

Multiple Distributed Indexing Scheme for Supporting Energy-efficient Range Query in Data-Centric Storage Sensor Networks

XuanTung Hoang, Younghee Lee

Information and Communications University

119 Munji-ro, Yuseong-gu, Daejeon, Republic of Korea, 305-714

{tung_hx, yhlee}@icu.ac.kr

Abstract: We envision that a sensor network would be deployed for multiple applications with different preference of range query processing. In that situation, multi-dimensional range query capability is useful since it can flexibly support various applications. We propose, in this paper, a data indexing scheme that improves energy-efficiency of such range query for Data-Centric Storage sensor networks. Our indexing scheme maintains multiple locality preserving sub-indices. Each sub-index is favorable, in terms of locality preserving, for a specific attribute. Querying is conducted on all those sub-indices to find the most efficient copies of data. Our simulation shows that our indexing scheme can reduce up to 50% of query cost.

Introduction

In sensor network applications, such as environment sensing, and habitat monitoring, data plays a more important role than identifiers of sensor nodes that process the data. For example, a data collector in an environment sensing application wants to know environmental measurements (temperature, humidity, or pressure) at a specific location more than the sensor that provides those measurements. That observation has motivated a route-to-data communication abstraction and Data-Centric Storage (DCS) architecture of sensor networks [2]. The data-centric communication abstraction of DCS sensor networks is characterized by the capability of routing messages to data-dependent destinations in the network. A sensor node detecting an event uses the data-centric communications to insert the event to the network under a name (or key). And nodes that are interested in the event will also use the data-centric communications to retrieve the event by sending a query under the corresponding name (key). The data-centric communications can be implemented efficiently with geographical routing such as GPSR [3]. Examples of DCS proposal for sensor networks are GHT [1, 2], DIM [4], DIFS [6] and DIMENSIONS [7].

Supporting multi-dimensional range query DCS sensor network currently is a challenging problem. As can be seen in database research, the common approach for solving multi-dimensional range query is considering data

space as a multi-dimension space. Each data object (or detected event) corresponds to a point in the multi-dimensional space. Coordinates of a point are numeric representations of event attributes (such as temperature, light intensity, pressure, or geographical coordinates). DIFS [6] and DIM [4] are first proposals for resolving range queries in sensor networks. However, to our best knowledge, the problem is still incompletely solved for sensor networks. DIFS [6] allows only 1-dimensional range queries since it enables range queries on a single key. And DIM [4] allows multi-dimensional range queries but suffers from inefficient query resolution when a query is split into multiple sub-queries. Those sub-queries are possibly scattered on the network area wasting message transmissions and node energy.

For handling range queries more efficiently, we propose a novel indexing scheme, called Multiple Distributed Indexing or MDI. Our proposed indexing scheme maintains several distributed sub-indices. Those indices complement each other to enhance the locality-preserving property, thus, mitigate negative effects of query splitting in DIM. For ease of exposition, we summarize the indexing scheme of DIM in section 2 before presenting the query splitting problem in more details (section 3) and our MDI scheme (section 4).

Preliminaries: DIM Indexing Scheme

Distributed Indexing for Multidimensional data (DIM) [3] is designed for supporting multi-dimensional range queries in DCS sensor networks. DIM protocol works on top of the well-known geographic routing GPSR. In DIM, sensor nodes are assumed to know their physical positions. That assumption is possible with GPS receivers equipped for all sensor nodes. Also, the sensor field where the sensor network is deployed is assumed to be a rectangle whose sizes are known to sensor nodes.

DIM defines a zone structure as follows: the sensor field is vertically divided into two equal level-1 zones: left and right. The left and right zones are respectively given zone codes 0 and 1. Each level-1 zone is then horizontally divided into upper and lower level-2 zones. Each level-2 zone is further divided vertically to form level-3 zones.

The division procedure is repeated until every node lies in a distinct zone. Zone code of a level- k zone is given by adding “0” to its “parent zone” code if it is the left or lower part of the parent zone. Otherwise, its zone code is obtained by adding “1” to its parent zone code. That tessellation can be performed distributedly as follows:

In the beginning, zone code 0 is assigned to all nodes. Each node includes its position and its zone code into beacon messages and broadcasts to neighbors. When node i receives such a beacon message from a neighbor j , if i and j have identical zone code, i updates its zone code according to the position of i and j . Specifically, i determines a zone border that separates i and j and takes the corresponding zone code it belongs to. After several rounds of exchanging beacons and refining nodes’ zone codes, each node can know the zone it owns.

Events detected by sensors in DIM are recognized as points in a multi-dimensional space whose dimensions are events’ attributes. That event space is also divided and coded in the same way as forming zones. Thus, a zone code also corresponds to a portion of the space. Every event is mapped to a smallest zone that corresponding to the data portion containing it, and is stored at a sensor node that owns the zone. For event insertion and query, DIM specifies a GPSR-based routing mechanism that gradually refines destination zone of events and query.

Query Splitting Problem

The rule that assign code to zone in DIM can be seen as mapping zones to a z-order Space Filling Curve [5] (SFC). Zone codes are z-values of indexed zones. Thus, DIM can be viewed as a mapping from high-dimensional space to 2-dimensional space with 2 z-order SFCs: one is used to index geographical area and the other is for indexing high-dimensional data space. The mapping is completed by matching z -values of the two z-order SFCs. Figure 1 shows an example of a sensor field and its zone structure. Zones are indexed in the order indicated by arrows.

The query splitting problem can be seen through an example. Assume that the zone structure of DIM is used to index events with 4 attributes X^1, X^2, X^3, X^4 (4-D space). Consider two hyper-cubes on the 4-D space of events: (C_1) whose coordinates all start with bit 0; and (C_2) whose coordinate X^1 starts with bit 1 while the other coordinates start with bit 0. If the order of partitioning the 4D space is X^1, X^2, X^3, X^4 , the two hyper-cubes C_1 and C_2 are mapped to zone 0000 and 1000 respectively. As can be seen in figure 1, the C_1 and C_2 are mapped to two separated zones even though they are adjacent in 4-D event space.

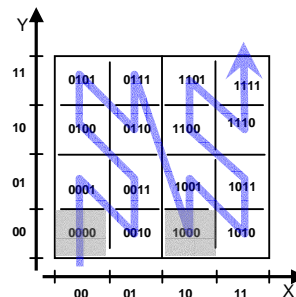


Figure 1: Example of zone structure of a sensor field

Multiple Distributed Indexing

Targeting the query splitting problem, we propose a multiple indexing scheme, called Multi Distributed Indexing (MDI), which enhances locality preserving characteristic.

Index Structure

Our proposed indexing scheme comprises of multiple sub-indices. Those sub-indices complement each other and form a combined index structure with better locality preserving property. Particularly, MDI indexes a k -dimensional data space $\lceil k/2 \rceil$ with $\lceil k/2 \rceil$ different z-order SFCs. Each z-order SFC uses a particular order of coordinates. Specifically, the order of coordinates of a SFC is obtained by shifting and circulating that of the previous SFC by two positions. For example, if the first SFC indexes data points using the order (X^1, X^2, \dots, X^k) , the coordinate order for the second SFC is $(X^3, X^4, \dots, X^k, X^1, X^2)$. In order to perform each SFC index distributedly, we simply adopt the DIM’s zone construction protocol to build a common zone complex for all sub-indices. Each event is stored $\lceil k/2 \rceil$ times in the sensor networks corresponding to $\lceil k/2 \rceil$ sub-indices. Each replica of an event is coded with the corresponding z-order SFC and mapped to the common zone complex. Thus, MDI can be seen as a combination of $\lceil k/2 \rceil$ different DIM indices.

Note that code of the zone where an event’s replica is stored depends on the order of coordinates that is used. If a sub-index uses $\langle X^1, X^2, \dots, X^{k-1}, X^k \rangle$ as the order for indexing event space, only adjacent hyper-cubes that are different in X^{k-1} or X^k will be mapped to two adjacent geographical zones. Adjacent hyper-cubes that differ in other coordinates will be separated from each other. Doing shift-and-circulate the order of coordinates $\lceil k/2 \rceil$ times guarantees that every coordinate will stand in one of two ending positions at least one. Thus, adjacent hyper-cubes will be mapped to adjacent zones in at least one DIM sub-index.

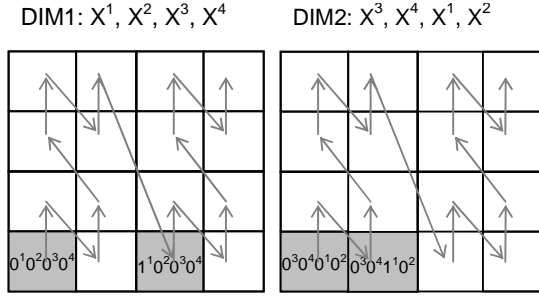


Figure 2: MDI for 4-D space (superscripts indicate coordinates).

Figure 2 illustrates MDI scheme for 4-dimensional data space. The MDI indexing comprises of two DIM sub-indices. DIM1 partitions event space according to the order $\langle X^1, X^2, X^3, X^4 \rangle$; and DIM2 uses the order $\langle X^3, X^4, X^1, X^2 \rangle$. In DIM 1, hyper-cubes (0, 0, 0, 0) and (1, 0, 0, 0) are mapped to zones 0000 and 1000, which are separated. In DIM2, they are mapped to adjacent zones (0000 and 0010).

Proposition 1: *Given a hyper-cube Z in k -D space, let $I^{(i)}(Z)$ be the geographical zone where Z is mapped to by the DIM indexing $I^{(i)}$. If Z and Z' are two adjacent hyper-cubes in the k -D data space, there exists one DIM sub-index $I^{(i)}$ such that $I^{(i)}(Z)$ and $I^{(i)}(Z')$ are adjacent in the geographical area.*

Proof (informal): Since Z and Z' are two adjacent hyper-cubes in event space, their coordinates are all identical except for a coordinate, say coordinate l . Also, coordinate l of the two hyper-cubes differ only in the least significant bit. From an order of k coordinates used by one of DIM sub-indices, repeatedly shift and circulate coordinates by two positions until the dimension l stands in one of the two ending position in the order. Because there are totally k dimensions, the number of shifting is less than or equal to $\lceil k/2 \rceil$. Thus, the resulting order is also used in one of $\lceil k/2 \rceil$ DIM sub-indices, say DIM sub-index $I^{(i)}$. Zone codes of Z and Z' are either $P\alpha$ and $P\beta$ or $P\alpha b$ and $P\beta b$. Here P is a common bit string; α, β, b are bits. Calculating X and Y coordinates of geographical zones $I^{(i)}(Z)$ and $I^{(i)}(Z')$ by bitwise de-interleaving zone codes reveals that $I^{(i)}(Z)$ and $I^{(i)}(Z')$ differ in either X coordinate or Y coordinate in the least significant bit. Thus, $I^{(i)}(Z)$ and $I^{(i)}(Z')$ are adjacent in the geographical area ■

Handling Insertion and Query

MDI inserts an event $\lceil k/2 \rceil$ times in the network, each for a DIM index of a specific coordinate order. A straightforward algorithm for insertion of an event to MDI is invoking $\lceil k/2 \rceil$ DIM insertions separately. However, such a semantic of inserting data to multiple places of MDI naturally matches a multicasting mechanism. Using multicast routing in event insertions will help in saving

message transmissions and thus improve efficiency. We adopt a simple multicast routing for event routing as follows:

Event message: an event message for an event contains the routed event (a tuple of attributes) and a bit mask indicating index replicas carried in the message. The bit mask has $\lceil k/2 \rceil$ bits in length. Initially its bits are all 1 meaning that the event message carries all $\lceil k/2 \rceil$ replicas.

Event message replication and forwarding: On receiving an event message, a node, say node i , decides to replicate, to forward or to store the event message by checking the common code of all replicas carried in the message. The common code is the longest common prefix of those replicas' codes and, in fact, corresponds to the smallest bounding zone of destination zones of replicas. If the smallest bounding zone contains i 's zone, i duplicates the event message by partitioning the set of replicas into two subsets. Each subset belongs to a child zone of the minimum bounding zone. The two copies of the event message is then forwarded or stored according to DIM's mechanisms. For details of event routing of DIM, we refer interested readers to [4].

Our proposed query resolution algorithm routes query message on multiple DIM sub-indices. Queries are also considered as tuples of attributes like events. Thus, each query has $\lceil k/2 \rceil$ destination zones where matched events can be found. In order to reduce query cost, a node i , on receiving a query, forwards the query toward a destination zone that is closest to the zone of i . Again, we adopt the query routing of DIM in forwarding query to the closest destination zone.

Performance Evaluation

We evaluate the efficiency of MDI through simulation with ns2 simulator [8]. Our simulation code of MDI is based on DIM source code that is provided by authors in [4].

Our evaluation is conducted on a square sensor field whose size is set to 200 meters. There are 150 sensor nodes uniformly deployed on that field. The radio transmission range is set to 40 m. Each event is a tuple of 8 attributes. An attribute is represented as an integer from 0 to 1023 inclusively. It means the space of events used in our simulation is a 8-dimensional space and there are 4 sub-indices in MDI indexing structure. Performance metrics for our evaluation are costs of insertion and query. Here, costs are calculated by the total number of messages generated by one insertion or query. Since the communications subsystem consumes most of energy for operations of sensor networks. Insertion and query cost reflect energy efficiency of our indexing protocol.

Insertion cost: Our first experiment is for evaluating the efficiency of multicast routing in event insertions. For each round of simulation, a number of random events are inserted into the network. The rate of event insertions is random numbers from 0.5 to 5 events per second. Averaging on insertion cost shows that insertion cost in

MDI is double as much as cost for inserting an event into a single DIM. Since there are 4 DIM indices in the index structure, without multicast routing, insertion of an event costs 4 times as much as a single DIM does. Thus, multicast routing in MDI reduces the cost of event insertion by half.

Query cost: Reduction in query cost of MDI is investigated in the second experiments. We compare query cost in MDI with that of the original DIM. In this set of experiments, we assume that the data-centric storage sensor network is “full”. It means the whole event space is stored in the sensor network. A number of random range queries are generated in a round of simulation. Query rate is also a random value between 0.5 to 5 queries per second. We vary the size of queries and plots corresponding query cost in figure 3. Each data point in the plot is averaged from 3 simulation runs.

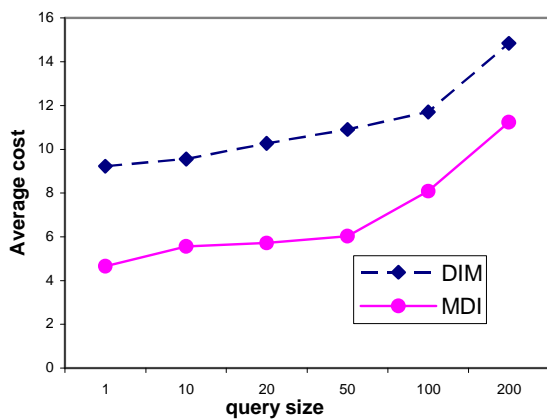


Figure 3: Query cost vs. query size

As can be seen from figure 3, MDI approximately saves 50% costs of small range queries. Although efficiency of MDI degrades when query range grows, we believe that huge range queries do not appear frequently in typical applications like intrusion detection or habitat monitoring. For query size of 200, which is about 20% of a dimension of the event space, MDI can still reduce query cost by 25%. It is also noteworthy that the cost reduction in query and trade-off in insertion cost ties. Thus, MDI is suitable for applications in that queries are more frequently occurred than event insertions.

Related Work

Data-Centric Storage is proposed for the first time in [2]. The paper clarifies that a data-centric communication abstraction is suitable for application scenarios where data retrievals can be requested from any querier inside sensor networks. Primitives of data-centric communication used in DCS networks are, in fact, identical to that of Distributed Hash Table [11, 12, 15]. Particularly, the most basic primitives are insertion of an event under a key, and query of events given that the events keys are known. In practice, the feasibility of DCS sensor networks depends

on efficiency of underlay routing protocol that supports data-centric communications primitives. One of such efficiency routing approach is geographical routing. Geographical routing, such as GPSR [3], exploits node positions to route packet to a destination location. This characteristic makes geographical routing matches the DHT abstraction since the routing primitive is location-base but not node-based. Also, GPSR is stateless. Information required for nodes to do routing is independent of network size. This allows geographical routing scales to very huge network.

The first DCS design introduced in [1, 2] is Geographic Hash Table (GHT). As the name implies, GHT hashes event names to geographic location with consistent hashing functions such as SHA-1. An event, after having an associated hash location, is routed to its location with GPSR and stored at nodes on the perimeter surrounding the hash location. Since consistent hashing maps events to almost random locations, GHT is suitable for point queries only.

Besides DCS proposals that utilize nodes’ physical positions and GPSR as the underlay routing, several attempts have been made to build DCS sensor networks that solely requires topological information and node connectivity. T-DHT [10] sets up a virtual coordinates system for each node by using a set of landmarks nodes. A greedy forwarding approach is used to forward packets to destinations. GEM [14] maps sensor nodes to a virtual polar coordinate system and routes packets between nodes on a ringed tree rooted at a single root node. Location of a node in GEM is determined by a distance (in hops) to the root and a virtual angle. Virtual angles can be assigned to nodes after the ringed tree is built. [9] and [13] presents topology-based DCS designs that work for sensor fields of complex shapes. Although those topology-based schemes provide solutions for DCS networks without geographical information, their routing mechanisms become fragile when there are nodes joint and leave the networks and when node mobility increases. Also, it is hard for topology-based DCS to support range queries.

Closest to our work would be [16]. In [16], each event is *decomposed* into attributes and each attribute is stored in one DIM. In other words, each event has k replicas; each replica is indexed by one of event’s attributes and stored in one DIM. Our proposed scheme differs from such an idea in the way of indexing each event replica. Every event replica is indexed by all k attributes. The difference in indexing different replicas is just the order of attributes. Our indexing scheme, in comparison to the data organization scheme in [16], needs to maintain only $\lceil k/2 \rceil$ DIM sub-indices. Also, our indexing structure employs multicast routing in event insertions to reduce insertion cost. Also, in case of the scheme in [16], if one event replica is lost due to node failures, that event is unable to retrieve by querying on the lost attribute. In case of MDI, an event is still possibly retrieved by querying on all attributes despite losses of some replicas. Thus, our indexing is also better in terms of data resiliency.

Conclusions

Since a single SFC-based index, such as DIM, cannot map all adjacent hyper-cubes in a k -D space, $k > 2$, to adjacent zones in 2-D space, we combine multiple indices to form a better locality-preserving index structure. For event insertions to MDI, a multicast routing scheme is used to reduce insertion cost. For resolving a query, query is routed to the closest replica among $\lceil k/2 \rceil$ replicas in DIM sub-indices. We have conducted several simulations to evaluate the benefit of our combined indexing approach in terms of energy-related cost in processing multi-dimensional range query. Our initial simulation results show that our indexing scheme can reduce up to 50% of query cost.

References

- [1] Sylvia Ratnasamy, Brad Karp, Li Yin, Fang Yu, Deborah Estrin, Ramesh Govindan, Scott Shenker, "GHT: A Geographic Hash Table for Data-Centric Storage," WSNA '02.
- [2] Scott Shenker, Sylvia Ratnasamy, Brad Karp, Ramesh Govindan, Deborah Estrin, "Data-centric storage in sensornets", ACM SIGCOMM Computer Communication Review, volume 33, issue 1, January 2003
- [3] Karp, B. and Kung, H.T., "Greedy Perimeter Stateless Routing for Wireless Networks," in Proceedings of the Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom 2000), 2000/08.
- [4] Xin Li, Young Jin Kim, Ramesh Govindan, and Wei Hong, University of Southern California, "Multi-dimensional Range Queries in Sensor Networks," Proceedings of the ACM Conference on Embedded Networked Sensor Systems, 2003/11.
- [5] Mohamed F. Mokbel, Walid G. Aref, and Ibrahim Kamel, "Performance of Multi-Dimensional Space-Filling Curves," Proceedings of the International Symposium on Advances in Geographic Information Systems, ACM GIS 2002
- [6] B. Greenstein, D. Estrin, R. Govindan, S. Ratnasamy, and S. Shenker, DIFS: A Distributed Index for Features in Sensor Networks, Proceedings of the First IEEE International Workshop on Sensor Network Protocols and Applications, May 2003
- [7] D. Ganesan, D. Estrin, and J. Heidemann, "DIMENSIONS: Why do we need a new Data Handling architecture for Sensor Networks?," In Proceedings of the First Workshop on Hot Topics In Networks (HotNets-I), October 2002
- [8] The Network Simulator - ns-2, <http://www.isi.edu/nsnam/ns/>
- [9] Bruck, J.; Gao, J. & Jiang, A. (A. (2005), MAP: medial axis based geometric routing in sensor networks, in 'MobiCom '05: Proceedings of the 11th annual international conference on Mobile computing and networking', ACM, New York, NY, USA, pp. 88--102.
- [10] Landsiedel, O.; Lehmann, K. & Wehrle, K., "T-DHT: topology-based distributed hash tables," Peer-to-Peer Computing, 2005. P2P 2005. Fifth IEEE International Conference on, 143-144.
- [11] Stoica, I.; Morris, R.; Karger, D.; Kaashoek, M. F. & Balakrishnan, H. (2001), 'Chord: A scalable peer-to-peer lookup service for internet applications', SIGCOMM Comput. Commun. Rev. 31(4), 149--160.
- [12] Ratnasamy, S.; Francis, P.; Handley, M.; Karp, R. & Schenker, S. (2001), "A scalable content-addressable network," in 'SIGCOMM '01: Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications', ACM, New York, NY, USA, pp. 161--172.
- [13] Fang, Q.; Gao, J. & Guibas, L. J. (April 2006), "Landmark-Based Information Storage and Retrieval in Sensor Networks", INFOCOM 2006.
- [14] Newsome, J. & Song, D. (2003), "GEM: Graph Embedding for routing and data-centric storage in sensor networks without geographic information," in 'SenSys '03: Proceedings of the 1st international conference on Embedded networked sensor systems', ACM, New York, NY, USA, pp. 76--88.
- [15] Rowstron, A. & Druschel, P. (2001), "Pastry: Scalable, Decentralized Object Location, and Routing for Large-Scale Peer-to-Peer Systems", Lecture Notes in Computer Science 2218,
- [16] Gummadi, R.; Li, X.; Govindan, R.; Shahabi, C. & Hong, W. (2004), "Energy-efficient data organization and query processing in sensor networks," in 'SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems', ACM, New York, NY, USA, pp. 273--274.