An Overlay Network for Forwarding Symbolically Addressed Geocast Messages

Frank Dürr
Institute of Parallel and Distributed Systems, Universität Stuttgart
Universitätsstraße 38, 70569 Stuttgart, Germany
phone: +49-711-7816-431 fax: +49-711-7816-424
frank.duerr@informatik.uni-stuttgart.de

***Abstract*— Geocast is a new and promising communication paradigm for a broad range of applications that allows to send messages to all hosts located in geographic target areas. Geocast target areas can be specified either by geometric figures or symbolic addresses, such as `/usa/fl/miami/market-street/`.**

In this paper, we present a novel geocast routing protocol for symbolically addressed messages. Compared to geocast protocols based on geometric information, our protocol can operate on simple symbolic location models, and message forwarding does not require costly geometric operations. The proposed protocol is based on an overlay network that is mapped to an IP-based network infrastructure. The overlay network is structured in a hierarchical fashion. However, a simple hierarchical architecture may lead to long message paths and bottlenecks at top-level routers. Therefore, we propose an optimized geocast routing approach that avoids these shortcomings. This approach is composed of two parts: First, an overlay network architecture that optimizes message forwarding by adding "shortcut" links to the hierarchically structured overlay network. These shortcuts lead to shorter overlay network paths and reduce router load by by-passing routers of the hierarchy. We present both, a static and a dynamic heuristics to add shortcuts to the hierarchical overlay network.

Secondly, routing is optimized by utilizing a light-weight layer 3 multicast protocol to create shortest path trees in the underlay network that are more efficient than the trees created by pure overlay routing. Although our approach does not rely on a layer 3 multicast protocol, we show how frequent message transfers to certain target areas can be optimized by establishing optimal multicast trees. Our evaluations show that these optimizations improve routing quality considerably.

***Index Terms*— geocast, overlay network, routing, communication network, location-based service**

I. Introduction

With geocast, messages can be sent to all mobile and stationary hosts currently located in a geographic target area. The availability of wireless communication technologies and various indoor and outdoor positioning systems enable a wide spectrum of promising geocast applications like disaster warning, location-based information services, or location-based advertising. In these different applications, the size of the target areas may vary greatly, like from a state, town, or street to a floor or room of a building.

Geocast messages can be addressed either by geometric figures like polygons, or by symbolic names like city names, room numbers, etc. A simple example of a symbolic address is `/usa/fl/miami/`, which denotes the city of Miami in the state of Florida in the USA. Although geometric figures are very flexible and can describe arbitrary areas, symbolic addressing also has several advantages over geometric addressing. With symbolic addressing, target areas can be specified by addresses similar to those people are using in everyday life. Therefore, symbolic addresses are more intuitive to use than geometric ones. Also people and applications tend to address messages to real-world locations, such as rooms or buildings, rather than to abstract spatial areas in a geometric coordinate system. Consequently, symbolic addressing is an important alternative to geometric addressing.

Even if symbolic addresses are used at the geocast service interface, routing can still be based on geometric addressing if symbolic addresses are mapped onto geometric ones. However, to be able to perform this mapping, we need a complex location model including geometric descriptions of *every* symbolically addressable location. For example, to address areas within a building, a three-dimensional geometric model including geometric representations of every room, floor, etc. would be required. Moreover, if geocast routing is based on geometric information, forwarding decisions require the comparison of geometric figures, which may be rather costly operations, even if approximated geometries as proposed in [1] are used. If geocast routing is based on symbolic addresses instead, the modeling effort is reduced to a minimum as location models can be represented by hierarchical name spaces without any need for geometric representation. Additionally, forwarding decisions only require simple prefix matching operations rather than costly geometric operations.

In this paper, we propose a geocast routing protocol purely based on symbolic addresses. The proposed protocol provides a global geocast service for any size of target area. Since this protocol is implemented in an overlay network being mapped onto an IP-based underlay network, it can be deployed gradually and without modifying existing IP network infrastructures. The overlay network is basically structured hierarchically according to a spatial partitioning of the world. However, a strict hierarchical architecture may lead to suboptimal message paths and bottlenecks at top level routers. Therefore, we propose optimizations of the hierarchical routing strategy that improve routing in terms of bandwidth-efficiency and scalability. Bandwidth-efficiency is increased by reducing the number of hops a message has to traverse. Scalability is increased by reducing the message load of routers and avoiding bottlenecks, especially at top-level routers of the hierarchy.

Our optimizations consists of two parts: First, we add shortcuts to the hierarchically structured overlay network. These shortcuts lead to shorter overlay network paths and reduce router load by by-passing routers of the hierarchy. We present both, a static and a dynamic heuristics to add shortcuts to the hierarchical overlay network that both reduce the overhead induced by the set up of shortcuts. Secondly, routing is optimized by utilizing a light-weight layer 3 multicast protocol to create shortest path trees in the underlay network that are more efficient than the trees created by pure overlay routing. Although our approach does not rely on a layer 3 multicast protocol, we show how frequent message transfers to certain target areas can be optimized by establishing optimal multicast trees. Our evaluations show that these optimizations improve routing quality considerably.

The remainder of this paper is structured as follows. In Sec. II, we present related approaches. Then, we introduce our addressing scheme in Sec. III. Afterwards, we describe our system model and the basic overlay network architecture in Sec. IV. The message forwarding algorithm is described in Sec. V together with a multicast-based optimization. In Sec. VI we state the requirements for the following optimizations of the overlay network architecture. In Sec. VII and Sec. VIII we present two heuristics for shortcut installation that optimize the basically hierarchical overlay network architecture. In Sec. IX, we present the evaluation of our approach, before the paper is concluded with a short summary and outlook onto future work in Sec. X.

II. Related Work

The geocast approaches that we consider to be closest to our work are the hierarchical geocast routing algorithms described in [2], [3], and our approach proposed in [4]. These algorithms forward geocast messages along a hierarchy of dedicated geocast routers that are responsible for forwarding messages to geographic areas. Forwarding decisions are made based on comparisons of symbolic or geometric target and service areas. While [2] strictly forwards messages along the router hierarchy, [3] and [4] introduce shortcuts to send messages directly to routers in certain areas. In [4], no strategy is describe to decide, which shortcuts should actually be installed. In [3], a comparably simple strategy is applied. Basically, shortcuts are installed top down, i.e., first shortcuts to country-level routers are installed, then to state-level routers, etc. All router addresses are propaged within the whole network leading to considerable overhead. In contrast to this approach, we try to limit the overhead by being more selective in installing shortcuts and avoiding network-wide address propagation. Furthermore, although our approach does not rely on a layer 3 multicast protocol, we show how to integrate a light-weight multicast protocol to increase efficiency.

Multicast-based geocast routing protocols are also proposed in [5]. With these approaches, each location is assigned an individual multicast address, and an IP multicast protocol, such as Multicast Open Shortest Path First (MOSPF [6]) or the Distance Vector Multicast Routing Protocol (DVMRP [7]), is used to forward messages to the corresponding location. In principle, either a source-based tree or shared-tree protocol can be applied, which both have their limitations. A source-based tree approach may cause significant overhead if messages are sent sporadically—like event notifications—to locations. In the worst case, each message may cause a new tree to be established in the network. While shared trees alleviate this problem, they are not optimal for all senders and add extra complexity for managing rendezvous points. In contrast to that, our geocast approach uses an overlay network for forwarding messages to locations rather than relying on the existence of an IP multicast infrastructure. Of course, this level of independence does not come for free. However, we will propose an optimization which sends the first messages of a sender over the overlay and then switches to a Source-Specific Multicast (SSM [8]) protocol. With this optimization, an individual SSM channel is set up for a sender forwarding a bulk of messages to a location. Due to its source specific nature, SSM avoids the problems associated with shared trees, and the combination of our overlay network and SSM reduces the overhead to set up source-based trees.

In [9], a directory stores the areas covered by subnetworks, so that it can be queried for the addresses of the subnetworks overlapping the target area of a geocast message. This approach only scales up to a small number of subnetworks per target area if messages are sent to the subnetworks via unicast. Using multicast instead leads to an approach similar to the multicast-based approach mentioned above.

It is important to stress that our approach relies on an infrastructure. In contrast to that, geocast approaches for mobile ad hoc networks like the ones described in [10] are based on fundamentally different assumptions. In contrast to these approaches, which typically operate in limited geographic domains, our approach is tailored to the implementation of a *global* geocast service, which is only feasible using a routing infrastructure.

III. Addressing Scheme

Our approach is based on a hierarchical symbolic location model similar to the ones described in [11] and [12]. Figure 1 shows such a symbolic location model.

The location hierarchy consists of a set of locations L and is structured according to the spatial containment relationship. For two locations l_1 and l_2 it holds $l_1 < l_2$, if l_2 spatially contains l_1. l_1 is called a *descendant* of location l_2, and l_2 is an *ancestor* of l_1. A direct descendant of location l is called a *child location* and a direct ancestor a *parent location*. $\mathrm{ancestors}(l)$, $\mathrm{children}(l)$ and $\mathrm{parent}(l)$ denote the set of ancestor locations, child locations, and the parent location, respectively. Each symbolic location has a unique *symbolic address*, which is constructed by concatenating the (relative) addresses on the path from the root of the hierarchy to the location. For instance, location l_1 in Fig. 1 has the symbolic address `/us/ny/new-york-city/`. Target areas may be any location in L identified by its symbolic address. Locations of the location model are used to define host positions and service areas of geocast routers and message servers.

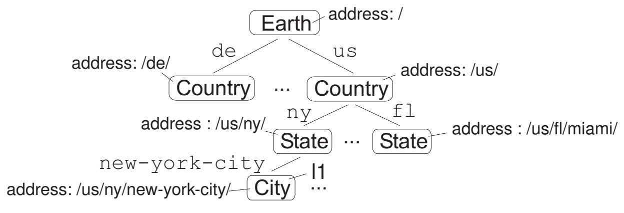
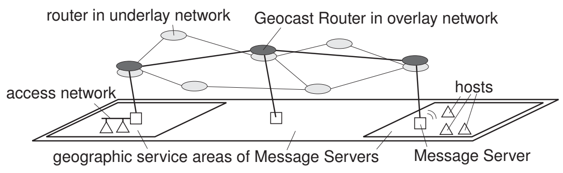


Fig. 1. Symbolic location model



router in underlay network
Geocast Router in overlay network
access network
hosts
geographic service areas of Message Servers
Message Server

Fig. 2. System model

Note that the prefix property of symbolic names reflects the spatial containment relationship. That means, the address of location l_1 is a prefix of the address of l_2, iff $l_1 \geq l_2$. Therefore, spatial inclusion can be determined efficiently by simple prefix matching operations.

IV. System Model and Basic Overlay Network Architecture

The three components of our architecture are hosts, message servers, and routers (cf. Fig. 2):

Geocast Hosts, which can be mobile or fixed, are the recipients and senders of geocast messages. If a message is sent to a given target area, the message has to be delivered to those hosts that currently reside in the target area. We assume that hosts are able to determine their geographic position. For stationary hosts, this position can be configured statically; mobile devices need some positioning system to determine their position.

Geocast Message Servers are responsible for distributing geocast messages to hosts within a certain access network. A message server has a geographic *service area* whose size is equal to the area covered by the access network. Such a service area can be as small as a single floor of a building on which a wireless LAN is installed or as large as a radio cell covering a whole city district. Since the same geographic area may be covered by different access networks, the service areas of message servers may overlap. Every party that wants to be able to distribute geocast messages to the hosts in their access network must set up a message server with an appropriate service area and register this message server with the geocast routers of the overlay network as described later.

Geocast Routers are responsible for forwarding geocast messages from the sender to the message servers whose service areas overlap with the target area of the message, so that these message servers can further distribute the message within their access networks. Geocast routers are arranged in an overlay network and exchange messages using the UDP service offered by the underlying IP-based Internet infrastructure.

In this paper, we focus on the forwarding of messages between access networks through an overlay network of geocast routers. The efficient local distribution of messages within an access network by message servers is beyond the scope of this paper. A message server may simply broadcast a received message within its access network and hosts filter messages according to their current location. A more efficient approach could utilize multicast to address only hosts in certain geographic parts of an access network.

A. Basic Overlay Network Architecture

Basically, the overlay network of geocast routers is structured hierarchically. Each location l is associated with one *designated geocast router*, say r. l is called the *service area* of r, denoted by $\mathrm{s}(r) = l$. A designated router of a location is set up by configuring the address of its service area manually. A designated router with service area l is also designated router of each location $l' < l$ if no other router has been set up for some location l'' with $l' \leq l'' < l$. For instance, a router that is designated router of a city is also designated router of a city district of this city as long as no city district router has been set up.

A router has at least links to the designated router of the parent location of its service area and to designated routers of child locations of its service area. For instance, the New York City router has links to the US state router (parent router) and to the borough routers of New York City, e.g., to the Manhattan router (child router). With these links, the basic overlay network is structured hierarchically resembling the location hierarchy.

The links of the overlay network hierarchy are established top down. For example, first the links from the earth router to the country routers are established, then links from country routers to state routers, etc. In order to join the overlay network, a router, say r_{new}, must know some other bootstrap router, $r_{\mathrm{bootstrap}}$, that is already part of the overlay network. This router can either be configured manually (e.g., the earth or country routers should be well known), or a mechanism like expanding ring search can be used in the underlay network to find this router. Consider for instance the New York City router (r_{new}) that wants to join the overlay network. The following steps are executed to integrate r_{new} into the overlay network:

1) r_{new} sends a "parent discovery" message containing its own UDP address and service area address $\mathrm{addr}(\mathrm{s}(r_{\mathrm{new}}))$ to $r_{\mathrm{bootstrap}}$ (Fig. 3 ❶).
2) The routers in the overlay network forward this message as described in detail in Sec. V to the router, r_p, whose service area address $\mathrm{addr}(\mathrm{s}(r_{\mathrm{p}}))$ is the longest prefix of $\mathrm{addr}(\mathrm{s}(r_{\mathrm{new}}))$ of all routers that are already part of the overlay network (❷). Consequently, r_p is the router with the smallest service area that contains r_{new}'s service area. In the example, this is the New York State router, whose service area address is a longer prefix of the New York City location address than the addresses of the USA or earth location.
3) When r_p receives a "parent discovery" message, it sends its own UDP address and its service area address $\mathrm{addr}(\mathrm{s}(r_p))$ to r_{new} (❸).

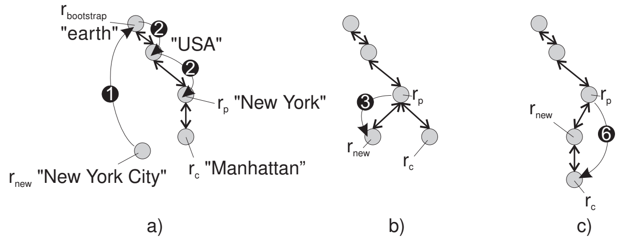


Fig. 3. Steps of overlay network construction. Only parent-child links and all links between r_p, r_c, and r_{new} are shown for clarity. Location addresses of service areas are given in parenthesis

4) r_{new} adds an entry $[\text{addr}(\text{s}(r_p)) \mapsto$ UDP addr. of $r_p]$ to its routing table to establish a new child-to-parent link.
5) r_p adds an entry $[\text{addr}(\text{s}(r_{\text{new}})) \mapsto$ UDP addr. of $r_{\text{new}}]$ to its forwarding table to establish the new parent-to-child link.

After the steps 1–5, r_{new} is integrated into the overlay network as shown in Fig. 3b. However, problems may arise if routers do not join the overlay in strict top down order. Consider for instance the Manhattan router joining the overlay *before* the New York City router. Then, the Manhattan router has already established a parent-child link with the New York State router when the New York City router sends the parent discovery message to the New York State router in order to join the overlay. Using the algorithm described so far, the Manhattan router would not notice that it has to change its parent router. Therefore, the parent router checks its own routing table for an entry $[\text{addr}(\text{s}(r_c)) \mapsto$ UDP addr. of $r_c]$, where $\text{addr}(\text{s}(r_{\text{new}}))$ is a prefix of $\text{addr}(\text{s}(r_c))$ when it receives a parent discovery message. If it finds such an entry, it sends a “new parent” message to r_c to notify r_c of its new parent router (**❻**). Then, r_c established a parent-child link with its new parent by sending a parent discovery message to r_{new}. In our example, the New York State router sends a new parent message to the Manhattan router when it receives the parent discovery message sent by the New York City router. Then, the Manhattan router will register with the New York City router.

In order to keep forwarding tables up-to-date and detect failed or disconnected child or parent routers, heartbeat messages between child and parent routers are exchanged periodically.

Additional *shortcut links* allow for direct connections between routers by by-passing routers of the hierarchy. A shortcut is a unidirectional overlay link between the source router of the shortcut and the target router. Only the source router knows about the shortcut, i.e., only the source router has an entry in its routing table for each shortcut starting at the source router. Thus, the messages can only traverse the shortcut in a source-to-target-router direction. The service area of the target router is called the target area of the shortcut. For instance, a shortcut from the city router of Berlin (source router) to the USA country router (target router) can be used to send messages directly from Berlin to the USA (target area of shortcut) by-passing for instance the country router of Germany. In Sec. VII and Sec. VIII, we describe in detail how shortcuts are selected and added to the overlay network.

V. Geocast Routing Algorithm

In this and the next two sections, we give a detailed description of our optimized geocast routing strategy. The strategy is composed of two parts: First, a routing algorithm to forward messages to all access networks in the target area. Secondly, heuristics to adapt the overlay network by adding shortcuts in order to optimize overlay geocast message paths. We first present the geocast routing algorithm in this section, and then two heuristics for overlay network adaptation in the following sections.

The routing algorithm described in the next subsection operates in the overlay network. It only requires unicast communication primitives to exchange messages between overlay network routers. However, we also present an optimized routing strategy in the second part of this section that utilizes a light-weight layer 3 multicast-protocol to optimize the bulk transfer of messages to certain target areas.

A. Overlay Network Routing Algorithm

The routing algorithm executed in the overlay network consists of three phases. In the first phase, the message is forwarded from the sending host via geocast routers of the overlay network to the designated geocast router, r_a, of the target area, say a. For instance, a message from Berlin to Manhattan is forwarded to the designated Manhattan router. In phase 2, the message is distributed among all geocast routers in the target area by simply forwarding the message down the hierarchy starting at r_a. In the example, the borough router of Manhattan forwards the message to all street routers in Manhattan, the street routers to all building routers in Manhattan, etc. The routers reached in phase 2 also forward the message to the message servers of access networks intersecting the target area, e.g., to a message server of an access network covering one floor of a a multi-story building in Manhattan. In the third phase, the message is distributed to the geocast hosts within access networks by the corresponsing message server. In this paper, we focus on the first two phases of routing, i.e., on the forwarding of messages to the access networks in the target area via the overlay network of geocast routers. We assume that within access networks messages are distributed using a broadcast mechanism and host-based filtering according to the host’s position.

Algorithm 1 shows the details of the routing algorithm. To find out the current phase of message forwarding, r first checks a flag in the message header (lines 2 and 16). If this flag signals phase 2, then r forwards the message to all child routers in the target area (line 18).

Otherwise, r determines the next hop router, r_{next}, by searching for the router whose service area address is the longest prefix of the target area address (line 3, S denotes the service areas of all routers in r’s routing table). If r finds itself to be the router with the longest prefix address, then r is the designated router of the target area that starts phase 2 (line 4–7). If r finds another router, message forwarding is still in phase 1, and r forwards the message to r_{next} that is closer to the designated router. r_{next} may be a router reached via a shortcut. If r cannot find a router whose service area address is a prefix of the target area address, then it simply forwards the message to its parent router. In this case, the message traverses the hierarchy.

Consider, for instance, a message to Manhattan (`/us/ny/new-york-city/manhattan/` reaching the Miami city router. If the Miami router has shortcuts to the USA country router (`/us/`) and the state router of New York (`/us/ny/`), then the longest prefix match will return the New York State router since `/us/ny/` is a longer prefix of `/us/ny/new-york-city/manhattan/` than `/us/`. If the Miami router has no shortcuts installed, it forwards the message to its parent router, namely the state router of Florida.

```
On receiving message $m$ with target area $a$ do
  if $\text{phase}(m) = 1$ then
    $r_{\text{next}} \leftarrow$ longest_prefix_match($a$, $S$)
    if $r_{\text{next}} = r$ then
      // $r$ is the designated router of the target area.
      // $r$ starts phase 2.
      $\text{phase}(m) \leftarrow 2$
    else
      // phase 1: forwarding to target area
      if $r_{\text{next}} = undef$ then
        $r_{\text{next}} \leftarrow$ parent router of $r$
      fi
      forward message to router $r_{\text{next}}$
    fi
  fi
  if $\text{phase}(m) = 2$ then
    // phase 2: distribution within target area
    forward $m$ to each child router $r_c$ with $\text{s}(r_c) \leq a$
    forward $m$ to each message server $gms$ with
      $\text{s}(r) = \text{s}(gms) \cap a$
  fi
```

Algorithm 1. Forwarding algorithm executed by router r

In order to forward messages to message servers in phase 2 (line 19), we assume that each message server, gms, registers with each router, r, with $\text{s}(r) \leq \text{s}(gms)$ by sending a register message via geocast to the target area $\text{s}(gms)$. For instance, a message server associated with an access network covering Manhattan registers with the Manhattan router, every street router in Manhattan, etc. In order to prevent a message server from receiving the same message multiple times in phase 2 from different routers, only router r with $\text{s}(r) = \text{s}(gms) \cap a$ actually delivers a message targeted at area a to gms. Note that the intersection $\text{s}(gms) \cap a$ can be calculated efficiently by simply chosing the shorter address of $\text{s}(gms)$ and a. For instance, a message to New York City is delivered once by the Manhattan router to a message server covering Manhattan rather than multiple times by street or building routers in Manhattan.

B. Optimized Multicast-based Routing Algorithm

The overlay network routing algorithm presented in the previous subsection does not rely on a layer 3 multicast protocol, and thus can be deployed without modification of the existing layer 3 infrastructure. However, it is clear that an overlay network approach can hardly distribute messages as bandwidth-efficient as a layer 3 approach. Therefore, we present now how our overlay network approach can be integrated with a layer 3 multicast protocol to optimize message forwarding.

The basic idea is to start delivering messages to message servers via the overlay network and then switch to layer 3 multicast. Since switching to a layer 3 multicast tree introduces additional overhead, such an optimization is especially useful if several messages are sent frequently to the same target area rather than only single messages that are sent sporadically. Therefore, we distinguish between two modes of geocast message forwarding that are offered to applications on a host. The *message mode* is tailored to communication patterns where applications sporadically send messages to frequently changing target areas, e.g., the transfer of short textual warning messages to different endangered areas. In this mode, messages are forwarded solely via the overlay network without optimization. The *streaming mode* is for scenarios where applications send streams of messages over a longer period of time to certain target areas, e.g., periodical advertisements including images and product descriptions that are sent from a company's host to potential customers in a shopping mall. The application explicitly requests to send a stream of messages to a target area and also signals when it has no more messages to send. In streaming mode, the protocol establishes a multicast tree that exists as long as the stream exist, i.e., as long as applications intend to send messages from the host to the target area.

A geocast stream can be identified by a (source,target area) pair, where "source" is the sending host. In order to get an efficient layer 3 multicast trees, the utilized layer 3 protocol should create a source-based tree, where the sending host is the root and the message servers in the target area are the leaves. The class of source-specific multicast (SSM [13]) protocols is well-suited for our requirements. SSM defines so-called channels (S, G), which are identified by unicast source address S and multicast address G. In our context, S is the sending host and G identifies the addressed target area. A SSM protocol like Protocol Independent Multicast (PIM) SSM (a subset of the PIM Sparse Mode protocol [8]) can build shortest path trees (SPT) rooted at S that can be used for efficient message transfer from S to all message servers in the target area.

In detail, a SPT is built as follows. Sender S starts sending geocast messages to target area a via the overlay network (Fig. 4 ❶). In the header of the message, S sets a "streaming mode bit" that tells the receiving message servers to switch to a SPT tree. Additionally, S maps a to a multicast address, say G, and includes G in the header. The only requirement for this mapping is that if S sends messages to different target areas *at the same time* then these areas must be mapped to different multicast addresses. Note that other senders can re-use the same multicast address for different target areas. Therefore, even a restricted layer 3 multicast address space is no problem in contrast to the multicast-based geocast approaches described in Sec. II where locations are mapped to globally unique multicast addresses. When message server, gms, receives the message, it tells its local designated PIM-SSM router to join channel (S, G) (❷). In PIM-SSM, a SPT for channel (S, G)

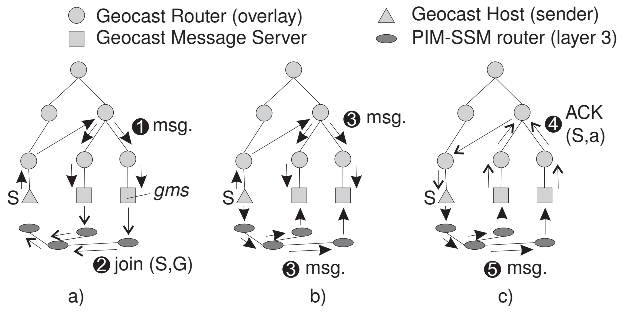


Fig. 4. Optimization using Source-Specific Multicast. a) Sender S sends message to message servers in target area via overlay. message servers join shortest path tree (SPT). b) Message is delivered via overlay and SPT. c) S stops using overlay.

is built by sending join messages from designated routers towards S. PIM-SSM routers traversed by this message become part of the SPT. When the first message server has joined the channel, S starts sending messages via the overlay network *and* via the SPT (❸). gms now receives the same messages via the overlay and the SPT. When gms starts receiving messages via the SPT, it sends an "ACK" message including sender S and target area a to the overlay router from which it also receives these messages (❹). The routers aggregate ACKs of all downstream message servers and routers and forward the aggregated ACKs upstream towards the sender. When all message servers receive the message via the SPT, S finally receives the aggregated ACK for all message servers in the target area from its router and stops sending messages via the overlay network. Now, messages are sent solely via the SPT to the target area (❺).

The usage of the overlay network for the initial message of a stream allows to utilize a light-weight SSM protocol without knowing the sender a priori. Since the SSM trees are source-based, messages can be distributed more efficiently than by using a shared-tree multicast protocol. Moreover, SSM is simpler than a shared tree-protocol. For instance, SSM does not need any rendezvous points. In contrast to source-based multicast protocols used for geocast in the related work, SSM leads to less overhead, e.g., no detailed topological knowledge or pruning of the multicast tree is required. Since we do not set up SPTs for sporadically sent messages, the overhead is further reduced.

VI. Overlay Optimization

In the previous section we showed, how our routing algorithm can forward messages along shortcuts towards the target area. Next, we present heuristics to add suitable shortcuts to the hierarchical overlay network to optimize message paths.

First, we state the requirements for shortcut selection in this section. Then, we present two heuristics in the following sections: a static and a dynamic heuristics.

The static heuristics assumes that messages are mostly sent to target areas close to the sender. Therefore, more shortcuts are set up for nearby target areas. The set of selected shortcuts is fixed.

The second heuristics dynamically adapts the selected shortcuts to the usage pattern of geocast communication. Shortcuts to currently "hot" geographic target areas according to their popularity are installed and constantly updated.

A. Requirements for Shortcut Selection

Before we present our heuristics, we first state the requirements for shortcut selection.

First of all, we want to maximize the utility of installed shortcuts. Each installed shortcut optimizes message paths to certain target areas. However, not all shortcuts are equally useful. The utility of a shortcut is influenced by different factors.

One important factor is the popularity of target areas. Since a shortcut is only effective for certain target areas, the popularity of a shortcut determines how often a shortcut will be used. For instance, it does not make sense to install a shortcut to a certain city in Germany, if the majority of messages are sent to the USA. Note that the popularity of target areas may differ from geocast router to geocast router. If, for instance, the Berlin router in Germany forwards many messages to Munich in Germany, it does not necesserily mean that also the Miami router in the USA forwards many messages to Munich. Therefore, the utility of shortcuts has to be determined individually by each router. Furthermore, the popularity of target areas may change over time. In this case, an adaptive shortcut selection strategy is better suited than a static one.

Also if two target areas have the same popularity, the utility of a shortcut to these areas may differ. The goal of shortut installation is to optimize the overlay network in terms of bandwidth-efficiency and scalability. Bandwidth-efficiency is increased by reducing the number of hops a message has to traverse. Scalability is increased by reducing the message load of routers, i.e., by distributing load among routers and avoiding bottlenecks, especially at top-level routers. Therefore, a shortut that saves many hops or by-passes heavily loaded routers has a great utility.

Beside the positive effects of using shortcuts, the management of shortcuts also causes additional costs:

- Communication overhead: In order to set up a shortcut to a certain geographic area, the designated geocast router of this area has to be determined by the source router setting up the shortcut. In order to determine this target router, the source router sends a query to other routers of the overlay network. Furthermore, a router has to collect metrics to decide which shortcut is worth to be installed by communicating with other overlay routers.
- Space overhead: To set up a shortcut, a router has to store the geographic target area of the shortcut and the UDP address of the target router in its routing table. Additionally, it has to store metrics and further management information to decide which shortcuts are to be installed, when a shortcut is to be refreshed, etc.
- Computational overhead: First, larger routing tables increase the time to make a forwarding decision. Secondly, the algorithm that decides which shortcut is to be installed adds further computational overhead.

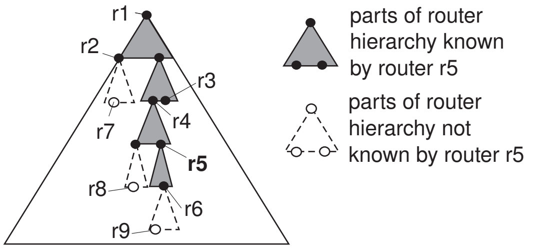

r1
r2
r3
r4
r5
r6
r7
r8
r9
parts of router hierarchy known by router r5
parts of router hierarchy not known by router r5

Fig. 5. Overlay routers known by router r_5

Obviously, these overheads should be minimized.

VII. Static Heuristics for Shortcut Selection

Our first heuristics is based on the assumption that most senders are located geographically close to the addressed target area. That means, nearby areas are more popular than distant areas. Based on this assumption, this heuristics adds more shortcuts to nearby areas than to distant areas. The selection of shortcut target areas is static in the sense that the set of shortcut target areas chosen by a router is fixed. In detail, router r with service area $\mathrm{s}(r)$ established shortcuts to the following set of geocast routers:

- Designated routers of ancestor locations in $\mathrm{ancestors}(\mathrm{s}(r))$. For instance, a city router of city c knows the earth router, and the routers of the country and the state in which c is located. In Fig. 5, r_5 has for example shortcuts to r_1 and r_4.
- Designated routers of child locations of r's ancestor locations, i.e., locations in $\{l \in L \mid \exists l' \in L : l' \in \mathrm{ancestors}(\mathrm{s}(r)) \wedge l \in \mathrm{children}(l')\}$). In the example, the city router all city routers of the state in which c is located, all state routers of the country in which c is located, etc. In Fig. 5, r_5 has shortcuts to r_2 and r_3, but not to r_7, r_8, and r_9.

Note that with this heuristics the routing table of a child router is a superset of its parent router's routing table. Therefore, a new router, r_{new}, joining the overlay network can establish its shortcuts without contacting the shortcut target routers itself by simply copying its parent's routing table. The parent router's routing table is sent to r_{new} as a response to the parent discovery message during the join procedure (cf. Sec. IV-A, step 3). To keep forwarding tables up-to-date, routers send forwarding table updates to child routers and they propagate changes down the router hierarchy.

VIII. Dynamic Heuristics for Shortcut Selection

In the previous subsection, we presented a static shortcut selection heuristics based on the assumption that most messages are sent to target areas close to the sender. Although this is a reasonable assumption for many applications, there might be situations where a significant portion of messages is sent to distant target areas whereas nearby areas are less popular. Moreover, the set of "hot" shortcut target areas may change over time. We now present a heuristics that dynamically adapts shortcuts to the current usage pattern of geocast communication.

A. Selection Algorithm

Our goal is to install only a limited set of shortcuts to relevant target areas. The dynamic selection algorithm has to decide which shortcuts are relevant. Algorithm 2 shows the basic shortcut selection algorithm.

```
On receiving message $m$ with target area $a$ do
  if shortcut $c$ with $\mathrm{targetarea}(c) = a$ does not exist then
    create new shortcut $c$
    $\mathrm{targetarea}(c) \leftarrow a$
    $\mathrm{targetrouter}(c) \leftarrow \mathrm{undef}$
    $\mathrm{refcnt}(c) \leftarrow 0$
    insert $c$ into cache
    if $\mathrm{size(cache)} > size_{\max}$ then
      evict shortcut with minimum utility from cache
    fi
  else
    update utility of shortcut $c$ with $\mathrm{targetarea}(c) = a$
    $\mathrm{refcnt}(c) \leftarrow \mathrm{refcnt}(c) + 1$
    if $\mathrm{refcnt}(c) > refcnt_{min}$ and
      $\mathrm{targetrouter}(c) = \mathrm{undef}$ then
      request shortcut target router with service area $a$
    fi
  fi
```

Algorithm 2. Shortcut selection algorithm executed by router r

The basic idea is to have a shortcut cache of limited size ($size_{\max}$) that is populated with relevant shortcuts. The size limit assures that the space overhead is limited since only information about a limited number of shortcuts has to be managed. Additionally, the computational overhead is limited since the selection and forwarding algorithm only operate on a limited set of shortcuts. Moreover, communication overhead is also limited since only a limited set of target routers needs to be determined and kept up-to-date. Each shortcut is associated with a utility metric that defines the relevance of a shortcut. If a new shortcut, c, is inserted into the cache and the cache has reached its maximum capacity, the shortcut with the smallest utility is evicted from the cache (lines 8–10).

The selection algorithm is executed each time a message is forwarded. First, the algorithm checks if a shortcut to the target area, a, of the message already exists (line 2). If such a shortcut exists, its utility is updated (line 12) as presented later.

Then, a reference[1] counter is increased that counts the number of messages addressed to the shortcut target area since the shortcut was brought into the cache (line 13). The idea is to set up a shortcut to a certain targer area only if the shortcut has proven to be relevant for a certain amount of time rather than when it is considered relevant for the first time. To set up a shortcut to an area, a, a "target router solicited" message is sent to the designated router of the target area using the routing algorithm presented in Sec. V-A (line 16). As response to this request, the target router sends its UDP address back to the source router, and the source router adds an entry in its routing table. Although the request can be piggy-backed onto forwarded traffic, the response is a separate message since we cannot guarantee that messages are also sent in the reverse direction. This leads to communication overhead that should be reduced. Setting up shortcuts to target areas that have been

[1]We say a target area is "referenced" if a message is sent to this area.

relevant for at least certain number of messages (refcnt_{min}) sorts out shortcuts that are only relevant for a short amount of time ("one hit wonders"), since these are evicted soon from the cache. However, this also reduces the reactivity to sudden changes of the relevance of shortcuts since an artificial delay is added.

Next, we show how the utility of a shortcut is defined that determines which shortcuts are considered relevant.

B. Least-unified Value

In order to decide which shortcuts are relevant, a metric is required that defines the utility of shortcuts. As stated in the requirements section the utility of a shortcut is influenced by the popularity of target areas. Since we cannot know for sure where message are sent to in the future, we estimate the future popularity of target areas based on their reference history. We consider two aspects of the reference history of target areas: reference recency and reference frequency. We assume that a target area that was addressed recently is likely to be addressed in the near future, and a target area that was addressed frequently in the past is likely to be addressed frequently in the future. Ideally, we can tune our heuristics to define the influence of recency and frequency on the popularity of target areas. Since one of our requirements is space efficiency, it is prohibitive to store detailed historic information like the reference time of every forwarded message. Instead, we only store aggregated information using an approach proposed by Lee et al. in [14]. Originally, this approach termed Least-Unified Value (LUV) was used for the caching of web pages. We now give a short description of LUV before we show how to adapt this approach for geocast.

Using LUV, the utility of a cached web page, say c, at the current time t is calculated based on the utility at the last reference time t' as follows:

$$\mathrm{utility}_t(c) = \mathrm{utility}_{t'}(c) \times \mathrm{F}(t - t') \qquad (1)$$

If c is referenced at time t, then its new utility is calculated based on the utility at the last reference time t':

$$\mathrm{utility}_t(c) = \mathrm{utility}_{t'}(c) \times \mathrm{F}(t - t') + \mathrm{weight}(c) \qquad (2)$$

With this approach, each reference in the past contributes to the current utility of c, although only the utility at the last reference time has to be remembered rather than detailed information about each single reference in the past. This recuces space overhead considerably. Function $\mathrm{F}(x) = \frac{1}{2}^{\lambda x}$ with tuning parameter λ defines the influence of reference recency and frequency. For $\lambda = 0$, only the frequency of reference is considered by simply counting the number of past references. That means, LUV behaves like a weighted least frequently used (LFU) strategy in this case. For large values of λ, the emphasis is put onto the recency of references. In this case, LUV behaves similar to a weighted least recently used (LRU) strategy.

Beside the popularity of target area in terms of reference frequency and recency, Equation 2 also considers the function $\mathrm{weight}(c)$ that defines the weight of cache entry c. Typically, in web caching, this weight reflects the cost of fetching a web page from the server. This weight function has to be chosen to optimize the desired performance metric. Below we will show how this function has to be adapted to the geocast scenario.

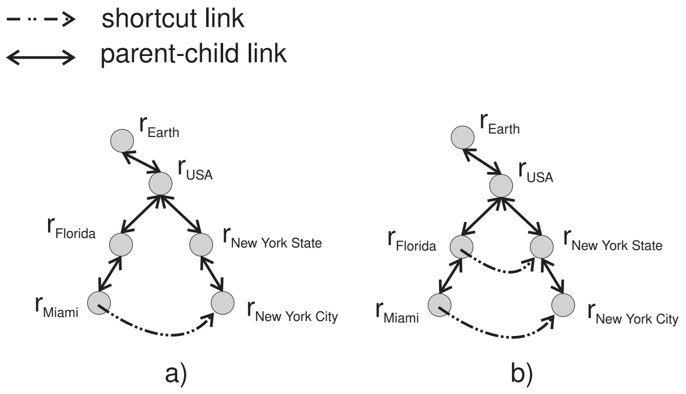


Fig. 6. Metric: number of saved hops.

C. Shortcut Utility

In our context, geographic target areas are referenced rather than web pages. However, we can easily adapt Equation 2 to determine the utility of shortcuts rather than cached web pages.

Let c_a be a shortcut to the designated router of target area a. Each time a geocast router forwards a message addressed to a it updates the utility $\mathrm{utility}(c_a)$ of c_a according to Equation 2.

Using this equation, the calculated utility reflects the popularity of target areas in terms of frequency and recency. The function $\mathrm{weight}(c_a)$ now defines the weight of a shortcut rather than the weight of a web page. For web caching, the weight of a cached web pages is typically calculated based on document size or download latency. However, these values are not very useful to calculate the weight of a shortcut. Instead, we define the weight of a shortcut based on another metric that is related to bandwidth-efficiency and router load: the number of saved overlay hops. This number of saved overlay hops is defined as the number of routers that are by-passed by a shortcut compared to forwarding a message without using the shortcut. Obviously, less overlay hops also means less hops in the underlay network. Thereby, bandwidth is saved. Moreover, less routers are involved in message forwarding, and load is taken from by-passed routers.

Consider for instance a message to New York City arriving at the Miami router in Florida, USA (cf. Fig. 6a). If the message is sent without using a shortcut via the router hierarchy, then 4 hops are required. If the Miami router has a shortcut to the New York City router, this message by-passes 3 routers, namely the Florida router, the USA router, and the New York State router. Therefore, the shortcut has a weight of 3. If another shortcut between the Florida router and New York State router exists (cf. Fig. 6b), then the shortcut from Miami to New York City has a weight of 2 since it only saves 2 hops.

Determining the weight of a shortcut is not trivial since in general a router does not know the distance to other designated routers a priori. Although the distance in terms of overlay hops can be determined easily by sending a probe message to the shortcut target router via the overlay network, this leads to communication overhead. In order to reduce this overhead, we use the following approach. If a router does not know the weight of a shortcut, c_a, to target area a, it first estimates its weight. If later the shortcut is considered to be relevant, the acutal distance is determined when the shortcut is set up by sending a "target router solicited" message to the target router, r_a. Each router that forwards this message increases a counter in the header of this message that counts the hops to router r_a. This hop count is returned together with the UDP address of the target router to the requesting source router of the shortcut. The source router replaces the estimated weight by the actual weight of the shortcut and updates its utility accordingly.

A remaining question is how to estimate the distance to the target router as accurate as possible. If the estimated distance is excessively high, then we overestimate the relevance of a shortcut and falsely replace a shortcut that is actually more relevant. This also increases communication overhead since overestimated shortcuts will soon be replaced when their weight is updated to the actual weight. If the estimated distance is excessively small, then we underestimate the shortcut's relevance and the shortcut will never be installed in favor of shortcuts that are actually less-relevant. This reduces the performance of the overlay network. A fairly accurate estimation can be made based on the service area address, $\mathrm{addr}(\mathrm{s}(r))$, of source router r and the shortcut target area address, $\mathrm{addr}(a)$:

$$\begin{aligned}\mathrm{weight}'(c_a) \;=\;& \mathrm{len}(\mathrm{addr}(\mathrm{s}(r))) + \\ & \mathrm{len}(\mathrm{addr}(a)) - \\ & 2 \times \mathrm{len}(\mathrm{compfx}(\mathrm{addr}(\mathrm{s}(r)), \mathrm{addr}(a))) \\ & -1 \end{aligned} \tag{3}$$

The estimated distance is denoted by Function $\mathrm{weight}'()$. Function $\mathrm{len}()$ calculates the number of address parts of a symbolic address. For instance, the address `/usa/fl/` has a length of 2. Function $\mathrm{compfx}()$ calculates the common prefix of two symbolic addresses. The estimated distance is equal to the path length between source and target router if the messages traverses the router hierarchy. This is the longest possible path between the source and target router. Therefore, the estimated weight will always be greater or equal to the actual weight. Consider, for instance, the Miami and the New York City router. According to Equation 3, the estimated weight is $\mathrm{len}(\text{/usa/fl/miami/}) + \mathrm{len}(\text{/usa/ny/new-york-city/}) - 2 \times \mathrm{len}(\text{/usa/}) - 1 = 3 + 3 - 2 - 1 = 3$. If a shortcut from the Florida router to the New York State router exists, this estimated weight will be updated to 2 when the response to the target router solicited message is received.

D. Hierarchical Shortcuts and Covered Shortcuts

Looking at our routing algorithm, we can see that not only a shortcut directly aiming at the target area router shortens the path to the target area, but also any other shortcut targeted at ancestor areas of the target area. We call these indirect shortcuts hierarchical shortcuts. Consider once more a message to New York City arriving at the Miami router. Obviously, the best shortcut is the direct shortcut from the Miami router to the New York City router. However, also shortcuts from the Miami router to the New York State router or the USA router are useful, since both by-pass for example the Florida State router.

Moreover, we can imagine situations where larger areas can be considered to be more popular than smaller areas below these larger areas in the hierarchy. If for instance the Miami router sends one message to each city Texas, then the cities themselves are not very popular target areas. However, the state Texas can be considered to be a hot target area since many messages are sent to areas within Texas.

Looking at these example, we modify our shortcut selection algorithm as follows: If a message to a target area, say a_t, arrives at the designated router r of area a_r, then all ancestor locations of a_t are referenced recursively up to the smallest area that contains a_t and a_r. That means, Algorithm 2 is executed for each target area between a_t and the least upper bound of a_t and a_r. The address of the least upper bound of a_t and a_r is simply the longest common prefix of the addresses of a_t and a_r. Consider for instance the message to New York City `/usa/ny/ny-city/` arriving the Miami router with the service area `/usa/fl/miami/`. Then, all areas between `/usa/ny/ny-city/` and the least upper bound `/usa/` are referenced, namely `/usa/ny/ny-city/`, `/usa/ny/`, `/usa/`.

However, sometimes it is sensible to stop referencing target areas recursively before reaching the least upper bound, especially if a router already has set up shortcuts to ancestor locations of the target area. In the example above, router r may already has set up a shortcut to the New York State router. In this case, an additional shortcut to the USA country router is useless for messages to New York City since the USA country router will be by-passed by the already existing shortcut to New York State. We say, the New York State shortcut covers the USA shortcut for messages to areas below New York State. In this case, we stop referencing areas recursively when we reach an area to which a shortcuts has already been set up. In the example, we will stop at New York State and do not reference the USA.

E. Implementation

The dynamic shortcut selection algorithm is executed for every forwarded message during runtime. Therefore, it is crucial that this algorithm can be implemented efficiently.

As described in Sec. VIII-B, the shortcut selection algorithm is based on the LUV algorithm. One important property of the LUV algorithm is that it can be implemented using a heap structure containing the shortcuts. With this heap, adding and replacing shortcuts has a complexity of $O(n)$ where n is the number of shortcuts. Therefore, the implementation can be considered to be efficient and capable of managing larger numbers of shortcuts also for high message rates.

The shortcut selection algorithm decides about the relevance of shortcuts based on their reference history. As described in Sec. VIII-B, LUV has to manage two simple values for each

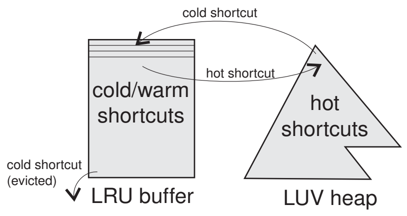


Fig. 7. Internal router architecture: The LUV heap of restricted size manages hot shortcuts. The LRU buffer contains cold and warm shortcuts ($\mathrm{sizeof}(\mathrm{LUV\ heap}) \ll \mathrm{sizeof}(\mathrm{LRU})$). Cold shortcuts evicted from the LUV heap are put into the LRU buffer. Warmed up shortcuts from the LRU buffer may enter the LUV heap when they become hot.

shortcut: the time of the last reference, and the utility at this time. For hot shortcuts, this information is managed by the heap mentioned above. The size of the heap determines the (maximum) number of hot shortcuts that can be installed. The size of this heap is strictly limited to limit the communication overhead that is necessary to set up shortcuts and keep their target router address up-to-date (the UDP address of shortcut target routers is refereshed periodically for instance every 60 seconds to determine failed or new routers). Usually, this heap only contains fewer than 100 hot shortcuts. However, this also means that historic information about cold and warm shortcuts is lost when they are evicted from the LUV heap. However, this may prevent a cold shortcut from becoming a hot one since it has no chance to "heat up" and probably replace a currently hot shortcut after some time.

In order to remember also historic information about such cold and warm shortcuts, we add a separate buffer (cf. Fig. 7). This buffer uses a simple least recently used (LRU) replacement strategy where insertions and replacement operations have a complexity of $O(1)$. Therefore, information about large numbers of shortcuts can be stored efficiently, and this buffer may take up the remaining space of a router's memory. The target router information for shortcuts in this LRU buffer is not requested or refreshed. That means, these shortcuts remain passiv until they become hot shortcuts and the target router of the shortcut is determined using a "target router solicited" message.

Every time a message is forwarded and thereby a target area is reference, the router executes Alg. 2. If the utility of the referenced shortcut is great enough it will be placed into the LUV heap possibly evicting another shortcut from the LUV heap (Alg. 2, line 9). The evicted shortcut is considered cold and placed at the front of the LRU buffer. The evicted shortcut therefore keeps its utility ("temperature") and needs not start from the very beginning ("warm up") the next time it is referenced.

IX. Evaluation

In this section, we evaluate our approach by simulating different scenarios with an implementation of our algorithm for the network simulator ns-2. We first describe the simulation set up, before we present the results in the following subsections.

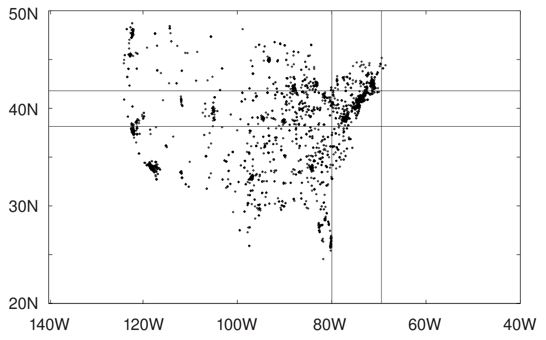


Fig. 8. Geographic positions of underlay network routers. The routers within the marked area with the corners 80W,38N and 70W,42N constitute the underlay network of the simulation.

A. Simulation Set Up

The underlay network topology for our evaluation consists of a real network topology from Liljenstam, Liu, and Nicol [15]. The whole topology consists of the backbones of 8 major Internet service providers in the USA. The topology models routers and their links. Moreover, routers are mapped to geographic positions (latitude, longitude) and symbolic addresses of the form `/country/state/city/`.

The geographic mapping is shown in Fig. 8. Since the whole data set consists of 44,223 routers and 68,681 links, we have to select a subset of routers to get a manageable network topology for our simulations. We only consider routers from one autonomous system that are within the geographic area marked in Fig. 8. We finally end up with 2942 underlay network routers connected by 3302 links.

The overlay network consists of country, state, city, and city district geocast routers. These overlay routers are placed at underlay network routers. Since an algorithm for automatic placement is beyond the scope of this paper, we use a simple heuristics for overlay router placement: A designated geocast router of a certain geographic area is placed at an underlay network router located in this area. For instance, we place the New York City geocast router at an underlay network router with the geographic position `/usa/ny/ny-city/`.

Geocast message servers are placed at each underlay network router with exactly one link. We assume that each message server is responsible for a different access network covering a city district.[2] In the evaluation, we consider the forwarding of messages from senders to message servers, i.e., to access networks in the target area.

In our simulations, we address target areas at the city district level, i.e., at the leaf level of the location hierarchy. Our heuristics are targeted at different communication patterns. The static heuristics assumes that most messages are sent to target areas close to the sender. The dynamic heuristics loosens this assumption and only assumes that certain "hot" target areas exist that are more popular than others. In order to see the influcence of these assumptions, we evaluated both heuristics with different communication patterns. First of all, we vary the intensity of hotspots:

[2]Since the geographic mapping of the underlay network only has city granularity, modeled city districts do not correspond to real city districts.

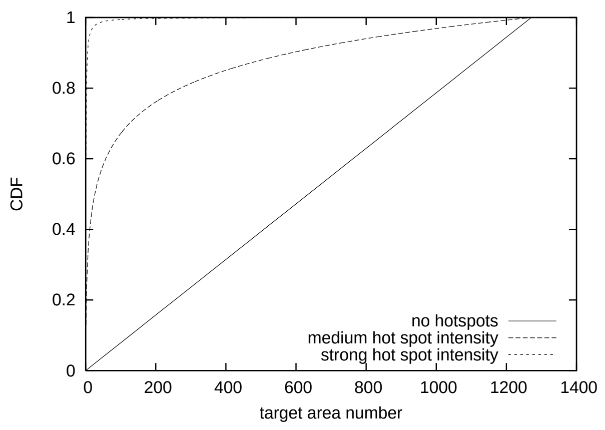


Fig. 9. hotspot intensity (cumulative distribution function; fraction of messages addressed to a certain number of target areas)

- no hotspots: all target areas are equally popular
- medium hotspot intensity: there are many hotspots, however, each one is only of medium popularity
- strong hotspot intensity: there are only few hotspots; these few hotspots are very popular

To create these different hotspot intensities, we use a Zipf distribution. Figure 9 shows the cumulative distribution function for different hotspot intensities. For medium hotspot intensity, for example, about 80% of all messages are sent to about 25% of the target areas (1271 city districts). Each of the senders choses its favorite hotspots independently according to the distribution shown in Fig. 9. That means, a sender in New York City may chose other hotspots than a sender in Arlington, Virginia. In order to evaluate the influence of the assumption of the static heuristics that most messages are sent to target areas close to the sender, we used two different geographic hotspot distributions:

- Local geographic hotspot distribution: Most messages are sent to target areas close to the sender. Each sender orders all possible target areas increasingly according to the geographic distance to the sender. Then, he choses hotspots according to the distribution functions shown in Fig. 9.
- Random geographic hotspot distribution: Messages to distant target areas are as likely as to target areas close to the sender. Each sender orders all possible target areas randomly and choses hotspots according to the distribution functions shown in Fig. 9.

B. Path Quality

Our shortcut heuristics are designed to minimize the number of hops traversed by a message from the sender to the designated router of the target area since the number of traversed hops is related to router load and bandwidth-efficiency as mentioned earlier. In this subsection, we evaluate the path quality achieved by our routing algorithm. As metric we use the *stretch factor*. The stretch factor denotes the factor by which the achieved underlay network path length is longer than the optimal path length in the underlay network. A stretch factor of 2 for instance means that the path is twice as long as the optimal path and therefore twice as much message copies (and bandwidth) are required as in the optimal case. The optimum is defined by single source shortest path trees (SPT). The source of a SPT is the sender of the message. Trees are pruned such that they only contain branches leading to message servers in the target area. The calculated SPTs are similar to source-based trees calculated by layer 3 multicast protocols like MOSPF or PIM-SSM. Therefore, using a SPT as reference gives a realistic comparison. At the same time, this shows the gain we can achieve by using the multicast optimization (SSM) presented in Sec. V-B.

Figure 10 shows the developing of stretch factors for the random geographic distribution. The simulation runs for 200 s. The dynamic heuristics starts with an empty shortcut cache that is populated during the simulation. The static heuristics has all shortcuts installed from the beginning, however, it does not adapt the chosen shortcuts for its static nature.

First of all, we see that the static heuristics has a constant stretch factor of about 2.5, independently of the intensity of hotspots. Since with the random geographic hotspot distribution often messages are sent to distant target areas, the static heuristics cannot achieve short paths. Often, only the shortcuts to state routers can be used (using the static heuristics, each city district router in the USA has shortcuts to all US state routers). Since the static heuristics does not adapt the chosen shortcuts, it achieves a constant stretch factor.

In contrast to the static heuristics, the dynamic heuristics adapts the chosen shortcuts during the simulation run. In all our simulations, each router has a shortcut cache of 32 hot shortcuts (LUV heap; cf. Sec. VIII-E) and 100 cold shortcuts (LRU buffer). The dynamic heuristics sets up shortcuts to state routers rapidly. Therefore, the stretch factor rapidly approaches 2.5, also if no hotspots at the city district level can be identified (curve “dynamic heuristics, no hotspots” in Fig. 10). In this case, the heuristics fills up the cache with the largest areas, i.e., states in our simulation. For medium and especially for strong hotspots intensity, the dynamic heuristics choses shortcuts to smaller areas that are considered hot. This leads to stretch factors that can come close to the optimum of 1 for strong hotspot intensity (note that the optimum 1 can hardly be achieved by an overlay approach).

The results are different if messages are mostly addressed to target areas close to the sender (Fig. 11). Now, the static heuristics benefits from the large number of shortcuts to nearby areas and comes close to the optimum of 1 if hotspot intensity is high. The dynamic heuristcs also adapts its shortcuts in this case and achieves results comparable to the random geographic distribution. This shows the independence of the dynamic heuristics of the geographic distribution of target areas. However, the dynamic heuristics does not reach the performance of the static heuristics in this case, which is due to the fact that the dynamic heuristics sets up fewer shortcuts than the static heuristics to reduce the overhead of shortcut set up.

These results show that the heuristics achieve good results, especially as expected for strong hotspot intensity. Depending on the communication pattern, the presented multicast-based

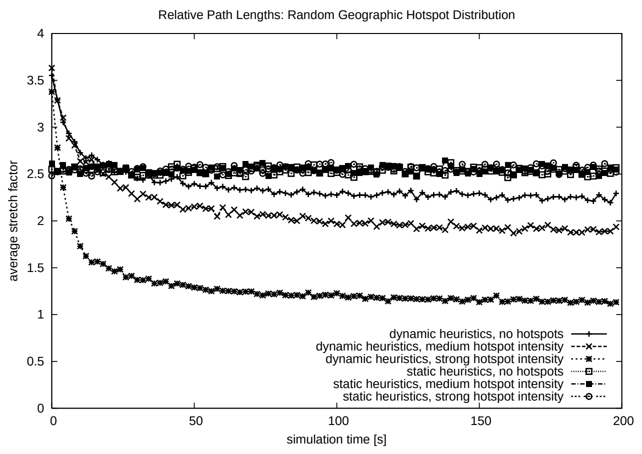


Fig. 10. Developing of stretch factors for random geographic target area distribution

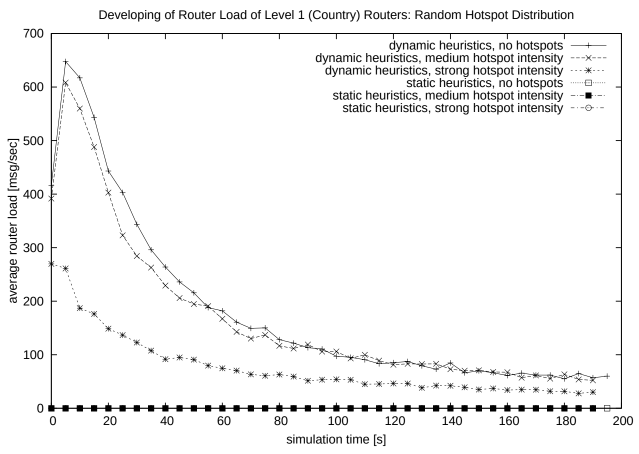


Fig. 12. Developing of router load of country router for random geographic hotspot distribution.

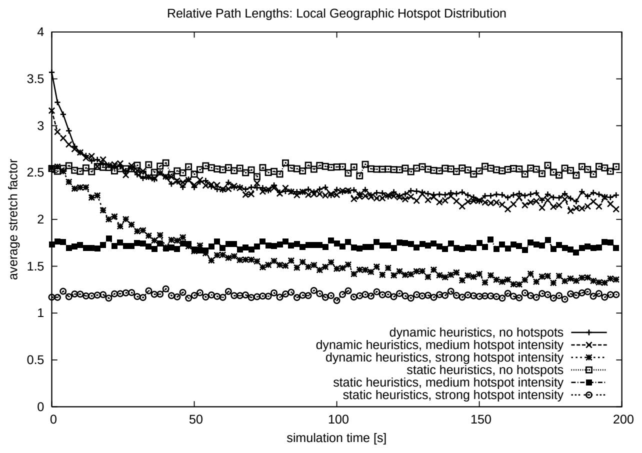


Fig. 11. Developing of stretch factors for local geographic target area distribution

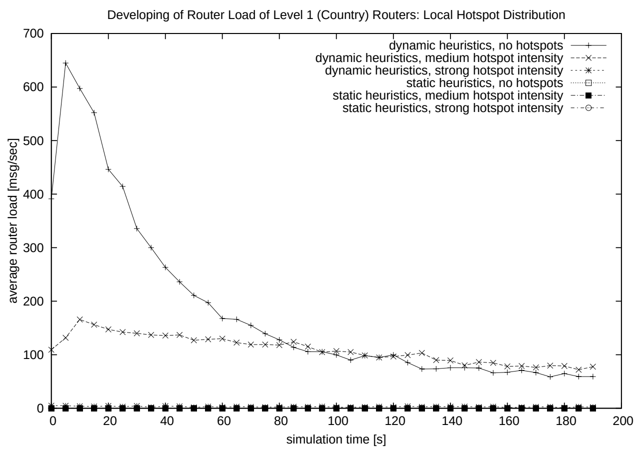


Fig. 13. Developing of router load of country router for local geographic hotspot distribution.

optimization can improve paths by a factor up to 2.5, which pays off especially if many messages are sent to a certain target area.

C. Router Load

One important goal of adding shortcuts to the router hierarchy is to prevent routers from becoming bottlenecks. Especially routers at the top of the hierarchy like country or state routers can become bottlenecks if they are traversed by many messages. These potential bottlenecks should be bypassed by shortcuts.

Figure 12 shows the load of the USA country router for the random geographic hotspot distribution. Router load is expressed by the number of messages that are forwarded by this country router per second on average.[3] We see that router load decreases rapidly as shortcuts are established if the dynamic shortcut heuristics is used. When the majority of shortcuts is established, the load settles at about 50 message per second. Compared to an overall message rate of 1271 message per second, only about 4% of all messages traverse the country router. Using the static heuristics, the USA country router forwards no messages at all since all routers within the USA have shortcuts to all US state routers.

Figure 13 shows the load of the USA country router if senders mostly address areas geographically close to them. This traffic pattern further reduces the USA country router's load since many message do not have to traverse the country router even without using a shortcut.

Figures 14 and 15 show the load of state routers for the random and local geographic hotspot distribution, respectively. For the dynamic heuristics, results are comparable to the load developing of the country router described above. As soon as shortcuts are set up, router load decreases, especially if few hotspots of high intensity exist. Since now load is distribute among multiple state routers rather than only one country router, the absolute message rate is smaller than the message rate of the country router, especially at the beginning of the simulation when no shortcuts exist.

If the static heuristics is used, results are different. This

[3]Since in our simulation messages are only sent from areas within the USA to targets within the USA, no messages traverse the root router of the hierarchy.

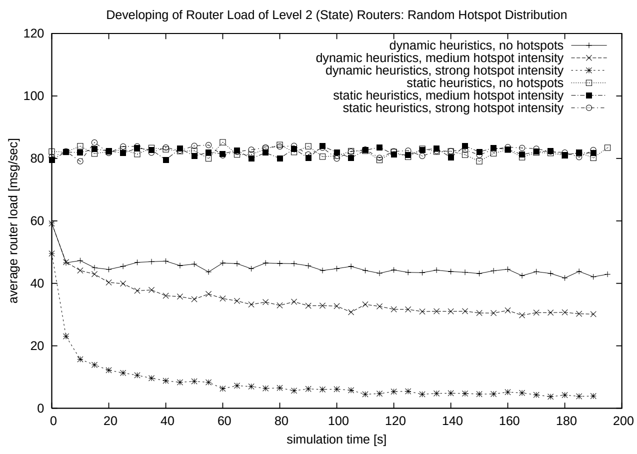


Fig. 14. Developing of router load of state routers for random geographic hotspot distribution.

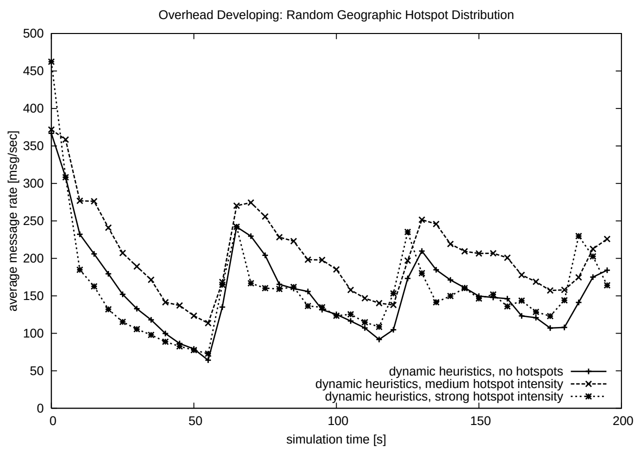


Fig. 16. Communication overhead developing for random geographic hotspot distribution (overall message rate of parent solicited messages)

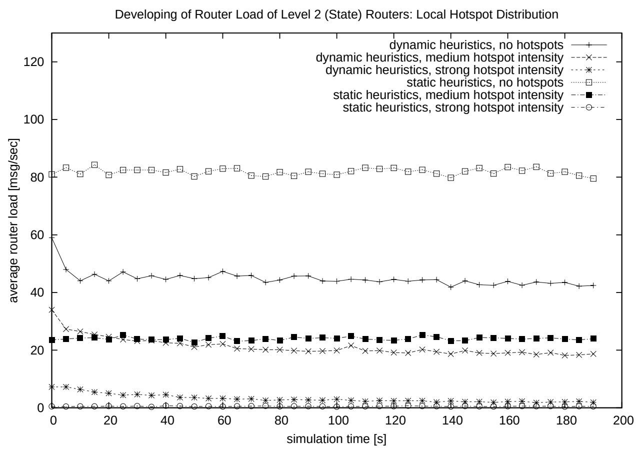


Fig. 15. Developing of router load of state routers for local geographic hotspot distribution.

heuristic only decreases the load of state routers, if hotspots close to the sender exist (Fig. 15, medium and strong hotspot intensity, static heuristics). Then, router load is smaller than the load achieved by the dynamic heuristics. However, if a significant portion of messages is sent to distant target areas, the static heuristics cannot significantly decrease router load, even if (distant) hotspots of strong intensity exist (Fig. 14).

D. Communication Overhead

1) Static Heuristics: If the static shortcut heuristics is used, the routing table of a child router is a superset of the routing table of its parent router. Therefore, the parent router sends its routing table to its child routers, and each child router simply copies the whole parent table. Updates of the routing table are propagated down the hierarchy. When a new router joins the overlay, the complete routing table of its parent is transfered to the new router once by bulk transfer to reduce communication overhead. Afterwards, only changes have to be propagated down the hierarchy. Therefore, the actual overhead depends on the size and dynamics of the overlay network. We may reasonably assume that large parts of the overlay network are almost static, i.e., only few routers join or leave the network. Moreover, changes may occure mainly at the leaf-level of the hierarchy. These changes only cause local update messages. Thus, the overall communication overhead is negligible.

2) Dynamic Heuristics: The dynamic heuristics permanently adapts its shortcuts to the current popularity of target areas. For new shortcuts, the target router has to be determined by sending a target router solicited message to this router. Moreover, the acutal number of saved hops is determined and periodically updated by these messages. Figure 16 and Figure 17 show the communication overhead induced by these messages for random and local geographic hotspot distribution, respectively. These figures show the overall message rate of target router solicited messages. We see that at the beginning of the simulation when shortcut caches are populated, the overhead is high. However, this overhead decreases rapidly when the routers have installed the majority of shortcuts. In our simulation, shortcuts are refreshed every 60 seconds. Therefore, the overhead rises periodically every 60 seconds. This periodic behaviour is due to the fact that all routers are started at the same time, thus many of them refresh their shortcuts at the same time. In reality, an almost constant developing can be expected since routers refresh shortcuts at different times. We also see that overhead settles down after some time when the majority of shortcuts do not change anymore. Compared to the total message rate of 1271 geocast messages per second, the remaining overhead of less than 200 messages per second is small.

E. Space Overhead

1) Static Heuristics: Using the static heuristics, a router installs shortcuts to ancestore locations and children of ancestor locations. This leads to a forwarding table size of $O(dc)$, where d is the depth of the location hierarchy and c is the maximum number of child locations. For a balanced hierarchical location model with n locations, d grows with $O(\log n)$. If we consider typical examples of location hierarchies, d is rarely greater than 10. For instance, a global location hierarchy consisting of the earth, countries, states, cities, streets, buildings, floors, and

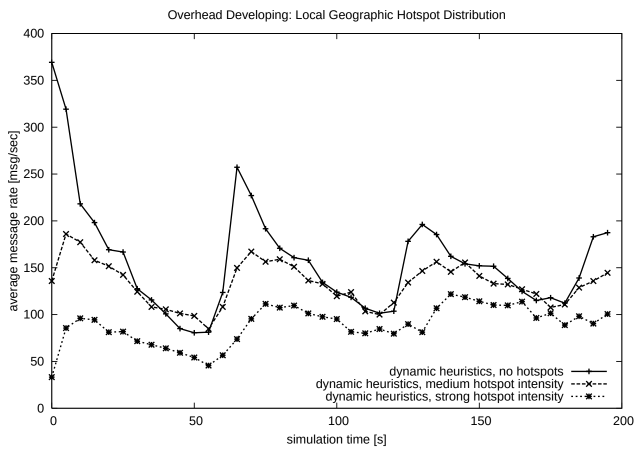


Fig. 17. Communication overhead developing for local geographic hotspot distribution (overall message rate of parent solicited messages)

rooms would lead to $d = 8$. Giving an estimate for c is more difficult. For instance, the number of states within a country is rarely much greater than 50, whereas the number of streets per city can exceed 1000 for large cities. However, big cities will very likely be partitioned into districts, which automatically reduces the number of child locations of the city location. To reduce the number of forwarding table entries, we can also insert artificial locations into the location model to decrease the number of child locations. Therefore, we expect a router to have at most a few 1000 forwarding table entries. Since a shortcut is defined by the target area and UDP address of the target router, it can be stored with with a few hundered bytes. Even for a few thousand shortcuts this poses no considerable challenge for a router.

2) Dynamic Heuristics: The shortcut selection algorithm decides about the relevance of shortcuts based on their reference history. As described in Sec. VIII-B, LUV has to manage two simple values for each shortcut: the time of the last reference, and the utility at this time. Additionally, the target area and UDP address of shortcuts must be stored. Overall, information about one shortcut can be stored with a few hundred bytes. Therefore, we can assume that even the management of thousands of shortcuts is no problem.

X. Summary

In this paper we presented a novel geocast routing approach tailored to symbolic addresses rather than geometric addressing. One advantage of using symbolic addresses is the reduced modeling effort. Our approach can be used to address different sizes of areas down to the room level an below *without* the need for a highly detailed three-dimensional geometric model. Moreover, symbolic addresses are easily understandable by the user and can be handled efficiently by routing components.

The proposed geocast routing protocol is based on an overlay network that is mapped to an IP-based network infrastructure. The overlay network is basically structured in a hierarchical fashion. To optimize bandwidth-efficiency and scalability the hierarchical overlay network is augmented with so-called shortcuts that allow for direct connections between certain locations. These shortcuts lead to shorter overlay network paths and reduce router load by by-passing routers of the hierarchy. We presented two heuristics to add shortcuts, which are aimed at different traffic patterns: First, a static heuristics for situations where most messages are sent to target areas close to the sender. Secondly, a dynamic heuristics that is independent of the geographic distribution of target areas. This heuristics constantly adapts shortcuts to the current popularity of target areas.

We further improved our approach by integrating a lightweight layer 3 multicast protocol. Although our approach does not rely on a layer 3 multicast protocol, we showed how frequent message transfers to certain target areas can be optimized by establishing optimal multicast trees.

Through simulations we showed that these optimizations greatly improve message paths and router load, and only induce small overhead.

References

[1] J. C. Navas and T. Imielinski, “On reducing the computational cost of geographic routing,” Rutgers University, Department of Computer Science, Tech. Rep. DCS-TR-408, Jan. 2000.

[2] Julio C. Navas and T. Imielinski, “Geocast – geographic addressing and routing,” in *Proceedings of the Third Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom ’97)*, Budapest, Hungary, Sept. 1997, pp. 66–76.

[3] J. Roth, “Semantic geocast using a self-organizing infrastructure,” in *Innovative Internet Community Systems (I2CS)*. Leipzig, Germany: Springer, June 2003, pp. 216–228, LNCS 2877.

[4] F. Dürr, C. Becker, and K. Rothermel, “Efficient forwarding of symbolically addressed geocast messages,” in *Proceedings of the 14th International Conference on Computer Communications and Networks (ICCCN 2005)*, San Diego, CA, USA, Oct. 2005, pp. 77–83.

[5] J. C. Navas, “Geographic routing in a datagram internetwork,” Ph.D. dissertation, Rutgers University, Department of Computer Science, May 2001.

[6] J. Moy, “Multicast extensions to OSPF,” IETF, RFC 1584, Mar. 1994.

[7] D. Waitzman, C. Partridge, and S. Deering, “Distance vector multicast routing protocol,” IETF, RFC 1075, Nov. 1998.

[8] B. Fenner, M. Handley, H. Holbrook, and I. Kouvelas, “Protocol independent multicast–sparse mode (PIM-SM): Protocol specification (revised),” IETF, Internet-Draft, Oct. 2004.

[9] T. Imielinski and J. C. Navas, “GPS-based geographic addressing, routing, and resource discovery,” *Communications of the ACM*, vol. 42, no. 4, pp. 86–92, 1999.

[10] C. Maihöfer, “A survey on geocast routing protocols,” *IEEE Communications Surveys and Tutorials*, vol. 6, no. 2, 2004.

[11] B. Brumitt and S. Shafer, “Topological world modeling using semantic spaces,” in *Proceedings of the Workshop on Location Modeling for Ubiquitous Computing, UbiComp 2001*, Sept. 2001, pp. 55–62.

[12] F. Dürr and K. Rothermel, “On a location model for fine-grained geocast,” in *Proceedings of the Fifth International Conference on Ubiquitous Computing (UbiComp 2003)*, Seattle, WA, Oct. 2003, pp. 18–35.

[13] H. Holbrook and B. Cain, “Source-specific multicast for IP,” IETF, Internet-Draft, Sept. 2004.

[14] H. Bahn, K. Koh, S. L. Min, and S. H. Noh, “Efficient replacement of nonuniform objects in web caches,” *Computer*, vol. 35, no. 6, pp. 65–73, 2002.

[15] M. Liljenstam, J. Liu, and D. M. Nicol, “Development of an internet backbone topology for large-scale network simulations,” in *Proceedings of the 2003 Winter Simulation Conference*, New Orleans, LA, Dec. 2003.