Towards Distributed Geolocation by Employing a Delay-Based Optimization Scheme

Michael Grey and David Schatz and Michael Rossberg and Guenter Schaefer
Technische Universität Ilmenau
[michael.grey, david.schatz, michael.rossberg, guenter.schaefer][at]tu-ilmenau.de

***Abstract*— To support position-dependent services, like matchmaking algorithms for online games or geographic backup routes, the estimation of peer locations became a key requisite for a range of applications, recently. However, exact localization may be impossible, e.g., due to nodes lacking Global Positioning System (GPS) access for reasons of cost, energy, or signal unavailability. Alternative approaches, e.g., by nearby WLAN BSSIDs or IP geolocation, rely on databases and normally contain large outliers, in particular when concerning underrepresented mapping locations. This led us to the study of a complementary idea: By embedding nodes on a sphere and periodically minimizing local positioning errors by delay-based multilateration, we efficiently estimate node positions by distributed means, given a fair amount of position hints. Based on simulations that rely on real-world PlanetLab latency data, we show that global-scope peer locations can be estimated with an accuracy of a few hundred kilometers, where the novel approach outperforms a previously proposed spring-mass-based method by about 50%.**

***Keywords*—Geolocation, Nonlinear Optimization, Resilience, Overlay Networks.**

I. Introduction

The number of distributed applications is continuously increasing, starting with entertainment applications like online games through to private overlay networks. Simultaneously, to allow for position-dependence, they start to rely on real geolocation: Matchmaking algorithms in online games depend on such information [1] and next-generation overlay applications may plan backup routes based on geographic coordinates [19] like illustrated in Fig. 1.

Unfortunately, exact locations are typically only available for some nodes: There may be nodes lacking GPS receivers for cost reasons or energy limitations [12], others might suffer from signal unavailability. Alternative approaches for localization, e.g., by nearby WLAN BSSIDs or mapping of IP addresses to geolocations, typically rely on central databases that are – if solely used – rather inaccurate [20], [18].

Taking into account previous work on a delay-based spring-mass model [8], we contribute an alternative approach, where nodes are embedded on a sphere and local positioning errors are periodically minimized by an iterative Internet delay-based multilateration technique. By including a fair amount of position hints, we efficiently estimate node positions by distributed means. Additionally, in order to treat the highly influencial structural heterogeneities that emerge in inter-continental wide-area networks, we augment the approach with dedicated neighborhood selection strategies. Furthermore, the potentials of multilateration-based approaches diminish in extremly sparse populated regions, occasionally leading to poor location estimates. Hence, by providing an outlier detection technique that identifies highly inaccurate estimates, we significantly increase the reliability of the algorithm. Potentials and limits of this approach are pointed out in the course of this article, in particular in terms of location accuracy.

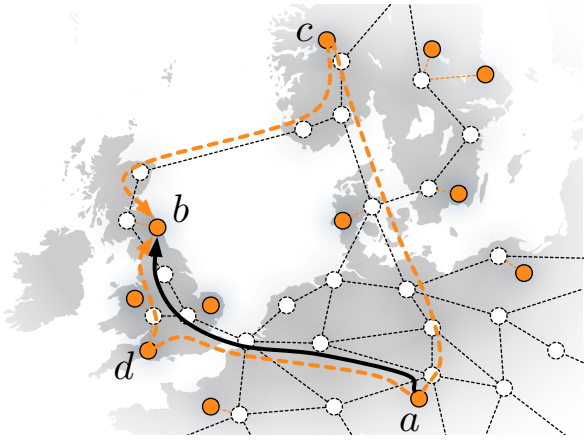

Fig. 1. Exemplary overlay on top of a transport network; as in real systems, overlay links typically often share physical connections. In case of infrastructural failures on the transport network path between overlay nodes a and b, an indirect backup path via node c is more promising than a path via node d

To provide robust, distributed geolocation for overlay nodes, we pursue the following steps: We summarize objectives for distributed geolocation approaches and use them to categorize existing methods in Sec. II, we present a novel, optimization-based approach for distributed geolocation including techniques for advanced neighborhood selection (Sec. III), and in Sec. IV we evaluate the accuracy of the method with delays from PlanetLab measurements. The article concludes in Sec. V.

II. System Objectives & Related Work

To provide a robust, globally distributed estimation of node locations the following objectives should be met:

Accuracy: The most substantial metric for geolocation is the accuracy of estimations in comparison to the real node locations. By design, the achievable quality of an estimation based on delay measurements cannot be as accurate as a localization by GPS, for example, but the absolute difference between real and estimated positions can be expected to be a few 100 kilometers.

SPONSORED BY THE
Federal Ministry of Education and Research

Scalability: The proposed technique shall address distributed applications that may have several thousands of participants. Accordingly, the service must scale with the number of participants and should depend on local knowledge only. Exposed entities, i.e., servers, should be avoided and the communication overhead should be limited.

Robustness & Reliability: The insensitivity to measurement failures and inaccuracies is another key objective of geolocation methods. Moreover, the proposed service should support applications that are sensitive to outliers. Such outliers are not only induced by measurement errors, but occur also when peers have weak preconditions for delay-based estimation, e.g., all neighbors being close together. The system should identify and mark outliers in order to increase the estimation reliability.

Limited dependency on external location information: External location informations can be assumed to be available for some nodes, e.g., by GPS receivers or location databases. However, an access to reliable information cannot be expected for all participants.

In the following, we will discuss existing approaches for geolocation regarding these objectives. Beyond the highly accurate GPS service, a variety of geolocation concepts emerged. On the one hand, several prevalent approaches rely on the same basic principle: Geolocation databases are queried with the public IP addresses of the peers. This includes extensions to DNS records [5], *DNS-LOC* [13], *GeoCluster* [17], and various other solutions. However, these approaches generally suffer from dynamic address assignment and the subsequent invalidation of database records [6]. [20] and [18] recently studied the accuracy of popular database-querying approaches and the results indicate that the corresponding localization may be unreliable, i.e., even failures of thousands of kilometers were observed. And even though, database-based approaches achieve valuable estimations for a major amount of hosts, the introduction of third-party dependencies, e.g., DNS records and centralized databases, is often not desirable for robustness and scalability.

On the other hand, alternative concepts emerged that consider geolocation by delay measurements: The authors of *GeoPing* [17] measure the delay to n designated anchors. The resulting values are compared in an n-dimensional vector space to previously defined references. The position belonging to the reference with minimum Euclidean distance is assumed to be the measured peer's position. Unfortunately, Arif et al. [3] found that the accuracy is not very well for more significant delays, i.e., global scenarios. Furthermore, *GeoPing* assumes the availability of all reference anchors, leading to limited robustness and scalability properties. The *Constraint-based Geolocation* [9] also relies on initial delay measurements, but applies linear regression to the collected data and estimates positions by multilateration. However, a centralized component is involved to perform the computations for host localization.

Beyond geographic localization, efficient and scalable distance estimations may be achieved by *network coordinate systems*. The authors of [10] classify them into two main classes: *Landmark-* and *Simulation-based schemes*. The former rely on a fixed number of landmarks to calculate the local virtual coordinate. A popular example is Global Network Positioning (GNP) [15], which works as follows: First, a set of reference anchors is chosen from all nodes. Second, the distances between all anchors are determined by initial RTT measurements. Third, the anchor positions within a virtual coordinate system are determined by global least-squares minimization of the measured distances between all anchors. Finally, an arbitrary node may determine the own virtual coordinate by RTT measurements towards each neighbor and an analogous minimization. However, even with extensions [16], the achieved scalability cannot compete with fully distributed systems. *Simulation-based schemes* typically scale better. The most popular example is perhaps *Vivaldi* [4] that makes use of a spring-mass-model and in our previous work [8], we studied a Vivaldi-inspired geolocation approach. However, the achievable accuracy was not entirely satisfying, in particular because the algorithm suffered from conceptual weaknesses of the spring-mass approach, e.g., frequent erroneous embedding at local minima.

III. Optimized Embedding by Iterative Multilateration

To provide estimates that directly lead to geographic representations, nodes embed themselves on a spherical surface, following the structure of the Earth. Initially, each node v without external position hints chooses a random point of the sphere (φ_v, λ_v), while nodes that are aware of a potential position initialize their coordinate according to that hint. Furthermore, each node v selects a random neighbor set, where a low constant number of neighbors is sufficient. The nodes then periodically measure the round-trip-time to each neighbor, which is simply assumed to be twice the delay to that node, neglecting routing asymmetries.

The measured delays are mapped to real distances by $f\colon \mathbb{R} \to \mathbb{R}$. As global latencies are not only affected by the propagation delay, but also other delay types (especially varying queuing delay) and potential inaccuracies [8] one cannot expect f to return accurate distances, but to derive meaningful approximates. For this article, we will assume a directly proportional relation between delay and distance. Accordingly, given the real earth radius r_{earth} and r_{delay}, the relation between an approximated delay-value $\frac{rtt}{2}$ and the resulting distance $\tilde{s}$ can be expressed by:

$$\tilde{s} = f\left(\frac{rtt}{2}\right) \simeq \frac{r_{earth}}{r_{delay}} \cdot \frac{rtt}{2} \tag{1}$$

The suitability of a single, fixed value for r_{delay} depends on the properties of the underlying transport network, but works for smaller distances on the globe. While it is possible to define more complex functions, e.g., by deriving high-level polynomials by regression fitting of measurement data, such functions have a higher complexity and may even result in higher errors depending on the suitability of the samples.

To cope with embedding errors despite the simple linear approximation, we rely on a meridian-arc-based mapping with two spherical and one additional non-negative Manhattan dimension. The additional dimension, the so-called *height*, is similar to work on the virtual coordinates [4] and essentially allows to model delays that a node experiences towards all neighbors, e.g., due to a high access delay.

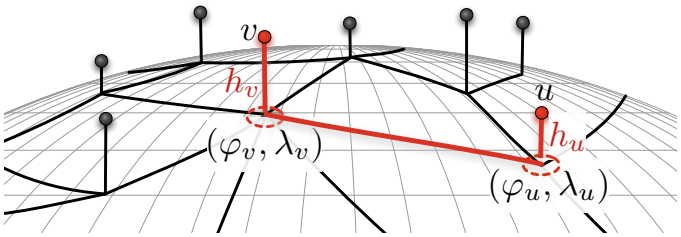

Fig. 2. Decomposition of delay between two nodes v and u into the distance between their coordinates on the sphere surface, i.e., (φ_v, λ_v) and (φ_u, λ_u), and the according heights h_u and h_v.

To perform the localization each node $v \in V$ periodically updates its position $(\varphi_v, \lambda_v, h_v)$, consisting of latitude, longitude, and height, by performing a multilateration involving all of its neighbors $N_v = 1, 2, ..., n$. The delay to each neighbor i is used to calculate the *measured* distance $\tilde{s}_{v,i}$ according to Eq. 1. Accordingly, the *estimated* distance $s_{v,i}$ to a neighbor is the sum of the height of the nodes h_v and h_i, as well as the spherical distance $d_{v,i}$ between (φ_v, λ_v) and (φ_i, λ_i), i.e.:

$$s_{v,i} = h_v + d_{v,i} + h_i \tag{2}$$

To let $s_{v,i}$ reflect $\tilde{s}_{v,i}$ a three step, iterative update procedure is performed (see Fig. 3): a coordinate transformation to Euclidean space, a multilateration-based adaptation, and a subsequent back-transformation. Each of these steps will be presented in the following sections.

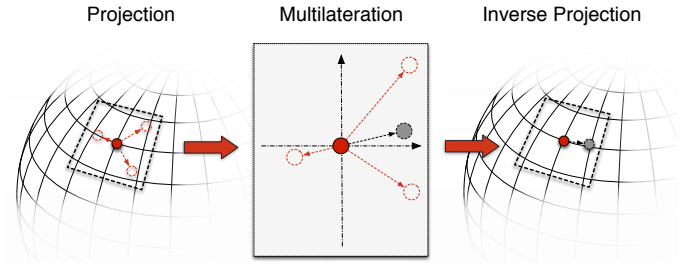

Fig. 3. Projection from earth surface towards local tangential plane and inverse projection after calculation of adaption.

A. Coordinate Projection

As an optimization-based multilateration based on non-linear coordinate systems is extremely computing intensive, we transform the problem: All neighbors are projected on the tangential plane that originates at the coordinate representation of the updating node v. This projection is comparable to an azimuthal equidistant projection, where not only angles from the center are preserved, but also distances from the center are proportionately correct.

To project a neighbor i on the tangential plane, the according spherical distance $d_{v,i}$ as well as the azimuth $\alpha_{v,i}$ are to be calculated: To do so, given the two coordinates $(\varphi_v, \lambda_v, h_v)$ and $(\varphi_i, \lambda_i, h_i)$ the enclosed angle $\sigma_{v,i}$ between two lines can be approximated, where each line originates the center of the sphere and passes one of the two spherical coordinates - not regarding the height component. This is done in a numerically stable way by applying a simplified version of the *Vincenty formula* [22]. Afterwards, the spherical distance $d_{v,i}$ is simply derived by multiplying $\sigma_{v,i}$ and the sphere radius r_{earth}:

$$\Delta\lambda = \lambda_i - \lambda_v \tag{3}$$

$$\sigma_{v,i}^2 = \frac{(\cos\varphi_i \sin\Delta\lambda)^2 + (\cos\varphi_v \sin\varphi_i - \sin\varphi_v \cos\varphi_i \cos\Delta\lambda)^2}{(\sin\varphi_v \sin\varphi_2 + \cos\varphi_v \cos\varphi_i \cos\Delta\lambda)^2} \tag{4}$$

$$d_{v,i} = \sigma_{v,i} \cdot r_{earth} \tag{5}$$

The azimuth $\alpha_{v,i}$ denotes the angle between a straight line leading from the coordinate of v to the coordinate of i and an imaginary straight line through the coordinate of v and the North Pole. It can be calculated in a numerically stable way as follows:

$$\alpha_{v,i} = \arctan\left(\frac{\sin(\Delta\lambda)\cos(\varphi_i)}{\cos(\varphi_v)\sin(\varphi_i) - \sin(\varphi_v)\cos(\varphi_i)\cos(\Delta\lambda)}\right) \tag{6}$$

Using the calculated values for distance and azimuth, the referenced neighbor i is projected on the tangential plane that originates at the coordinate representation of the to-be-adapted node v. The Cartesian coordinate $(x_i, y_i)^T$ is obtained:

$$\begin{pmatrix} x_i \\ y_i \end{pmatrix} = d_{v,i} \begin{pmatrix} \cos\alpha_{v,i} \\ \sin\alpha_{v,i} \end{pmatrix} \tag{7}$$

Note that $\alpha_{v,i}$ is interpreted in reference to the vertical axis, leading to a rotation of the coordinate during projection. However, this does not affect the following calculations.

The projection is performed for all neighbors in the first stage of each update step.

B. Multilateration-based Adaptation

Using the mapped Cartesian coordinates of the neighbors, we can calculate the estimation errors $\Delta d_{v,i}$:

$$\Delta d_{v,i} = \left|\left|\begin{pmatrix} x_i - x_v \\ y_i - y_v \end{pmatrix}\right|\right| + h_v + h_i - \tilde{s}_{v,i} \tag{8}$$

Using these errors and an error-driven weight factor w_i, we can now apply multilateration to calculate a new Cartesian coordinate for node v, i.e., minimize the error of the following least squares optimization problem:

$$\begin{pmatrix} \hat{x}_v \\ \hat{y}_v \\ \hat{h}_v \end{pmatrix} = \underset{x_v, y_v, h_v}{\operatorname{argmin}} \sum_{i=1}^{n} w_i \left[\frac{\Delta d_{v,i}}{\tilde{s}_{v,i}}\right]^2 \tag{9}$$

We rely on an objective function that minimizes quadratic relative differences for two reasons: First, large outliers are avoided due to the quadratic influence of the error values. Therefore, some virtual network coordinate systems like GNP [15] also minimize the squared error, but they rely on a simpler multi-dimensional Euclidean coordinate spaces. Second, the optimization is influenced at varied intensity according to the neighbor distance due to the use of relative errors, i.e., errors related to less distant neighbor references are more influential when compared to equal-valued errors related to more remote neighbors. This can be considered beneficial as the influence of transport network characteristics that induce estimation errors tends to increase as the distance increases.

The formulated nonlinear optimization problem can be locally solved by applying the Nelder-Mead technique [14], which is basically a heuristic search method. To maximize the

chance of convergence to a global minimum, the starting point to be passed should be reasonably close to such a solution. We found that an appropriate starting point can be determined by initially solving the unweighted problem after linearization, i.e., by the Linear Least Squares technique. According to [11], the underlying equation system can be linearized by subtracting the average over all n left-hand-side terms from each of the left-hand-side terms and subtracting the average over all right-hand-side values from each right-hand-side value, accordingly. Thereby an $n \times 2$ matrix A as well as an n-dimensional vector b can be derived, representing the following equation system with $\tilde{d}_{v,i} = max(\tilde{s}_{v,i} - h_v - h_i, 0)$:

$$\overbrace{\left(x_i - \frac{1}{n}\sum_{j=1}^{n} x_i\right)}^{a_{i,1}} x_v + \overbrace{\left(y_i - \frac{1}{n}\sum_{j=1}^{n} y_j\right)}^{a_{i,2}} y_v \quad = \underbrace{\frac{1}{2} \cdot \left[x_i^2 - \frac{1}{n}\sum_{j=1}^{n} x_j^2 + y_i^2 - \frac{1}{n}\sum_{j=1}^{n} y_j^2 - \left(\tilde{d}_{v,i}^2 - \frac{1}{n}\sum_{j=1}^{n} \tilde{d}_{v,j}^2\right)\right]}_{b_i}$$

Note that the partial conversions are not equivalent as the distances are of approximate character. Furthermore, the formulated equation set involves absolute errors rather than relative errors and weighting factors w_i. Thus, the solution of the derived linear problem does not directly lead to sufficient estimations. However, the result is a well-suited starting point for the Nelder-Mead technique. Formally, the starting point calculation for the updated Cartesian coordinate and height component is subject to the following problems:

$$(x_v^{(0)}, y_v^{(0)})^T = \underset{\vec{\mathrm{x}}}{\operatorname{argmin}} \|A\vec{\mathrm{x}} - b\|_2^2 \tag{10}$$

$$h_v^{(0)} = \underset{h_v}{\operatorname{argmin}} \sum_{i=1}^{n} \left[h_v - (\tilde{s}_{v,i} - d_{v,i} - h_i)\right]^2 \tag{11}$$

Using the first objective function, the following normalized equation can be derived that can be solved in a numerically stable way by QR-decomposition:

$$A^T A\vec{\mathrm{x}} = A^T b \tag{12}$$

As the graph of the second objective function is a parabola that opens upwards, the global minimum can be easily found at $t'(h) = 0$, resulting in:

$$h_v^{(0)} = \frac{1}{n}\sum_{i=1}^{n}(\tilde{s}_{v,i} - d_{v,i} - h_i) \tag{13}$$

After calculating the initialization values, a node v determines both new Cartesian coordinates and a new height by applying the Nelder-Mead technique to find an approximate solution for the nonlinear optimization problem. The result of the calculation is the coordinate of an optimal embedding of the node v. However, switching the node's position to the calculated point, may result in instabilities, when neighboring nodes change positions simultaneously. Therefore, we rely on incremental changes: The local coordinate is adapted towards the calculated position, where the amount (i.e, the Euclidean distance towards the calculated Cartesian coordinate) is weighted by a step size parameter δ_{update}. The step size parameterization influences stability and convergence properties. On the one hand, small step sizes increase the chance of convergence by avoiding "jumping" between estimations. On the other hand, a larger size decreases time to convergence, when other nodes do not change positions quickly. Height values may be neglected in the damping as this was not found to lead to stability problems and an inclusion significantly delays convergence.

Similar to a technique used in [4], we propose an error-driven step size parameterization. Given the own relative error e_v, the actual average error over all neighbors involved in the actual update procedure and an additional damping constant γ, the step size δ_{update} for a given update procedure can be calculated as follows:

$$\delta_{\text{update}} = \gamma \cdot \frac{e_v}{e_v + \frac{1}{n}\sum_{i=1}^{n} e_i} \tag{14}$$

Besides, as an additional extension, we propose to weight each squared relative difference denoted in Eq. 9 by an error rate derived from remote error e_i and e_v, i.e., $e_v(e_v + e_i)^{-1}$. A more specialized weighting that takes geographic effects into account may give even better results, but it is out of scope and left for further study.

C. Inverse Coordinate Projection

In the last stage of an update step, node v adapts the own spherical coordinate estimation by applying an inverse map projection, i.e., a mapping from the tangential to the spherical plane. Given the former coordinate (φ_v, λ_v), the angle α between adaptation vector $\vec{\mathrm{x}}_v$ and x-axis as well as the length of the adaptation vector denoted by $||\vec{\mathrm{x}}_v||$ the resulting coordinate after adaption $(\hat{\varphi}_v, \hat{\lambda}_v)$ can be determined as follows (see Fig. 3):

$$\sigma = ||\vec{\mathrm{x}}_v||\, r_{earth}^{-1} \tag{15}$$

$$\hat{\varphi}_v = \arcsin\left(\sin\varphi_v \cos\sigma + \cos\varphi_v \sin\sigma \cos\alpha\right) \tag{16}$$

$$\hat{\lambda}_v = \lambda_v + \arctan\left(\frac{\sin\alpha \sin\sigma \cos\varphi_v}{\cos\sigma - \sin\varphi_v \sin\hat{\varphi}_v}\right) \tag{17}$$

Both, projection and inverse projection introduce inherent errors, especially if the coordinates are close to the poles or the distances are large. However, this can be neglected due to the damping as described above and the reliance on incremental updates.

D. Improving Accuracy: Advanced Neighborhood Selection

Using the outlined technique, nodes are able to estimate their geographic coordinates. However, the estimation quality can be significantly increased by adapting the neighborhood after an initial configuration. Note, that height values are not considered in the following section, as they have no impact on the geographic multilateration but only model delay effects. Given a set of potential neighbors $M_v = 1, 2, \ldots, m$ and a subset of neighbors to be chosen N_v, we propose the following two selection strategies.

(a) β = 0.8 (b) β = 1.0 (c) β = 1.25

Fig. 4. β-skeleton: Relevant regions $R_{u,v}$ in dependency of parameterization.

1) Distance- & Angle-based Selection: Intuitively, neighbors may be picked according to their distance as the absolute accuracy typically benefits from short-distance references: Thus, the potential neighbors are first ordered by their distances $d_{v,i}$. Neighbors are then selected according to a distribution function that is chosen such that nearby nodes have a higher chance of being selected, i.e., by sampling by a Weibull density function with shape parameter W_α and scale parameter W_β. We propose to use a fixed value W_α, but to adapt W_β according to the remaining number of candidate nodes after each selection. Generated random numbers that exceed the available number of candidates are discarded. However, solely relying on the distance may lead to the selection of nodes that are grouped very closely together. Grouped neighbors are considered disadvantageous as the employed multilateration scheme benefits from references with angular diversity. Hence, we propose an angle-based approach to identify and select neighbors with high diversity:

1) To prevent computational intensive calculations on the sphere surface, all potential neighbors are projected on a tangential plane as described in Sec. III-A, first.
2) Given all Cartesian coordinates, a β-skeleton graph may be constructed, and the neighbor set N_v corresponds to the peers that are connected to the node v within the β-skeleton structure. Given a set of points on a geometric plane P, the β-skeleton defines a undirected graph $G = (P, E)$, where the nodes correspond to the points. Depending on the parameter β the edges E are derived as follows [2]:

$$\Theta = \begin{cases} \sin^{-1}(1/\beta), & \text{if } \beta \geq 1 \\ \pi - \sin^{-1}(\beta), & \text{if } \beta < 1 \end{cases} \qquad (18)$$
$$E = \left\{ \{u, v\} \in \binom{P}{2} \mid \nexists w \in P : \angle(uwv) > \Theta \right\}$$

   Note that $\angle(uwv)$ denotes the angle between $\overline{wu}$ and $\overline{wv}$. Fig. 4 illustrates the graph definition: Both points u and v define a region $R_{u,v}$. The resulting shape corresponds to either the intersection (Fig. 4a) or the union (Fig. 4c) of two circular areas. Accordingly, E contains a specific edge $\{u, v\}$ if and only if there is no point $w \in P$ that is located within $R_{u,v}$.
3) Depending on the parameterization of β, the determined neighbor set does not correspond to the desired size. Hence, to attain the desired number of neighbors, nodes are either randomly removed from N_v or additional neighbors are included in N_v, where peers with smaller distances are again preferred.

2) Cluster-based Selection: Any delay-based location technique will significantly benefit from homogeneous networks

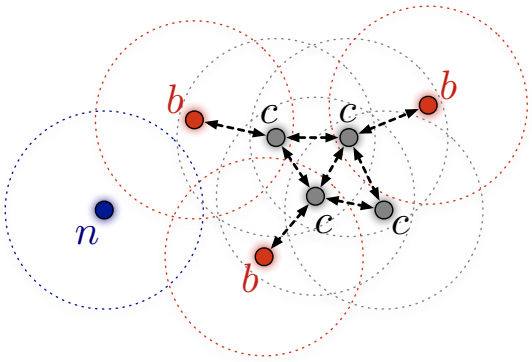

Fig. 5. Examplary DBSCAN object categorization ($minPts = 2$): Core objects c, border objects b and one noise object n.

conditions, i.e., where traversed network paths are subject to similar characteristics. While the propagation delay is typically not subject to significant variations in real world networks, other delay types heavily depend on properties of the underlying network, e.g., the overall switching delay correlates with the number of intermediate hops and the deployed access technology dominates the last-hop delay. These properties are considerably heterogeneous in global networks and typically depend on geographic, political, and economic area. The impact on the localization must be considered to be significant, at least if major anomalies are involved that occur in topologically sparse regions, e.g., oceans or deserts. To give an example, in contrast to landline connections (e.g., on the European continent) of similar length, traversed submarine connections (e.g., between Europe and North America) induce negligible switching delay. These inhomogeneities are neither treated by the linear delay-distance mapping nor the height model and thus lead to lower accuracy.

To reduce the influence of heterogeneities we designed a cluster-based neighborhood re-selection technique to identify and separate homogeneous areas, such that nodes only select neighbors that are located in the same area. Areas are identified by employing a heuristic method that is based on a local-knowledge derivation of DBSCAN [7], a density-based cluster algorithm. The basic DBSCAN algorithm operates on objects $p \in P$ and a given distance function $dist$: $P \times P \to \mathbb{R}$. With respect to our application, the former is defined by the peer coordinates and the latter is defined by the distances between estimations of peer positions. In addition, the input parameters $minPts$ and ϵ are defined to control the minimum number and density of objects within a cluster. Furthermore, the so-called ϵ-neighborhood of an object is defined by $N_\epsilon(p) = \{q \in P \,|\, dist(p, q) \leq \epsilon\}$. Moreover, DBSCAN differentiates three object categories, where p will be either:

- a *core object* if $|N_\epsilon(p)| \geq minPts$,
- a *border object* if it is not core object, but there is a core object $q \in P$ ($q \neq p$) for which $p \in N_\epsilon(q)$, or
- a *noise object* if it is neither a core nor a border object.

Subsequently, a cluster group C can be defined by recursion as follows: C contains at least one core object *and* for all core objects $q \in C$ applies $N_\epsilon(q) \subseteq C$. Based on this basic algorithm, a distributed heuristic method that solely relies on local-knowledge is employed to derive the local cluster

C_v of node v. Given the precondition that each node of the system is aware of its ϵ-neighborhood, the local cluster C_v is identified as follows: First, node v requests $N_\epsilon(u)$ from each node $u \in N_\epsilon(v)$. Second, considering the obtained sets, the closest core object $w \in N_\epsilon(v)$ is identified, i.e., the core object with minimum distance $dist(\varphi_v, \lambda_v, (\varphi_w, \lambda_w)$. Third, if node v cannot find any core object in its ϵ-neighborhood, it will be implicitly classified as noise object and randomly selects new neighbors. Otherwise, from the existence of a core object k (which may be node v itself without adverse consequences) can be concluded that v is located within a cluster C_k. Consequently, initialized by request of node v, the whole cluster C_v can be determined by a recursive version of DBSCAN as illustrated in Algorithm 1. After C_v is determined, node v additionally checks the presence of position hints within the cluster. Without C_v containing any position hint, node v is marked to be a noise object and selects a random neighborhood again. Otherwise, node v is aware of its local cluster and selects the neighbors from C_v by applying another strategy, i.e., distance and angle-based selection.

Algorithm 1 Recursive algorithm for calculation of C_v

```
function RECURSIVE_DBSCAN(Node u, Cluster C)
    if |N_ε(u)| ≥ minPts then
        for all w ∈ N_ε(u) do
            if w ∉ C then
                C ← C ∪ {w}
                C ← RECURSIVE_DBSCAN(w, C)
    return C
```

E. Improving Reliability: Outlier Detection

A number of applications that benefit from position estimations, e.g., wide-area backup path planning, seriously suffer from incorrect estimates. Thus, outliers in the estimation must be treated and the following approach is applied: For one thing, nodes that are classified by the clustering to be noise objects are marked as outliers. For another thing, nodes are also marked as outliers if their height exceeds a predetermined limit h_{max} as extreme heights reflect extraordinary delays to all neighbors. This is the case when the access delay is high, but also when the geographic distance of the according node to all other nodes is extraordinary large, which typically leads to inaccurate estimates.

IV. Evaluation

To assess the proposed approach, its characteristics with respect to the objectives from Sec. II are discussed first. Afterwards, the geolocation accuracy is subject to a quantitative evaluation with the help of simulations.

Scalability: The proposed system is considered to scale over the number of nodes for the following reasons: The computational and communication effort of each node solely depends on a constant number of neighbors. For the cluster-based neighborhood selection presented in Sec. III-D2 one call of RECURSIVE_DBSCAN(u, C) only implies one request to node u, when neglecting the initialization. Additionally, each return value is sent to a constant number of neighbors over the network. However, the induced overhead can be considered low as the selection mechanism is rarely executed compared to coordinate adaptions, and the derived cluster information is distributed within the cluster to avoid repetitions.

Robustness & Reliability: The proposed mechanism does not introduce exposed entities and tolerates fail-stop errors as well as measurement inaccuracies. Even the cluster-based neighborhood selection does not introduce single-points-of-failure as any node is capable of triggering the cluster identification even though the latter is typically only initiated one time per cluster and period. The mechanisms for outlier detection treat outliers explicitly and thus increase reliability.

Minor dependency on external location information: The proposed mechanisms do not rely on the amount nor the accuracy of specific external positions. Nonetheless, a fair amount of nodes with access to external location information is required to provide meaningful results in terms of absolute positioning. However, given scenarios without any external position information, the system will still be capable of estimating relative distances.

Accuracy: The absolute and relative errors of estimates – being the most significant metrics – are evaluated for different fractions of nodes with external location sources with the help of quantitative measurements. To do so a simulative study was performed utilizing a dataset of delays between *PlanetLab* sites. This allows for verification of the occurring positioning errors as the reference locations of each *PlanetLab* measurement site are publicly available. The presented method is simulated on top of this real-world dataset in a simulation environment that is based on OMNeT++ [21]. If not stated otherwise, the following set of relevant parameters is applied:

Nodes	195	Neighbors	32
Sphere Radius	0.08 seconds	γ (step size)	0.5
Cluster: ϵ	1000 km	Cluster: minPts	5
β-Skeleton: β	0.8	β-Skeleton: W_α	0.5

Furthermore, the following charts show means over 64 simulation runs and indicate mean and median values, where error bars denote 99% confidence intervals and interquartile range, respectively.

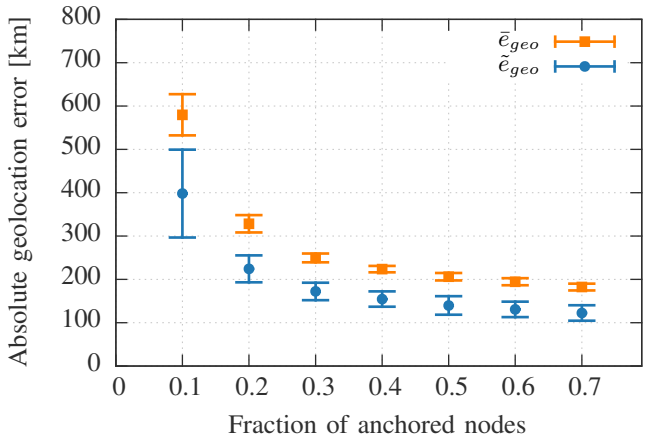

Fig. 6. Absolute geographic positioning error

The absolute distance between real-world node positions i_{act} and estimated peer positions i_{est}, the *geographic estimation error* e_{geo} is used to evaluate the approach. Nodes with access to position hints are excluded from evaluation. Fig. 6 illustrates the mean and median error of e_{geo} for North American and European nodes depending on the fraction of position hints. The depicted graph indicates that a minimum ratio of anchored nodes with external hints is required to obtain reasonable estimations for all the other nodes. Furthermore, the estimation errors strictly decrease with an increasing ratio of location hints, i.e, given a fair ratio of 30% anchors, the median error is 172 km, where the mean error equals 250 km. At a first glance, the curve shapes are comparable to a previous study [8], but the reached accuracy is considerably higher. This applies in particular to the dataset, consisting of North American and European nodes: In contrast to the early approach that suffered from inhomogeneities of the underlying network, the introduced neighborhood selection strategies reduce the corresponding influence now.

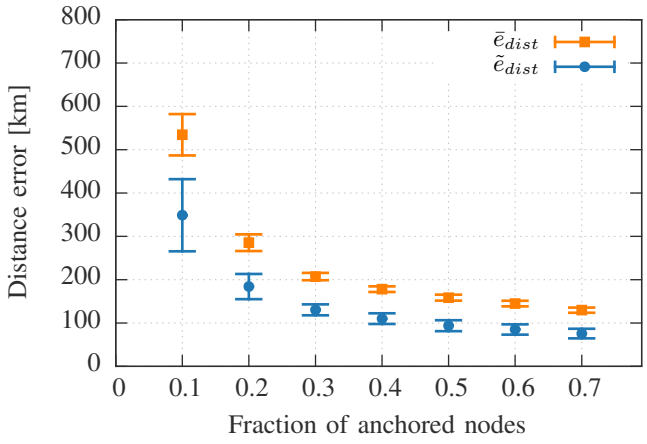

Fig. 7. Average and median error of relative distances between nodes

For some applications the relative positioning between nodes is more significant. Thus, the average distance error $e_{dist}(i,j) = |d(i_{act}, j_{act}) - d(i_{est}, j_{est})|$ was evaluated. Fig. 7 shows the mean and median of the average distance error e_{dist} in dependence of the fraction of involved position hints. While the shape characteristic is similar to the absolute error, the distance errors are of smaller amount when compared to absolute errors. Again, given a fair ratio of 30% anchors, the average median distance error equals 130 km and the average mean error is 207 km. The absolute and distance errors confirm the premise that the achievable accuracy cannot compete with external location services, e.g., GPS, and does not fit the requirements of a range of classic location-based applications. However, the proposed approach can be considered reasonable for applications with less stringent localization accuracy requirements, e.g., services for matchmaking applications or third-party location database verification.

V. Conclusion

Pursuing previous work, we addressed the feasibility of fully distributed geographic position estimations of peers by delay measurements in this article. In terms of localization accuracy, the presented optimization-based approach outperforms the previously proposed spring-mass-based method by about 50%. Moreover, we provided additional means to avoid pitfalls, i.e., reliable detection of localization outliers and additional neighborhood selection strategies that neglect the influence of network inhomogeneities. While the achievable localization accuracy is not comparable to precise external geolocation services such as GPS, the method satisfies the requirements of a number of applications with less stringent requirements, e.g., the planning of backup paths, and is conceptually complementary to alternative techniques.

In future research, we plan to further increase the geolocation quality by introducing dynamic parameter adaptation according to network characteristics. In addition, the proposed technique will be combined with a fully distributed backup path selection service on the basis of geographic peer positions, which is expected to increase the reliability of a wide range of overlay applications, e.g. virtual private networks.

References

[1] S. Agarwal and J. Lorch. Matchmaking for online games and other latency-sensitive P2P systems. In *ACM SIGCOMM Computer Communication Review*, volume 39, pages 315–326, 2009.

[2] N. Amenta, M. Bern, and D. Eppstein. The crust and the β-skeleton: Combinatorial curve reconstruction. *Graphical models and image processing*, 60(2):125–135, 1998.

[3] M. Arif, S. Karunasekera, S. Kulkarni, A. Gunatilaka, and B. Ristic. Internet host geolocation using maximum likelihood estimation technique. In *IEEE AINA*, pages 422–429, 2010.

[4] F. Dabek, R. Cox, F. Kaashoek, and R. Morris. Vivaldi: A Decentralized Network Coordinate System. In *ACM SIGCOMM*, pages 15–26, 2004.

[5] C. Davis, I. Dickinson, T. Goodwin, and P. Vixie. A means for expressing location information in the domain name system. 1996.

[6] Z. Dong, R. Perera, R. Chandramouli, and K. Subbalakshmi. Network measurement based modeling and optimization for IP geolocation. *Computer Networks*, 56(1):85–98, 2012.

[7] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD*, volume 96, pages 226–231, 1996.

[8] M. Grey, M. Rossberg, M. Backhaus, and G. Schaefer. On Distributed Geolocation by Employing Spring-Mass Systems. *IEEE GIIS*, 2013.

[9] B. Gueye, A. Ziviani, M. Crovella, and S. Fdida. Constraint-Based Geolocation of Internet Hosts. In *4th ACM SIGCOMM conference on Internet measurement*, pages 288–293. ACM, 2004.

[10] J. Ledlie, P. Gardner, and M. Seltzer. Network Coordinates in the Wild. In *4th USENIX Symposium on Networked Systems Design & Implementation*, pages 299–311, 2007.

[11] Z. Li, W. Trappe, Y. Zhang, and B. Nath. Robust statistical methods for securing wireless localization in sensor networks. In *IEEE IPSN*, pages 91–98, 2005.

[12] J. Liu, B. Priyantha, T. Hart, H. Ramos, A. Loureiro, and Q. Wang. Energy Efficient GPS Sensing with Cloud Offloading. 2012.

[13] J. Muir and P. Oorschot. Internet geolocation: Evasion and countereva-sion. *ACM Computing Surveys (CSUR)*, 42(1):4, 2009.

[14] J. A. Nelder and R. Mead. A simplex method for function minimization. *The computer journal*, 7(4):308–313, 1965.

[15] T. E. Ng and H. Zhang. Global network positioning: a new approach to network distance prediction. *Computer Communication Review*, 32(1):61, 2002.

[16] T. E. Ng and H. Zhang. A network positioning system for the internet. In *USENIX Annual Technical Conference*, pages 141–154, 2004.

[17] V. Padmanabhan and L. Subramanian. An Investigation of Geographic Mapping Techniques for Internet Hosts. In *ACM SIGCOMM Computer Communication Review*, volume 31, pages 173–185. ACM, 2001.

[18] I. Poese, S. Uhlig, M. A. Kaafar, B. Donnet, and B. Gueye. IP geolocation databases: Unreliable? *ACM SIGCOMM Computer Communication Review*, 41(2):53–56, 2011.

[19] S. Savage, T. Anderson, A. Aggarwal, D. Becker, N. Cardwell, A. Collins, E. Hoffman, J. Snell, et al. Detour: Informed Internet Routing and Transport. *IEEE Micro*, 19(1):50–59, 1999.

[20] Y. Shavitt and N. Zilberman. A Geolocation Databases Study. *IEEE JSAC*, 29(10):2044–2056, December 2011.

[21] A. Varga et al. The OMNeT++ discrete event simulation system. In *European Simulation Multiconference (ESM)*, volume 9, 2001.

[22] T. Vincenty. Direct and inverse solutions of geodesics on the ellipsoid with application of nested equations. *Survey review*, 23(176):88–93, 1975.