RDRIB, LBIB, and Hop/SHU. As the network density increases, the retrieval message overhead of HD decreases due to an increase in the number of nodes that store the data-of-interest, and in addition, the retrieval message overhead of HD decreases far more slowly than the number of nodes that store the data-of-interest in HD increases due to a reason analogous to that for the replication message overhead.

6.5 Retrieval Latency

In HD and LBIB, only one message for data retrieval is forwarded; therefore, the retrieval latency equals the retrieval message overhead. By contrast, in RDRIB and Hop/SHU, two messages for data retrieval are forwarded concurrently, resulting in a smaller retrieval latency than the retrieval message overhead.

6.6 Construction Message Overhead

Compared with RDRIB, HD, and LBIB, Hop/SHU has the greatest construction message overhead because a large amount of messages are required to identify the network boundary. In HD, each node must identify the neighboring clusters at level $k$ by broadcasting a message within $\alpha \cdot 2^{k+1}$ hops, resulting in a greater construction message overhead than for RDRIB and LBIB. As the network density increases, the construction message overhead in HD and Hop/SHU increases due to increases in the number of neighboring clusters and the number of nodes used to identify the network boundary, respectively. In addition, the difference between RDRIB and LBIB is negligible.

6.7 Simulation Results of GHT in the UDG Model

As anticipated, GHT guarantees successful data retrieval in the packet-level simulation (PLS). The replication message overhead of GHT is equal to 2 because the data of a producer are stored only in the producer and the node closest to the hashed geographic location. In a network with a lower density, GHT has a greater replication message overhead, a greater retrieval message overhead, and a greater retrieval latency. These increases occur because the message for data replication/retrieval is forwarded along a face boundary if the message encounters a node with no neighbor closer to the destination (a concave node), and the message encounters a concave node with a higher probability in a network with a lower density. In our simulation, the message containing the neighbor information periodically exchanged between neighbors is not counted in the construction message overhead, and thus, the construction message overhead of GHT is equal to 0.

6.8 Simulation Results in the QUDG Model

The retrieval rate of RDRIB is almost 100%. In the QUDG model, RDRIB cannot guarantee successful data retrieval because the consumer cannot retrieve the data of the producer even if the path of the data replication intersect. (Lemma 1 does not hold in the QUDG model.) Except for the retrieval rate, the simulation results of RDRIB in the QUDG model are analogous to that of RDRIB in the UDG model. GHT has the same performance in both the UDG model and the QUDG model, in terms of the retrieval rate and the replication memory overhead. In the QUDG model, GHT incurs a great deal of message overhead to construct faces using CLDP, and thus, GHT has a far greater construction message overhead compared with RDRIB. As the network density increases from 11 to 31, the replication message overhead, the retrieval message...
overhead, the retrieval latency, and the construction message overhead all dramatically increase because the average sizes of the face boundaries constructed by CLDP increase sharply. Note that GHT has a great replication message overhead, a great retrieval message overhead, and a great retrieval latency in the network with density 6, which results from the observation that the message for data replication/retrieval traverses a face boundary with a higher probability in a network with a lower density.

6.9 Simulation Results in NS-2

The retrieval latency is characterized by hops (the left-side scale) in the packet-level simulation, and by seconds (the right-side scale) in the NS-2 simulation, as illustrated in Fig. 8(e); the left-side scale for 1 hop corresponds to the right-side scale for 0.004 seconds, which is the average time for forwarding a message per hop equal to the average delay for forwarding a message (0.0025 seconds) plus the packet transmission time for 32-byte data in IEEE 802.11 (0.0015 seconds). Except for the retrieval rate, the simulation results of RDRIB (or GHT) in NS-2 are analogous to that of RDRIB (or GHT) in the packet-level simulation. As anticipated, neither RDRIB nor GHT guarantees successful data retrieval due to the packet loss resulting from the packet collisions. The retrieval rate of RDRIB fluctuates as the network density increases. By contrast, as the network density decreases from 31 to 6, the retrieval rate of GHT decreases noticeably. This result occurs because in a network with a lower density, GHT has a greater number of packet collisions due to a longer path of the message for data replication/retrieval. In a network with a density equal to 6, compared with GHT in the UDG model on the packet-level simulator, GHT in NS-2 has a smaller replication memory/message overhead due to the loss of the messages for data replication, and conversely, has a greater retrieval message overhead because the message for data retrieval traverses the whole face boundary surrounding the hashed geographic location with a greater probability. In addition, with GHT in NS-2, the retrieval message overhead is greater than the number of hops corresponding to the retrieval latency, which occurs because the hops traversed by the message for data retrieval are not counted in the retrieval latency if the message traverses the whole face boundary surrounding the hashed geographic location such that the consumer fails to retrieve the data-of-interest in the NS-2 simulation.

7 Conclusion

In this paper, we propose a reliable double-ruling-based information brokerage scheme, RDRIB, to replicate and retrieve the data, in which each node is not required to possess the geographic location information and no boundary detection algorithms are used. In RDRIB, the virtual boundary consisting of a cycle of four axes is constructed. The data are first replicated (or retrieved) toward the interior of the virtual boundary until a node in the interior of the virtual boundary is reached; subsequently, the data are replicated (or retrieved) along two paths in the interior of the virtual boundary toward the first and third axes (or the second and fourth axes). As a result, RDRIB guarantees successful data retrieval,
requires a small number of nodes to store the data, and demands a small message overhead to replicate and retrieve the data. In addition, the message for data replication (or retrieval) is required to carry only the data (or data-of-interest). Thus, RDRIB requires a small-sized message for data replication (or retrieval).

We first evaluated, using a packet-level simulator, the retrieval rate, the replication memory overhead, the replication message overhead, the retrieval message overhead, the retrieval latency, and the construction message overhead of RDRIB, HD [14], LBIB [15], and Hop/SHU [16] in networks with densities ranging from 6 to 31 in square regions with side lengths equal to 100 times that of the transmission range of a node in the UDG model. The simulation shows that RDRIB and HD each guarantee successful data retrieval, while LBIB and Hop/SHU do not. This result occurs because LBIB and Hop/SHU need to select a good set of landmarks and identify a perfect network boundary, respectively, that are difficult to obtain in the actual practice. Because HD stores the data in a large number of nodes, it has the best performance in terms of the retrieval latency while it has the worst performance in terms of the replication memory overhead and the replication message overhead. Hop/SHU has the worst performance in terms of the construction message overhead because it requires a large amount of messages to identify the network boundary. LBIB has the worst performance in terms of the retrieval latency because it forwards only one message for data retrieval. RDRIB has the best performance in terms of the replication message overhead and the retrieval latency, is equivalent to LBIB and Hop/SHU for the best performance in terms of the replication memory overhead, and is equivalent to LBIB for the best performance in terms of the construction message overhead. In terms of the retrieval message overhead, the performance of RDRIB is less satisfactory than that of HD, and better than that of LBIB and Hop/SHU. In addition, we evaluated the performance of RDRIB and GHT [8] in the UDG and QUDG models using a packet-level simulator, and in the UDG model using the network simulator NS-2. In the QUDG model, the retrieval rate of RDRIB is almost 100%, and compared with GHT, RDRIB has a better performance in terms of the replication message overhead, the retrieval message overhead, the retrieval latency, and the construction message overhead. In NS-2, neither RDRIB nor GHT guarantees successful data retrieval due to the packet collisions. Furthermore, RDRIB has greater and comparable retrieval rates in networks with low and high densities, respectively, compared with GHT.

In RDRIB, all sensors are assumed to be static and have the same transmission range. It will be an interesting subject for future research to modify RDRIB for sensor networks in which sensors are mobile or have different transmission capability. Another future research includes the study of extending RDRIB to a wireless sensor network in three-dimensional space.

REFERENCES


