A Node Virtualization Scheme for Structured Overlay Networks Based on Multiple Different Time Intervals

Tomoya Kawakami

Graduate School of Engineering, University of Fukui, 3-9-1 Bunkyo, Fukui 910-8507, Japan; tomoya-k@u-fukui.ac.jp

Received: 24 October 2020; Accepted: 27 November 2020; Published: 30 November 2020

Abstract: Sensor data which relate to the specific geographical positions, areas, and time are strongly expected in IoT. The author has studied overlay networks to efficiently process interval queries which have specific time intervals and the actual users tend to request. However, unfairness and a concentration of the loads occur for the specific processing computer (node) in the previous method because the density of data or those generators/providers is different from those related values. In this paper, the author proposes the enhanced scheme for structured overlay networks based on multiple different time intervals. The proposed method uses node virtualization to equalize the loads of each real (physical) node. The simulation results showed that the proposed method can increase the fairness of the number of the assigned data among physical nodes.

Keywords: temporal data; interval query; virtual node; load balancing; ring-shaped overlay network; distributed data management

1. Introduction

The number of IoT (Internet of Things) devices [1,2] is growing rapidly, and also the published data obtain a huge number and great variety. As an example of the published data, sensor data are expected such as temperature, power consumption, and camera images. The sensor data are temporal and spatial data which relate to the specific information about geographical positions, areas, and time. In addition, those users and systems search and utilize sensor data based on such related data. To accommodate a huge number of devices (nodes), data, and users, therefore, the high-scalable mechanisms are necessary to search the required data efficiently.

As the techniques to realize high scalability, overlay network construction techniques have been proposed such as distributed hash tables (DHTs), skip graphs, and geographical-based ones [3–20]. Those techniques construct logical networks among nodes based on the specific information of each node such as a one-dimensional ID and multi-dimensional attribute values. Nodes are discovered by sending queries with the specific value or range called “key”. Each node forwards the queries to the destination node based on its own routing table. In addition, the existing techniques assign territories to the nodes for distributed management and search of sensor data. On the other hand, the actual users and systems are expected to request sensor data with the specific time intervals such as “I want to know the data for each one month from the specific day”, “I want to show the data in the same day of week”. In addition, the specified time intervals are expected to have several patterns at the same time. Figure 1 shows the example of such queries as sensor data collection. In this paper, we call those queries “interval queries”.
We call “interval queries”

Figure 1. Users’ queries which request data with the specific time interval.

In previous work, the author has proposed an overlay network construction method which can efficiently process the queries based on multiple different time intervals [21]. The proposed method assumes the ring-shaped topology and assigns the nodes to the key space based on one-dimensional information about time. In addition, to reduce the forwarded messages for the assumed queries, each node constructs shortcut links with the specific intervals which the actual users tend to request. However, the previous method causes unfairness and a concentration of the loads to the specific node because the density of data or those generators/providers is different from those related key values. Therefore, we proposed the enhanced scheme to construct structured overlay networks based on multiple different time intervals [22]. The proposed scheme uses node virtualization to equalize the loads of each real (physical) node. By the proposed scheme, each physical node manages multiple virtual nodes with those assigned keys (time). Load concentration to the specific physical node can be reduced because a large number of virtual nodes are managed by various physical nodes and widely placed on the whole of the key space. As an example of the applications run by virtual nodes, distributed content management systems are expectable for time-related contents such as a photograph, video, and sensor data.

This paper is an extension of [22] to show new experiment results obtained in another simulation environment. In addition, comparisons with related work and discussion are also described in Sections 2.1 and 4, respectively. The contributions of this paper are: (1) clarification of “interval queries” with problems in the existing techniques, (2) establishment of a node virtualization scheme for structured overlay networks based on multiple different time intervals, and (3) the experiment results in the viewpoints of fairness and a communication load among nodes.

In the following, Section 2 describes the materials and methods which contain the related work and the proposed scheme. The evaluation of the proposed scheme and discussion is summarized in Sections 3 and 4, respectively. This paper is concluded in Section 5.
2. Materials and Methods

2.1. Related Work

2.1.1. Chord

Chord [3] is a typical construction technique for ring-shaped DHTs, and also the expanded techniques have been studied [4–6]. In Chord, each node is assigned to the specific position on a one-dimensional key space based on hashed information such as its ID, IP address, and so on. The next node of each node on the key space is called its “successor”, and the previous node is called its “predecessor”. Each node constructs links to its successor and predecessor when the node joins the overlay network. In addition, each node manages the partial space from the predecessor to its position as a territory for data management. In Chord, moreover, each node constructs shortcut links called “finger table” to reduce the number of hops for node searching in the overlay network. The finger table consists of the links to the nodes which have additional $2^1, 2^2, 2^3, \ldots, 2^{m-1}$ bit-keys from each node while $m$ denotes the bit length of the key space. The link for an additional $2^0$ bit-key corresponds to the successor. Figure 2 shows an example of the finger table when the length of the key space is $2^6$ bits (from 0 to 63). In Figure 2, the numbers within the circles show the nodes with their keys, and the arrows show the shortcut links of the node assigned to key 1. The upper right table shows a part of the linked nodes for each key.

![Figure 2. An example of the constructed shortcut links in Chord.](image)

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

2.1.2. Other Overlay Network Techniques

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

2.2. Overlay Networks Based on Multiple Different Time Intervals

In this paper, the author proposes an enhanced scheme for overlay networks which
can efficiently process the queries based on multiple different time intervals such as a year,
month, week, day, and time. The overviews and details of our proposed scheme are described below.

To process the queries such as “interval queries”, the author has proposed a construction method
of Chord-like overlay networks based on multiple different time intervals [21].

2.2.1. Idea

The proposed method assumes the ring-shaped topology similar to Chord and assigns the nodes
to the key space based on one-dimensional information about time. The one-dimensional information
is a bit value which represents the minimum unit such as a year, month, week, day, and time. In addition,
the one-dimensional information has a length as a key space similar to Coordinated Universal
Time (UTC), UNIX time, and so on. As a finger table in Chord, each node constructs shortcut
links to the nodes assigned to $2^i$ bits from its own key ($i = 1, 2, \ldots, m - 1$). $m$ denotes the bit
length of the key space. In our proposed method, on the other hand, each node constructs shortcut
links with the specific intervals which the actual 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 (one year) from its own key. In this paper, we consider
the year, month, week, day, and hour. Table 1 shows the types, keys, and number of the constructed
shortcut links. The minimum unit is an hour in Table 1, and the bit length of the key space is denoted
as $m$. The minimum unit can be set to other elements such as a minute and second.

| Type   | Key (Interval)           | Number |
|--------|--------------------------|--------|
| Year   | $2^0, 2^1, \ldots, 2^{m-1}$ | $m$    |
| Month  | 6, 3, 1                  | 3      |
| Day    | 15, 7 (one week), 3, 1   | 4      |
| Time   | 12, 6, 3, 1              | 4      |
| Neighbor| Successor, predecessor  | 2      |
Figure 3 shows the assumed flow to forward interval queries. Although Figure 3 is simplified without ring-shaped topology, the interval queries are recursively forwarded to the related nodes via the shortcut links. The query received nodes reply the matched data if the nodes have them.

2.2.2. Construction of Shortcut Links

Figure 4 shows an example of the constructed shortcut links described in Section 2.1.1. In Figure 4, the minimum unit is a day from 1 January 2001, and the length of the key space is 64 days (from 1 January to 5 March). The numbers within 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 a part of the linked nodes for each key. In the example, in Figure 4, the node assigned to key 1 constructs the shortcut links which have the following intervals:

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

Compared to the finger table in Chord, our proposed method constructs the shortcut links to the nodes assigned to the keys for the next month and week. Therefore, our proposed method has few messages and hops to forward the queries such as “I want to know the data for each one month
from the specific day”, “I want to show the data in the same day of week”. In addition, there are four shortcut links in Figure 4, and it is not a significant difference compared to Chord, which constructs \( m-1 \) shortcut links as the maximum case.

2.2.3. Node Virtualization

To equalize the loads of each node in Chord, Ref. [3] describes a technique called “virtual nodes”. Each real (physical) node runs multiple virtual nodes and places them into the key space. The assignment influence of the specific single key is reduced because each physical node is assigned to the multiple ranges (territories) of keys by its virtual nodes. If all the physical nodes evenly place the same and enough virtual nodes, the numbers of the assigned data are probabilistically equalized among physical nodes. In addition, adaptive assignment can also be realized by changing the virtual nodes based on the performance or the situation of each physical node.

In this paper, the author enhances the method described in [21] by the virtual node technique mainly to equalize the loads of each physical node. Figure 5 shows an example of virtual node placement in the newly proposed method by the dashed arrows from physical nodes. In Figure 5, physical node 1 runs the virtual nodes assigned to keys 1, 18, and 42, and physical node 2 runs the virtual nodes assigned to keys 11 and 26. In this case, physical node 1 has three territories of keys and manages the data assigned to the keys.

3. Results

In this paper, the author evaluates the proposed method by simulation.

3.1. Simulation Environment

Table 2 shows the simulation environment. In the simulation environment, the minimum unit of the key space is an hour, and the length of the key space is 16 bits \( (2^{16} = 65,536 \text{ h}) \). The number of physical nodes is from 250 to 1000, and all the physical nodes run the same number of virtual nodes from \( 2^1 \) to \( 2^4 \). Each virtual node is assigned to the specific key at random and constructs shortcut links to other virtual nodes based on the method described in [21]. In this paper, the author compares the proposed method to the case of no virtual nodes (or each physical node runs only one virtual node) because the proposed scheme aims to highly distribute the loads of physical nodes by node virtualization.
Table 2. Simulation environment.

| Item                                         | Value                                      |
|----------------------------------------------|--------------------------------------------|
| The min. unit of the key space               | 1 h                                        |
| The length of the key space                  | 16 bits ($2^{16} = 65,536$ h)             |
| The number of nodes                          | 250, 500, 750, 1000                       |
| The number of virtual nodes per physical node| 2, 4, 8, 16                                |
| Node’s key                                   | Assigned at random                         |
| Comparison method                            | No virtual nodes                           |
| The number of data                           | 10,000                                     |
| Data pattern                                 | Uniform, normal distribution, Zipf distribution |
| Query pattern                               | 12 keys with a 720 h interval             |
| Simulation count                             | 20                                         |
| Evaluation items                             | Fairness index, Avg. number of messages    |

First, this simulation measures the fairness index ($FI$) of the number of the assigned data among physical nodes. The fairness index is calculated by the following:

$$FI = \frac{\left(\sum_{i=0}^{n} x_i\right)^2}{n\sum_{i=0}^{n} x_i^2} \tag{1}$$

where $0 \leq FI \leq 1$, and it indicates that the numbers of the assigned data to each physical node, $x_1, \ldots, x_n$, are more similar if it is closer to 1. When $FI = 1$, $x_1 = \ldots = x_n$. The number of the assigned data are 10,000, and those assigned keys are determined based on three patterns, random, Gaussian, and Zipf’s. The first pattern is determined at random within all of the key space. The Gaussian pattern is determined based on Gaussian distribution whose mean is the center of the key space (32,768) and the standard deviation is $65,536/10 = 6553.6$. The Zipf’s pattern is determined based on Zipf’s law (Zipf distribution) [23,24]. Figures 6–8 show examples of each data key distribution, respectively. In addition, the simulation measures the number of messages among virtual nodes as communication loads to process interval queries. The sent query is only one, but it contains 52 keys from the random point and forwarded them to those assigned nodes while those intervals are 168 h (the total length is 8736 h = 52 weeks). The author executes the simulation 20 times for each environment and calculates the average of the measured values.

![Figure 6](image-url).

*Figure 6.* The number of messages among virtual nodes for one query.
3.2. Simulation Results

Figures 9–11 show the fairness index of the number of the assigned data among physical nodes when the keys of data are determined at random, based on Gaussian distribution, and Zipf distribution, respectively. Each lateral axis represents the number of physical nodes.

In Figure 9, $FI$ is increased by the number of virtual nodes. This means that the differences of the number of assigned data and unfairness are reduced. In addition, $FI$ is slightly decreased by the number of physical nodes. In addition, in Figure 10, $FI$ is increased by the number of virtual nodes. Although all of the results are lower than Figure 9, the number of virtual nodes has a larger influence on the differences. On the other hand, the keys are extremely unbalanced in Zipf distribution and the whole of $FI$ is significantly low in Figure 11 even if the number of virtual nodes is large. This seems because the loads are concentrated on the specific physical node and the number of virtual nodes is
not enough to distribute the loads. These results can be generally applied for other overlay network techniques such as Chord.

Figure 9. The fairness index of the number of the assigned data among physical nodes (random).

Figure 10. The fairness index of the number of the assigned data among physical nodes (Gaussian).

Figure 11. The fairness index of the number of the assigned data among physical nodes (Zipf’s).
Figure 12 shows the number of messages among virtual nodes for one interval query while the lateral axis represents the number of physical nodes. In Figure 12, the number of messages is basically increased by the number of physical nodes or virtual nodes because the scale of the overlay network becomes larger. On the other hand, the number of message among virtual nodes can be reduced by optimization of query routing on each physical node or among physical nodes.

![Figure 12. The number of messages among virtual nodes for one query.](image)

4. Discussion

As described in the last paragraph in Section 3.2, the optimization of query routing is effective in the proposed scheme because the number of messages among physical nodes can be reduced. In overlay networks, virtual node techniques have been researched in order to avoid load concentration into the specific physical node [25,26]. In [25,26], Shao et al. also assume “flash crowd” where many requests are concentrated to the specific service in a short time [27]. The load-concentrated node asks low-load nodes for help. The asked nodes generate virtual nodes with the same key to the load-concentrated node and add the virtual nodes into the overlay network. By this process, not only the loads to the specific node but also the loads for the keys around the specific node can be distributed. Although this paper assigns keys to virtual nodes at random in Sections 2 and 3, the proposed method can employ the same approach.

As another limitation of the proposed scheme, a specific physical node failure immediately affects the multiple virtual nodes assigned to the failure node. To enhance the reliability and feasibility of overlay networks, the related techniques have also been researched [28–38]. First, a variety of replication schemes, such as path replication, have been proposed for unstructured P2P networks [28], where nodes search for content by forwarding queries to publishing nodes via neighboring links; and the path replication schemes replicate the replied content on the nodes between the publishing and requesting nodes. Related to the path replication schemes, a number of methods have been proposed based on specific factors such as the number of queries, the probability to put replicas, churn situations, and so on [29–31]. In addition, replication schemes have also been proposed for structured P2P networks, in order to increase the efficiency of replica maintenance and searches. Scalaris, for example, an Erlang implementation of a distributed key/value store [32], 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 [33]. Here, the 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 [34], with each block having a root node that manages the metadata of replicas on other nodes in their own different data blocks. Although these techniques do not assume interval queries, the proposed scheme can employ the same or similar ideas such as node/data replication to enhance its reliability.

5. Conclusions

This paper described the enhanced scheme for structured overlay networks based on multiple different time intervals. In the previous method, unfairness and concentration of the loads occur for the specific node because the density of data or those generators/providers is different from those related key values. Therefore, the proposed scheme uses node virtualization to equalize the loads of each real (physical) node. The simulation results showed that the proposed scheme can increase the fairness of the number of the assigned data among physical nodes.

As future work, the author will study optimization techniques of query routing on each physical node or among physical nodes to reduce message forwarding among virtual nodes. In addition, node/data replication techniques are expected to enhance the reliability and feasibility because specific physical node failure immediately affects the multiple virtual nodes assigned to the failure node. After that, the author will implement the proposed scheme to P2P agent platform, PIAX [39], or testbed systems to evaluate it in practical systems.

Funding: This work was supported by JSPS KAKENHI Grant No. 18K11316, G-7 Scholarship Foundation, and Research Grants from the University of Fukui.

Conflicts of Interest: The author declares no conflict of interest.

References

1. Lin, J.; Yu, W.; Zhang, N.; Yang, X.; Zhang, H.; Zhao, W. A Survey on Internet of Things: Architecture, Enabling Technologies, Security and Privacy, and Applications. *IEEE Internet Things J.* 2017, 4, 1125–1142. [CrossRef]
2. Čolaković, A.; Hadžialić, M. Internet of Things (IoT): A Review of Enabling Technologies, Challenges, and Open Research Issues. *Comput. Netw.* 2018, 144, 17–39. [CrossRef]
3. Stoica, I.; Morris, R.; Liben-Nowell, D.; Karger, D.R.; Kaashoek, M.F.; Dabek, F.; Balakrishnan, H. Chord: A Scalable Peer-to-Peer Lookup Protocol for Internet Applications. *IEEE/ACM Trans. Netw.* 2003, 11, 17–32. [CrossRef]
4. Shu, Y.; Ooi, B.C.; Tan, K.L.; Zhou, A. 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, 31 August–2 September 2005; pp. 173–180.
5. Tetarave, S.K.; Tripathy, S.; Ghosh, R.K. V-Chord: An Efficient File Sharing on LTE/GSM Network. In Proceedings of the 19th International Conference on Distributed Computing and Networking (ICDCN 2018), Varanasi, India, 4–7 January 2018.
6. Cheklat, L.; Amad, M.; Omar, M.; Boukerram, A. Energy Efficient Physical Proximity Based Chord Protocol for Data Delivery in WSNs. In Proceedings of the 2018 International Conference on Applied Smart Systems (ICASS 2018), Médéa, Algeria, 24–25 November 2018.
7. Gu, T.; Hong, S.; Uhm, S.; Lee, K. Membership Management based on a Hierarchical Ring for Large Grid Environments. *J. Inf. Process. Syst.* 2007, 3, 8–15. [CrossRef]
8. Zhang, W.; Tang, H.; Byna, S.; Chen, Y. 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, Limassol, Cyprus, 1–4 November 2018; pp. 1–12.
9. Aspnes, J.; Shah, G. Skip Graphs. *ACM Trans. Algorithms* 2007, 3, 1–25. [CrossRef]
10. Takeuchi, S.; Shinomiya, J.; Shiraki, T.; Ishi, Y.; Teranishi, Y.; Yoshida, M.; Shimojo, S. A Large Scale Key-Value Store Based on Range-Key Skip Graph and Its Applications. In Proceedings of the 15th International Conference on Database Systems for Advanced Applications (DASFAA 2010), Tsukuba, Japan, 1–4 April 2010; Volume II, pp. 432–435.

11. Banno, R.; Fujino, T.; Takeuchi, S.; Takemoto, M. SFB: A Scalable Method for Handling Range Queries on Skip Graphs. *IEICE Commun. Express* 2015, 4, 14–19. [CrossRef]

12. Boshrooyeh, S.T.; Ozkasap, O. Guard: Secure Routing in Skip Graph. In Proceedings of the 2017 IFIP Networking Conference (IFIP Networking) and Workshops, Stockholm, Sweden, 12–16 June 2017.

13. Hassanazadeh-Nazarabadi, Y.; Ozkasap, O. 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 2017), Istanbul, Turkey, 5–8 June 2017.

14. Aoki, Y.; Ohnishi, M.; Shudo, K. Ballistic Skip Graph: A Skip Graph-Style Constant-Degree Structured Overlay. In Proceedings of the 23rd IEEE Symposium on Computers and Communications (ISCC 2018), Natal, Brazil, 25–28 June 2018.

15. Mondal, A.; Lifu, Y.; Kitsuregawa, M. P2PR-Tree: An R-Tree-Based Spatial Index for Peer-to-Peer Environments. In Proceedings of the International Workshop on Peer-to-Peer Computing and Databases in Conjunction with the 9th International Conference on Extending Database Technology (EDBT 2004), Crete, Greece, 14–18 March 2004; pp. 516–525.

16. Tanin, E.; Harwood, A.; Samet, H. Using a Distributed Quadtree Index in Peer-to-Peer Networks. *VLDB J.* 2007, 16, 165–178. [CrossRef]

17. Araujo, F.; Rodrigues, L. GeoPeer: A Location-Aware Peer-to-Peer System. In Proceedings of the 3rd IEEE International Symposium on Network Computing and Applications (NCA 2004), Cambridge, MA, USA, 30 August–1 September 2004; pp. 30–46.

18. Ohnishi, M.; Inoue, M.; Harai, H. Incremental Distributed Construction Method of Delaunay Overlay Network on Detour Overlay Paths. *J. Inf. Process.* 2013, 21, 216–224. [CrossRef]

19. Teranishi, Y.; Takeuchi, S.; Harumoto, K. HDOV: An Overlay Network for Wide Area Spatial Data Collection. In Proceedings of the 26th ACM Symposium on Applied Computing (SAC 2011), Taichung, Taiwan, 21–25 March 2011; pp. 506–513.

20. Ragab, K. An Autonomic <K, D>-Interleaving Registry Overlay Network for Efficient Ubiquities Web Services Discovery Service. *J. Inf. Process. Syst.* 2008, 4, 53–60.

21. Kawakami, T. 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-si, Korea, 19–21 February 2019; pp. 81–86.

22. Kawakami, T. Node Virtualization for Structured Overlay Networks Based on Multiple Different Time Intervals. In Proceedings of the 2020 World Congress on Information Technology Applications and Services (World IT Congress 2020), Seoul, Korea, 24–26 February 2020.

23. Nair, T.R.G.; Jayarekha, P. A Rank Based Replacement Policy for Multimedia Server Cache Using Zipf-Like Law. *J. Comput.* 2010, 2, 14–22.

24. Fan, X.; Cao, J.; Mao, H.; Wu, W.; Zhao, Y.; Xu, C. Web Access Patterns Enhancing Data Access Performance of Cooperative Caching in IMANETs. In Proceedings of the 17th IEEE International Conference on Mobile Data Management (MDM 2016), Porto, Portugal, 13–16 June 2016; pp. 50–59.

25. Shao, X.; Jibiki, M.; Teranishi, Y.; Nishinaga, N. A Virtual Replica Node-Based Flash Crowds Alleviation Method for Sensor Overlay Networks. *J. Netw. Comput. Appl.* 2016, 75, 374–384. [CrossRef]

26. Shao, X.; Jibiki, M.; Teranishi, Y.; Nishinaga, N. An Efficient Load-Balancing Mechanism for Heterogeneous Range-Queriable Cloud Storage. *Future Gener. Comput. Syst.* 2018, 78, 920–930. [CrossRef]

27. Fujita, S. Flash Crowd Absorber for P2P Video Streaming. *IEICE Trans. Inf. Syst.* 2019, 102, 261–268. [CrossRef]

28. Lv, Q.; Cao, P.; Cohen, E.; Li, K.; Shenker, S. Search and Replication in Unstructured Peer-to-Peer Networks. In Proceedings of the 16th ACM International Conference on Supercomputing (ICS 2002), New York, NY, USA, 22–26 June 2002; pp. 84–95.

29. Gopalakrishnan, V.; Silaghi, B.; Bhattacharjee, B.; Keleher, P. Adaptive Replication in Peer-to-Peer Systems. In Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS 2004), Tokyo, Japan, 24–26 March 2004; pp. 360–369.
30. Yamamoto, H.; Maruta, D.; Oie, Y. Replication Methods for Load Balancing on Distributed Storages in P2P Networks. *IEICE Trans. Inf. Syst.* 2006, 89, 171–180. [CrossRef]

31. Puttaswamy, K.P.N.; Sala, A.; Zhao, B.Y. Searching for Rare Objects Using Index Replication. In Proceedings of the 27th IEEE International Conference on Computer Communications (INFOCOM 2008), Phoenix, AZ, USA, 15–17 April 2008; pp. 1723–1731.

32. Schütt, T.; Schintke, F.; Reinefeld, A. Scalaris: Reliable Transactional P2P Key/Value Store. In Proceedings of the 7th ACM SIGPLAN Workshop on ERLANG (ERLANG 2008), Victoria, BC, Canada, 27 September 2008; pp. 41–48.

33. Shen, H.; Zhu, Y. A Proactive Low-Overhead File Replication Scheme for Structured P2P Content Delivery Networks. *J. Parallel Distrib. Comput.* 2009, 69, 429–440. [CrossRef]

34. Legtchenko, S.; Monnet, S.; Sens, P.; Muller, G. RelaxDHT: A Churn-Resilient Replication Strategy for Peer-to-Peer Distributed Hash-Tables. *ACM Trans. Auton. Adapt. Syst.* 2012, 7, 1–18. [CrossRef]

35. Kim, H.; Kang, S.; Yeom, H.Y. Server Selection Schemes Considering Node Status For a Fault-Tolerant Streaming Service on a Peer-to-Peer Network. *J. Inf. Process. Syst.* 2006, 2, 6–12. [CrossRef]

36. Cao, Q.; Fujita, S. Cost-Effective Replication Schemes for Query Load Balancing in DHT-Based Peer-to-Peer File Searches. *J. Inf. Process. Syst.* 2014, 10, 628–645. [CrossRef]

37. Zahid, S.; Abid, S.A.; Shah, N.; Naqvi, S.H.A.; Mehmood, W. Distributed Partition Detection With Dynamic Replication Management in a DHT-Based MANET. *IEEE Access* 2018, 6, 18731–18746. [CrossRef]

38. Hassanzadeh-Nazarabadi, Y.; Küpü, A.; Özkasap, O. Decentralized and Locality Aware Replication Method for DHT-Based P2P Storage Systems. *Future Gener. Comput. Syst.* 2018, 84, 32–46. [CrossRef]

39. Teranishi, Y. PIAX: Toward a Framework for Sensor Overlay Network. In Proceedings of the 6th Annual IEEE Consumer Communications and Networking Conference (CCNC 2009), Las Vegas, NV, USA, 10–13 January 2009; pp. 1–5.

**Publisher’s Note:** MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

© 2020 by the author. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).