A P2P Algorithm for a Smart Information System

Agostino Forestiero(✉) and Giuseppe Papuzzo

CNR - ICAR, Institute for High Performance Computing and Networking,
Via P. Bucci 7/11 C, Rende, CS, Italy
{forestiero,papuzzo}@icar.cnr.it

Abstract. Last generation applications, often, generate a huge amount of data also characterized by high variety and velocity, like those Internet of Things (IoT) paradigm based. Big data analytics increasingly drive decision making in several fields and, then, new approaches and methodologies are needed to improve management operations. Mobile, social and ubiquitous services, comprising data sources and middleware, can be considered a dynamic content, and an effective solution to manage contents are the Content Delivery Networks (CDNs). However, CDNs show limits in dynamic and large systems in which a large amount of data is managed. To deal with these weaknesses and exploit their benefits, new algorithms and approaches have to be designed and employed. This paper introduces *RadixOne*, a distributed and self-organizing algorithm, to build an information system in pervasive and dynamic environments. Thanks to autonomous and local operations performed by hosts of a distributed system, a logically organized overlay network emerges and the resource discovery operations faster and efficient. Preliminary experimental results show the effectiveness of our approach.

Keywords: Distributed algorithm · Self-organization
Overlay network · Content delivery networks

1 Introduction

In the last years, the number of objects connected to Internet is steadily growing and large amounts of data are generated, especially due to the explosion of the Internet of Things paradigm. Health, environment, traffic, vehicles, aviation, manufacturing, defense, home automation, communication are some examples of the application domains that use IoT technologies. New models and innovative approaches are needed to investigate data characterized by high velocity, volume and variability. Common big data applications allow users and devices to upload data to data centers and share them with other users or devices in real time. The Internet backbone needs to forward massive distributed data to data centers with high throughput, and deliver processed data to users from data centers with low latency. As such, high-performance end-to-end links are required for uploading,

Á. Rocha et al. (Eds.): WorldCIST′18 2018, AISC 745, pp. 671–677, 2018.
https://doi.org/10.1007/978-3-319-77703-0_67

and efficient content distribution networks are required for downloading [8]. Content delivery networks (CDNs) is one of the successful virtual networks rapidly developed over the last decade with the specific advantage of optimizing the Internet. CDNs use edge servers to cache contents, achieving better web browsing experience with lower latency and maintaining data consistency among them. Nowadays, the CDNs has become one of the most important parts of the Internet architecture for content distribution [9]. IoT, mobile and social services, comprising data sources and middleware, can be considered a dynamic content. It takes inputs from devices (sensors) and responses to requests from machine-to-machine applications after middleware processing the requests [4]. A centralized mechanism to provide contents and IoT services when requested, is necessary. CDNs with limited size can be managed with a centralized approach, but in large and dynamic systems, such as IoT environment, their limits emerge. Thanks to their inherent scalability and robustness, peer-to-peer approaches can be useful exploited to design algorithm and protocol to manage and discovery resources in distributed systems [3] and also in CDNs [5]. The aim of these approaches is to rapidly locate the server that stores given contents. The content stored in a server can be represented through metadata, that is a syntactical description of the content and/or an ontology description. In peer to peer systems, metadata are often indexed through bit vectors, or *keys*, which can have different meanings. One is that each bit represents the presence or absence of a given *topic* [2,7]. This method is suitable for contents like documents, because it is possible to identify the different topics existing in the documents. Otherwise, a metadata can be mapped through a hash function into a binary vector. The hash function has to be designed locality preserving [1,6], thus, neighbor vectors are assigned to contents with neighbor/similar characteristics. In this paper, a decentralized algorithm that exploit the concepts above mentioned and build a P2P information system in IoT environment, namely *RadixOne*, is proposed. The algorithm organizes logically the CDN servers with the aim of improving the rapidity and effectiveness of discovery operations. Each CDN server autonomously executes simple operations, an ordered overlay network emerges at global level and the discovery operations become efficient and faster. In the following, in Sect. 2 the algorithm is detailed while in Sect. 3, some preliminary results to show the effectiveness of our approach, are illustrated.

2 *RadixOne*: A Self-organizing Algorithm to Build an Information System in Dynamic Environments

The main objective is to build a overlay network logically ordered of CDN servers in order to improve discovery operations of contents, such as services or data, in pervasive and dynamic environments. To achieve this aim, each CDN server autonomously executes simple operations, based on local information, and a logical sorting at global level emerges. The servers are represented through binary keys (i.e. bit vectors) obtained by applying of a locality preserving hash function to its own content. In this way, by defining of a metric, it is possible realize

a sorting among servers based on their binary key. Once all servers are ordered, the discovery operations of services or contents become faster. A list of servers logically ordered means that each server is logically linked to only other two servers, the server having the binary key value immediately lower and the server having the binary key value immediately higher. Obviously, the server having the binary key with the absolute minimum/maximum value of all binary keys of the network, has a unique linked server. Algorithm 1 reports the sequence of the operations performed by each server in order to generate the overlay. In Figs. 1(a) and 1(b) an example of how the algorithm works, is reported.

```
while true do
    compute Hlist: the list of all linked servers having value of binary key higher
     than the current server;
    compute Llist: the list of all linked servers having value of binary key lower
     than the current server;
    while Llist.length() > 1 do
        identify, in Llist, the servers having the minimum value and the
         sub-minimum value of binary key;
        create a virtual link between them;
        notify both the servers about their new virtual neighbors;
        remove from Llist the server having the minimum value of binary key;
    end
    while Hlist.length() > 1 do
        identify, in Hlist, the servers having the maximum value and the
         sub-maximum value of the binary key;
        create a virtual link between them;
        notify both the servers about their new virtual neighbors;
        remove from Hlist the server having the maximum value of binary key;
    end
end
```

Algorithm 1. Pseudo-code of the algorithm performed by each server.

Figure 1(a) shows an example of a CDN network with the related bit vector for each server. For the sake of simplicity, in round bracket is reported the decimal value of bit vector. In Fig. 1(b) the emerging overlay network. In particular, the dot lines represent the real links, whereas the solid lines indicate the virtual links generate by the algorithm. Notice that, at the end of the operations, each server is linked, through a real or virtual link, to two servers: the server with the value of bit vector immediately lower than its own and the server with the value of binary vector immediately higher than its own, of the all network. Obviously, the servers with the highest/lowest value of bit vector has only one link. In the emerging overlay, the servers are ordered and then the discovery procedure becomes informed and faster. Moreover, the overlay spontaneously adapts to the joins and departs of servers and to the changing of the characteristics of the contents. In particular, when a server leaves the network it must notify to

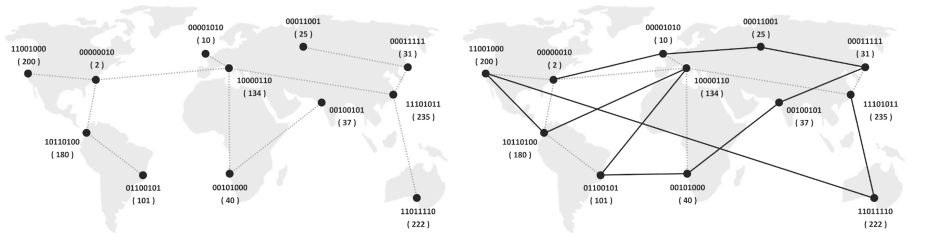

Fig. 1. (a) An example of CDN network. For each server its own bit vector and the corresponding decimal value, in round bracket, are reported; and (b) the outcome of the algorithm in a CDN network. The solid lines represent the virtual links of the overlay network generated thanks to the algorithm.

its own linked servers of the overlay network the information each other. While when a new server want to join to the network, it must simply communicate its arrival to the neighbors, by exchanging the bit vectors, and successively executes the algorithm. Automatically it will be involved in the logical organization.

2.1 A Discovery Mechanism to Exploit the Emerging Overlay Network

Thanks to the ordering performed by the algorithm, a simple and intuitive search mechanism can be designed. The discovery algorithm exploits an *informed* mechanism, since queries are delivered towards servers with bit vector describing the content more and more similar to the target content. Moreover, also for the discovery procedure, a centralized service does not exist, it is fully distributed and the messages are forwarded only on the basis of local information. The ordering of the servers achieved by algorithm guarantees to individuate the target content. When a query is issued by a user for a "target content", a search procedure for a server having the bit vector as more as close to the bit vector representing the target content, starts. Thanks to the logical sorting achieved by the algorithm, the query can be simply forwards, at each step, towards the "best neighbor server". The best neighbor server is the linked server, belonging to the overlay o real network, with the minimum distance from the target bit vector. This procedure guarantees that, at each step, an approaching to the target content is achieved. When no bit vector of the neighbor servers is closer than the bit vector of the local server, it is not advantageous forward the query: the search procedure finishes. At this point, a message is issued towards the requesting server/user with the location of the target content. Table 1 reports the path performed by a user request from the CDN server with bit vector [1100101] (decimal value equals 101) for a content located in the CDN server with bit vector [11011110] (decimal value equals 222). The query is issued step by step towards the neighbor server with the bit vector having the minimum distance

Table 1. An example of path performed by a user request from server with bit vector [1100101] (decimal value equals 101) for content located in server with bit vector [11011110] (decimal value equals 222).

STEP	Local server	Selectable linked server	Distance from target	Selected server
1	101	(40, 134, 180)	(182, 88, 42)	180
2	180	(2, 101, 134, 200)	(220, 121, 88, 22)	200
3	200	(2, 180, 222)	(220, 42, 0)	222 (Target)

between itself and the target bit vector. It can be noticed that the number of steps to get to the target server is very limited.

3 Experimental Results

In order to investigate the effectiveness of the algorithm, a Java simulator was implemented in which the characteristics of real CDNs were careful considered. Initial simulations were executed to evaluate the traffic generated by the algorithm, that is the average number of messages managed by servers to achieve a stable logical organization. Figure 2(a) shows the mean number of messages managed by each server for different CDN network size, that is for different number of CDN serves involved in the logical reorganization. The experiments were executed for different values of $AvgNears$, that is the average number of real connections/neighbors of each server. It can be noticed that, with a limited number of messages, the algorithm reaches a stable situation and an overlay is obtained. In Fig. 2(b) the total number of messages exchanged by all servers per

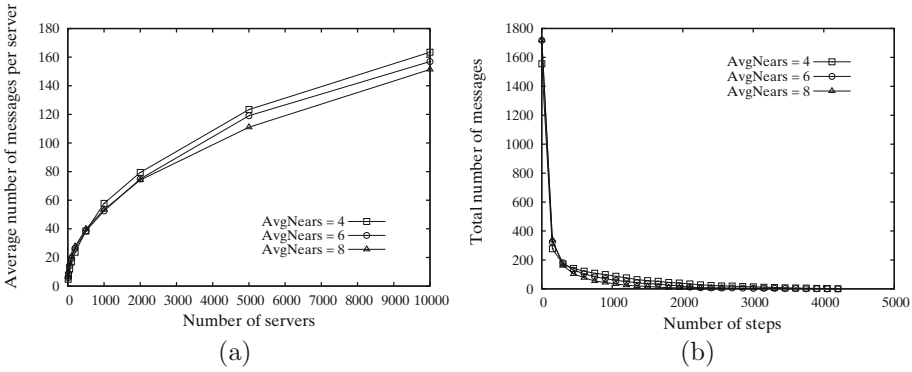

Fig. 2. (a) The average number of messages managed by each servers to obtain the logical sorting, for different values of network size; and (b) the total number of messages exchanged by all servers per step to obtain the sorting in a network having 5000 servers.

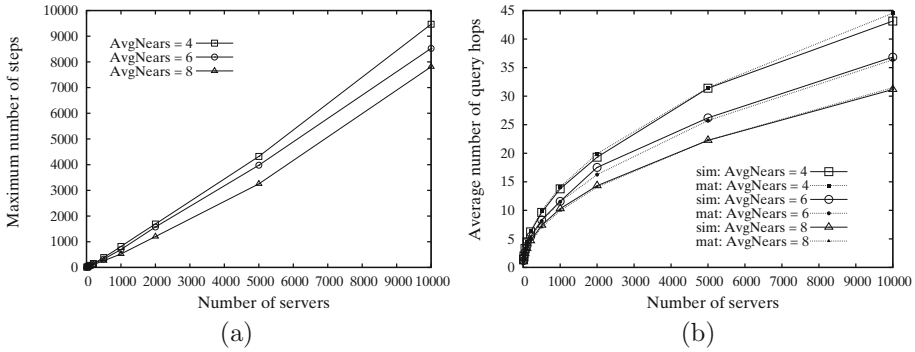

Fig. 3. (a) The maximum value of possible steps (worst case) necessary to obtain the logical sorting; and (b) the average number of query hops performed by the algorithm to locate the target server. Simulation results, with solid lines, and mathematical results, with dot lines.

step to achieve the global logical organization in a network with 5000 servers, was showed. It is possible to note that the number of messages decreases exponentially and the algorithm converges in a finite number of steps. Successively, the worst case to obtain the organization, i.e. the maximum possible number of steps needed to each server to individuate its own neighbors, was calculated and showed in Fig. 3(a). Even in this case, the experiments were executed for different values of the average number of connections/neighbors of each server, when the network size changes. The interesting result is that the maximum number of steps needed to each server to individuate its own neighbors is always very low and limited to first application of the algorithm in which no previous sorting exists. To prove the usefulness of the overlay structure achieved by the algorithm, a further set of experiments was performed. Figure 3(b) shows - with the solid lines - the average number of steps performed by a query to individuate the CDN server having the desired content, N_{hops}. The experiments were performed for different network sizes and for different average number of connected servers (neighbors) to each CDN server. We can see that a limited number of steps is sufficient to reach the target server even when the network size grows. With the aim of comparison, in Fig. 3(b) is also reported - with the dot lines - the curves obtained through a mathematical expression reported in formula 1.

$$N_{hops} = \sqrt{\frac{N_{servers}}{1.26 * AvgNears}} \tag{1}$$

The mathematical expression was derived empirically from the experiments performed. It is possible to remark how both values are practically equals. In formula 1, $N_{servers}$ represents the number of CDN servers, and it can be noticed how the mean number of query hops performed by a user request is directly related to the square root of the ratio between the network size and the average number of neighbors.

4 Conclusions

In this paper a distributed algorithm to build an information system to improve discovery operations in pervasive and dynamic environments, was proposed. Each servers containing the data is indexed through a bit vector obtained by exploiting of a hash function locality preserving to the metadata representing the content. Thanks to simple operations autonomously performed by each server, a sorted overlay network emerges. In this way, the path of the user requests to reach the desired target content or services becomes informed and faster. The experimental results proved the effectiveness of the algorithm and how the discovery operations are proportional to square root of network size.

References

1. Cai, M., Frank, M., Chen, J., Szekely, P.: Maan: a multi-attribute addressable network for grid information services. J. Grid Comput. **2**(1), 3–14 (2004)
2. Crespo, A., Garcia-Molina, H.: Routing indices for peer-to-peer systems. In: Proceedings of the 22nd International Conference on Distributed Computing Systems ICDCS 2002, pp. 23–33 (2002)
3. Forestiero, A., Mastroianni, C., Spezzano, G.: Building a peer-to-peer information system in grids via self-organizing agents. J. Grid Comput. **6**(2), 125–140 (2008)
4. Li, J., Shvartzshnaider, Y., Francisco, J.A., Martin, R.P., Nagaraja, K., Raychaudhuri, D.: Delivering internet-of-things services in mobility first future internet architecture. In: Proceedings of the 3rd International Conference on the Internet of Things (IOT), pp. 31–38. IEEE (2012)
5. Lu, Z., Wang, Y., Yang, Y.R.: An analysis and comparison of cdn-p2p-hybrid content delivery system and model. J. Commun. **7**(3), 232–245 (2012)
6. Oppenheimer, D., Albrecht, J., Patterson, D., Vahdat, A.: Design and implementation tradeoffs for wide-area resource discovery. In: Proceedings of the 14th IEEE International Symposium on High Performance Distributed Computing. Research Triangle Park, NC, USA (2005)
7. Platzer, C., Dustdar, S.: A vector space search engine for web services. In: Proceedings of the Third European Conference on Web Services ECOWS 2005, p. 62, Washington, DC, USA. IEEE Computer Society (2005)
8. Xiaomeng, Y., Liu, F., Jiangchuan, L., Hai, J.: Building a network highway for big data: architecture and challenges. IEEE Netw. **28**(4), 5–13 (2014)
9. Yin, H., Liu, X., Zhan, T., Sekar, V., Qiu, F., Lin, C., Zhang, H., Li, B.: Design and deployment of a hybrid CDN-p2p system for live video streaming: experiences with livesky. In: Proceedings of the 17th ACM International Conference on Multimedia MM 2009, pp. 25–34. (2009)