Dynamische Teilrouten zur anonymen Navigation

Christian Roth, Lukas Hartmann, Katharina Issel, Dogan Kesdogan

Universität Regensburg, Universitätsstr. 31, 93053 Regensburg,
{vorname.nachname}@ur.de

Abstract. Um Privacy bei Location Based Services (LBS) zu gewährleisten, muss auf Anwendungsebene verhindert werden, dass der Service Provider bei Navigationsabfragen die Pfade des Nutzers nachverfolgen kann. Gleichzeitig muss die Funktionsfähigkeit des Service erhalten bleiben. Durch Stückelung der Route und anonyme Abfrage der Teilrouten lässt sich ein Privacy Enhanced Routing unter Einschränkung der Genauigkeit realisieren.

Keywords: Anonymität, Navigation, Graphen, l-Diversity

1 Motivation

Nutzer von Online-Anwendungen, wie einer Navigationslösung für den Straßenverkehr, sind sich kaum bewusst, was sie durch Ortungs- und Routen-Informationen Persönliches über sich preisgeben. Die Informationelle Selbstbestimmung [1] beinhaltet jedoch, dass Individuen selbst darüber entscheiden können, welche personenbezogenen Daten offenbart werden. In der physischen Welt kann ein Nutzer leichter erkennen, in wie weit er sich im öffentlichen Raum bewegt. Die Informationspreisgabe kann weitestgehend gesteuert werden. Bewegt sich derselbe Nutzer nun in der digitalen Welt, kann er nicht gleich oder gar nicht erkennen, welche Informationen über ihn zum einen öffentlich zugänglich werden [2], zum anderen welche Schlüsse auf seine Einstellung und Persönlichkeit gezogen werden können. Rückschlüsse auf die Person ist selbst bei einem sonst anonymen Nutzer allein durch Bekanntgabe von Start- und Zielort möglich [7], wodurch bspw. Bewegungsmuster erstellt werden können (z.B. Tracking).

Unser Vorschlag soll das Erkennen einer vollständigen Route (von Start- zu Zielort) eines Nutzers durch Dritte verhindern. Nur der Nutzer kennt Start- und Zielpunkt, jedoch nicht die Route dazwischen. Wird die Route in Teilrouten zerlegt und anonym abgefragt, kann der Nutzer anonym zum Ziel geführt werden. Die Teilrouten sollen nicht miteinander verkettbar sein, sodass ein Dritter (z.B. ein Location Based Service Provider wie Google) nicht einmal feststellen kann, welche Teilroute Start- und Endpunkt der Gesamtroute des Nutzers enthält. Die Funktionsfähigkeit der Navigation soll trotz Anonymität erhalten bleiben, weil nutzerseitig die passenden Teilrouten zur

Gesamtroute zusammenfügt wird. Dieses Paper stellt daher als Basis zunächst einen Routenalgorithmus vor, der Routen anhand von Teilrouten erstellen kann. Des Weiteren wird der Overhead dieser kombinierten Routen im Vergleich zu direkter Routenfindung untersucht.

2 Related Work

Ortsanonymität ist besonders im akademischen Bereich ein relevantes Thema. So zeigen bspw. Wang und Liu [3] anhand eines Graphen-basierten, skalierbaren Modell wie Ortsanonymität und Ortsdiversität hergestellt werden können. Für die Ortsanonymität verwenden diese k-Anonymität, indem sie ein Set an anonymen Nutzern verwenden, dass die einzelnen Nutzer innerhalb des Sets voneinander ununterscheidbar werden lässt. Das setzt zwingend voraus, dass eine bestimmte Anzahl an aktiven Nutzern im Set vorhanden sein muss. Ortsdiversität wird durch l-Diversität realisiert, bei welcher der preisgegebene Ort des Nutzers l unterschiedliche Straßen-Segmente enthält.

Pingley et al. [4] fokussieren sich in ihrer Arbeit auf die Anonymität von Abfragen von Location Based Services (LBS), um die Privatsphäre der abgefragten Service-Attribute (wie z.B. Bar, Krankenhaus, Kirche) zu gewährleisten, selbst wenn die Identität des Nutzers entdeckt wurde. Zur Lösung verwenden sie Dummy-Abfragen für verschiedene Services, ohne dass eine Trusted Third Party benötigt oder die Abfrage-Genauigkeit beeinflusst wird.

Im MobiMix-Ansatz von Palanisamy und Liu [5] wird Ortsanonymität durch Unterteilung des Straßennetzes in Mix Zonen erreicht, innerhalb welcher der Nutzer nicht nachverfolgt werden kann. Ein- und Austrittspunkte aus einer Mix Zone sind zwar sichtbar, aber der Nutzer ist durch unverkettbare Pseudonyme an diesen Stellen anonym. Beim Festlegen der Mix Zonen sind verschiedene Aspekte sorgfältig zu berücksichtigen (Geometrie der Zonen, statisches Verhalten der Nutzergruppe, Ortsbeschränkung Bewegungspfade der Nutzer), was einen gewissen Aufwand bedeutet und zusätzliche Komplexität erfordert.

3 Privacy Enhanced Routing

Der vorgeschlagene Algorithmus soll den Nutzer vor einem passiven Angreifer (bspw. LBS Anbieter) schützen, der Inhalte der Routenabfrage mitlesen kann. Dafür wird das Straßennetz zunächst durch einen Graphen $G = (V, E)$ mit Knoten V und Kanten E dargestellt, wobei jeweils nur dem Nutzer Start- und Endknoten (v_S bzw. v_T) bekannt sind.

Um dem Angreifer keine Möglichkeit zu bieten, die vollständige Route R in Erfahrung zu bringen, wird die Gesamtroute iterativ aus dynamischen Teilrouten R_i zusammengesetzt. Dazu wird zum Schutz der Anonymität ein Teilgraph G_i angefragt, der mindestens einen Knoten mit Knotengrad ≥ 3 beinhaltet. Anschaulich gesprochen enthält G_i somit mindestens eine Straßenkreuzung und besitzt folglich mindestens zwei weitere Randpunkte $\partial V(G_i)$, für die jeweils eine Anfrage beim LBS gestellt wird und die GPS Differenz zu v_T berechnet wird. Der Endknoten mit der minimalen Distanz wird vom Nutzer gewählt und dient als Startknoten für den nächsten Teilgraphen.

Die Abfrage der jeweiligen Teilgraphen erfolgt anonym, sodass ein Angreifer die Teilrouten nicht verketten und einem Nutzer zuschreiben kann. Auch kann vom Angreifer nicht bestimmt werden, welche Teilroute den Start- bzw. Zielpunkt beinhaltet. Angriffe über die Verweildauer in einzelnen Teilrouten o.ä. werden durch zusätzliche Maßnahmen erschwert/verhindert. Da der Algorithmus auf bestehende LBS aufsetzt, werden Kartenmaterialen von Dritten (z.B. Google Maps, OpenStreetMap) genutzt. Die Abfrage kann somit über unterschiedliche Anbieter gestreut werden, was zusätzlich die Unverkettbarkeit und Anonymität fördert.

3.1 Datenimport

Der Algorithmus verwendet aktuell Daten von OpenStreetMap (OSM). OSM sind öffentlich zugängliche Kartendaten, die kostenfrei eingesetzt werden können. Die Karten werden von Community-Mitgliedern erstellt, gepflegt und geprüft, sie folgen dem Crowd-Sourcing Prinzip. Damit einher geht, dass die Datenqualität schwanken kann, man aber annehmen darf, dass die semantische Korrektheit durch die aktive Community sichergestellt wird. Auf Grund der kostenlosen Verwendbarkeit und der sehr ausführlichen Informationen der Karten fiel daher die Wahl auf OSM.

Flexibilität bietet der vorgestellte Algorithmus jedoch hinsichtlich der darunterliegenden Datenbasis: Der Kartenprovider kann, ausreichende Informationen vorausgesetzt, jederzeit ausgetauscht werden. Dazu werden die Rohdaten gereinigt und in eine Datenbank importiert. Um zukünftige Anwendungsgebiete nicht zu beschränken, wurde im Projekt die in Java implementierte Graphdatenbank neo4j verwendet, die sowohl lokal als auch serverseitig betrieben werden kann. neo4j speichert Daten in Form von Knoten und Kanten. Knoten können über Kanten miteinander verbunden werden. Sowohl Knoten als auch Kanten können verschiedene Eigenschaften zugewiesen werden (genannt Properties und Labels). Die Speicherdarstellung ähnelt folglich der eines Straßennetzes bzw. eines Graphen.

Beim Import der Daten von OSM in neo4j wurden nur solche Straßen betrachtet, die von KFZ frei befahren werden können. Zur Filterung wurden die vom OSM Projekt vorgegeben Straßentypen verwendet. Von Nutzern selbstständig definierte Straßentypen wurden gänzlich ignoriert.

Als Besonderheit bei OSM ist zu beachten, dass eine Straße (Way) aus mehreren Nodes zusammengesetzt ist. Dies ist notwendig um die geographischen Eigenschaften, wie bspw. Kurven o.ä. zu repräsentieren. Für den verwendeten Einsatzzweck sind die geographischen Eigenschaften jedoch irrelevant, sodass Straßen zu einer Start- und einer Endnode verbunden über eine Kante zusammengefasst wurden. Folglich entspricht jede Node einem Endpunkt oder einer Kreuzung.

3.2 Algorithmus

Der Algorithmus zur dynamischen Anfrage von Teilrouten verwendet wie bereits erwähnt einen Graphen $G = (V, E)$ mit Knoten V und Kanten E, der das Straßennetz repräsentiert. Die Knoten entsprechen dabei in bekannter Weise Wegpunkten und die Kanten Wegen zwischen diesen Punkten. Der Startknoten v_S und der Endknoten v_T einer Gesamtroute sind nur dem Nutzer bekannt und werden in unterschiedlichen unverkettbaren Teilrouten anonym abgefragt.

Die Gesamtroute R zwischen v_S und v_T wird iterativ aus dynamischen Teilrouten R_i zusammengesetzt. Für die Teilroute R_i werden der Start- und Zielknoten mit v_{S_i} bzw. v_{T_i} bezeichnet. Die Teilrouten miteinander zu einer validen Strecke von v_S nach v_T verbunden, sodass stets $v_{T_i} = v_{S_{i+1}}$ gelten soll. Für den Startpunkt v_{S_i} der Teilroute R_i berechnet der Algorithmus zunächst einen Teilgraphen G_i von G, der so beschaffen ist, dass er von v_{S_i} ausgeht und mindestens einen weiteren Knoten mit Knotengrad ≥ 3 enthält. Anschaulich gesprochen enthält G_i also mindestens eine Kreuzung und somit mindestens zwei weitere Knoten, die als v_{T_i} in Frage kommen können.

Sei G^* der Minor von G, der entsteht, in dem alle Grad 2-Knoten – außer v_S und v_T – entfernt werden (G ist somit eine Unterteilung von G^*). Durch das iterative Verfahren ist v_{S_i} ebenfalls ein Knoten von G^*. Der Teilgraph G_i^* von G^* enthält nun alle Knoten, die zu v_{S_i} höchstens einen gewissen Abstand in G^* haben. Dieser Abstand ist als Parameter im Algorithmus vorhanden und wird mit *Hops* bezeichnet. Es gilt soweit für alle Knoten $v \in G_i^*$: $dist_{G^*}(v_{S_i}, v) \leq Hops$.

Bei Knoten, die bereits auf den bisherigen Teilrouten liegen, wird die Breitensuche abgebrochen und diese Knoten werden nicht in den Teilgraphen G_i^* aufgenommen. Die Knoten $v \in G_i^*$ mit $dist_{G^*}(v_{S_i}, v) = Hops$ und $v \neq v_{S_i}$ werden nachfolgend als *Randknoten* $\partial V(G_i^*)$ bezeichnet. Abschließend werden die Grad 2-Knoten, die auf den Kanten von G_i^* liegen, wieder eingefügt. Der resultierende Graph ist der gewünschte Teilgraph G_i. Die Randknoten von G_i sind somit identisch mit denen von G_i^*, d.h. es gilt $\partial V(G_i) = \partial V(G_i^*)$.

Für alle Randknoten $v \in \partial V(G_i)$ wird nun eine Routenabfrage von v_{S_i} nach v beim verwendeten LBS gestellt. Da aufgrund der genannten Bedingung die Konstruktion G_i

mindestens eine Kreuzung und somit mindestens zwei Randknoten enthält, werden folglich für jeden Teilgraphen mindestens zwei Routen angefragt. Für jeden $v \in \partial V(G_i)$ wird zusätzlich die geometrische Distanz (Luftlinie) zwischen v und dem Zielpunkt der gesamten Route v_T berechnet. Es wird derjenige Knoten v als nächster Zwischenschritt v_{T_i} ausgewählt, der die geringste Distanz zum Ziel v_T besitzt. Die bereits beim LBS angefragte Route von v_{S_i} nach v_{T_i} wird als nächste Teilroute R_i benutzt.

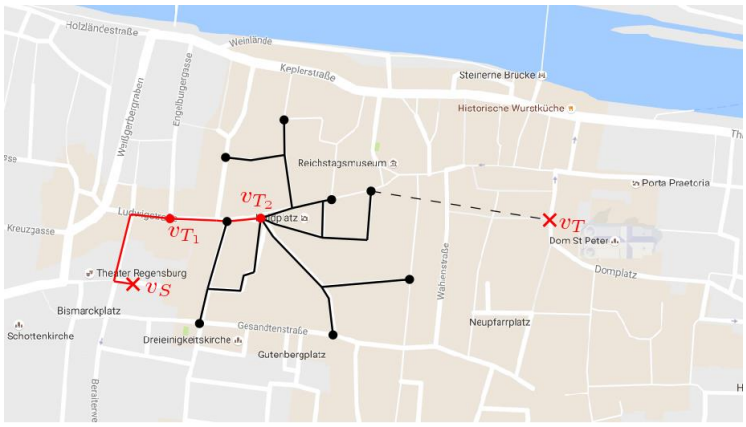

Abbildung 1: Der Teilgraph G_3 wurde im dritten Iterationsschritt berechnet.

Abbildung 1 zeigt beispielhaft den dritten Iterationsschritt einer Routenanfrage vom Theater Regensburg (v_S) zum Dom St. Peter (v_T). Start- und Zielknoten sind als rote Kreuze und die bereits fertiggestellten Teilrouten R_1 und R_2 ebenfalls in Rot dargestellt. Der berechnete Graph G_3 ist schwarz gezeichnet. Dabei sind die Randknoten $\partial V(G_3)$ als dicke schwarze Punkte gemalt. Für alle diese Knoten wird eine Routenanfrage beim LBS gestellt. Der Knoten rechts oben besitzt die geringste geometrische Distanz zum Zielknoten v_T, sodass dieser als nächstes Zwischenziel v_{T_3}verwendet wird. Als nächster Routenabschnitt ergibt sich somit die in Abbildung 2 dargestellte Route.

Wurde die Teilroute R_i gefunden, so startet das Verfahren nun erneut mit $v_{S_{i+1}} = v_{T_i}$. Iterativ werden so lange Teilrouten angefragt, bis der endgültige Zielknoten v_T in einem Teilgraphen G_j enthalten ist. Dabei ist zu beachten, dass die Anfrage der jeweiligen Teilrouten beim LBS unverkettbar erfolgen muss. Dies kann z.B. durch die Verwendung von verschiedenen Pseudonymen oder IP-Adressen erreicht werden. Durch die Unverkettbarkeit weiß der LBS nicht, welche Anfragen die wahren Start- und Zielknoten v_S bzw. v_T enthalten und welche der angefragten Routen verwendet wurden.

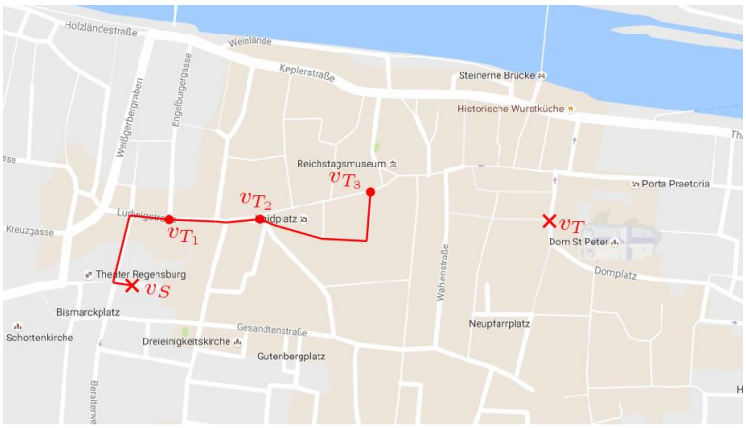

Abbildung 2: Die Teilrouten $\boldsymbol{R_1}$, $\boldsymbol{R_2}$ und $\boldsymbol{R_3}$, die vom LBS erstellt wurden, ergeben zusammen eine Route von $\boldsymbol{v_S = v_{S_1}}$ nach $\boldsymbol{v_{T_3}}$ ergeben

3.3 Der Parameter Hops

Der Parameter *Hops* wurde bereits im vorigen Abschnitt erwähnt. Er diente dazu, die Größe des Teilgraphen G_i festzulegen. Anschaulich gesprochen enthielt G_i diejenigen Knoten als Randknoten, die *Hops* Kreuzungen vom Startpunkt v_{S_i} der Teilroute entfernt liegen.

Abbildung 3: Verschiedene Hops-Werte für einen Teilgraphen:
Links *Hops* = 1, Mitte *Hops* = 2, Rechts *Hops*= 3.

In Abbildung 3 ist beispielhaft ein Teilgraph G_i für *Hops*-Werte 1, 2 und 3 dargestellt. Der Startknoten v_{S_i} ist als rotes Kreuz markiert, während die Randknoten $\partial V(G_i)$ als dicke Punkte dargestellt sind. Es ist erkennbar, dass mit größer werdender *Hops*-Anzahl der Teilgraph mehr Randknoten erhält und somit pro auszuwählenden

Teilgraph mehr Anfragen beim verwendeten LBS erstellt werden. Die verwendeten Teilrouten werden ferner länger, sodass insgesamt weniger Teilrouten-Stücke angefragt werden müssen. Damit verbunden enthalten längere Teilrouten potentiell mehr Informationen über die Strecke, die der Nutzer zurücklegt und liefern dem LBS dadurch mehr personenbezogene Daten. Andererseits kann bei einer längeren Teilroute mehr von den zusätzlichen Funktionalitäten eines online-basierten LBS profitiert werden. So lässt sich z.B. eine verkehrsabhängige Routenführung nutzen oder Alternativrouten können vom LBS für die jeweilige Teilroute berechnet werden. Gerade letzteres ist bei geringerer *Hops*-Anzahl und damit verbunden bei kurzen Teilrouten oft nicht möglich, da es im Maßstab von nur wenigen Kreuzungen keine sinnvollen Alternativrouten gibt.

Als Extrembeispiel ließe sich $Hops = \infty$ wählen. Dies würde dazu führen, dass der erste Teilgraph G_1 bereits den Zielpunkt der Route beinhaltet und somit direkt das Paar v_S und v_T beim LBS angefragt würde. Der Nutzer erhielte in diesem Fall zwar eine optimale Route mit allen zusätzlichen Funktionalitäten des LBS, im Gegenzug wäre jedoch keine Anonymität mehr gewährleistet.

Insgesamt kann der Parameter *Hops* als Einstellparameter zwischen Qualität der Route und Anonymität angesehen werden, bei dem ein Nutzer selbst bestimmen kann, wie viel Anonymität er aufgeben möchte.

3.4 Grenzfälle

Die Routenplanung auf Basis von Teilrouten hat eine begrenzte Sichtweite für den möglichen, weiteren Verlauf der Route. Daher können während der Routenfindung zwei Grenzfälle auftreten:

a) Einerseits ist es möglich, dass keine weiteren Teilrouten mehr gefunden werden können. Man spricht von einer Sackgasse.
b) Andererseits kann es vorkommen, dass zwar weitere Teilrouten gefunden werden können, diese aber die Distanz zum Ziel nicht verringern und dieses in Konsequenz nicht erreicht wird.

Um diese Fehler zu erkennen und zu beheben, verwendet der dargestellte Algorithmus das Backtrackingverfahren, bei dem entweder die zuvor gewählte Teilroute oder die aktuelle Teilroute verworfen werden.

Fall a) ist der triviale Fall. Sobald für eine Teilroute R_i keine zukünftigen Teilrouten R_{i+1} gefunden werden können, wird R_i verworfen und zurück zu R_{i-1} gesprungen. Anschließend startet der Algorithmus erneut, wobei die Route R_i, die in eine Sackgasse führte, als nicht möglich markiert wird. Dieser Vorgang wird solange wiederholt, bis eine sinnvolle Teilroute gefunden wird.

Um Fall b) abzufangen wird ein Trendsystem eingesetzt, das erkennt, ob sich die Distanz zum Ziel mit der Zeit verringert. Zu diesem Zweck wird das Simple Moving

Average (MA) Verfahren [6] eingesetzt, das einen Mittelwert über eine bestimmte Spannweite von vergangenen Distanzen zum Ziel berechnet. Durch dieses Verfahren können auch Teilrouten gewählt werden, bei denen sich die Distanz zunächst erhöht und mit zukünftigen Teilrouten wieder sinken kann.

Ein einfaches Beispiel soll diese Notwendigkeit verdeutlichen: Um ein Ziel zu erreichen, muss zunächst ein Umweg zur Autobahnauffahrt in Kauf genommen werden. Anschließend kann über die Autobahn das Ziel schneller erreicht werden. Würde man in diesem Fall nicht auf den Trend, sondern auf einen einzelnen Distanzwert von v_{T_i} zu v_T setzen, so würde der Umweg zur Autobahnauffahrt verworfen werden. Zur Anwendung des MA muss ein Schwellwert definiert werden, welcher bestimmt, ab welcher Mehrdistanz die Teilroute verworfen wird. Wird dieser Schwellwert überschritten, so wird die potentielle Route R_{i+1} verworfen, als Kandidat gesperrt und ausgehend von R_i eine neue Teilroute gewählt.

4 Evaluation

Im Folgenden wurde der Algorithmus hinsichtlich seiner Mehrdistanz zur Referenzroute untersucht. Die Mehrdistanz (Overhead) ist dabei wie folgt definiert:

$$overhead = \frac{len(R)}{len(R_{ideal})} = \frac{\sum_i dist(v_{S_i}, v_{T_i})}{dist(v_S, v_T)}$$

Der Overhead kann dabei Werte von 1 bis theoretisch ∞ annehmen. Ist $overhead = 1$ so hat die berechnete Route, die sich aus Teilrouten zusammensetzt, keine Mehrdistanz ggü. der idealen Referenzroute. Es wurde somit die optimale Route gefunden. In Fällen $overhead > 1$ muss eine Mehrdistanz in Kauf genommen werden.

Die Evaluation wurde auf Basis von OSM Daten für die Region Oberpfalz durchgeführt. Diese Kartenbasis wurde gewählt, da sie sowohl städtische Regionen als auch ländliche Gegenden mit verschiedenen Straßentypen enthält. Als Werte für *Hops* wurde 1, 3, 6 und 12 gewählt. Der Wert 1 bietet dabei die höchste Privatsphäre, da es möglichst kleine Teilgraphen erstellt.

Das Vorgehen für die Versuche war wie folgt: Es wurden zufällig Start und Endknoten (v_S, v_T) ausgewählt, die über eine bestimmte Anzahl an Kreuzungen zu erreichen waren, d.h. für die $dist_{G^*}(v_S, v_T) = k$ gilt. Dabei wurde mit $k \in [10{,}25]$ getestet. Für jede Distanz wurden $n = 250$ Wiederholungen mit zufälligen Punktepaaren untersucht.

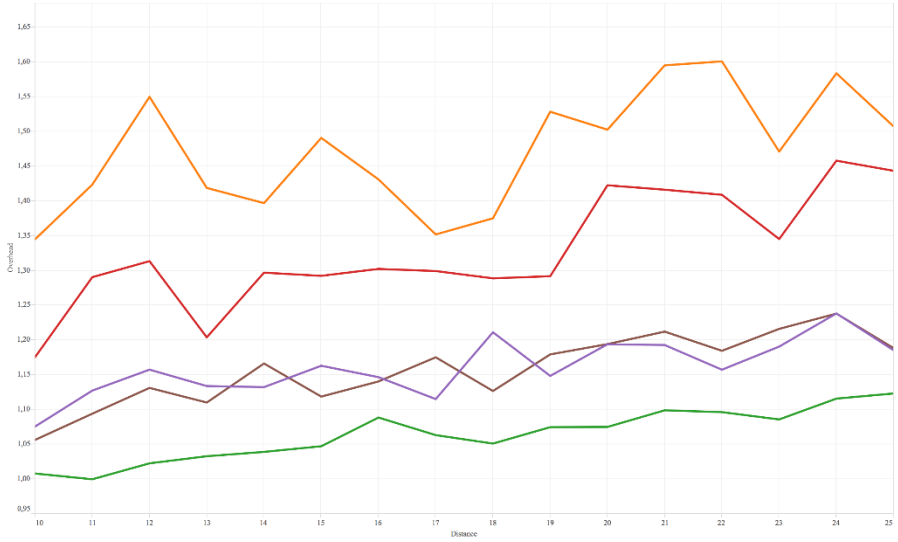

Abbildung 4: Messwerte für $\boldsymbol{n = 250}$, $\boldsymbol{Hops} = [\mathbf{1}; \mathbf{3}; \mathbf{6}; \mathbf{12}]$ (orange, rot, lila, grün; MIXED braun)

Abbildung 4 zeigt, dass $Hops = 12$ (grün) erwartungsgemäß die beste Route bestehend aus Teilrouten zusammenfügt. Für die Distanzen 10, 11 und 12 entspricht $Hops = 12$ der Gesamtroute, da keine Teilgraphen gebildet werden müssen (der berechnete Teilgraph enthält bereits Start- und Zielpunkt). Daher ist kein Overhead messbar. $Hops = 1$ (orange) liefert die schlechtesten Routen mit dem höchsten Overhead, da die Route aus ebenso vielen Teilgraphen wie die Distanz zusammengesetzt ist. $Hops = MIXED$ (braun) wählt aus den möglichen $Hops$ einen zufälligen Wert für jeden Teilgraphen und schneidet ähnlich gut ab wie $Hops = 6$.

5 Ausblick

Der vorgestellte Algorithmus erlaubt die Routenplanung mittels Teilrouten, die von verschiedenen Routenprovidern zur Verfügung gestellt werden. Dadurch ist es einem Nutzer möglich, die lokale Routenplanung mit Online-Informationen, wie bspw. Verkehrsinformationen, zu kombinieren. Weiterhin wird er durch die verschiedenen *Hops* Parameter zwischen Anonymitätsgrad und Routenqualität abwägen können.

Die prototypische Implementierung zeigt, dass sowohl Performance als auch Routenqualität für gängige Anwendungsfälle akzeptabel sind. In der Regel kann für fast alle Start- und Zielknoten eine erfolgreiche Route gefunden werden. Der maximale, mittlere Overhead beträgt 60%, womit die anonyme Teilroute für $Hops = 1$ etwas länger ausfällt.

Für die zukünftige Arbeit ist geplant, das Verfahren an verschiedenen Stellen zu modifizieren. Darunter fällt einerseits die Robustheit, da bisher bei kleinen *Hops* Werten für lange Strecken keine erfolgreiche, akzeptable Route gefunden werden kann. Dieses Problem sollte sich durch Justierung der Paramater wie bspw. Moving Average beheben lassen. Andererseits soll überprüft werden, welche weiteren Kriterien als Abbruchbedingung bei der Routenfindung dienen können. Weiterhin soll der optimale *Hops* Wert abhängig von der Routenlänge untersucht werden. Die Anbindung an weitere LBS ist für eine konkurrenzfähige Routenführung notwendig, um beispielsweise auch aktuelle Verkehrsinformationen einbinden zu können; OSM liefert diesbezüglich keinerlei Information. Durch die Modularität des vorgestellten Verfahren kann dieses Feature jedoch auf einfachem Weg hinzufügen werden. In einem nächsten Schritt soll ebenfalls die Sicherheit des skizzierten Verfahrens hinsichtlich Anonymität untersucht werden. Darunter fallen Fragestellungen wie die optimale Anzahl an Teilrouten oder aber der Schutz gegen externe Beobachter.

References

[1] Zoche, P., et al.: Selbstdatenschutz (Whitepaper). Schriftenreihe Forum Privatheit und selbstbestimmtes Leben in der digitalen Welt. 2. Aufl., 2014

[2] Kapadia, A., Henderson, T., Fielding, J., Kotz, D.: Virtual Walls: Protecting Digital Privacy in Pervasive Environments. Proceedings of 5th International Conference, PERVASIVE 2007, Toronto, Canada, May 13-16, 2007.

[3] Wang, T, Liu, L.: Privacy-Aware Mobile Services over Road Networks. In: Proceedings of the Very Large Data Base Endowment, 2(1), S. 1042-1053, ACM, New York 2009

[4] Pingley, A., et al.: Protection of Query Privacy for Continuous Location Based Services. In Proceedings of IEEE INFOCOM, 2011

[5] Palanisamy, B., Liu, L.: MobiMix: Protecting Location Privacy with Mix-zones over Road Networks. In Proceedings of 2011 IEEE 27th International Conference on Data Engineering, 2011

[6] L. Rong, Y. Qing. Trends analysis of news topics on Twitter. International Journal of Machine Learning and Computing, 2(3):327–332, 2012.

[7] P. Golle, K. Partridge. On the anonymity of home/work location pairs. Pervasive, 2009.