J Inf Process Syst, Vol.16, No.6, pp.1447~1458, December 2020
https://doi.org/10.3745/JIPS.04.0198

ISSN 1976-913X (Print)
ISSN 2092-805X (Electronic)

JOURNAL OF INFORMATION PROCESSING SYSTEMS JIPS

A Structured Overlay Network Scheme Based on Multiple Different Time Intervals

Tomoya Kawakami*

Abstract

This paper describes a structured overlay network scheme based on multiple different time intervals. Many types of data (e.g., sensor data) can be requested at specific time intervals that depend on the user and the system. These queries are referred to as "interval queries." A method for constructing an overlay network that efficiently processes interval queries based on multiple different time intervals is proposed herein. The proposed method assumes a ring topology and assigns nodes to a keyspace based on one-dimensional time information. To reduce the number of forwarded messages for queries, each node constructs shortcut links for each interval that users tend to request. This study confirmed that the proposed method reduces the number of messages needed to process interval queries. The contributions of this study include the clarification of interval queries with specific time intervals; establishment of a structured overlay network scheme based on multiple different time intervals; and experimental verification of the scheme in terms of communication load, delay, and maintenance cost.

Keywords

Communication Load Reduction, Distributed Data Management, Interval Query, Ring-Shaped Overlay Network, Sensor Data, Temporal Data

1. Introduction

The number of Internet of Things (IoT) devices is growing rapidly [1,2], leading to an ever larger volume of published data of great variety. One example is sensor information consisting of temperature, power consumption, and camera images. Sensor information contains temporal and spatial data, including geographical positions, areas, and times, which users and systems can search. To accommodate the many devices (nodes), data, and users, a highly scalable mechanism is necessary to search the available data efficiently.

Various techniques have been proposed to realize high scalability of overlay networks, such as distributed hash tables (DHTs), skip graphs, and geographical-based data [3–20]. Those techniques construct logical networks between nodes based on the specific information of each node (e.g., a one-dimensional ID or a multi-dimensional attribute). Nodes are discovered by sending a query, referred to as a "key", with a specific value or range. Each node sends the queries to the destination node according to its routing table. In addition, current techniques assign territories to the nodes for distributed

Manuscript received April 1, 2019; first revision November 12, 2019; accepted January 14, 2020.
Corresponding Author: Tomoya Kawakami (tomoya-k@u-fukui.ac.jp)
* Graduate School of Engineering, University of Fukui, Fukui, Japan (tomoya-k@u-fukui.ac.jp)

management and searches of sensor data. Users are likely to request sensor data with specific time intervals as “I want to know the data for each one month from the specific day” or “I want to show the data on the same day of other weeks”. However, current techniques cannot efficiently process queries with multiple different time intervals, which are expected to have several patterns simultaneously. Fig. 1 shows an example of such queries for sensor data collection. These queries are referred to as “interval queries” herein.

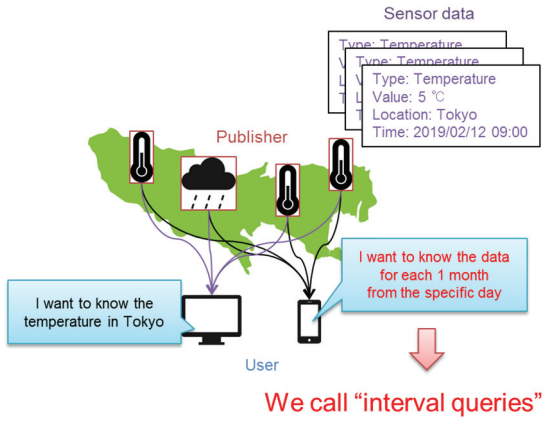

Fig. 1. User’s query requesting data with a specific time interval.

In a previous study, we proposed a method for constructing overlay networks that can efficiently process queries based on multiple different time intervals [21]. This paper presents a method that assumes a ring topology and assigns each node to a keyspace based on one-dimensional time information. In addition, to reduce the number of forwarded messages for queries, each node constructs shortcut links for each interval that users are likely to request. This paper presents an extension of the earlier work and provides new experimental results. The contributions of this study are as follows: clarification of interval queries with specified time intervals; establishment of a structured overlay network scheme based on multiple different time intervals; and experimental verification in terms of communication load, delay, and maintenance cost.

The extant technique is essentially based on a Chord network and is described in Section 2. The proposed method is described in Section 3, and its evaluation is summarized in Section 4. The related work is described in Section 5, and the conclusions presented in Section 6.

2. Chord

Chord [3] is a typical protocol and algorithm for ring-shaped DHTs for which expanded techniques have been studied [4–6]. Chord assigns each node to a specific position on a one-dimensional keyspace based on hashed information such as its ID or IP address. The node that follows a given node in the keyspace is called a “successor”, and the previous node is called a “predecessor”. Each node constructs links to its successor and predecessor when it is inserted into the overlay network. In addition, each node manages a partial space from the predecessor in its position as a territory for data management by Chord.

Chord directs each node to construct shortcut links in what is called a "finger table" to reduce the number of hops for node search in the overlay network. The finger table consists of links to the nodes that have 2^1, 2^2, 2^3, ..., 2^{m-1} -bit additional keys from each node where m is the bit length of the keyspace. The link to an additional 2^0-bit key corresponds to the successor. Fig. 2 shows an example of a finger table when the length of the keyspace is 2^6 bits (values from 0 to 63). The numbers in the circles in Fig. 2 refer to the nodes with their keys, and the arrows show the shortcut links of the node assigned to key 1. The upper right table shows part of the linked nodes for each key.

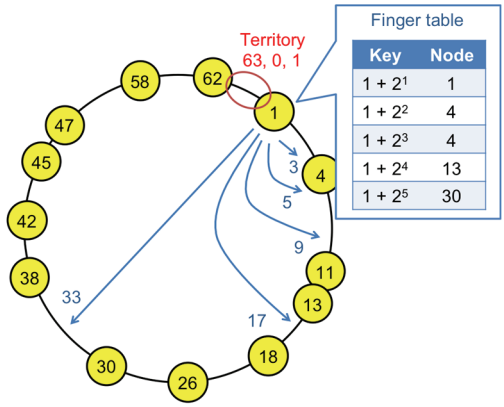

Fig. 2. An example of the constructed shortcut links in Chord.

In Chord, the finger table enables $\boldsymbol{O}(\log N)$ hops for node searching using a specified key while keeping the size of the finger table under $m + 1$ (N denotes the number of nodes). In addition, Chord can be applied to distributed data management, and its expanded techniques have been proposed for multi-dimensional information [4]. Users and systems are expected to request sensor data with specific time intervals such as "I want to know the data for each one month from the specific day" or "I want to show the data on the same day of other weeks". In addition, the specified time intervals are expected to have several patterns at the same time. However, existing techniques cannot efficiently process queries with multiple different time intervals.

3. Proposed Method

In this paper, an overlay network construction method that can efficiently process queries based on multiple different time intervals such as a year, a month, a week, a day, or the time is proposed. An overview and details of the proposed method are in the following section.

3.1 Overview

The proposed method assumes a ring topology similar to that of Chord and assigns nodes to the keyspace based on one-dimensional time information. The one-dimensional information is a bit value that represents the unit of the query (e.g., a year, a month, a week, a day, or the time). In addition, the one-dimensional information has a length as a keyspace similar to coordinated universal time (UTC) or

UNIX time. In a Chord finger table, each node constructs shortcut links to the nodes assigned to 2^i bits from its own key (i = 1, 2, ..., m - 1). m denotes the bit length of the keyspace. In contrast, the proposed method has each node construct shortcut links with the specific intervals that users tend to request. In this case, each node constructs shortcut links to the nodes assigned to one year, one month, one week from its own key, and so on. Moreover, each node constructs shortcut links to reduce the hop number (e.g., one month, three months, six months, and 12 months from its own key); the year, month, week, day, and hour were considered in this study. Table 1 shows the types, keys, and number of the constructed shortcut links. The smallest unit in Table 1 is 1 hour, and the bit length of the keyspace is denoted by m. The minimum unit can be set to other elements such as a minute or a second.

Fig. 3 shows the assumed flow for forward interval queries. Although Fig. 3 is simplified and does not show the ring topology, the interval queries are recursively forwarded to the related nodes via the shortcut links. If a node that receives the query has matched data, a reply is sent by the node.

Table 1. The types, keys, and number of constructed links

Type	Key (interval)	Number
Year	2^0, 2^1, 2^2, ..., 2^{m-1}	m
Month	6, 3, 1	3
Day	15, 7 (one week), 3, 1	4
Time	12, 6, 3, 1	4
Neighbor	Successor, predecessor	2

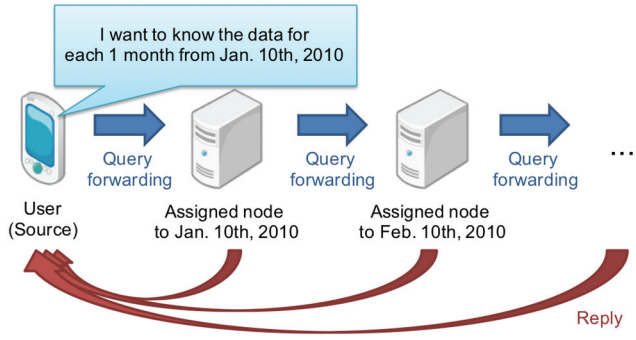

Fig. 3. The assumed forwarding of interval queries.

3.2 Construction of the Shortcut links

Fig. 4 shows an example of the constructed shortcut links described in Section 3.1. In Fig. 4, the minimum unit is a day, started from January 1, 2001, and the length of the keyspace is 64 days (from January 1 to March 5). The numbers in the circles show the nodes with their keys (days), and the arrows show the shortcut links of the node assigned to key 1. The upper right table shows part of the linked nodes for each key. The node assigned to key 1 constructs the shortcut links with their intervals:

- Month: 1 (31 days)
- Day: 15, 7 (one week)
- Neighbors: successor, predecessor

Unlike the finger table used in Chord, shortcut links are constructed in the proposed method to the

nodes assigned to the keys for the next month and week. This method generates few messages and hops to forward queries such as “I want to know the data for each month from the specific day” or “I want to show the data on the same day of other weeks”. In addition, Fig. 4 shows four shortcut links, which is not significantly different than the Chord technique, which constructs $m - 1$ shortcut links in the worst case.

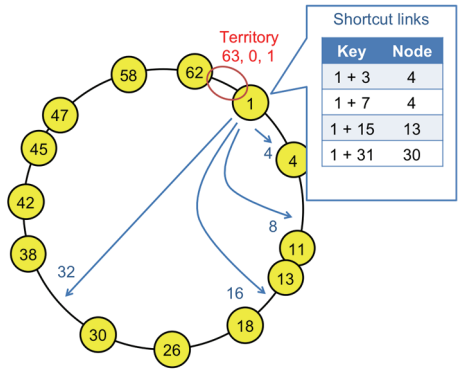

Fig. 4. An example of the constructed shortcut links in the proposed method.

4. Evaluation

In this study, a simulator was used to evaluate the proposed method. It was implemented using the Java programming language. Table 2 shows the simulation parameters, the details of which are described as follows.

Table 2. Simulation parameters

Parameter	Value
The minimum unit of the key space	1 hour
The length of the key space	16 bits (2^{16} = 65,536 hours)
The number of nodes	1000, 2000, 3000, 4000
Node keys	Assigned at random
Comparison method	Chord-like shortcut links (exponential)
The number of queries	1
The number of keys with a query	12, 52, 120
Query patterns	12 keys with a 720-hour interval, 12 keys with random intervals, 54 keys with a 168-hour interval, 54 keys with random intervals, 120 keys with 72-hour interval, 120 keys with random intervals
Simulation count	20
Evaluated items	Average number of messages, maximum number of hops, average number of entries in the routing table

4.1 Simulation Environment

The smallest unit of time in the keyspace in the simulation is 1 hour, and the length of the keyspace is 16 bits (the total is 2^{16}, or 65,536 hours). There are 1000 to 4000 nodes; each node is assigned to a key at random and constructs shortcut links to other nodes. The proposed method was compared to the Chord-like shortcut links (exponential time). Only one query is sent; however, it contains several keys and is forwarded to the assigned nodes. There are six key patterns for the query. The first pattern is 12 keys from a random point at intervals of 720 hours (the total is 8,640 hours, or 360 days). The second pattern is also 12 keys; however, they are determined at random. The third pattern is 52 keys from a random point while those intervals are 168 hours (the total is 8,736 hours, or 52 weeks). The last pattern is 52 keys from a random point, but each key is determined at random. The fifth pattern is 120 keys from a random point while those intervals are 72 hours (the total is 8,736 hours, or 52 weeks). The last pattern is 120 keys from a random point, but each key is determined at random. The simulation was run 20 times under each environment and calculated the average number of messages sent to the assigned nodes as communication loads in a distributed system.

4.2 Results of Query Forwarding

Fig. 5 shows the number of messages sent to the assigned nodes in the first and second query patterns for 12 keys. The x-axis represents the number of nodes. "Exponential (Random query)" and "Proposed (Random query)" refer to the Chord-like comparison method and the proposed method with random intervals, respectively. In addition, "Exponential (Interval query)" and "Proposed (Interval query)" mean the Chord-like comparison method and the proposed method with a specified interval, respectively. Fig. 5 shows that the proposed method achieved the lowest number of messages when the 12 keys of the query had an interval of 720 hours. However, the proposed method had more messages than the comparison method when the 12 keys of the query were determined at random. In this environment, the comparison method for the interval query had the same result as the proposed method for a random query. In addition, Fig. 6 shows the maximum number of hops to the assigned nodes in the first and second query patterns for 12 keys. Although the proposed method had the worst result when the 12 keys of the query were determined at random, the method also had the lowest hop count when the 12 keys of the query had a specific interval.

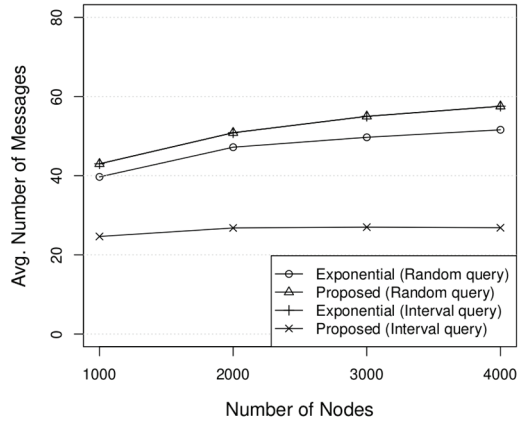

Fig. 5. Number of messages to assigned nodes for each query with 12 keys.

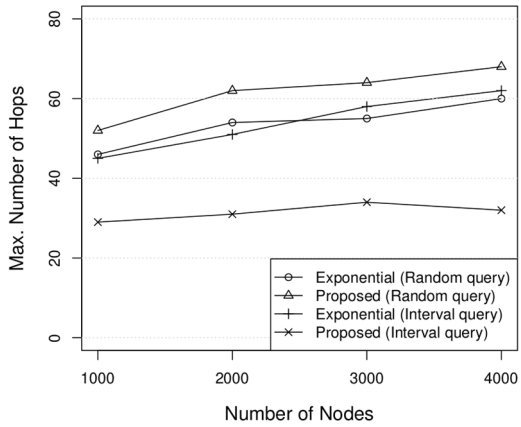

Fig. 6. Maximum of hops to assigned nodes for each query with 12 keys.

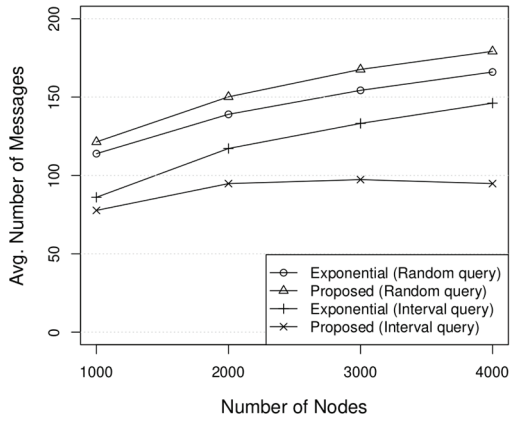

Fig. 7. Number of messages to assigned nodes for each query with 52 keys.

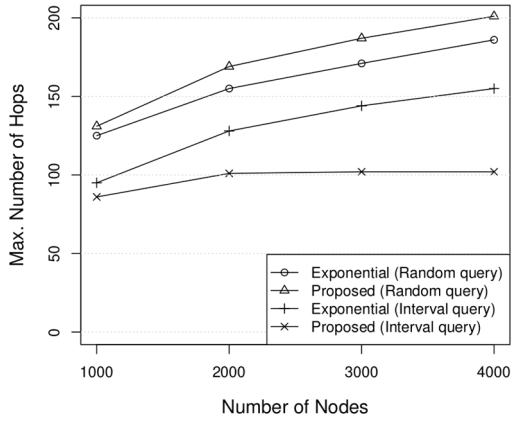

Fig. 8. Maximum number of hops to assigned nodes for each query with 52 keys.

Figs. 7 and 8 show the number of messages and the maximum number of hops in each query pattern for 52 keys, respectively. In addition, in Fig. 7, the proposed method has the lowest number of messages even if the number of keys in the query is increased to 52. In comparison with the results in Fig. 5, the

comparison method for the interval query returns a different result from the proposed method for a random query. Fig. 8 shows that the method had the lowest hop count when the keys of the query have a specific interval, although the proposed method is worse when the keys of the query are determined at random. Figs. 9 and 10 show the number of messages and the maximum number of hops in each query pattern for 120 keys, respectively. Although those curves by the number of nodes are different, they are placed in the same order. These results indicate that the proposed method reduces the number of messages needed to process queries related to the specific time intervals that users tend to request.

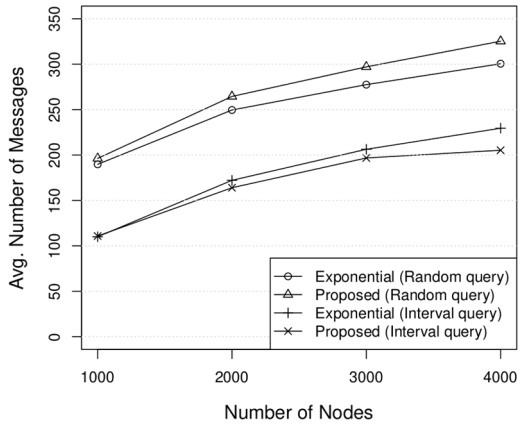

Fig. 9. Number of messages to assigned nodes for each query with 120 keys.

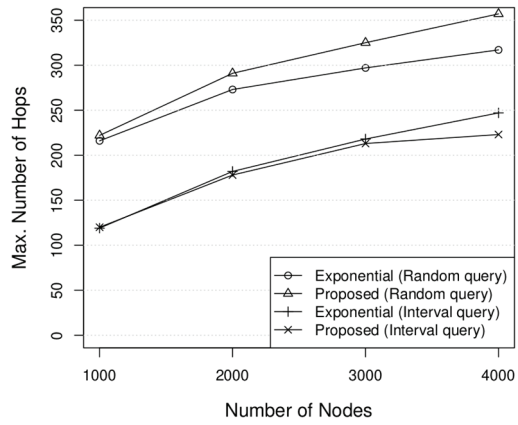

Fig. 10. Maximum number of hops to assigned nodes for each query with 120 keys.

4.3 Results of Entries in Routing Table

Fig. 11 shows the average number of entries in each node's routing table, and the x-axis represents the number of nodes. The two lines in Fig. 11 show the number of entries by registered keys; however, the other two lines show the number of unique destinations in routing table entries.

The number of registered keys is affected by the length of the keyspace but not by the number of nodes. In this simulation, there were 19 registered keys in the comparison method and 16 in the proposed method. In addition, the number of unique destinations in the proposed method is less than that in the

comparison method. This result indicates that the proposed method can reduce the maintenance costs to keep the routing table updated for nodes' join/leave.

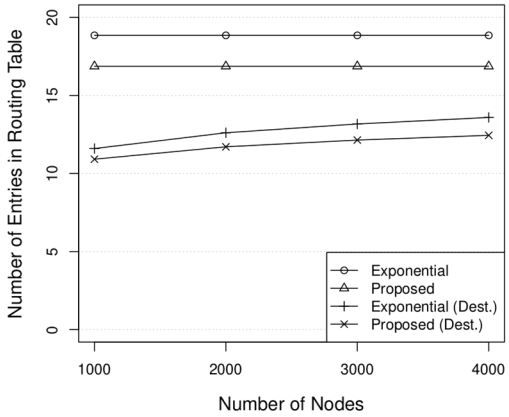

Fig. 11. Average number of entries in each node's routing table.

4. Related Work

Except for Chord or DHTs, many overlay network techniques have been researched [6–20]. Skip graphs are overlay networks in which a skip list is applied in the P2P model [9–14]. The nodes in skip graphs are sorted in ascending order of those keys, and bidirectional links are created among the nodes. The "membership vector", is assigned to each node when the peer joins and is used to create hierarchical (multi-level) links among the nodes. Skip graphs can process range queries that specify the beginning and end of the keys to be searched, and the queries are forwarded to the node whose key is within that range, or less than the end of the range. The number of hops in the key search is represented by $\boldsymbol{O}(\log n)$ where n is the number of nodes. The average number of links on each peer is represented by $\log n$. As one of the expanded techniques of the original skip graph, the Ballistic Skip Graph reduces the degree of each node to $\boldsymbol{O}(1)$ by limiting the number of shortcut links to a single level at random [14]. In addition, both one-dimensional and multi-dimensional range queries have been researched in many overlay network techniques [15–19]. Those techniques can be applied to the management of location/area-based data on 2D and 3D maps, some of which are "geographical overlay networks". Many of the geographical overlay networks are based on geometrical techniques such as the R-tree, quadtree, and Delaunay triangulation (Voronoi diagram). However, these techniques do not assume "interval queries" and cannot forward the queries as efficiently as the proposed method.

To enhance the reliability and feasibility of overlay networks, related techniques have been researched [22–32]. First, a variety of replication schemes, such as path replication, have been proposed for unstructured P2P networks [22], where nodes search for content by forwarding queries to publishing nodes via neighboring links. The path replication schemes replicate the replied content on the nodes between the publishing and requesting nodes. Related to path replication schemes are the many methods based on specific factors such as the number of queries, probability to put replicas, and churn situations [23–25]. In addition, replication schemes have also been proposed for structured P2P networks to increase

the efficiency of replica maintenance and searches. One example is Scalaris, an Erlang implementation of a distributed key/value store [26]; it uses replication for data availability and majority-based distributed transactions for data consistency. Plover is a proactive low-overhead file replication scheme with replication among physically proximate nodes based on their available capacities [27]. Here, physically proximate nodes are grouped in clusters, each of which has a supernode with high capacity and rapid connections. In RelaxDHT, nodes are divided into data blocks [28], with each block having a root node that manages the metadata of replicas on other nodes in their own data blocks. Although these techniques do not assume interval queries, the proposed method can adapt the same or similar ideas, such as node/data replication, to enhance its reliability.

5. Conclusion

This paper describes a structured overlay network scheme based on multiple different time intervals. The overlay network construction method assumes a ring topology and assigns nodes to a keyspace based on one-dimensional time information. In addition, to reduce the number of forwarded messages for the queries, each node constructs shortcut links with the specific intervals that users tend to request. The proposed method was confirmed to reduce the number of messages needed to process queries related to time intervals.

In future studies, virtual node techniques will be employed to reduce the load differences among the nodes because the amount of the registered or requested data differs with the keys that represent time. In addition, node/data replication techniques will be used to enhance the reliability and feasibility of the proposed scheme mentioned in Section 4. Subsequently, the proposed scheme will be implemented for the P2P agent platform, PIAX [33], and testbed systems to evaluate the scheme for practical applications.

Acknowledgement

This work was supported by JSPS KAKENHI (Grant No. 16K16059), Hoso Bunka Foundation, and Research Grants from the University of Fukui.

References

[1] J. Lin, W. Yu, N. Zhang, X. Yang, H. Zhang, and W. Zhao, “A survey on Internet of Things: architecture, enabling technologies, security and privacy, and applications,” *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1125-1142, 2017.

[2] A. Colakovic and M. Hadzialic, “Internet of Things (IoT): a review of enabling technologies, challenges, and open research issues,” *Computer Networks*, vol. 144, pp. 17-39, 2018.

[3] I. Stoica, R. Morris, D. Liben-Nowell, D. R. Karger, M. F. Kaashoek, F. Dabek, and H. Balakrishnan, “Chord: a scalable peer-to-peer lookup protocol for Internet applications,” *IEEE/ACM Transactions on Networking*, vol. 11, no. 1, pp. 17-32, 2003.

[4] Y. Shu, B. C. Ooi, K. L. Tan, and A. Zhou, “Supporting multi-dimensional range queries in peer-to-peer systems,” in *Proceedings of the 5th IEEE International Conference on Peer-to-Peer Computing (P2P 2005)*, Konstanz, Germany, 2005, pp. 173-180.

[5] S. K. Tetarave, S. Tripathy, and R. K. Ghosh, "V-Chord: an efficient file sharing on LTE/GSM Network," in *Proceedings of the 19th International Conference on Distributed Computing and Networking (ICDCN)*, Varanasi, India, 2018, pp. 1-8.

[6] L. Cheklat, M. Amad, M. Omar, and A. Boukerram, "Energy efficient physical proximity based Chord protocol for data delivery in WSNs," in *Proceedings of the 2018 International Conference on Applied Smart Systems (ICASS)*, Medea, Algeria, 2018, pp. 1-6.

[7] T. Gu, S. J. Hong, S. Uhmn, and K. M. Lee, "Membership management based on a hierarchical ring for large grid environments," *Journal of Information Processing Systems*, vol. 3, no. 1, pp. 8-15, 2007.

[8] W. Zhang, H. Tang, S. Byna, and Y. Chen, "DART: distributed adaptive radix tree for efficient affix-based keyword search on HPC systems," in *Proceedings of the 27th International Conference on Parallel Architectures and Compilation Techniques (PACT)*, Limassol, Cyprus, 2018, pp. 1-12.

[9] J. Aspnes and G. Shah, "Skip graphs," *ACM Transactions on Algorithms*, vol. 3, no. 4, article no. 37, 2007.

[10] S. Takeuchi, J. Shinomiya, T. Shiraki, Y. Ishi, Y. Teranishi, M. Yoshida, and S. Shimojo, "A large scale key-value store based on range-key Skip Graph and its applications," in *Database Systems for Advanced Applications*. Heidelberg, Germany: Springer, 2010, pp. 432-435.

[11] R. Banno, T. Fujino, S. Takeuchi, and M. Takemoto, "SFB: a scalable method for handling range queries on skip graphs," *IEICE Communications Express*, vol. 4, no. 1, pp. 14-19, 2015.

[12] S. T. Boshrooyeh and O. Ozkasap, "Guard: secure routing in skip graph," in *Proceedings of the 2017 IFIP Networking Conference (IFIP Networking) and Workshops*, Stockholm, Sweden, 2017.

[13] Y. Hassanzadeh-Nazarabadi and O. Ozkasap, "ELATS: energy and locality aware aggregation tree for skip graph," in *Proceedings of the 2017 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom)*, Istanbul, Turkey, 2017.

[14] Y. Aoki, M. Ohnishi, and K. Shudo, "Ballistic skip graph: a skip graph-style constant-degree structured overlay," in *Proceedings of the 23rd IEEE Symposium on Computers and Communications (ISCC)*, Natal, Brazil, 2018, pp. 341-346.

[15] A. Mondal, Y. Lifu, and M. Kitsuregawa, "P2PR-Tree: an R-tree-based spatial index for peer-to-peer environments," in *Current Trends in Database Technology - EDBT 2004 Workshops*. Heidelberg, Germany: Springer, 2004, pp. 516-525.

[16] E. Tanin, A. Harwood, and H. Samet, "Using a distributed quadtree index in peer-to-peer networks," *The VLDB Journal*, vol. 16, no. 2, pp. 165-178, 2007.

[17] F. Araujo and L. Rodrigues, "GeoPeer: a location-aware peer-to-peer system," in *Proceedings of the 3rd IEEE International Symposium on Network Computing and Applications (NCA)*, Cambridge, MA, 2004, pp. 30-46.

[18] M. Ohnishi, M. Inoue, and H. Harai, "Incremental distributed construction method of Delaunay overlay network on detour overlay paths," *Information and Media Technologies*, vol. 8, no. 2, pp. 556-564, 2013.

[19] Y. Teranishi, S. Takeuchi, and K. Harumoto, "HDOV: an overlay network for wide area spatial data collection," in *Proceedings of the 26th ACM Symposium on Applied Computing (SAC)*, TaiChung, Taiwan, 2011, pp. 506-513.

[20] K. Ragab, "An autonomic <K, D>-interleaving registry overlay network for efficient ubiquities web services discovery service," *Journal of Information Processing Systems*, vol. 4, no. 2, pp. 53-60, 2008.

[21] T. Kawakami, "A construction method for structured overlay networks based on multiple different time intervals," in *Proceedings of the 2019 World Congress on Information Technology Applications and Services (World IT Congress 2019 Jeju),* Jeju, South Korea, 2019, pp. 81-86.

[22] Q. Lv, P. Cao, E. Cohen, K. Li, and S. Shenker, "Search and replication in unstructured peer-to-peer networks," in *Proceedings of the 16th ACM International Conference on Supercomputing (ICS)*, New York, NY, 2002, pp. 84-95.

[23] V. Gopalakrishnan, B. Silaghi, B. Bhattacharjee, and P. Keleher, "Adaptive replication in peer-to-peer systems," in *Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS)*, Tokyo, Japan, 2004, pp. 360-369.

[24] H. Yamamoto, D. Maruta, and Y. Oie, "Replication methods for load balancing on distributed storages in P2P networks," *IEICE Transactions on Information and Systems*, vol. 89, no. 1, pp. 171-180, 2006.

[25] K. P. N. Puttaswamy, A. Sala, and B. Y. Zhao, "Searching for rare objects using index replication," in *Proceedings of the 27th IEEE International Conference on Computer Communications (INFOCOM)*, Phoenix, AZ, 2008, pp. 1723-1731.

[26] T. Schütt, F. Schintke, and A. Reinefeld, "Scalaris: reliable transactional P2P key/value store," in *Proceedings of the 7th ACM SIGPLAN Workshop on ERLANG*, Victoria, Canada, 2008, pp. 41-48.

[27] H. Shen and Y. Zhu, "A proactive low-overhead file replication scheme for structured P2P content delivery networks," *Journal of Parallel and Distributed Computing*, vol. 69, no. 5, pp. 429-440, 2009.

[28] S. Legtchenko, S. Monnet, P. Sens, and G. Muller, "RelaxDHT: a churn-resilient replication strategy for peer-to-peer distributed hash-tables," *ACM Transactions on Autonomous and Adaptive Systems*, vol. 7, no. 2, article no. 28, 2012.

[29] H. Kim, S. Kang, and H. Y. Yeom, "Server selection schemes considering node status for a fault-tolerant streaming service on a peer-to-peer network," *Journal of Information Processing Systems*, vol. 2, no. 1, pp. 6-12, 2006.

[30] Q. Cao and S. Fujita, "Cost-effective replication schemes for query load balancing in DHT-based peer-to-peer file searches," *Journal of Information Processing Systems*, vol. 10, no. 4, pp. 628-645, 2014.

[31] S. Zahid, S. A. Abid, N. Shah, S. H. A. Naqvi, W. Mehmood, "Distributed partition detection with dynamic replication management in a DHT-based MANET," *IEEE Access*, vol. 6, pp. 18731-18746, 2018.

[32] Y. Hassanzadeh-Nazarabadi, A. Kupcu, and O. Ozkasap, "Decentralized and locality aware replication method for DHT-based P2P storage systems," *Future Generation Computer Systems*, vol. 84, pp. 32-46, 2018.

[33] Y. Teranishi, "PIAX: toward a framework for sensor overlay network," in *Proceedings of the 6th Annual IEEE Consumer Communications and Networking Conference (CCNC)*, Las Vegas, NV, 2009, pp. 1-5.

Tomoya Kawakami **https://orcid.org/0000-0002-5394-3067**

He received his B.E. degree from Kinki University in 2005 and his M.I. and Ph.D. degrees from Osaka University in 2007 and 2013, respectively. From 2007 to March 2013 and from July 2014 to March 2015, he was a specially appointed researcher at Osaka University. From April 2013 to June 2014, he was a Ph.D. researcher at Kobe University. From April 2015 to February 2020, he was an assistant professor at Nara Institute of Science and Technology. Since March 2020, he has been a senior assistant professor at the University of Fukui. His research interests include distributed computing, rule-based systems, and stream data processing. He is a member of the Information Processing Society of Japan (IPSJ) and IEEE.