Future Generation Computer Systems ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Contents lists available at ScienceDirect

Future Generation Computer Systems

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

PeerAppear: A distributed geospatial index supporting collaborative world model construction and maintenance

Andrew J. Compton *, John M. Pecarina, Alan C. Lin, Kenneth M. Hopkinson

2950 Hobson Way, Wright–Patterson AFB, OH 45433, United States

HIGHLIGHTS

- A location-aware framework for decentralized collaborative mapping is proposed.
- The framework adapts a proven p2p network topology for location-based addressing.
- Contributors store data locally and publish content summaries to a distributed index.
- Search is facilitated by efficient index lookup and direct p2p signature-based query.

ARTICLE INFO

Article history:
Received 24 April 2017
Received in revised form 27 November 2017
Accepted 22 December 2017
Available online xxxx

Keywords:
Peer-to-peer
Distributed index
Location-based search
Overlay network
Geospatial index
Visual navigation

ABSTRACT

This paper addresses the problem of scalable location-aware distributed indexing to enable the leveraging of collaborative effort for the construction and maintenance of world-scale maps and models. These maps and models support numerous activities including navigation, visual localization, persistent surveillance, and hazard or disaster detection. We approach a solution through the creation of PeerAppear, a location-aware framework for peer-to-peer indexing, search and retrieval. Due to the dynamic nature of the world, the problem of constructing and maintaining relevant world-scale models generally requires significant effort to be spent on mapping. PeerAppear offers a decentralized solution which enables the leveraging of collaborative effort through the implementation of a peer-to-peer middleware framework which automates the indexing and sharing of sensed geospatial information captured and stored in the local repositories of participants. The PeerAppear network achieves scale through a Kademlia-like overlay network which indexes data based on location by adapting Google's S2 hierarchical geographic segmentation scheme to a globally addressable distributed geographic table. Our communications primitives allow search queries to be formed and executed, enabling the discovery of information published in a specified geographic area. An evaluation of the framework is presented demonstrating excellent retrievability of published data, logarithmic efficiency and global scalability.

Published by Elsevier B.V.

1. Introduction

In recent decades our portable electronic devices have gained significant awareness of their environments through the inclusion of an array of new sensors. The prevalence of these sensors has been spurred by miniaturization and massive declines in cost, with many chips now including a miniature array of various sensors in a single inexpensive package. Most modern smart phones include multiple imaging sensors, an accelerometer, gyroscope, magnetometer, barometer, light sensor, microphones, GPS receiver, and various RF transceivers. This leads to unprecedented awareness of the local environment and the sensor's position in it, which can be exploited for visual [1], magnetic [2], and WiFi [3] mapping applications. With significant adoption of these devices occurring within the United States [4] and world-wide [5], the aggregate reach of these sensors and associated mapping capabilities is rapidly approaching global scale, thereby offering the potential for systems supporting the collaborative construction of world-scale maps and models. However, the realization of such a system must address three primary challenges.

(1) Global availability: While the aggregate reach of individual sensor platforms may represent near global coverage, the challenge of combining individual local mappings into global-scale world models requires careful consideration. Previous efforts to collaboratively construct world models have principally focused on centralized aggregation, assembly and distribution of model data. While this approach may be suitable for compact sensor

* Corresponding author.
E-mail addresses: andrew.compton.1@us.af.mil (A.J. Compton), john.pecarina@us.af.mil (J.M. Pecarina), alan.lin@us.af.mil (A.C. Lin), kenneth.hopkinson@us.af.mil (K.M. Hopkinson).

https://doi.org/10.1016/j.future.2017.12.025
0167-739X/Published by Elsevier B.V.

data, the bandwidth required for images and other types of less compact data generally makes the practice infeasible. In addition, centralization poses other challenges such as single-point failure modes and scalability in response to model growth, number of users and bandwidth needs. Our early attempts [6,7] at building visual world models from aggregators of crowd-sourced imagery also demonstrated that centralized solutions lack sufficient uniform density because manually volunteered images tend to be densely clustered near landmarks and points of interest, but sparse elsewhere.

(2) Publish and search in dynamic environments: The problem of aggregating a world-scale database of multimedia sensor data is further compounded by the dynamic nature of the world. Natural and human-driven change constantly reshapes the world, necessitating updates to maps and world models on a regular basis. This change often comes in the form of geologic activity, climate change, weather, disasters, and human activity. As changes occur, the global database must be maintained up-to-date in order to remain relevant. This requires both the inclusion of updated geospatial mappings as well as the expiry of those which are outdated.

(3) Scalability: The scale of the world poses a significant challenge to the task of collecting, storing, and indexing globally-sourced localized data. The impact of scale can be addressed in terms of *(i) lookup complexity* stemming from quantity of data and number of world-wide contributors and in terms of granularity and global reach of *(ii) geospatial addressing*. The system must be resilient and adaptable to growth and variation in both.

To overcome these challenges, our intuition leads us to seek a decentralized, distributed and collaborative solution to the problem of creating and maintaining world-scale maps and models for various types of localized data [8]. The practical goal of this approach is to enable a dynamic mapping capability which leverages the passive effort of collaborators to more rapidly update maps and models, thereby allowing for more accurate representation of the physical state of the world. The types of maps and models could vary widely depending on the needs of the applications which make use of their data, however potential examples include roadway maps, street-level visual mappings, and mappings of the locations of RF signal sources used for localization. Meeting this objective requires the realization of a peer-to-peer middleware framework that utilizes a globally scalable universal addressing scheme for indexing geospatial data.

For these reasons, we present the PeerAppear framework. PeerAppear enables world-scale information discovery through the implementation of a peer-to-peer middleware framework which automates the indexing and sharing of sensed geospatial data stored in a decentralized manner within the local repositories of its contributing users.

PeerAppear achieves *global availability* of data, depicted in Fig. 1, by allowing contributors and consumers of sensed geospatial data to discover (1) and interact with other global participants (2) in order to construct a distributed index supporting publish (3) and search (4) operations. By enabling a distributed mapping solution, PeerAppear is able to exploit a high degree of parallelism in order to frequently re-map often-traversed areas in order to provide for *publish and search in dynamic environments*. PeerAppear achieves *scalability* in terms of *lookup complexity* through an efficient peer-to-peer distributed indexing scheme which enables contributors to publish location-based summaries of their local content to a searchable distributed geographic index. The framework achieves *scalability* in terms of global reach and granularity through the application of an elastic *geospatial addressing* and filtering scheme based on Google's S2 Geometry Library which enables cell-based hierarchical addressing of the earth. PeerAppear's index constitutes a distributed geographic table (DGT) which enables the collaborative construction and maintenance of world-scale maps and models in an efficient and scalable manner.

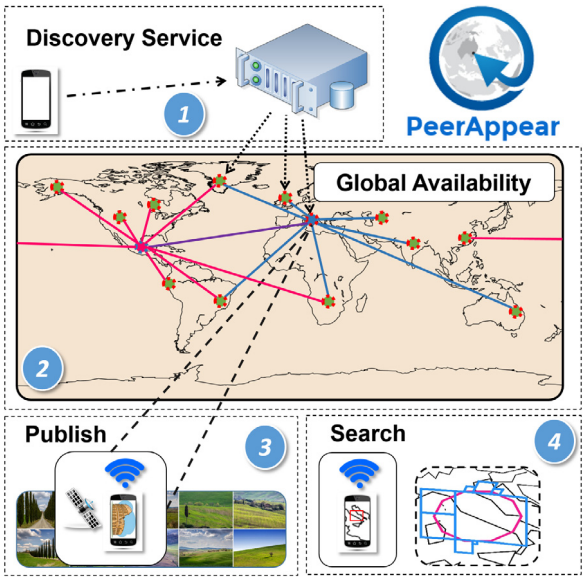


Fig. 1. The PeerAppear middleware framework enables global availability of indexed data through a globally interconnected network topology and publish-search architecture.

The paper makes the following contributions:

- It proposes a novel decentralized middleware framework supporting discovery and retrieval of geospatial information across a peer-to-peer overlay network, motivated by application-based use cases (Section 3.1).
- It describes a distributed geographic index which tightly integrates a locality-preserving addressing scheme to enable variably granular geospatial representations of nodes and data within the framework (Section 4.1).
- It presents a network topology enabling efficient and scalable logarithmic-time location-based search using novel geospatial filtering techniques and a heavily modified variant of the Kademlia overlay topology (Sections 4.2–4.3).

This paper is organized as follows. Related works is presented in Section 2. In Section 3 we present an overview of the PeerAppear framework including a high-level system description and an application-based use case. A low-level description of geospatial addressing, filtering, and network operations is presented in Section 4. An evaluation of the system is presented in Section 5. Conclusions and future work are presented in Section 6.

2. Related work

The challenges associated with world model creation and maintenance leads to the idea of leveraging the efforts of a large network of volunteer contributors to achieve maximal scale and information density. The approach is expected to improve coverage and responsiveness but also introduces the challenges of contributor scalability, quality and security from their reports. The creation of a collaborative network for mapping is not an entirely new concept. The OpenStreetMap project [9] was founded in 2008 to enable crowd-sourcing geospatial information on a world-scale. The system's largest shortcoming however is the manual effort required for end-users to contribute. CrowdAtlas [10] attempted to solve this shortcoming by automating the process of identifying particularly important GPS traces and uploading them to the OpenStreetMap database. While the two combined efforts have proven to be effective for creating a geospatial information database for

Please cite this article in press as: A.J. Compton, et al., PeerAppear: A distributed geospatial index supporting collaborative world model construction and maintenance, Future Generation Computer Systems (2018), https://doi.org/10.1016/j.future.2017.12.025.

roadways, they are still entirely reliant on GPS and the data they provide does not effectively enable an extensible capability for the collection, extraction, and sharing of sensor data. In addition, the OSM project relies on a centralized back-end to handle the processing of user queries and contributions. This architecture, while common, presents a single point of failure by requiring all OSM network traffic to pass through the OSM servers and centralizing the storage of all OSM data.

Enabling decentralized capabilities on the internet has been the focus of many recent research efforts. One of the most popular structured overlay network topologies developed recently is the Kademlia DHT [11]. Kademlia organizes clients and data using a shared 160-bit address space. Clients participating in the network select their own address according to a uniform random distribution. Data is indexed into this address space using the SHA-1 hash of the data, which is also 160-bit. These 160-bit hashes form compact signatures which represent the data from which they are derived, thereby producing a key which uniquely links to the data source. Key store and lookup activities within the DHT have proven worse-case complexity of $O(\log n)$ due to a bucket system used to store lists of peers for each client. These buckets are structured in a manner that enables a client to maintain more fine-grained knowledge about peers which reside locally in the address space than those that are more distant. When locating nodes for key pair storage or retrieval, a client performs a series of recursive lookups to identify the peers with addresses closest to that of the key. Those identified peers can then be sent *STORE* or *FIND_VALUE* commands to facilitate key storage or retrieval. One of Kademlia's most novel features is the *XOR* distance measure used to evaluate distance between pairs of addresses. This measure ensures that convergence within the network always occurs in a repeatable manner. Every communication that takes place between peers in the network includes the sender's address, thereby enabling peer buckets and the interconnectedness of the network to grow passively.

While Kademlia has proven to be an effective topology for the distributed indexing of files, it is not designed to support location-based search. To meet this need, Kovacevic et al. developed Globase.kom [12] in order to enable efficient peer-to-peer location-based search on a global scale. Globase, short for geographical location based search, enables peer-to-peer service and content discovery using a super peer topology which is organized based on client locations within rectangular geographic partitions on a Plate Carree projection. Each rectangular partition was managed by a single super peer responsible for handling queries and responses to and from nodes in other geographic partitions. A hierarchical partition structure enables overloaded super peers to subdivide their partition and assign the newly created partition to an appropriate peer within it. Fault tolerance is achieved through the maintenance of a super peer backup list for each partition. The system was evaluated using a network simulator and metrics recorded included number of hops, operation duration and relative delay penalty. The network's performance was also evaluated for various churn levels, thereby accounting for the effects of one of the primary obstacles faced by peer-to-peer networks. While the topology demonstrated excellent efficiency in terms of overhead and operational complexity, if did not provide full retrievability of globally published data.

Building upon the concepts of the Kademlia DHT, Picone et al. introduced GeoKad [13], the first fully decentralized location-based peer-to-peer topology. GeoKad adapted the Kademlia DHT to use two-dimension spatial coordinates instead of Kademlia's single-dimension file hash descriptors. In doing so, the authors coined the term distributed geographic table (DGT) which was more appropriate when applied to geospatial indexing applications. While GeoKad adopted most of Kademlia's features, the representation of addresses using two-dimension spatial coordinates was not compatible with Kademlia's XOR distance measure. Therefore, Kademlia's buckets were re-envisioned as concentric rings called GeoBuckets which store lists of peers at various distances, measured in Euclidean space, from the node. The system was designed for mobile participants. To account for mobility the address assumed by each node is determined based upon the node's current position. The authors present an evaluation of their system which was completed using a discrete event simulation tool. Metrics evaluated in their simulation include message rate, miss ratio, number of peers, and other qualitative factors. The authors present additional papers [14–18] which build upon GeoKad and demonstrate its use for vehicular and city-based infrastructure applications.

Following a similar approach to that used by GeoKad, Gross et al. created Geodemlia [19] to add search features and make refinements to the addressing scheme. While buckets in both GeoKad and Geodemlia are defined using concentric partitions at a specified radii, Geodemlia further subdivides the partitions based on cardinal directions. The resulting buckets produce better locality for the clients contained within because they share a common bearing from the client. An additional feature implemented by Geodemlia is the reproduction of key-pairs stored locally by neighboring peers. The evaluation of Geodemlia was accomplished using PeerfactSim, and demonstrated better retrievability than Globase.kom, but at higher latency and with greater overhead.

For Globase.kom, GeoKad and Geodemlia clients and data are discovered through radius-based search. While the concept of searching for data near a specified center point may seem intuitive to humans, it restricts the shape of search areas and imposes the requirement of perfect locality preservation on the address space. Brunisholz et al. proposed a solution to overcome this limitation in DataTweet [20], a network addressing scheme designed to enable location-based messaging. Brunisholz's addressing scheme leverages the Morton Z-Order curve to sequence hierarchical subdivisions of 2D space in order to assign addresses to users within spatial subdivisions such that addresses within the same subdivision share a common binary prefix. This location-derived address is then encoded within the IPv6 address of localized clients, thereby enabling efficient geocast for the broadcasting of location-based messages. In a similar manner the work described herein, PeerAppear, enables location-based indexing through the use of an addressing scheme which is derived from a hierarchical subdivision of the earth sequenced with a space filling curve. PeerAppear primarily distinguishes itself from DataTweet in purpose and approach. DataTweet enables IP geocast by embedding the location address within a client's IPv6 address while PeerApear enables the indexing of localized data through the construction of a Kademlia-inspired overlay network where data and client addresses are based on their location.

3. PeerAppear middleware overview

Many applications, such as simultaneous localization and mapping and structure from motion, require the collection and accumulation of environmental data in order to construct maps and models of the local environment. While these activities are traditionally accomplished independently, there is great potential for improvements in efficiency and awareness through passive collaboration with peers. PeerAppear facilitates this passive collaboration by enabling clients to effectively leverage the knowledge of their peers in order to extend their own environmental awareness while making their own knowledge available to be leveraged by others.

Please cite this article in press as: A.J. Compton, et al., PeerAppear: A distributed geospatial index supporting collaborative world model construction and maintenance, Future Generation Computer Systems (2018), https://doi.org/10.1016/j.future.2017.12.025.

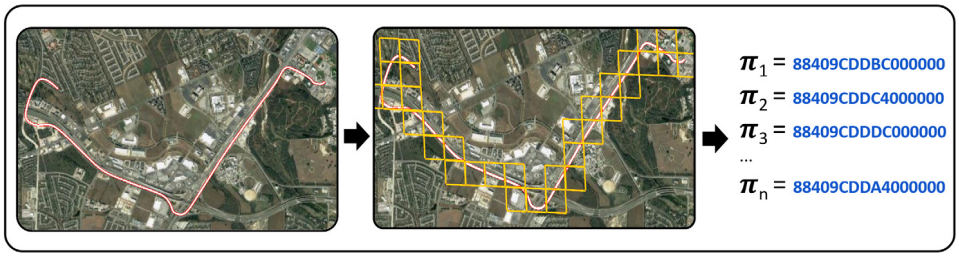


Fig. 2. Populating the index: The process of summarizing local data into a set of cell addresses to be published to the DGT. Image capture locations (left) covered as cells (middle) to reduce the number of published addresses (right).

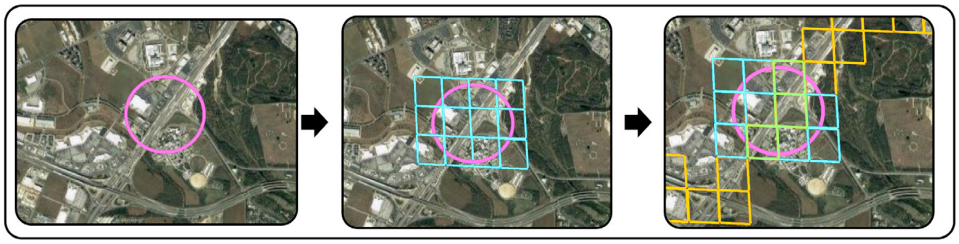


Fig. 3. Search: The process of performing a DGT lookup for keys matching a specified search filter. The potential search location (left) is covered with cells (middle) used to narrow potential repositories for search (right). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

3.1. Publish and search use cases

This motivational application involves localizing visual content. An image captured at an unknown location is used as a query to match against localized images within a repository. Successful matches, once validated, offer inference opportunities for the location of the query image.

(1) Populating the index: In order to enable a distributed search capability PeerAppear first requires the construction of an index which facilitates the identification of peers within the network who have images that might be relevant for a specified search area. Fig. 2 shows the process of summarizing a local inventory into a publishable list of cell identifiers. The process begins by constructing a list of locations for all localized data within a client's local repository. The red and white line represents the locations where images in the client's local repository were captured. A cell-base covering, depicted as yellow boxes, is then applied in order to summarize the area of the client's data and to reduce the number of addresses which must be published. Finally, each of the cell identifiers is published into PeerAppear's distributed index along with the node's network address to enable direct peer-to-peer communication.

(2) Search: Fig. 3 depicts the process of using PeerAppear to find relevant image matches. This process begins with an image and an approximate location of where it was captured, possibly derived through rudimentary cell tower localization, represented by the magenta circle. A covering of this area is applied in order to construct a list of cell identifiers (teal boxes) for use as a search filter within the global index. A lookup for each identifier within the index then produces a comprehensive and unranked list of clients which possess potentially relevant imagery (green boxes). Because keys published into the distributed index are paired with the network address of the owner within the underlying network topology, direct communication with the client can be accomplished to exchange both the image query and potential results. In prior work [7,6] we successfully demonstrated the use of bag of visual words (BoVW) image signatures to facilitate this final step, including the ranking of image matches based on cosine similarity of image signatures and numbers of matched point features.

This motivational use case specifically describes a distributed image capture, publish, and search capability. However, the PeerAppear framework has extensible elements enabling any type of localized data to be used so long as the individual elements can be represented using PeerAppear's S2-based geospatial cells, which are further described in Section 3.3.1. This paper focuses primarily on the technical underpinnings of PeerAppear's distributed geospatial index. A more thorough description of the PeerAppear extensible middleware structure can be found in our previously published works [7,6].

3.2. System requirements and assumptions

The primary purpose of the PeerAppear framework is to enable a decentralized information retrieval system allowing clients to identify and retrieve potentially relevant volunteered geospatial data made available by other peers. To achieve this purpose, PeerAppear maintains an index which keeps track of participating peers and summaries of the data that each possesses. This index, called a distributed geographic table (DGT), organizes data based on its geographic location. Unlike centralized indexes or centralized aggregation, the selection of a decentralized approach avoids many common pitfalls such as single point failure modes, bandwidth limitations, and inability to scale with growth in users, traffic, and storage requirements.

The operation of the PeerAppear framework relies on several key assumptions. The first is that all clients are able to communicate directly with all other clients using the underlying network topology, as is the case with internet protocol (IP) enabled communications infrastructure. The second is that all clients are able to localize themselves with some reasonable degree of accuracy. This is most often accomplished using GPS, WiFi SSID lookup, or cell tower triangulation. Finally all clients must be able to sense their local environment and store localized data about the environment in a database for their own use and to share with others.

While the collaborative nature of the framework presumes network connectivity, execution and storage elements are structured in a manner allowing local mappings to be constructed and used independently in the absence of network connectivity. Each participant maintains their own independent database containing local knowledge of environments which they have mapped. This data is retained locally to the extent their storage capacity allows, and data is not replicated by the network as a means of preventing the propagation of malicious or erroneous contributions. The

Please cite this article in press as: A.J. Compton, et al., PeerAppear: A distributed geospatial index supporting collaborative world model construction and maintenance, Future Generation Computer Systems (2018), https://doi.org/10.1016/j.future.2017.12.025.

Table 1
Framework elements used to meet system requirements.

	Global availability	Dynamic environments	Scalability	
			Elastic global addressing	Lookup complexity
S2 Addressing	×		×	
Network Topology	×	×		×
Publish and Search Protocol	×	×	×	×

framework instead relies on mass adoption to achieve a user base significant enough to maintain a mapped presence for most well-traveled areas.

Realizing a system fulfilling the goals and requirements of the framework described above necessitates an efficient location-aware overlay network topology which makes use of an appropriate and compatible geospatial addressing scheme. PeerAppear meets this need through the development and tight integration of a highly modified variant of the widely used Kademlia topology and the Google S2 Geometry Library. In the next two sections we address the overall system requirements, shown in Table 1, which drive the selection of PeerAppear's addressing scheme and network topology as well as the implications of and adaptations needed as a result of their combined integration.

3.3. Middleware overview

The PeerAppear middleware is comprised of three primary elements including an addressing scheme, network topology, and protocols supporting the publishing and searching of localized data represented within the distributed index.

3.3.1. S2 addressing

The PeerAppear framework overcomes the challenges of data availability and scalable coverage through the integration of Google's S2 geometry library as a principle means of localizing users and data in both local repositories and globally in the distributed index. Indexing data using multi-dimension location addresses, such as latitude and longitude, has traditionally posed significant challenges for information retrieval systems. Many of these systems, which generally rely on single-dimension keys for indexing, are not well-suited for multi-dimension indexing. Multi-dimension coordinate systems are generally a better fit for spherical addressing, but require pair-wise comparisons to determine relevance in response to area or radius-based search queries. Such comparisons must account for spherical distortion, polar anomalies, the equator and meridian. Solutions using grid-based hashing schemes, such as GeoHash [21], have gained some popularity as a means of partitioning localized data in order to simplify these comparisons, but suffer from non-uniform distribution of data across the address space because spherical distortion is still unaccounted for, leaving grid cells to cover greater area at the equator than at the poles. The S2 addressing scheme overcomes these challenges by mapping the Earth using a quadrilateralized spherical cube and further mapping the faces of the cube using quad trees sequenced by a Hilbert space-filling curve, resulting in a cell-based mapping with near-equal-area cell sizes for each hierarchical level. An example of this mapping can be seen on a global scale in Fig. 4 and on a local scale in Fig. 5.

3.3.2. Network topology

PeerAppear maintains an index mapping the locations of data collections in the network to individual users to enable clients to identify which peers have data that may be relevant for a location-based search. Due to the dynamic nature of the data available on the network at a given time, the aggregation and maintenance of a single centralized index would be challenging and costly. PeerAppear therefore adopts a Kademlia [11] inspired DGT approach for maintaining a distributed index of clients and representations of the locations of the data collections they store. The routing table structure established by the DGT ensures that data within the index is globally available and provides for efficient lookup complexity in order to support the high operations throughput necessary to enable frequent remapping necessary for dynamic environments.

3.3.3. Publish and search protocol

PeerApper uses a publish and search protocol which adapted from Kademlia in order to support the integration of S2 addressing the execution of location-based operations. The protocol enables data to be published and queries to be executed using 64-bit S2 addresses for all hierarchical levels. While Kademlia's publish and search protocols are only capable of supporting exact key matches, PeerApper enables mixed-level publish and search in order to allow the discovery of corresponding data based on the hierarchical relationship of query and publish cells.

4. Peer-to-peer distributed geospatial indexing

In this section we describe the technical underpinnings of the PeerAppear framework and present the manner in which it integrates underlying technologies. The includes PeerAppear's geospatial addressing and indexing mechanisms, publish and search protocols, and the implications of their integrated application.

4.1. S2 geospatial indexing

PeerAppear indexes data using 64-bit keys. These keys are simply the S2 addresses which correspond to the location where the data was captured. PeerAppear strips the leaf bits from addresses to prevent ambiguity within client routing tables, however the addresses are otherwise unmodified. Client and data addresses therefore are representative of location only, leaving the search for specific content to a second-stage search facilitated through direct communication with clients who have been identified as having location-relevant content. The S2 addressing scheme enables variably granular representations of the locations of data through hierarchical partitioning. These keys can be used to describe the locations of data within a repository in a compact manner suitable for publishing to the distributed index. They are also used when submitting queries to the index.

4.1.1. Elastic granularity

The S2 library enables a one-dimension global addressing scheme by subdividing the Earth using a hierarchical cell-based partitioning scheme. The partitioning begins with a quadrilateralized spherical mapping which divides the surface of the earth into 6 equal area-faces. Each face is further subdivided into near-equal-area cells in a hierarchical fashion using a quad tree. The sequential addressing of each cell is accomplished using a Hilbert space filling curve, as shown in Fig. 4.

The size of cells in the hierarchical subdivision range from approximately 1 cm^2 for the smallest (level 30) cells to approximately 85,000,000 km^2 for the largest (level 0) cells covering an entire cube face. All addresses in the S2 library use 64-bit representations, with the hierarchical structure of the quad trees represented in bit order in the address. This enables smaller cells to be grouped

Please cite this article in press as: A.J. Compton, et al., PeerAppear: A distributed geospatial index supporting collaborative world model construction and maintenance, Future Generation Computer Systems (2018), https://doi.org/10.1016/j.future.2017.12.025.

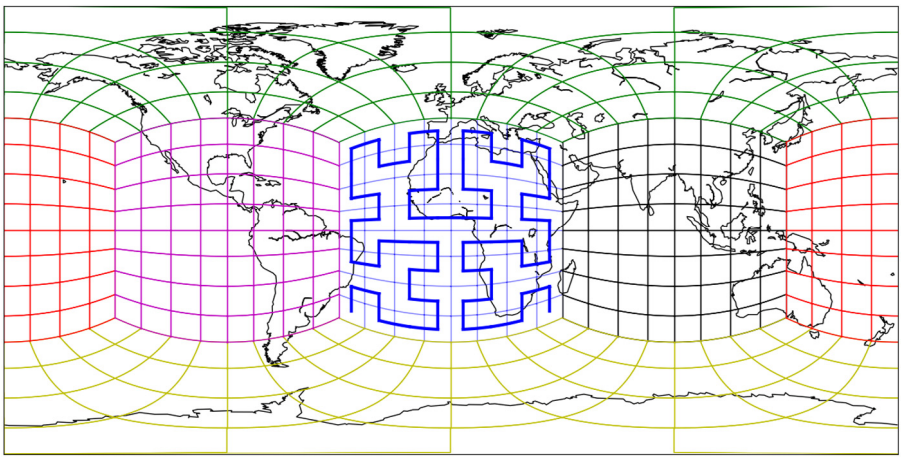


Fig. 4. A mapping of the globe using Google's S2 Spherical Geometry library with level 3 cells. Center cube face (Africa) shows sequential addressing using Hilbert mapping.

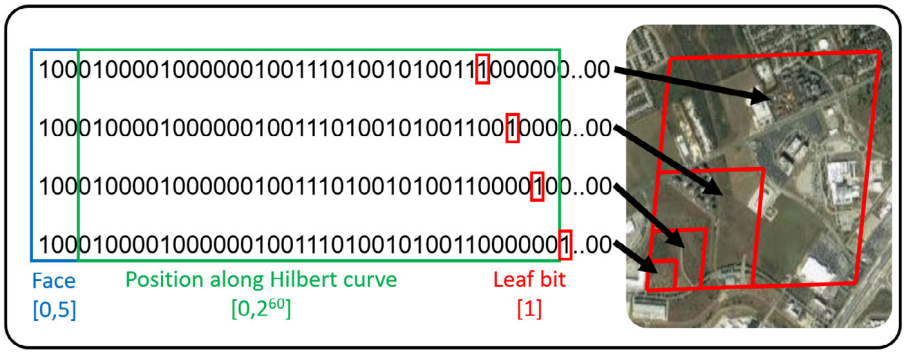


Fig. 5. A depiction of hierarchical S2 cells and corresponding cell prefixes.

and represented by a larger cell address using the largest common prefix of the set of smaller cells, as shown in Fig. 5. The S2 address encoding enables PeerAppear to represent the locations of users and user data with varying levels of granularity in order to provide sufficient fidelity to target search queries while still maintaining an abbreviated representation suitable for efficient peer-to-peer indexing.

Because of the varying granularity offered by the S2 cell partitioning, geospatial filters can be constructed which conform closely to most polygons using a heterogeneous set of S2 cells. Fig. 3 depicts an S2 cell covering corresponding to a search radius. An item found to have a matching prefix with any member of the search set would be identified as being within the search area. In the figure shown the search filter is comprised of 9 S2 cells. By using an increased number of smaller cells the conformity of the filter to the search radius can be improved with a penalty to the efficiency of the search. Conversely, a filter consisting of fewer cells can be formed in order to improve efficiency at the cost of loss of conformity with the specified search area.

4.1.2. Compact bitwise radix tree inventories

PeerAppear maintains the cell address inventory for each node in a radix tree data structure. The inventory includes cell addresses for data stored in the node's local repository as well as cell addresses published to the node by peers. This radix structure enables efficient lookup, insertion, and deletion with $O(\log n)$ time complexity. Lookup operations are especially efficient for heterogeneous level cell operations because of the inherent parent–child relationships used by the radix tree for cells with matching prefixes.

4.2. Networking in an S2-integrated Kademlia variant

PeerAppear's DGT primarily differentiates itself from Kademlia through its use of the S2 library's 64-bit addresses instead of Kademlia's 160-bit addresses. Kademlia's addresses, which are generated using the SHA-1 hash function for files and randomly assigned for users, bear no useful information about the data they represent or the users to which they are assigned. Their primary function is to ensure uniqueness, enabling files shared on Kademlia-powered networks to be uniquely identified and indexed with little chance of collision. This presents challenges however because the 160-bit identifier of a file must be known prior to searching for it on the DHT. Because S2 cells represent known geographic space, the mapping of geographic space to S2 cells enables search to be effected using a search filter comprised of cell addresses without prior knowledge of what has been published to the DGT. This promotes global availability of data within the network.

4.2.1. Global availability in a peer routing table

In order to facilitate a contiguous communications capability across the PeerAppear network, each node stores contact information about other peers in lists organized by address. These lists, which are each limited in length to k peers, are structured to ensure that space for knowledge about other peers whose addresses are distributed across the network's address space is reserved. These lists are not uniformly distributed across the address space. Lists are instead organized to ensure that nodes have increasingly greater knowledge of peers near their own assigned 64-bit address, as determined by the XOR distance between the node's

Please cite this article in press as: A.J. Compton, et al., PeerAppear: A distributed geospatial index supporting collaborative world model construction and maintenance, Future Generation Computer Systems (2018), https://doi.org/10.1016/j.future.2017.12.025.

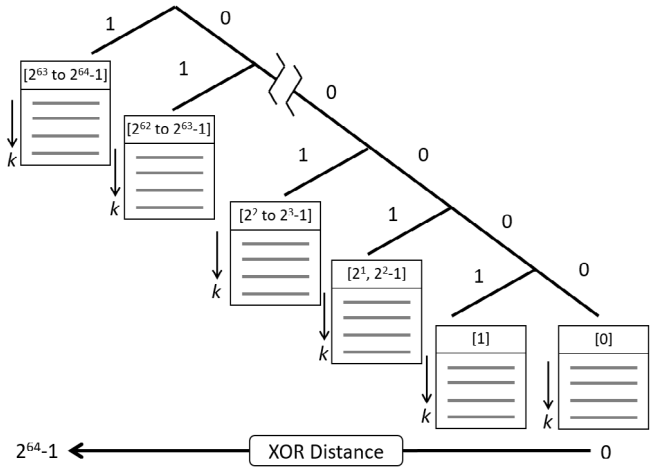


Fig. 6. The tree structure of fully populated peer lists of length k. Nodes with the furthest XOR distance are on the left and closest XOR distance are on the right.

own address and those of the known peers, as shown in Fig. 6. Because each node maintains fine-grained knowledge about peers near its own address, the likelihood of being able to respond to a *FIND_NODE* or *FIND_VALUE* RPC with nodes closer to the target address is greatly increased. This routing structure, which closely matches that used by Kademlia, is also key for enabling logarithmic time complexity for DGT operations.

PeerAppear nodes use an adaptive peer list structure which organizes peers based on their XOR distance from the node, $PEER\_ADDR = NODE\_ID \oplus PEER\_ID$, where *PEER_ID* is the 64-bit address of the peer and *NODE_ID* is the 64-bit address of the node which owns the list. Initially, only a single peer list is maintained. When the list exceeds k peers it is split into two lists each covering half of the range of the list they replace. This process of splitting lists continues as additional peers are added, however only the list with range including $PEER\_ADDR = 0$ can be split. This rule ensures that nodes track greater numbers of peers near their own *NODE_ID*. A geographic mapping of the adaptive list structure for a fully populated and enumerated peer list is shown in Fig. 7. For each known peer, a node stores the necessary information required to enable direct communications within the underlying network, including *(PEER_ADDR, [PEER_ID, IP_ADDR, PORT])*.

4.2.2. XOR compatible geospatial sharding

One of Kademlia's key contributions to the area of distributed indexing is the introduction of bitwise XOR as a means of determining distance between pairs of binary addresses. The use of XOR enables nodes to determine how "close" an address is to their own or that of another node, with the XOR distance effectively denoting the magnitude of the difference between two addresses. The XOR measure also provides for unidirectionality. With Kademlia's address space envisioned as a binary tree, selecting nodes based on smallest XOR distance ensures that convergence occurs along the same path, thereby supporting full retrievability. Organization of a node's routing table is also accomplished based on the XOR distance to known peers. The peer "buckets" created by this organization enable sharding for the distributed index.

The XOR distance measure also pairs well with S2 addresses because of their bitwise representation of hierarchical relationships. This makes XOR especially useful for determining relationships between pairs of S2 cell addresses within PeerAppear because it enables the identification of parent–child relationships within S2's hierarchical structure. The smaller the magnitude of the XOR of 2 cell addresses, the nearer they are to one another in the tree.

As with Kademlia, nodes in the PeerAppear network have addresses which are drawn from the same address space used to identify data. Nodes are responsible for indexing data with addresses that are "close" to their own address. Because addresses for files indexed by Kademlia are based on the file's SHA-1 hash, which generally follow a uniform random distribution, it makes sense that node addresses in Kademlia are generated according to a uniform random distribution. Addresses within PeerAppear are representative of data location, therefore random node address assignment is inappropriate. PeerAppear clients instead assume a 64-bit S2 address corresponding to their location at time of startup rather than a randomly generated one. The address is reset for every client restart, enabling the client to bootstrap its routing table and index inventory such that fine-grained knowledge is always local. This benefits both lookup complexity of nearby addresses as well as load balancing for the DGT because areas with the greatest numbers of users also collectively have the largest data repositories, thereby enabling a type of geospatial sharding for the distributed geographic index.

4.3. Publish and search protocol

The PeerAppear framework enables DGT access through heavily modified variants of Kademlia's three primary remote procedure calls (RPCs): *FIND_NODE*, *FIND_VALUE*, and *STORE*. These RPCs are further described in Table 2. Each RPC sent within the PeerAppear network includes the *[PEER_ID, IP_ADDR, PORT]* triplet containing information about the sender. This allows the recipient to add the sender to their peer list if room is available. PeerAppear incorporates these RPCs into its three main operations: *Node Lookup*, *Area Lookup*, and *Publish Inventory*.

The *Node Lookup* operation enables search to be effected in order to find network clients whose identifiers fall closest to a given query address. The lookup operation begins with a node sending the *FIND_NODE* RPC to the α nodes it knows of which are closest to the query address. The operation continues iteratively until the closest k nodes it has discovered to the query address have been contacted without yielding any closer contacts.

The *Area Lookup* operation enables search to be effected in order to find keys which have been published into the network for data matching a query's specified location filter. This operation takes as input a list of S2 addresses which collectively form a search area. For each S2 address, peers with matching content are discovered using the *FIND_VALUE* RPC. These peers are aggregated within results list and duplicates are excluded.

The *Publish Data* operation enables clients to publish key pairs which identify them as owners of data which fall within the bounds of a specified area. For each S2 address from the list which summarizes the node's local data, the *FIND_NODE* RPC is used to build the list of k peers responsible for indexing data at that location followed by the *STORE* RPC to each to instruct the nodes to record the values into their local radix tree inventory.

Because the index is expected to be subjected to high churn rates, the information stored in peer lists and index inventories must be refreshed periodically. Peer lists are maintained using a *ping* for peers a node has not communicated with in t_p minutes. Nodes that fail to respond to the *ping* are removed. Inventories are maintained through a perish and re-publish process. Inventory items older than t_{rp} are automatically removed. Nodes must therefore re-publish the list of cells representing data retained locally every t_{rp} minutes. Our system defaults to a t_p of 5 min and a t_{rp} of 15 min.

When a PeerApear node first starts it bootstraps its peer list by inserting nodes it saved from previous sessions or nodes it learns about through a centralized discovery service. The node then initiates a *Node Lookup* operation using its own node address as the

Please cite this article in press as: A.J. Compton, et al., PeerAppear: A distributed geospatial index supporting collaborative world model construction and maintenance, Future Generation Computer Systems (2018), https://doi.org/10.1016/j.future.2017.12.025.

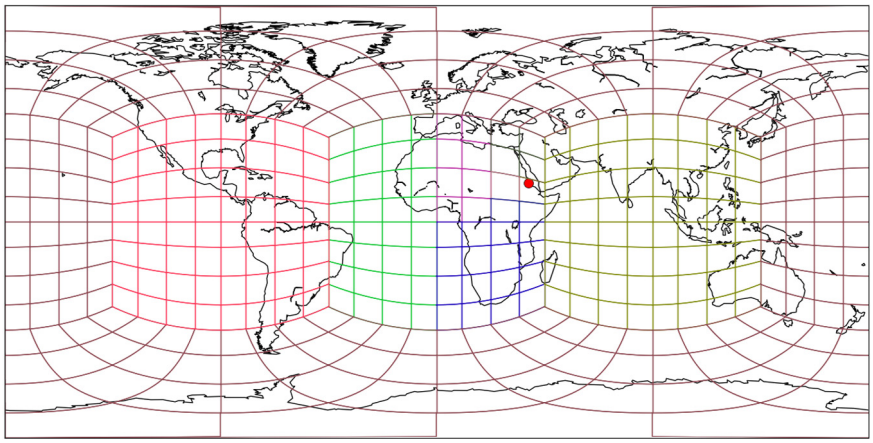


Fig. 7. The global mapping of individual peer list coverage areas, each containing up to k peers, shaded by color for a randomly generated node (red dot).

Table 2
Remote procedure call (RPC) functions used to manage nodes and data within PeerAppear.

RPC Name	Description
FIND_NODE	The *FIND_NODE* RPC returns *[PEER_ID, PEER_IP_ADDR, PEER_PORT]* for the k nodes from its peer lists which are closest to a specified lookup address.
FIND_VALUE	The *FIND_VALUE* RPC performs the same function as *FIND_NODE*, but also returns any items from the receiving node's own index inventory which match the specified lookup address. For an inventory item to be considered a match, it must meet one of three criteria: the addresses must be equal, the lookup address must be a child of an inventory cell address, or the lookup address must be a parent of an inventory cell address. These matching criteria ensure that lookups with addresses specified at any cell level successfully match to corresponding inventory cells which can also be published at any level.
STORE	The *STORE* RPC instructs a node to insert information about a published key pair into the local index inventory. This information includes the cell address and information about the node that published it, including *(CELL_ADDR, [NODE_ID, IP_ADDR, PORT])*.
PING	The *PING* RPC is used to determine if peers listed in a node's routing list are still connected to the network.

query address. This process serves both as a means of discovering nodes to insert into its own list as well as a means of informing other nodes of its existence. Finally, in order to bootstrap the locally-maintained list of keys published by peers, the node sends a request to each of its k closest nodes for a compressed inventory transmittal. These inventories are merged, excluding duplicates, and inserted into a radix tree.

5. Performance evaluation

In this section we present a simulation-based evaluation of the PeerAppear framework. The evaluation is divided into two subsections, with the first demonstrating the performance of the system with respect to its addressing scheme and the second characterizing the performance of the system with respect to its search capability using both single and multi-level cell operations.

5.1. PeerAppear Simulation Engine (PASE)

To enable incremental development, parameter tuning and performance evaluation, the PeerAppear framework was implemented in a custom Python-based discrete event simulation environment which we have named the PeerAppear Simulation Engine (PASE). PASE simulates and records all interactions between peers on the simulated framework. Peers and their local repositories can be randomly generated or built from a pre-existing dataset, enabling both large-scale global evaluation and small-scale application-driven evaluation. Each simulated client's local repository, peer inventory, routing table, and local system variables are maintained in a central database. Databases for various network sizes and tunable parameters can be constructed and saved, thereby facilitating evaluations to be executed multiple times for an array of parameter values using the same dataset.

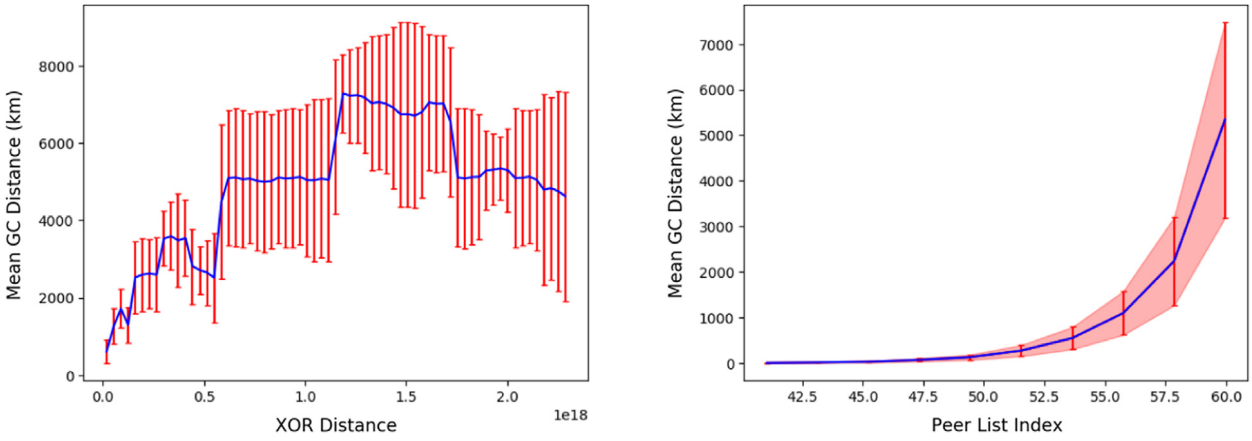
5.2. Addressing performance

The following section documents our use of PASE to evaluate the performance of the framework's geospatial addressing scheme to preserve locality and the uniformity of data storage and traffic across the framework's address space.

5.2.1. Locality preservation

Locality preservation means relative distances between locations in one address space are preserved when mapped into an alternate address space. The mapping of two-dimension spherical coordinates to one-dimension S2 addresses can be considered a type of reversible dimensionally-reducing hash function. A desirable property of hash functions which reduce dimensionality is preservation of locality. For a hash function to be locality preserving, it must preserve relative distances between input values and represent them in their corresponding output values. Because the S2 mapping is reversible, exact relative distances can always be determined by converting S2 addresses back to their corresponding spherical coordinates and calculating great-circle distance. We

Please cite this article in press as: A.J. Compton, et al., PeerAppear: A distributed geospatial index supporting collaborative world model construction and maintenance, Future Generation Computer Systems (2018), https://doi.org/10.1016/j.future.2017.12.025.



(a) The relationship between XOR and great-circle distance.

(b) Mean distance of peers by peer list index.

Fig. 8. Representation of great-circle distance in the PeerAppear address space and routing table.

have already shown S2 can provide global availability and elastic global addressing; however this process is computationally expensive.

To reduce the computational costs associated with calculating distances between cell addresses, PeerAppear establishes a network topology that adopts the XOR distance measure used by Kademlia. The XOR distance measure functions as a computationally low-cost heuristic for great-circle distance, enabling peer routing tables to be organized based on the magnitude of the XOR distance between a node's address and those of known peers. Because PeerAppear does not support proximity search, instead requiring search filters to be specified using S2 cell addresses, locality preservation within the address space is not explicitly required. Locality preservation does however benefit low-latency communications because nodes communicate primarily with peers whose addresses are nearby in the address space.

To evaluate the ability of the XOR heuristic to preserve locality within the PeerAppear address space and routing table, we generated 1,000,000 pairs of S2 addresses from the same level 0 parent cell using a random uniform distribution for analysis. The great-circle distances of the pairs' corresponding spherical coordinates were then plotted against the XOR distance of their S2 addresses. The resulting plot is shown in Fig. 8(a). This plot demonstrates a general trending correlation between XOR and great-circle distance, however portions of the address space do not conform to this trend due the manner in which the Hilbert mapping sequences cell addresses. In Fig. 8(b) we can see that these inconsistencies are mitigated by the XOR-based binning employed by the PeerAppear routing table. This graph demonstrates that while the great-circle distances for individual cells from adjacent peer lists within the routing table may not exhibit consistent total ordering when arranged by increasing XOR distance, the mean great-circle distances do demonstrate a consistent monotonic increasing relationship for lists with XOR distances which are further from the source node.

Next, we evaluated the effect of XOR's locality preservation on routing table composition and complexity of index lookup operations by effectively disabling locality preservation for comparative analysis. We accomplished this by building a network test set which "incorrectly" uses randomly generated 64-bit node addresses which do not correspond to the node's actual location. A second "correct" network test set was created using 64-bit S2-based node addresses corresponding to the locations of the simulated nodes. The PASE parameters for both network test sets are listed in Table 3.

The mean number of peers contained in the routing tables of both sets of test networks is plotted against the great-circle distance to the peers in Fig. 9(a). The graph demonstrates the network's ability to build routing lists which include fine-grained knowledge (greater numbers) of peers in local proximity while still maintaining access to more distant nodes. Numbers of randomly addressed peers correlate with the surface area of bands at specific range intervals, increasing up to one quarter of the earth's circumference before decreasing again. The mean lookup complexity for both sets of simulated networks is depicted in Fig. 9(b). The network using correctly assigned S2-based node addresses demonstrated logarithmic lookup complexities which correlate well with great-circle distance to the lookup address. The network using randomly generated node addresses performed worse for node lookups at distances less than 7500 km and only marginally better for further distances. The slight advantage held by random addressing for distant queries is likely due to uniform density of nodes in the routing table. While the network using location-based node addresses has greater numbers of peers in close proximity, the randomly network with randomly assigned node addresses has uniform knowledge of nodes at all distances and can perform remote lookups for distant addresses by contacting fewer nodes.

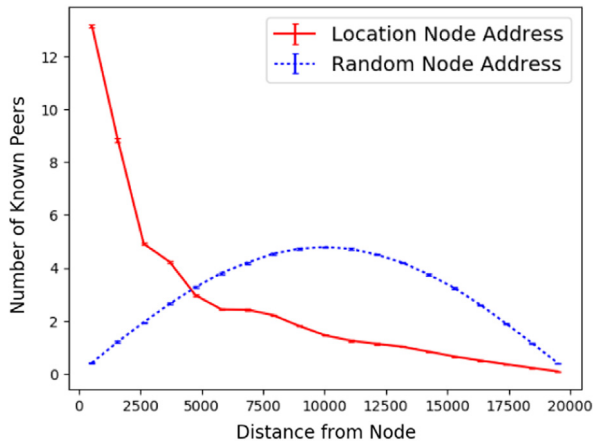
Table 3
PASE network parameters for locality analysis.

Parameter	Value
Environmental Parameters	
Peer distribution	Global
Size of area	510.1 M km^2
Number of peers	5000
Number of data items	50,000
System parameters	
Lookup concurrency (α)	X
Replication (k)	X
Node locations	S2 (Uniform random)
Node addressing	S2, Random
Data locations	Gaussian $\sigma = 5$ km

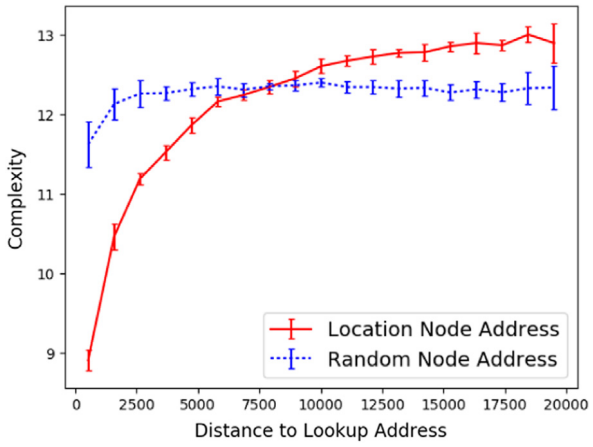
5.2.2. Uniformity of key storage

The use of S2 addressing within the PeerAppear network leads to two types of non-uniformity of communications traffic and storage of published keys which impact publish and search performance. The first occurs because nodes within the network are likely to be geospatially distributed such that the density of nodes is proportional to population density. This leads to high density in urban environments, low density in most less populous and infrequently traveled areas, and extreme sparsity near the poles

Please cite this article in press as: A.J. Compton, et al., PeerAppear: A distributed geospatial index supporting collaborative world model construction and maintenance, Future Generation Computer Systems (2018), https://doi.org/10.1016/j.future.2017.12.025.

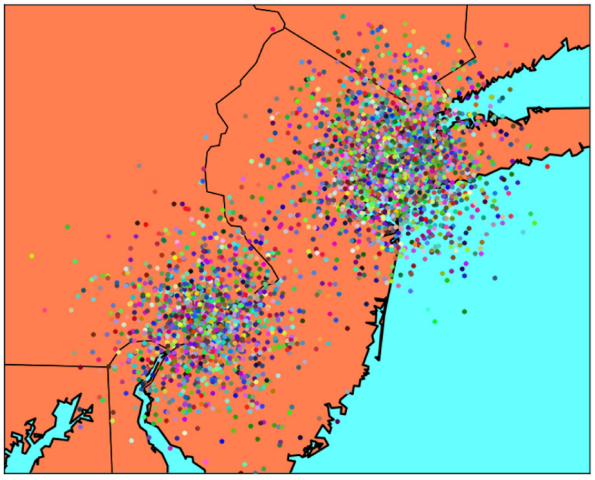


(a) Histogram of peer list by distance.

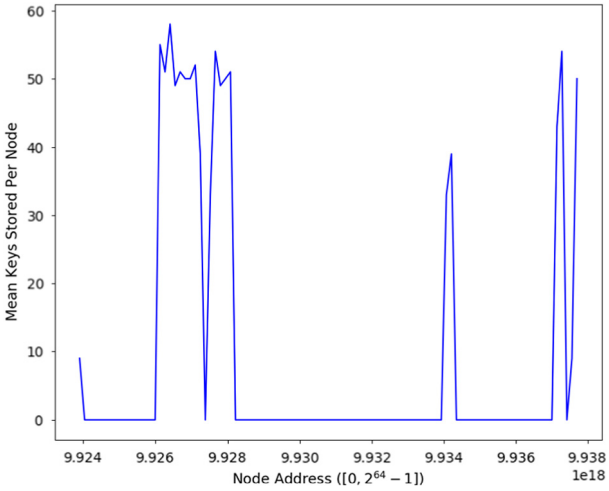


(b) RPC complexity vs distance.

Fig. 9. Spatial locality analysis for peer lists and RPC execution.



(a) Map of nodes distributed geographically according to a Gaussian mixture distribution.



(b) Distribution of keys across the address space for a Gaussian mixture distribution.

Fig. 10. Non-uniform distribution of key storage across address space caused by non-uniform geospatial distribution of nodes.

and water covered areas. An example of this type of non-uniform distribution can be seen in Fig. 10. Fig. 10(a) shows a mapping of nodes with locations assigned to New York City and Philadelphia based on a Gaussian mixture distribution and Fig. 10(b) depicts the density of keys stored by nodes within the area's 1-dimension S2 address space. Because nodes assume addresses corresponding to their location at startup, this type of non-uniformity does not negatively impact the network because the number of peers responsible for indexing an area grows as the number of keys published in a given area grows (demonstrated in Section 5.3).

The second type of non-uniform key distribution occurs as a result of publishing and querying cell addresses which do not span the full resolution of the 64-bit address space, as is the case with larger (low level) S2 cells. The negative consequences of this type of non-uniform key distribution include increased network traffic and storage requirements levied on clients with addresses closest to the address space subdivisions corresponding to the addresses of larger (shorter prefix) S2 cells. This can be seen in Fig. 11, where 50,000 keys were published to 5000 nodes on the North American cube face at cell levels $C_L = \{5, 4, 3, 2, 1\}$. As the size of the published cell address prefix shrinks (lower cell level) the distribution of keys across PeerAppear's address space forms ever tighter clusters.

The non-uniform distribution of stored network keys has an additional negative consequence which can affect retrievability within the system under certain circumstances. Due to the XOR distance between overlapping cells of different levels, the potential exists for a query to be submitted using an S2 address which is sufficiently distant from a matching address such that the k closest nodes to the query address do not hold the matching cell address in their inventories. To mitigate this possibility for systems employing a heterogeneous mix of cell levels, we propose the use of an adaptive-k system. Under adaptive-k, all results which overlap (child or parent) a specified query address are returned. A minimum of k results, if available, are returned by the *FIND_NODE* and *FIND_VALUE* RPCs. The performance of mixed-level operations and the adaptive-k approach is further evaluated in Section 5.3.2.

5.3. Network performance

We conducted a series of simulations using PASE with parameter values specified in Table 4 to evaluate the performance of PeerAppear's DGT operations. These parameters were selected in order to explore the effect that their variability could be expected to have on the performance of non-simulated implementations of

Please cite this article in press as: A.J. Compton, et al., PeerAppear: A distributed geospatial index supporting collaborative world model construction and maintenance, Future Generation Computer Systems (2018), https://doi.org/10.1016/j.future.2017.12.025.

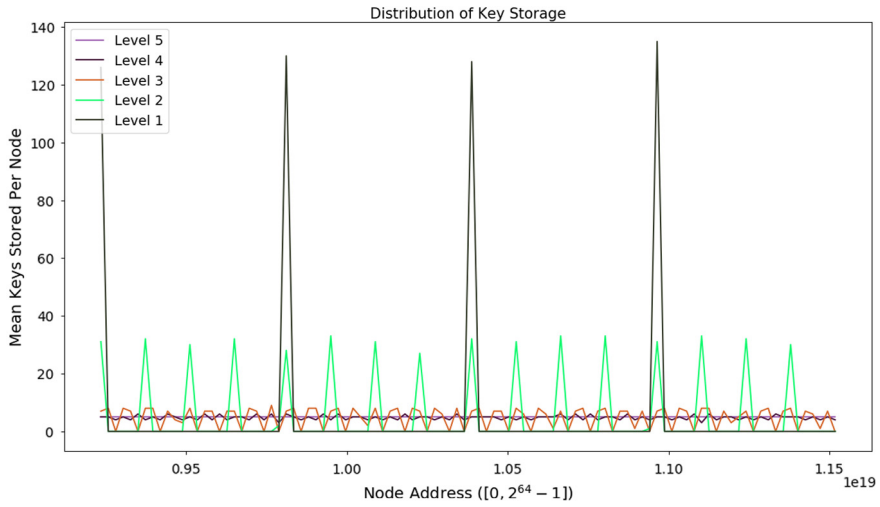


Fig. 11. Non-uniform distribution of key storage across address space caused by cell level prefix size ($C_L = \{1, 2, 3, 4, 5\}$).

Table 4
Environmental and system parameters used to execute experiments within PASE.

Parameter	Value
Environmental parameters	
Peer distribution	NYC and Philadelphia
Size of area	Approximately 62,500 km^2
Number of peers (n)	50–5000–20,000
Number of data items	$n * 10$
System parameters	
Lookup concurrency (α)	1–3–8
Replication (k)	1–7–8
Node locations	Gaussian $\sigma = 30$ km
Node addressing	S2
Data locations	Gaussian $\sigma = 5$ km

the PeerAppear framework. For evaluations which vary a single network parameter, the remaining parameters are fixed using the underlined value.

The locations of simulated network nodes and the locations of the data they publish to the network are generated according to a Gaussian mixture distribution, resulting in node maps similar to the one depicted in Fig. 9(a). The nodes are distributed such that 65% are assigned node addresses near New York city ($\mu = (40.7128\ \text{N}, 74.00592\ \text{W})$, $\sigma = 30$ km) and 35% are assigned node addresses near Philadelphia ($\mu = (39.9526\ \text{N}, 75.1652\ \text{W})$, $\sigma = 30$ km). This distribution was selected to simulate expected real-world conditions with a mix of dense urban and sparse rural node locales. The locations for data owned and published by each node is also Gaussian distributed based on the location represented by the node's address ($\mu = (\text{Owner Lat, Lon})$, $\sigma = 5$ km). All plots depict mean results and 95% confidence intervals from 5 trials completed using the same network parameters. Nodes for each trial are loaded and bootstrapped into the network sequentially, with contact information for up to 5 previously connected nodes being provided to new nodes by a simulated discovery service.

Network performance evaluations are divided into two subsections: 5.3.1) Single Cell-Level Operations and 5.3.2) Mixed Cell-Level Operations. The first evaluates the performance of the network for k and α parameter variation in terms of recall and search complexity for various numbers of nodes when keys are published and queried using a single cell level. This evaluation tests the network's ability to perform exact-match search for S2 keys. The second evaluates the performance of the network for various search cell levels in terms of recall and search complexity for various numbers of nodes and k values, including adaptive-k operations. This evaluation tests the network's ability to perform search for network keys using filters comprised of multiple cell levels.

5.3.1. Single cell-level operations

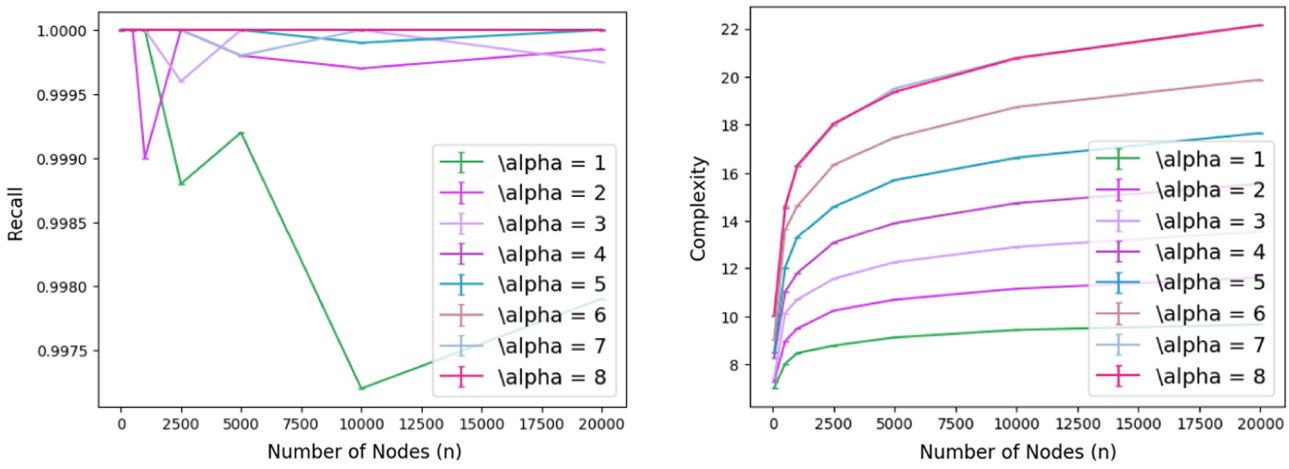
Single cell-level operations test the network's ability to find publishers of keys where the query address is exactly equal to the published key. This evaluation was accomplished using publish and search cell level $C_L = 16$. Representing an area approximately equal to 20,000 square meters or 1 city block, we consider level 16 cells to be a satisfactory compromise for granularity of data representation. The search recall and complexity of networks with numbers of peers between 50 and 20,000 for α values between 1 and 8 is shown in Fig. 12. The search complexity values indicate the number of clients contacted in order to completely execute a query. Recall results, depicted in Fig. 12(a) show that α values as low as 1 still return near full recall, however values above 1 produced more consistent results. The complexity analysis for the examined range of α values, shown in Fig. 12(b), showed that increases in α, up to the value of k, resulted in significantly greater numbers of clients contacted in order to fully execute search queries.

Using the same level for publish and search cells ($C_L = 16$), the experiment was repeated with a fixed k value of 3 and α values between 1 and 8. The results are depicted in Fig. 13. Fig. 13(a) shows that full recall is achieved for all network sizes with k values of 3 and greater. Fig. 13(b) shows marginal increases in complexity as k and the size of the network grow beyond 3. For all values of k network complexities exhibit logarithmic complexity growth as network size increases.

5.3.2. Mixed cell-level operations

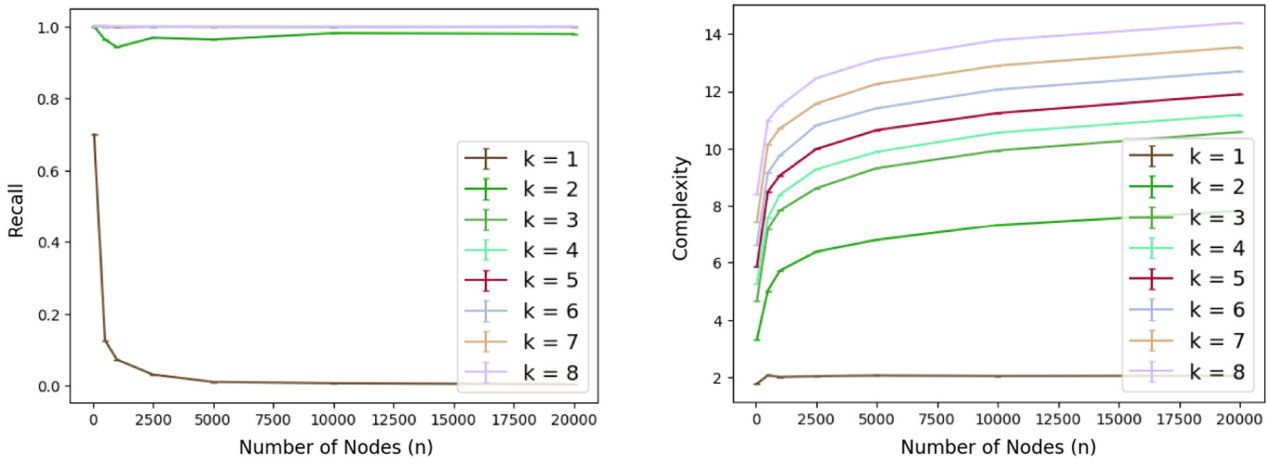
Mixed cell-level operations test the network's ability to find cells which share corresponding areas (share a full prefix) when the publish and search cell levels are not the same. For this evaluation, keys are published at $C_L = 16$ and queries are submitted across the range of $C_L = [5, 19]$. The search recall and complexity of networks with numbers of peers between $n = 50$ and $n = 20{,}000$ for the specified search levels is shown in Fig. 14. The results show search recall for a specified search cell level and network size in Fig. 14(a) and corresponding complexity in terms of number of nodes searched in Fig. 14(b). Recall results demonstrate marginally higher recall for smaller network sizes when search is performed using lower level cells, however recall for all network sizes converges nearly 1 for cell levels at and beyond level 14.

Please cite this article in press as: A.J. Compton, et al., PeerAppear: A distributed geospatial index supporting collaborative world model construction and maintenance, Future Generation Computer Systems (2018), https://doi.org/10.1016/j.future.2017.12.025.



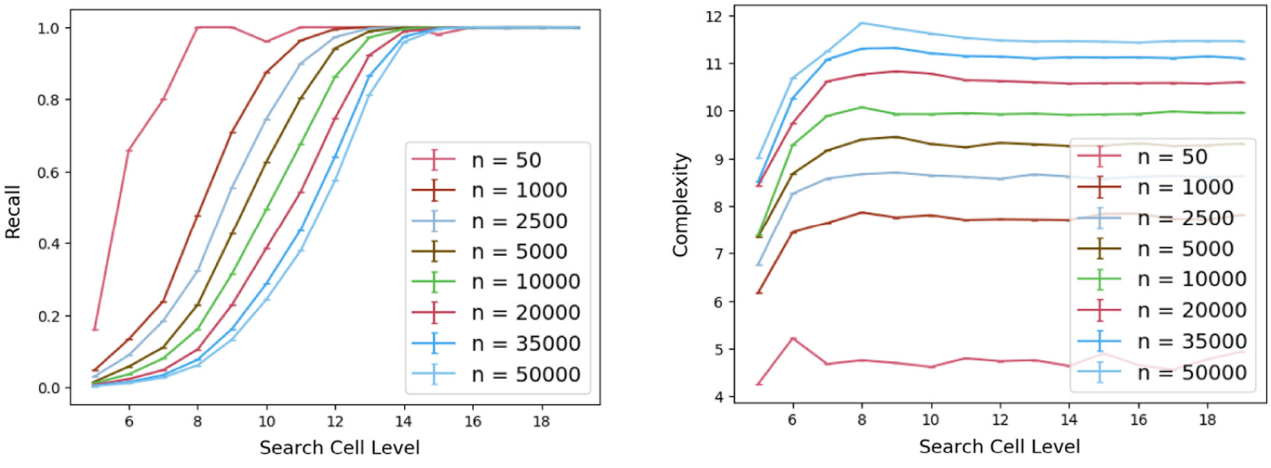
(a) Search recall. (b) Search complexity.

Fig. 12. Effect of numbers of peers on recall and complexity for various alpha values.



(a) Search recall. (b) Search complexity.

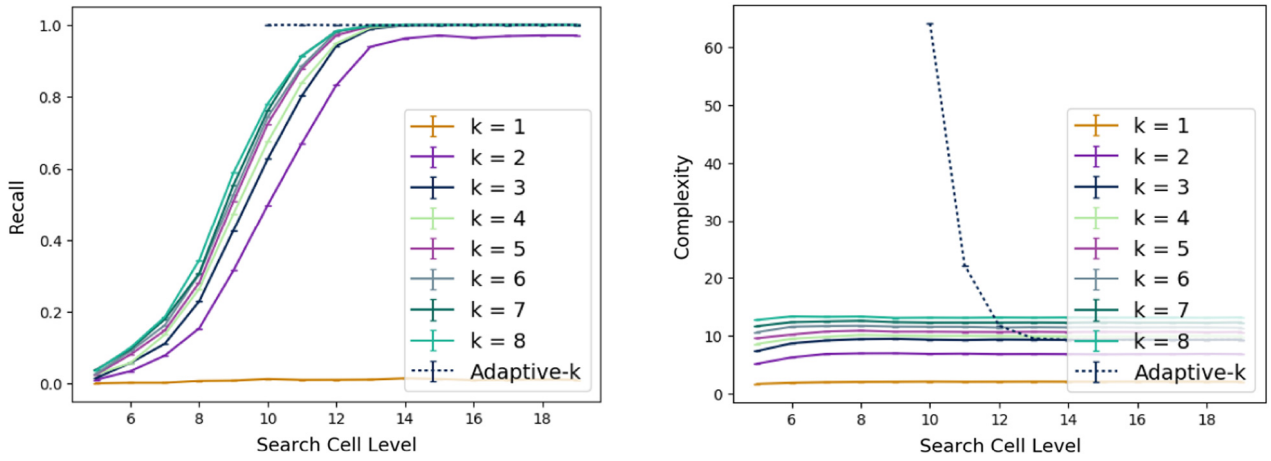
Fig. 13. Effect of numbers of peers on recall and complexity for various *k* values.



(a) Search recall. (b) Search complexity.

Fig. 14. Evaluation of search recall and complexity at different search cell levels for various network sizes.

Please cite this article in press as: A.J. Compton, et al., PeerAppear: A distributed geospatial index supporting collaborative world model construction and maintenance, Future Generation Computer Systems (2018), https://doi.org/10.1016/j.future.2017.12.025.



(a) Search recall.

(b) Search complexity.

Fig. 15. Evaluation of search recall and complexity at different search cell levels for various k values.

Using the same cell levels, the experiment was repeated for a fixed network size of $n = 5000$ and k values ranging between 1 and 7. In addition, the adaptive-k method described in Section 5.2.2 was also employed using a base-k value of 3. The results of this experiment are shown in Fig. 15. Fig. 15(a) demonstrates near-perfect recall for search cell levels at or above $C_L = 13$ using k values of 3 and higher. Below $C_L = 13$ recall rapidly drops with higher-k value networks consistently faring marginally better. Fig. 15(b) shows that complexity for fixed-k operations is a function of k value with little variation across the tested range of search levels. While the adaptive-k search evaluation demonstrated perfect recall, complexity proved nearly intractable as the size of the search space grew to encompass the entire dataset.

6. Conclusion and future work

In this paper we presented and evaluated PeerAppear, a middleware framework based on the S2 geometry library and the Kademlia peer-to-peer overlay network. The paper describes a system designed to be efficiently scalable in order to support the construction and maintenance of world-scale geospatial models. Our evaluation demonstrates that the overlay topology pairs well with the S2 addressing scheme, enabling efficient logarithmic time operations and full retrievability of published data for symmetric cell-level operations and high levels of retrievability for non-symmetric cell level operations.

Future work will include adapting operations and RPCs to support cell lists and extending the addressing space by adding additional leading bits to represent different data models and planets. An application-based evaluation of the framework for collaborative visual localization will also be conducted using a heterogeneous mix of live real-time and virtual nodes within PASE. This evaluation is expected to serve as demonstration platform showcasing the framework's potential to enable collaborative mapping in an increasingly sensor-rich and connected digital environment.

References

[1] G. Zhou, A. Liu, K. Yang, T. Wang, Z. Li, An embedded solution to visual mapping for consumer drones, 2014, pp. 670-675.

[2] M. Frassl, M. Angermann, M. Lichtenstern, P. Robertson, B.J. Julian, M. Doniec, Magnetic maps of indoor environments for precise localization of legged and non-legged locomotion, in: IEEE International Conference on Intelligent Robots and Systems, 2013, pp. 913–920.

[3] L. Sørensen, L. Jacobsen, J. Hansen, Low cost and flexible UAV deployment of sensors, Sensors 17 (1) (2017) 1–13.

[4] M. Anderson, Technology device ownership: 2015, Pew Reaserch Center, August 2015. URL http://www.pewinternet.org/2015/10/29/technology-device-ownership-2015/. (Retrieved 16 February, 2017).

[5] J. Poushter, Smartphone ownership and internet usage contributes to climb in emerging economies, Pew Research Center, February 2016. URL http://www.pewglobal.org/2016/02/22/smartphone-ownership-and-internet-usage-continues-to-climb-in-emerging-economies/. (Retrieved 18 February, 2017).

[6] A. Compton, J. Pecarina, N. Lesch, PeerAppear: A location-aware framework for extensible image annotation and peer-to-peer discovery, in: 2016 International Conference on Collaboration Technologies and Systems (CTS), IEEE, 2016, pp. 225–232.

[7] A. Compton, J. Pecarina, PeerAppear: A P2P framework for collaborative visual localization, in: Proceedings of 29th International Technical Meeting of the Satellite Division of the Institute of Navigation, ION GNSS+ 2016, ION, 2016, pp. 1080–1090.

[8] J. Raquet, whats next for practical ubiquitous navigation? Inside GNSS (Sep./Oct.) (2013) 61–69.

[9] M. Haklay, P. Weber, OpenStreetMap: User-generated street maps pervasive computing, IEEE Pervasive Comput. 7 (4) (2008) 12–18.

[10] Y. Wang, X. Liu, H. Wei, G. Forman, C. Chen, Y. Zhu, Crowd Atlas: Self-updating maps for cloud and personal use, in: MobiSys 2013, 2013, p. 27.

[11] P. Maymounkov, D. Mazieres, Kademlia: A peer-to-peer information system based on the xor metric, in: Peer-To-Peer Systems, Springer, 2002, pp. 53–65.

[12] A. Kovačević, N. Liebau, R. Steinmetz, Globase.KOM — A P2P overlay for fully retrievable location-based search, in: Proceedings — P2P — 7th IEEE International Conference on Peer-To-Peer Computing, 2007, pp. 87–94.

[13] M. Picone, M. Amoretti, F. Zanichelli, GeoKad: A P2P distributed localization protocol, in: 2010 8th IEEE International Conference on Pervasive Computing and Communications Workshops, PERCOM Workshops 2010, 2010, pp. 800–803.

[14] G. Brambilla, M. Picone, M. Amoretti, F. Zanichelli, An adaptive peer-to-peer overlay scheme for location-based services, in: Proceedings — 2014 IEEE 13th International Symposium on Network Computing and Applications, NCA 2014, 2014, pp. 181–188.

[15] M. Picone, M. Amoretti, F. Zanichelli, G. Conte, Location-aware overlay scheme for vehicular networks, Researchgate.net (2011).

[16] M. Picone, M. Amoretti, F. Zanichelli, Proactive neighbor localization based on distributed geographic table, Int. J. Pervasive Comput. Commun. 7 (3) (2011) 240–263.

[17] M. Picone, M. Amoretti, M. Martal, F. Zanichelli, G. Ferrari, Combining geo-referencing and network coding for distributed large-scale information management, Concurr. Comput. 27 (13) (2015) 3295–3315.

[18] M. Picone, M. Amoretti, F. Zanichelli, Evaluating the robustness of the DGT approach for smartphone-based vehicular networks, in: Proceedings — Conference on Local Computer Networks, LCN, 2011, pp. 820–826.

[19] C. Gross, D. Stingl, B. Richerzhagen, A. Hemel, R. Steinmetz, D. Hausheer, Geodemlia: A robust peer-to-peer overlay supporting location-based search, in: 2012 IEEE 12th International Conference on Peer-to-Peer Computing, P2P 2012, 2012, pp. 25–36.

[20] P. Brunisholz, F. Rousseau, A. Duda, DataTweet for user-centric and geo-centric IoT communications, in: Proceedings of the 2nd Workshop on Experiences in the Design and Implementation of Smart Objects, ACM, 2016, pp. 29–34.

[21] G. Neimeyer, Geohash Tips & Tricks, 2013. http://geohash.org/site/tips.html.

Please cite this article in press as: A.J. Compton, et al., PeerAppear: A distributed geospatial index supporting collaborative world model construction and maintenance, Future Generation Computer Systems (2018), https://doi.org/10.1016/j.future.2017.12.025.

Andrew Compton is a Ph.D. student studying electrical engineering at the Air Force Institute of Technology and an active duty Captain in the United States Air Force. His research interests include embedded systems, computer vision, and artificial intelligence. Andrew previously attended the Air Force Institute of Technology where he received a masters degree in computer engineering with an emphasis in artificial intelligence and California State University Long Beach where he received a bachelor's degree in computer engineering.

John M. Pecarina, Ph.D., is an Assistant Professor of Computer Science at the Air Force Institute of Technology teaching courses in Systems: Operating Systems, Distributed Systems and Data Security/Cryptography. He advises research for several graduate students in content based image retrieval, distributed systems, software engineering, and pattern recognition. Maj. Pecarina also serves as the curriculum chair for the cyber operations and cyber security graduate Master's Degree programs, which designates AFIT as the Center of Academic Excellence in Cyber Operations by the NSA.

Alan C. Lin is an Assistant Professor of Computer Science in the Department of Electrical and Computer Engineering at the Air Force Institute of Technology, Wright–Patterson AFB, OH. His research focuses on cyber training and education, digital forensics, software engineering, and space systems. He is an acquisition officer (SPRDE-III, STM-II) and in the space cadre.

Kenneth Hopkinson is a Professor of Computer Science at the Air Force Institute of Technology. His research interests include distributed systems, computer security, networking, and simulation. Dr. Hopkinson's research centers on the construction of secure and reliable distributed systems geared towards mission-critical environments, such as network-centric warfare platforms for use in military tacticals scenarios, monitoring and control software targeted towards the electric power grid, remote sensing, sensor fusion, cryptography, space applications, and group-communication systems designed for the specific needs of cyber-physical systems.

Please cite this article in press as: A.J. Compton, et al., PeerAppear: A distributed geospatial index supporting collaborative world model construction and maintenance, Future Generation Computer Systems (2018), https://doi.org/10.1016/j.future.2017.12.025.