Geodemlia: Persistent Storage and Reliable Search for Peer-to-Peer Location-Based Services

Christian Gross⋆, Björn Richerzhagen⋆, Dominik Stingl⋆, Christoph Münker⋆, David Hausheer†, Ralf Steinmetz⋆
⋆Multimedia Communications Lab, †Peer-to-Peer Systems Engineering,
Technische Universität Darmstadt, Germany
Email: {chrgross, richerzhagen, stingl, cmuenker, steinmetz}@kom.tu-darmstadt.de
{hausheer}@ps.tu-darmstadt.de

***Index Terms*—location-based search, overlay, replication, churn, peer-to-peer**

I. Introduction

Location-based services have become increasingly popular in the recent years due to the vast deployment of position-aware devices such as smartphones and tablet PCs and the ubiquitous availability of fast Internet connectivity. Existing location-based services are realized as cloud services, which cause considerably high costs. Furthermore, they are not location-aware leading to unnecessary long transmission paths between the users and the cloud infrastructure. The concept of Peer-to-Peer has proven to be a valid alternative for realizing the functionality of location-based services, which resulted in a plethora of approaches for location-based search [1], [4], [5]. Existing concepts, however, suffer from two major drawbacks: (i) they are not robust against high peer churn and (ii) they do not allow for the persistent storage of location-based data. To this end, in this demo we present the prototype of the overlay Geodemlia [3], which allows for both: the persistent storage of location-based information as well as the reliable search even under high churn rates. Location-based information in Geodemlia is stored in a location-aware way, reducing the length of the transmission path for store and search operations.

II. Geodemlia

During the design of Geodemlia the following assumptions have been made: (i) the world is modeled as a sphere and each peer is located at exactly one point l_p on that sphere. (ii) Each peer is able to determine its own location $l_p = (\phi, \psi)$, with longitude $\phi \in [-180°, 180°]$ and latitude $\psi \in [-90°, 90°]$ using well known localization techniques such as WiFi footprints or GPS. (iii) For calculating the distance $d(l_p, l_q)$ between peers p and q on the sphere, the well known Haversine formula [6] is used. (iv) For the underlay we assume that all peers are interconnected via a TCP/IP network such that two arbitrarily peers are able to communicate with each other as long as they know each others IP address and port. (v) All location-based data object o generated by peers are bound to a fixed location l_o and are tagged with a set of search terms s. (vi) Peers in the system execute area search queries that have a circular shape and are parameterized by a central point l_s and a radius r. (vii) Finally, all peers and data objects in the system are identified by a unique identifier $i \in [0, 2^{160-1}]$.

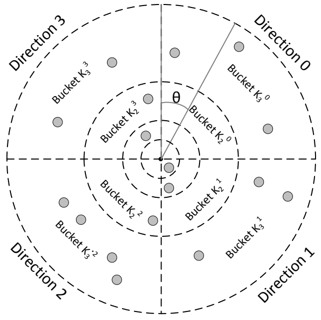


Figure 1. Routing table structure of a Geodemlia peer.

The idea behind Geodemlia is that location-based data objects are stored on the k geographically closest peers surrounding that object. To determine the set of the k closest peers for a given location-based data object, peers maintain a routing table with a structure as shown in Figure 1. The table is organized in direction segments j based on the bearing angle θ, which are further divided into exponentially increasing distance buckets K_i^j. Based on the above assumptions, Geodemlia provides the following functional primitives:

(i) FIND_NODES(l_s, k, b) allows to search for the k closest peers for a given search location l_s by using an iterative routing scheme. Therefore, a requesting peer queries its routing table for the k closest contacts and sends out α parallel search requests. Peers receiving the request also query their routing table and reply with the k contacts closest to the given query location l_s. The requesting peer p then merges each received contact c into a list, which is sorted by the distance $d(l_c, l_s)$ to the search location l_s. Afterwards, it checks, whether the top k contacts in the list have changed since the last iteration and whether all k node have already been queried. If a node has not been queried yet, peer p sends another search request. The procedure continues until the set of k closest nodes has converged with all k nodes being queried. To avoid receiving already known contacts twice, the requesting peer attaches a Bloom filter b to all its requests, which is computed based on the IDs of already received contacts.

(ii) STORE(o, l_o) allows to store a given location-based data object o with location l_o persistently into the overlay.

This work has been [co-]funded by the German Research Foundation (DFG) in the Collaborative Research Center (SFB) 1053 “MAKI – Multi-Mechanism-Adaptation for the Future Internet”.

Therefore, FIND_NODES(l_s, k, b) determines the set of k closest nodes, which then store the data object o.

(iii) AREA_SEARCH(l_s, r, s, b) allows to search for all stored data objects o that match the search term s and are located within the search area specified by the location l_s and the radius r. The method works similar to FIND_NODES(l_s, k, b), but in contrast, receiving nodes do not only return contacts close to the search location, but also check their internal storage for stored objects that match the search criteria. Furthermore, the querying node sends out requests to all contacts in his sorted list that fall into or are close to the search area.

III. Architecture

The architecture of Geodemlia is organized in modules as shown in Figure 2. Geodemlia is implemented in the discrete event-based overlay network simulator PeerfactSim.KOM [7]. The simulator comes with a network abstraction layer, which allows for the execution of Geodemlia in a simulation mode using a network model as well as in a prototype mode over a real network executing the *same* source code of Geodemlia. This increases the comparability of results obtained in simulations and testbed experiments. The functionality of the overlay is realized by a set of components. This comprises components for the (i) routing table, (ii) topology maintenance, (iii) message routing, (iv) replication, and (v) storage. Finally, Geodemlia includes a component for the fast up- and downloading of larger multimedia files in BitTorrent-like way termed GeoSwarm [2]. The overlay provides the interface for storing and searching of location-based information as explained in Section II. The interface is currently used by two different workload models: (i) a Twitter-like workload model, which was presented in [3] comprising data of 220.000 Twitter users, and (ii) a random workload model, which generates random store and search requests.

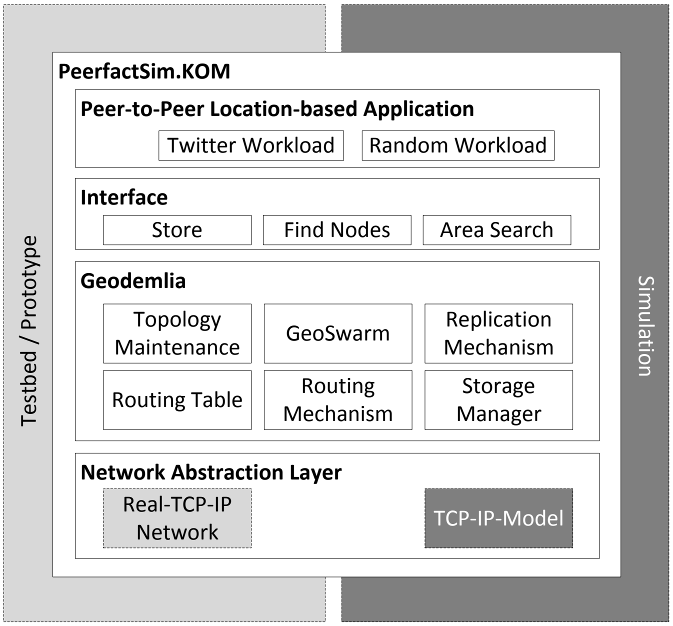
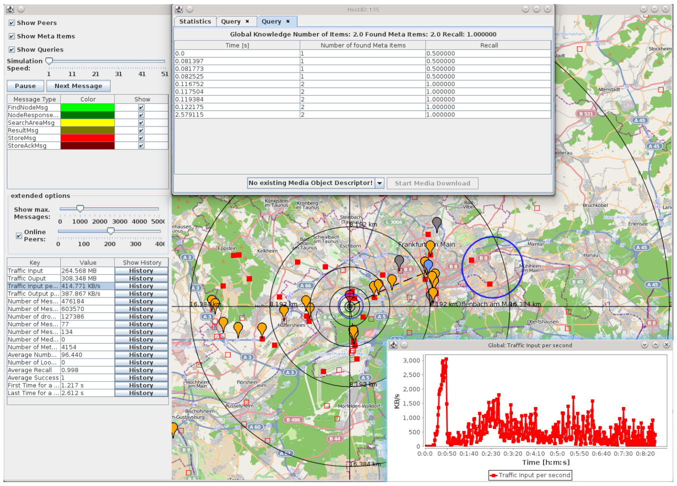


Figure 2. Architecture of Geodemlia.



Message Type	Color	Show
FindNodeMsg	🟩	☑
NodeResponse...	🟩	☑
SearchAreaMsg	🟨	☑
ResultMsg	🟫	☑
StoreMsg	🟥	☑
StoreAckMsg	🟥	☑

Key	Value	Show History
Traffic Input	264.568 MB	History
Traffic Ouput	308.348 MB	History
Traffic Input pe...	414.771 KB/s	History
Traffic Output p...	387.867 KB/s	History
Number of Mes...	476184	History
Number of Mes...	603570	History
Number of dro...	127386	History
Number of Mes...	77	History
Number of Mes...	134	History
Number of Med...	0	History
Number of Met...	4154	History
Average Numb...	96.440	History
Number of Loo...	0	History
Average Recall	0.998	History
Average Success	1	History
First Time for a ...	1.217 s	History
Last Time for a ...	2.612 s	History

Global Knowledge Number of Items: 2.0 Found Meta Items: 2.0 Recall: 1.000000

Time [s]	Number of found Meta Items	Recall
0.0	1	0.500000
0.081397	1	0.500000
0.081773	1	0.500000
0.082525	1	0.500000
0.116752	2	1.000000
0.117504	2	1.000000
0.119384	2	1.000000
0.122175	2	1.000000
2.579115	2	1.000000

Figure 3. Screenshot of the Geodemlia visualization.

IV. Demonstration Scenario

Figure 3 shows a screenshot of the Geodemlia demo running in simulation mode. The distribution of peers and published location-based data items is visualized using OpenStreet-Map data. The dashed lines visualize messages exchanged between peers. The goal of this demo is to demonstrate Geodemlia's reliable search and persistent storage under various levels of churn. Therefore, the GUI allows for the adjustment of the churn level during the simulation. Peers either join and leave the overlay automatically based on a predefined scenario description or can be added and remove to the simulation manually by hand. Store and search requests are generated automatically using a workload generator, which is based on traces of Twitter [3]. In addition, the user can manually insert store and search requests via the GUI. To ease the analysis of the behavior of the overlay, the GUI offers various options to further customize the visualization of the overlay and to plot system-wide metrics as well as the performance and cost observed by single peers.

References

[1] Y. Chawathe, S. Ramabhadran, S. Ratnasamy, A. LaMarca, S. Shenker, and J. Hellerstein, "A Case Study in Building Layered DHT Applications," *ACM SIGCOMM Computer Communication Review*, vol. 35, no. 4, 2005.

[2] C. Groß, B. Richerzhagen, D. Stingl, J. Weber, D. Hausheer, and R. Steinmetz, "GeoSwarm: A Multi-Source Download Scheme for Peer-to-Peer Location-based Services," in *International Conference on Peer-to-Peer Computing (P2P)*. IEEE, 2013.

[3] C. Groß, D. Stingl, B. Richerzhagen, A. Hemel, R. Steinmetz, and D. Hausheer, "Geodemlia: A Robust Peer-to-Peer Overlay Supporting Location-Based Search," in *International Conference on Peer-to-Peer Computing (P2P)*, 2012.

[4] V. Kantere, S. Skiadopoulos, and T. Sellis, "Storing and Indexing Spatial Data in P2P Systems," *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 2, pp. 287–300, 2009.

[5] A. Kovacevic, N. Liebau, and R. Steinmetz, "Globase.KOM - A P2P Overlay for Fully Retrievable Location-Based Search," in *International Conference on Peer-to-Peer Computing (P2P)*. IEEE, 2007.

[6] R. Sinnott, "Virtues of the Haversine," *Sky and Telescope*, vol. 68, p. 158, 1984.

[7] D. Stingl, C. Groß, J. Rückert, L. Nobach, A. Kovacevic, and R. Steinmetz, "PeerfactSim.KOM: A Simulation Framework for Peer-to-Peer Systems," in *International Conference on High Performance Computing & Simulation (HPCS)*. IEEE, 2011.