Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

Replica-aware, multi-dimensional range queries in Distributed Hash Tables

Antony Chazapis *, Athanasia Asiki, Georgios Tsoukalas, Dimitrios Tsoumakos, Nectarios Koziris

Computing Systems Laboratory, School of Electrical and Computer Engineering, National Technical University of Athens, Greece

A R T I C L E I N F O

Article history:
Received 7 May 2009
Received in revised form 4 September 2009
Accepted 26 January 2010
Available online 1 February 2010

Keywords:
Peer-to-peer systems
Distributed Hash Tables
Replication
Multi-dimensional data
Range queries

A B S T R A C T

In this paper, we present and evaluate a protocol that enables fast and accurate range-query execution in Distributed Hash Tables (DHTs). Range queries are of particular importance when the network is populated with groups or collections of data items, whose respective identifiers are generated in a way that encodes semantic relationships into key distances. Contrary to related work in the same direction, our proposed query engine is aware of data replicas at the DHT level and by grouping related nodes into replica *neighborhoods*, resolves queries with the minimum amount of messaging overhead. Moreover, we suggest pairing respective operations with the core DHT routing mechanics, which allows for reusing existing management and monitoring structures and automatically adapting the query path to the dynamic characteristics of the overlay. We also present an application scenario and the respective deployment details of a prototype implementation in the context of the *Gredia* project.

1. Introduction

Peer-to-peer computing has been widely accepted as a robust, easily deployable paradigm on which fully distributed, scalable applications can be built. Respective protocols focus on defining the transactions between autonomous network endpoints that collectively form an extensible and resilient substrate – a *network overlay*, which can operate unsupervised and dynamically adjust itself to unadvertised node arrivals and departures, or sudden network black-outs. *Structured* systems, also commonly known as *Distributed Hash Tables* (DHTs), represent a major class of peer-to-peer arrangements that abstract the aggregate network as a key-value based indexing and storage facility. All available designs, share the common concept of *hashing* both peers and data values to identifiers that represent slots of a pre-defined virtual fabric. In the identifier space, each node takes on the responsibility of storing values and managing operations that refer to data with IDs "close" to its own. DHT implementations are designated by the shape of the identifier space used and consequently the corresponding distance function [1]. Kademlia [2] uses an XOR metric, which implies placement at the leaves of a binary tree, while Chord [3] organizes all IDs clockwise around a circle.

DHTs pose as perfect candidates for replacing "traditional" centralized or layered global-scale databases of information that can be easily transformed to a key-value form (i.e. DNS records [4]). Their simple *put*/*get* interface allows the direct application of traditional programming models to a global scale. Nevertheless, their applicability is limited, due to the lack of inherent support for processing related subsets, or *ranges* of data identifiers. Expressing *range queries* in multiple simpler primitive lookups for all intermediate IDs is not scalable. Ideally, such operations should be internally handled by the overlay, while both respecting and sustaining the scalability and fault-tolerance mechanics that have made DHTs a prominent platform for supporting distributed applications.

This also reflects a common focus of many recent research initiatives, that propose *advanced* or *complex* DHT lookup functions – functions which exploit semantic relationships between data items, including similar or nearest neighbor searching or processing aggregate sets of closely related items. As group operations require data ordering, the hash function for placing values into the identifier space is usually replaced with a locality-preserving mapping scheme. When the data to be distributed is already tied to one or more indices, such a scheme can transform their numeric representations to a single identifier, while maintaining their original organizational and clustering properties. Example mapping functions for *multi-dimensional* values may be based on the *Z* or Hilbert *space-filling curves* (SFCs).

Motivated to implement a DHT capable of performing range queries on multi-dimensional data, we have investigated numerous systems presented in related work and found that most exhibit shortcomings that originate from their two-level architecture, choice of range-query resolution algorithms, or both. Using separate layers for query handling and routing complicates operations,

* Corresponding author. Tel.: +30 210 7722867; fax: +30 210 7721292.
E-mail addresses: chazapis@cslab.ece.ntua.gr (A. Chazapis), nasia@cslab.ece.ntua.gr (A. Asiki), gtsouk@cslab.ece.ntua.gr (G. Tsoukalas), dtsouma@cslab.ece.ntua.gr (D. Tsoumakos), nkoziris@cslab.ece.ntua.gr (N. Koziris).

0140-3664/$ - see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.comcom.2010.01.024

as each layer's data structures must be properly maintained and synchronized to propagate new values or node membership changes. Additionally, the process of translating multi-dimensional ranges to DHT identifier segments of the routing layer may be a very demanding processing task. A common trend to avoid the processing load is to distribute it among the nodes of the DHT, via a *query refinement* process. Such systems match the recursive translation algorithm to a tree-like ordering of peers. This may incur heavy loads at the nodes placed at the higher levels of the overlay's structure and by design does not solve the problem of computing the ID segments corresponding to each multi-dimensional range request. Furthermore, to our knowledge, no range-query capable system is compatible with DHT protocols that inherently handle data replication. While pioneering DHTs could only ensure data reliability via well-defined node disconnect procedures, modern overlays self-replicate each stored value to a dynamic group of close peers. Related multi-dimensional query platforms usually build upon a simple DHT substrate, optimized for routing, and delegate data management and replication to a separate layer. However, when such copies are already available in the overlay, the query engine should account for them accordingly.

In this paper, we present the necessary protocol enhancements that enable a DHT to handle range queries. We do not propose a new peer-to-peer architecture, but rather exploit existing internal structures to introduce a novel routing strategy for looking-up ranges of identifiers. In contrast to other designs, we do not employ a recursive multi-dimensional range to segment conversion, but apply a serial, segment-to-segment hopping algorithm that in addition to its improved efficiency allows the query process to be easily distributed and parallelized. Analogous query resolution mechanisms have been extensively used in database-related contexts, but not in DHTs.

Moreover, by working directly at the DHT level, we achieve several benefits. First, range queries self-manage the path taken throughout the network and avoid visiting nodes which would otherwise reply with duplicate values – *replica awareness* has been an important challenge throughout the design process. To this end, we introduce the notion of *peer neighborhoods*, as groups of peers that store the same replica set. Neighborhoods interact directly with the query engine and serve as a tool to help us estimate and measure the performance characteristics of the network, as well as balance the load over peers of the same group. Inherent DHT replication is not considered an obstacle, but is exploited in favor of faster query resolution. Second, the resulting DHT requires no extra management to support range queries. Since we do not introduce any new routing tables or other internal structures, there is no need for additional continuous or periodic maintenance. Therefore, we stress the importance of embedding range-query support at the core of the protocol. Implementing advanced functions at the lowest level, allows us to exploit the underlying DHT mechanics and let the query automatically adjust its execution depending on the ongoing node and data-placement relationships in the overlay. This degree of integration is also one of the main contributions of this work.

The remainder of this paper is organized as follows: In the next section we discuss on related work in the area. Section 3 introduces our protocol starting with the single-dimensional case, before moving on to multiple dimensions. In Section 4 we present both a formal and an empirical approach to estimate the performance of the proposed algorithm, show evaluation results and elaborate on its behavior under different overlay runtime parameters and data distributions. We then describe the scenario that has motivated us to deal with the problem, the final implementation and deployment details and conclude. A working prototype has been incorporated in the EU-funded *Gredia* project [5] infrastructure to provide an efficient and scalable solution for metadata search.

2. Related work

DHT overlays [2,3,6–8] have been established as an effective solution for data placement and *exact match* query routing in scalable network infrastructures. In respective protocols, values can only be located if the their exact keys are known in advance, while keys – randomly generated by a cryptographic hash function – do not contain any semantic information about the content. As applications demand more complex types of queries, capable of exploiting inter-relationships between data, research efforts have focused on developing corresponding algorithms and mechanisms. Resolving range queries in a peer-to-peer network requires the ordering of values, which comes in conflict with the random assignment of items to nodes. The methodology used to solve this problem, provides a basic categorization of proposed systems encountered in the bibliography:

- Overlays that rely on an existing DHT protocol and either modify/replace the hashing function or add additional indexing structures on top of the DHT.
- Overlays that distribute the dataset to peers, while not directly utilizing any existing DHT protocol.

In the rest of the section, we elaborate on related work from the field of structured peer-to-peer overlays, starting from systems supporting range queries for data items described by a single attribute, before proceeding to platforms that support multi-attribute range operations.

An early approach to support range queries in a DHT overlay is presented by Gupta et al. [9]. The authors store partitions of a relational database in a DHT and enable querying ranges one attribute at a time, by mapping them to identifiers via a special hash function. Nevertheless, hashing ranges is not efficient, as it is possible that similar or overlapping spaces are mapped to different nodes. To avoid flooding the network or following various distributed indices to reach the requested data, they introduce *Locality Sensitive Hashing*, so mappings preserve locality with high probability. However, the probabilistic scheme returns approximate results, whose quality depends on the complexity of the function. Additionally, there are load-balancing issues. Another method of adjusting hashing to enable range queries is described by Andrzejak and Xu [10], where the Hilbert space-filling curve is used for mapping ranges of an attribute's values to parts of a CAN-based overlay. The SFC offers the advantage that nearby attribute ranges are mapped to nearby CAN zones. A query is initially routed to the node responsible for the middle point of the range and then recursively propagated according to three proposed and evaluated 'flooding' strategies.

A common solution for the execution of range queries is the exploitation of trees as additional indexing structures on top of DHT-based, independent routing substrates. Gao and Steenkiste [11] implement a Range Search Tree (RST) on top of a DHT overlay, such as Chord. Each level of the tree corresponds to a different granularity of data partitioning, while content is registered with all or various levels of the RST and the corresponding physical nodes of the DHT. Distributed Segment Trees [12] are similar. P-Trees [13] are also decentralized indexing structures, maintaining parts of semi-independent B+ trees on top of a Chord ring. Their purpose is to help route range queries to the appropriate nodes. Because of complex cross-structure management, the authors note that every query cannot be guaranteed to terminate – for example, if a node crashes. Other efforts target the distribution of *tries*. Tries are a generalization of trees for storing and processing strings in which there is a node for every common prefix. In tries, the actual data is stored in the leaf nodes and thus lookups are resolved by finding the leaf whose label is a prefix of the queried value. The