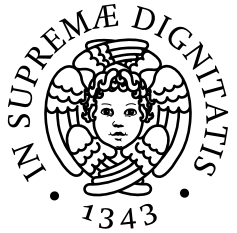


Università di Pisa

Facoltà di Ingegneria

Corso di Laurea Magistrale in Ingegneria Informatica

Tesi di Laurea

Una tecnica per la geolocalizzazione di host Internet basata su crowdsourcing e dispositivi mobili

Relatori:
Prof. Luciano Lenzini
Ing. Alessio Vecchio
Gloria Ciavarrini

Candidato:
Francesco Disperati

ANNO ACCADEMICO 2014/2015

Ai miei genitori, per la pazienza.
A Juana, por el apoyo moral.

Sommario

Il rilevamento della posizione geografica di un host Internet è un argomento di studio particolarmente interessante per le molteplici applicazioni, sia in ambito di ricerca che in ambito commerciale. Negli studi condotti fino ad oggi sono stati utilizzati sistemi dedicati per il sondaggio e il rilevamento della posizione degli host, ottenendo buoni risultati nelle regioni del mondo dove questi sistemi sono presenti (tipicamente Europa e Nord America). In questa tesi viene presentato un nuovo approccio al problema basato sull'utilizzo di dispositivi mobili connessi a reti wireless, utilizzando i dati raccolti in *crowdsourcing* dal progetto Portolan. Analizzando i dati collezionati in due anni di attività è stato elaborato un metodo di geolocalizzazione che tenesse conto delle limitazioni imposte dall'uso di dispositivi mobili. L'obiettivo è disporre di una quantità di sonde superiore di vari ordini di grandezza rispetto ai precedenti metodi, specialmente nelle zone del mondo dove le reti di ricerca non sono ancora sviluppate.

Indice

Elenco delle figure

Elenco delle tabelle

Capitolo 1

Introduzione

1.1 Motivazioni

La possibilità di determinare con precisione la posizione geografica di un host Internet (processo chiamato *geolocalizzazione IP*) trova molteplici applicazioni, dagli usi commerciali alla ricerca tecnologica:

- *localizzazione automatica* – visualizzazione dei contenuti nella lingua di provenienza dell'utente;
- *restrizioni sui contenuti* – distribuzione di contenuti limitata a certi paesi per ragioni politiche o di copyright. Le restrizioni sui contenuti vengono tipicamente applicate da servizi di streaming come Netflix, Spotify, YouTube;
- *Geomarketing* – utilizzo delle informazioni sulla posizione dell'utente per analizzare, pianificare ed attuare attività di marketing online;
- *controlli di sicurezza basati sulla posizione* – ad esempio permettere l'uso di certi servizi solo all'interno della sede di un'azienda, o impedire attività provenienti da posizioni insolite per un particolare utente (per citare un caso concreto, Google monitora la posizione geografica dei suoi utenti e blocca gli accessi da posizioni inusuali [1]);

- *mappaggio di Internet* – localizzazione di router e nodi intermedi di Internet per conoscerne la struttura.

L'ultimo punto è il più interessante, perché se per le applicazioni destinate all'utenza di massa possiamo ricavare la posizione tramite l'hardware del dispositivo (ricevitori GPS e antenne radio), non esistono metodi diretti per trovare la posizione dei nodi interni di Internet.

1.2 Studi precedenti

Nel corso dell'ultimo decennio sono stati proposti vari metodi per la geolocalizzazione di indirizzi IP arbitrari. Possiamo suddividere queste tecniche in due grandi famiglie: *metodi basati su registro* e *metodi basati su misurazione attiva*.

I *metodi basati su registro* cercano di inferire la posizione da database precompilati (tipicamente raccogliendo le informazioni di `DNS LOC` e `Whois`).

I *metodi basati su misurazione attiva* utilizzano un set di host di cui è nota la posizione per effettuare misurazioni di rete verso gli host che si intende geolocalizzare. Si suppone di avere il controllo diretto degli host noti ma non dell'host bersaglio, le misurazioni sono pertanto unidirezionali. Nel seguito chiameremo *landmark* gli host di riferimento e *target* gli host di cui vogliamo conoscere la posizione.

1.2.1 Metodi basati su registro

Il protocollo DNS mette a disposizione un tipo di record specifico denominato `DNS LOC` attraverso il quale i possessori di un nome di dominio possono dichiarare la posizione di host e sottoreti [2]. Eseguendo query per questo tipo di record è teoricamente possibile conoscere le coordinate associate al dominio (espresse in latitudine, longitudine e altitudine), ma in pratica l'adozione del record `LOC` è rimasta limitata a pochi casi specifici (ad esempio per i satelliti geostazionari), in quanto enti ed istituzioni preferiscono non divulgare questo tipo di informazioni.

Similmente tramite i database `Whois` è possibile risalire all'indirizzo della sede dell'organizzazione che controlla un dato blocco di indirizzi IP, tecnica utilizzata da strumenti come NetGeo [3]. Tuttavia le informazioni reperibili possono essere scorrette o non aggiornate. In molti casi le posizioni reali di un indirizzo IP sono completamente differenti dall'indirizzo riportato per la sottorete di appartenenza, specialmente nel caso di blocchi di IP molto grandi intestati ad un'unico ente [4, 5].

Altri tecniche cercano di compilare in maniera esaustiva un database con intervalli di indirizzi IP associati alle rispettive posizioni geografiche, raccogliendo i contributi degli utenti o degli ISP, come ad esempio GeoURL [6] e altri servizi commerciali [7, 8]. In ogni caso è molto difficile mantenere un elenco preciso ed aggiornato.

Proprio per la difficoltà di ottenere informazioni aggiornate le tecniche basate su registro sono generalmente considerate obsolete, e oggi rimangono di supporto per validare alcune fasi delle tecniche basate su misurazione attiva.

1.2.2 Geoping

L'idea di utilizzare il Round Trip Time (RTT) misurato dai landmark come indizio per il posizionamento di un target nasce con Geoping, una delle tre tecniche di geolocalizzazione proposte da IP2Geo [9]. Geoping individua la posizione di un target associandolo con il landmark più vicino in termini di RTT, supponendo che due host vicini geograficamente riportino RTT simili.

Ogni landmark conosce la posizione degli altri landmark e il tempo di propagazione verso ciascuno, memorizzato in un vettore di tempi caratteristico. Per geolocalizzare un indirizzo IP arbitrario viene misurato il RTT da tutti i landmark verso il target e confrontato con i vettori dei RTT di ogni landmark: la posizione stimata è la posizione del landmark con i vettori dei tempi più simili. Viene quindi fornita una risposta appartenente ad uno spazio discreto di soluzioni. Con questa tecnica possiamo avere anche landmark passivi (cioè che non eseguono misurazioni) che servono ad aumentare l'accuratezza del processo di localizzazione.

1.2.3 Constraint-Based Geolocation

Invece di mappare il target alla posizione di un singolo landmark, l'algoritmo Constraint Based Geolocation (abbreviato CBG [10]) utilizza i tempi di propagazione misurati da più landmark per generare un insieme di vincoli da cui dedurre la posizione.

I tempi di propagazione vengono convertiti in distanze tramite una funzione caratteristica di ogni landmark (*bestline*); tali distanze costituiscono il raggio di regioni circolari in cui è ragionevole ipotizzare che il target risieda. Il target è arbitrariamente stimato nel *baricentro geometrico* dell'intersezione di tutte le regioni, e l'area dell'intersezione fornisce una misura dell'incertezza del calcolo.

Per calcolare la *bestline* ogni landmark misura il tempo di propagazione verso gli altri landmark; successivamente viene trovata la linea più aderente a tutte le coppie di valori `(RTT, distanza)` con una funzione di fitting. La *bestline* viene calcolata offline e aggiornata periodicamente per rispondere ai cambi topologici della rete.

1.2.4 Topology Based Geolocation

Topology Based Geolocation (abbreviato TBG [11]) riprende le intuizioni dell'algoritmo CBG, aggiungendo le informazioni sulla topologia di rete.

Anziché calcolare la *bestline*, TBG opera in maniera più cauta ed utilizza la velocità massima di trasmissione per avere un margine maggiore nelle regioni di ogni landmark. Tramite `traceroute` vengono raccolti i tempi di propagazione verso il target e i router intermedi, individuando i percorsi comuni a più landmark con un algoritmo di *clustering* delle interfacce di rete che sono posizionate sullo stesso router. Dopo una fase di validazione e incorporamento delle informazioni derivanti dai record `DNS`, viene risolta la posizione di ogni nodo minimizzando una funzione obiettivo in termini di errore.

TBG si basa su una funzione di ottimizzazione globale applicata indistintamente a router e target, con lo svantaggio di introdurre errori di posizionamento dei target nel tentativo di minimizzare l'errore dei router intermedi.

Inoltre l'algoritmo deve necessariamente essere eseguito su tutti i target, rendendone impossibile l'utilizzo in tempo reale per singole richieste.

1.2.5 Octant

A differenza dei precedenti metodi che stimano una regione dove *è plausibile* che si trovi il target, Octant [12] considera anche le zone dove *non è possibile* che risieda il target.

Come CBG e TBG viene utilizzata l'intersezione delle *regioni di plausibilità* di ogni landmark, ma in questo caso la regione è rappresentata da un anello circolare delimitato dal massimo e dal minimo RTT. Alle regioni vengono assegnati dei pesi considerando anche informazioni sulla demografia e sulla geografica del luogo (per esempio assegnando pesi inferiori alle aree oceaniche o poco popolate). I vincoli sono implementati da curve Bézier per una rappresentazione precisa e a basso costo computazionale.

1.2.6 Spotter

Spotter [4] abbandona l'approccio puramente geometrico dei precedenti lavori per descrivere il rapporto RTT–distanza attraverso una funzione di probabilità spaziale. La funzione è costruita sulla base di osservazioni di dati raccolti precedentemente ed è unica per tutti i landmark. Unendo le funzioni di probabilità di tutti i landmark viene calcolata la regione con la più alta probabilità di contenere il target.

1.3 Utilizzo dei dispositivi mobili: un approccio bottom-up

Negli studi finora condotti i landmark sono server appartenenti a reti di ricerca (tipicamente PlanetLab e CAIDA), connessi da reti ad alte prestazioni. Risultati sempre più accurati sono stati raggiunti nel corso del tempo anche grazie a connessioni veloci e bassa latenza.

L'approccio comune è di tipo *top-down*: le misure sono effettuate dai nodi centrali di Internet, fortemente interconnessi, verso le propaggini della

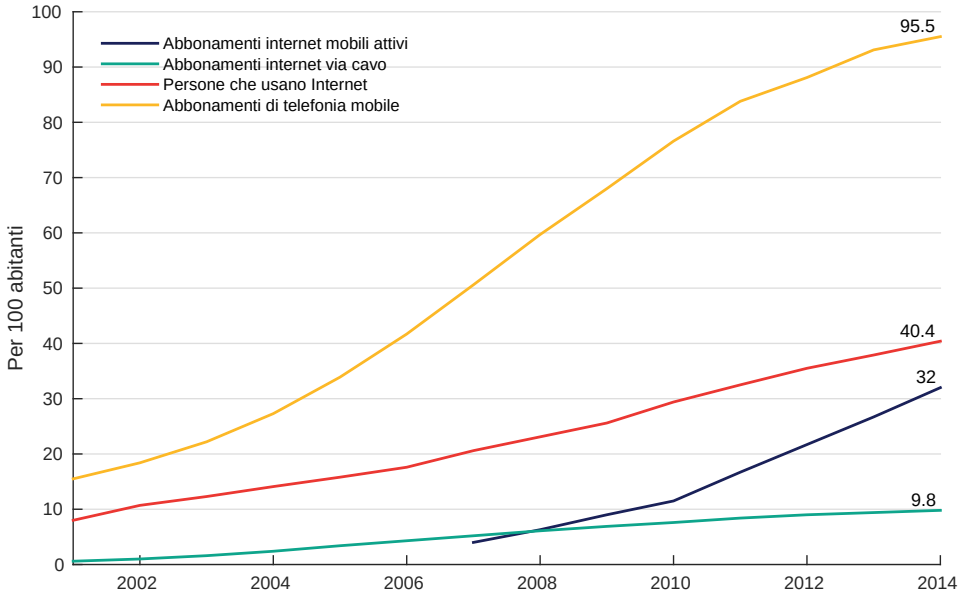


Figura 1.1: Sviluppo globale dell'ICT nel periodo 2001-2014

rete. Lo svantaggio principale di questo approccio è la necessità di installare nodi nelle regioni dove devono essere condotte le misurazioni, tant'è che in tutti gli studi analizzati risultati soddisfacenti sono stati ottenuti solo in Europa e Nord America, dove la rete PlanetLab è più sviluppata.

1.3.1 Caratteristiche dei dispositivi mobili

Il mercato dei dispositivi mobili è in continua crescita. Nel 2014 gli smartphone venduti in tutto il mondo hanno sorpassato il miliardo di unità [13], e gli analisti prevedono che le vendite continueranno ad incrementare per i prossimi anni. Il numero di contratti dati per rete mobile ha sorpassato quelli per accesso da rete fissa nel 2008, arrivando a 32 contratti ogni 100 abitanti [14] (si veda Figura 1.1). Un dispositivo mobile odierno vanta performance molto vicine a quelle di PC desktop; inoltre può essere localizzato facilmente utilizzando l'hardware presente (GPS, antenne radio cellulari, Wi-Fi).

Le caratteristiche e la diffusione fanno degli smartphone uno strumento perfetto per condurre analisi di rete distribuita. L'idea è quella che viene

definita *crowdsourcing*, cioè l'esternalizzazione di un'attività ad un insieme indefinito di persone non organizzate inizialmente, e possibilmente amplio. Poter utilizzare questi dispositivi come *vantage point* per l'analisi di Internet è l'idea alla base del progetto Portolan.

1.3.2 Il progetto Portolan

Portolan [15] è un progetto di ricerca dell'Università di Pisa e dell'Istituto IIT/CNR di Pisa che ha lo scopo di aumentare la conoscenza della struttura di Internet. Il progetto nasce nel Gennaio 2013 ed è il primo sistema crowdsourcing basato su smartphone (Android) per esplorare mediante `traceroute` la struttura di Internet. Nei due anni di attività sono stati analizzati dati provenienti da centinaia di dispositivi sparsi per il globo. L'obiettivo è di estendere l'uso dell'applicazione ai milioni di utenti di smartphone in tutto il mondo, al fine di ottenere una visione completa e precisa della struttura di Internet.

1.4 Obiettivi e contenuto della tesi

Le parti di Portolan dedicate alla raccolta dei dati sono già state implementate e testate con successo [16, 17]. Il lavoro presentato in questa tesi consiste nello studio e successiva implementazione di un metodo per la geolocalizzazione di host Internet, utilizzando i dati raccolti da Portolan, per poterlo integrare nella piattaforma esistente.

È stata effettuato un'accurata analisi dei dati raccolti per comprenderne possibili relazioni ed è stato elaborato un algoritmo di geolocalizzazione in cui confluiscono tutte le osservazioni fatte. L'algoritmo è ispirato agli approcci geometrici visti precedentemente (in particolare CBG), migliorati ed adattati all'ambito wireless.

La tesi è strutturata in cinque capitoli: in seguito a questa introduzione nel Capitolo 2 verranno delineate le basi teoriche e verrà spiegato il funzionamento di questo nuovo metedo di geolocalizzazione. Nel Capitolo 3 verranno analizzate in dettaglio le relazioni tra RTT e distanza, e tra RTT

e altri parametri quali il tipo di rete di accesso, la distanza per numero di hop e di Autonomous System attraversati; vedremo quindi le iterazioni svolte per integrare questi dati e migliorare i risultati. Nel Capitolo 4 verrà presentata in dettaglio l'implementazione del prototipo con cui sono stati effettuati i test; infine nel Capitolo 5 verranno valutati i risultati di questo studio e verrà offerta una panoramica sui possibili sviluppi futuri.

Capitolo 2

Geolocalizzazione IP basata su crowdsourcing e dispositivi mobili

In questo capitolo verrano spiegate come sono state utilizzate le misurazioni dagli smartphone per elaborare l'algoritmo. Partendo dai RTT vedremo come è possibile stimare la distanza e costruire dei vincoli per identificare la posizione del target.

2.1 Portolan e le campagne di misurazione

Le misurazioni vengono lanciate dal server di controllo di Portolan e propagate ai dispositivi dotati dell'applicazione Portolan client. In ogni *campagna di misurazione* vengono effettuate delle esecuzioni di `ping` e `traceroute` (nella versione Paris Traceroute [18]) verso uno o più target designati, georeferenziando i dati raccolti grazie a ricevitori GPS o altri sistemi di posizionamento presenti negli smartphone. La mobilità dei dispositivi gioca un ruolo chiave perché permette di condurre la stessa misurazione da posizioni e reti differenti, aumentando le informazioni disponibili.

In Figura 2.1 vediamo schematizzato il processo di raccolta dati in Portolan. Il server di controllo invia un comando per l'inizio di una campagna

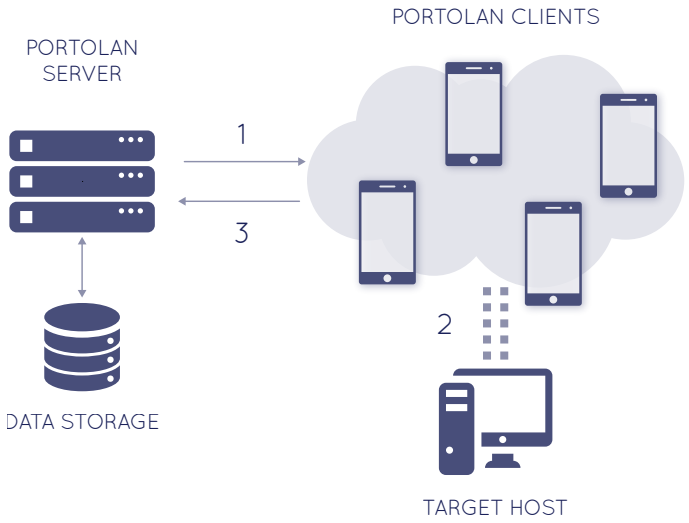


Figura 2.1: Schema semplificato di una campagna di misurazione in Portolan

di misurazione a tutti i dispositivi *online* (fase 1). I dispositivi che ricevono il segnale effettuano le misurazioni, inviando una serie di probe UDP per il traceroute o ICMP per il ping, verso i target selezionati (fase 2). Ogni dispositivo raccoglie le informazioni richieste e alla fine del processo comunica al server i risultati (fase 3), che organizza e archivia i dati ricevuti.

2.1.1 Descrizione dei dati utilizzati

Il set di dati utilizzato in questa tesi è relativo ad una serie di campagne condotte tra luglio 2013 e febbraio 2015, comprendenti 532 dispositivi dislocati su ogni continente. I 401 target delle campagne sono nodi appartenenti a host fissi o ad altri smartphone di cui è nota la posizione geografica. I dati sono organizzati in due tabelle, `ping` e `traceroute`, contenenti informazioni registrate dagli strumenti omonimi, e dalla tabella `target`.

Ping

La tabella di `ping` contiene una entrata per ogni campione di RTT, ed è descritta da:

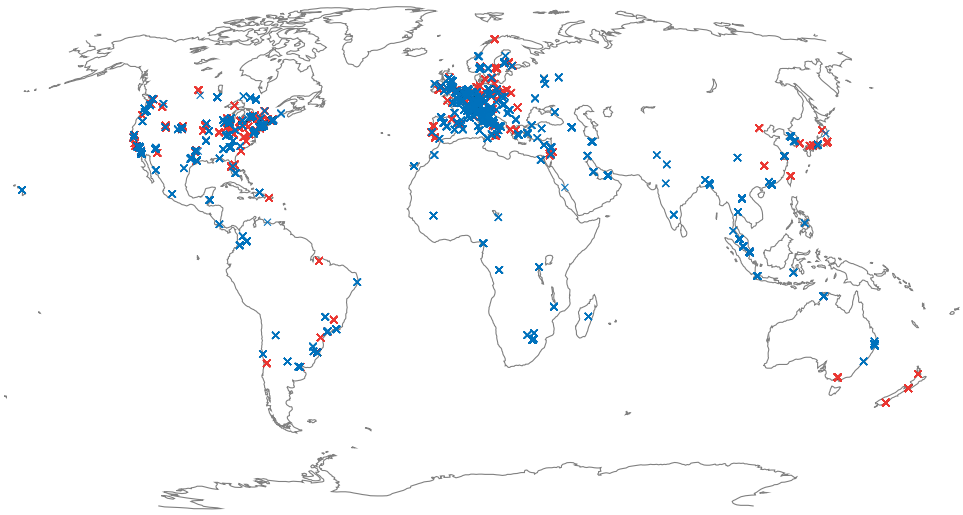


Figura 2.2: Posizione geografica di target e landmark del dataset (in rosso e blu, rispettivamente)

pid – Phone ID. Codice univoco del terminale che ha effettuato la misura.

cid – Codice identificativo della campagna di misura.

timestamp – Data e ora del momento in cui viene lanciato il ping.

position – Posizione geografica del terminale che ha lanciato il ping. La posizione è espressa in latitudine e longitudine secondo il modello `WGS84`.

accuracy – Precisione della posizione in metri.

rtt – `Round Trip Time` del ping in millisecondi.

net_type – Tecnologia di accesso (ad esempio WIFI, GSM, HSDPA, etc.).

target_id – Riferimento al target del ping.

Traceroute

La tabella di `traceroute` contiene una entrata per ogni hop attraversato, ed è descritta da:

Dato	Campioni totali	Campioni validi	Rapporto
Target	1.452	401	27,61 %
Ping	1.362.422	759.539	55,74 %
Traceroute	759.679	294.850	38,81 %

Tabella 2.1: Dettaglio dei dati utilizzati

pid – Phone ID. Codice univoco del terminale che ha effettuato la misura.

cid – Codice identificativo della campagna di misura.

timestamp – Data e ora del momento in cui viene lanciato il `traceroute`.

hop – Numero di hop del particolare trace.

interface_ip – Indirizzo IP dell'interfaccia a cui si era arrivati all'hop precedente.

nexthop_ip – Indirizzo IP del prossimo hop riferito all'hop corrente.

rtt – RTT relativo al particolare hop.

AS_number – Numero dell'Autonomous System a cui appartiene `interface_IP`.

target_id – Riferimento al target del trace.

Target

I target sono semplicemente descritti da:

ip – Indirizzo IP da geolocalizzare.

position – Posizione geografica del terminale che ha lanciato il ping. La posizione è espressa in latitudine e longitudine secondo il modello `WGS84`.

Un *landmark* (identificato dal `pid`) può aver eseguito più misurazioni verso un determinato target, e in più campagne differenti. Un singolo *burst*

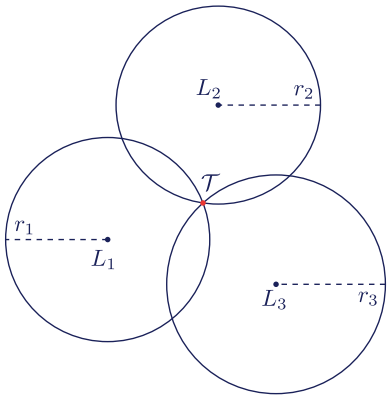


Figura 2.3: Trilaterazione su superficie piana utilizzando tre ancore

di ping viene individuato dalla tupla `(pid,cid,target_id)`. Per semplicità nel seguito chiameremo questa tupla con i termini *misure di ping* o *valori di RTT* o *RTT*.

Molte misurazioni sono risultate non valide o non utilizzabili. Sono state scartate tutte le entrate prive di informazioni geografiche, e i RTT con misure nulle o negative (Tabella 2.1).

2.2 Trilaterazione in Internet

La posizione geografica di un punto può essere stimata utilizzando un numero sufficiente di misurazioni verso delle ancore di cui è nota la posizione. Il processo è conosciuto in geometria come *triangolazione* o *trilaterazione*, a seconda che vengano utilizzate le misure degli angoli o delle distanze. In Figura 2.3 vediamo un esempio di trilaterazione in uno spazio bidimensionale: conoscendo le distanze dai punti L_i, è possibile posizionare il punto $\mathcal{T}$ all'interno dell'intersezione dei tre cerchi con raggio r_i pari alla distanza $\overline{L_i\mathcal{T}}$.

Il problema principale della trilaterazione in Internet è come convertire le misure di di `ping` o di `traceroute` in informazioni di distanza. Il sistema di geolocalizzazione GPS (Global Positioning System) utilizza il tempo di propagazione di segnali radio per ricavare le distanze ed eseguire la trilaterazione. La distanza ricevitore–satellite è calcolata misurando il *tempo di volo*

(anche detto TOF, Time of Flight) di un segnale inviato a intervalli regolari da ogni satellite. Il segnale contiene la posizione del satellite e l'istante esatto di emissione; conoscendo la velocità di propagazione del segnale, un ricevitore GPS è in grado di calcolare il TOF ed individuare la sua posizione in uno spazio tridimensionale. Per risolvere la posizione è richiesta la presenza di almeno quattro satelliti: tre per le coordinate di latitudine, longitudine e altezza, più uno per sincronizzare l'orologio del ricevitore con quello dei satelliti.

Nel caso di GPS i segnali viaggiano in linea retta e i satelliti trasmettitori sono equipaggiati con precisi orologi atomici; il mezzo di trasmissione è l'atmosfera, che nonostante possa alterare la propagazione delle onde per gli effetti della ionosfera o per la presenza di vapore acqueo, pioggia e pulviscolo, è descrivibile con un modello matematico accurato. Il risultato è un errore di qualche metro nei casi più estremi, errore generalmente trascurabile per le applicazioni civili [19, 20].

Nel caso di Internet i segnali sono i pacchetti di ping trasmessi dai landmark verso il target, e il mezzo di trasmissione non è noto a priori. Il percorso target–landmark è composto da innumerevoli cammini di varia natura (fibra ottica, cavi in rame, onde radio, etc.), selezionati in maniera non predicibile dai router intermedi. Spesso il percorso è asimmetrico, cioè il ping percorre differenti strade nel viaggio di andata e in quello di ritorno [21].

In particolare il tempo di propagazione è scomponibile in due parti:

- *ritardo fisso* – comprende il tempo di propagazione e tempo minimo di processamento del pacchetto a router scarico;
- *ritardo stocastico* – code nei router, tempo di processamento addizionale.

Anche per il ritardo fisso ci sono alcune problematiche:

- *assenza di percorsi diretti* – i collegamenti raramente seguono il percorso più breve tra host (*distanza di cerchio massimo*), ma possono essere più o meno tortuosi a seconda della geografia o degli accordi tra Autonomous System [22];

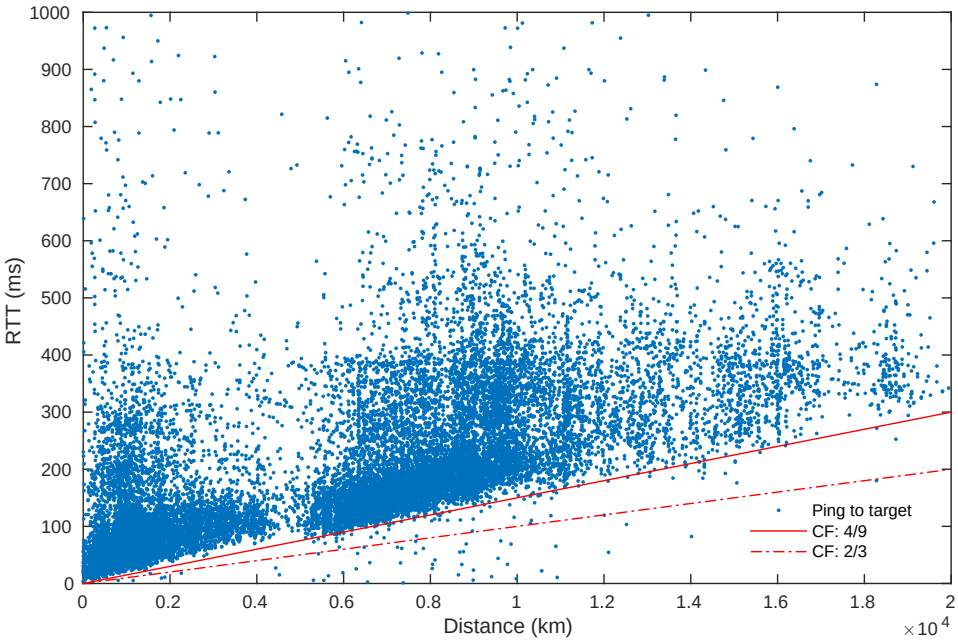


Figura 2.4: Grafico a dispersione dei RTT minimi di ogni landmark in relazione alla distanza dal target

- *ultimo miglio wireless* – il dispositivo mobile è tipicamente connesso ad una rete di accesso wireless che aumenta sensibilmente il tempo di propagazione nell'ultimo tratto. I mezzi di trasmissione radio sono anche caratterizzati da una varianza maggiore rispetto ai mezzi cablati, che rende questo ritardo meno predicibile.

Possiamo però fissare un limite inferiore al RTT oltre il quale non è fisicamente possibile arrivare, ed è il tempo che impiegherebbe il segnale per viaggiare su un ipotetico cavo in fibra ottica steso lungo il percorso di cerchio massimo tra i due punti. Sappiamo che questa velocità equivale a $\frac{2}{3}c$ (con c velocità della luce nel vuoto), ma in realtà nei test effettuati è stato rilevato un limite inferiore pari a circa $\frac{4}{9}c$, risultato confermato anche da TBG [11].

$\frac{2}{3}$ e $\frac{4}{9}$ sono due valori della costante denominata *Conversion Factor* (CF), che è definita come:

$$CF = \frac{2d}{RTT \cdot c} \tag{2.1}$$

con d distanza landmark–target. Analizzeremo meglio la funzione di questa costante nelle prossime sezioni.

Nel grafico di Figura 2.4 vediamo rappresentati i dati di ping collezionati da Portolan. Ogni punto rappresenta il minimo RTT estratto da ogni *burst* di ping, rapportato alla distanza di cerchio massimo dal landmark al target. La linea continua è la retta rappresentate questo limite superiore, mentre la linea tratteggiata è la velocità di propagazione teorica nella fibra ottica. Dei 21.200 campioni presi in esame, oltre il 99 % rispetta questo limite empirico, mentre appena 82 sono limitati da CF $\frac{2}{3}$. I 48 punti che vediamo al di fuori delle due rette sono con molta probabilità dovuti a errori di misurazione.

2.3 Descrizione dell'algoritmo

La relazione RTT–distanza è usata per tracciare la *regione di plausibilità* di ogni landmark, cioè lo spazio continuo in cui è probabile che si trovi il target.

Consideriamo un set di K landmark $\mathcal{L} = \{L_1, L_2, \ldots, L_K\}$. Ogni landmark L_i ha posizione nota e conterrà una o più misure di RTT verso un determinato target $\mathcal{T}$ che chiameremo $\mathcal{R}_i = \{RTT_{1L_i \to \mathcal{T}}, RTT_{2L_i \to \mathcal{T}}, \cdots, RTT_{jL_i \to \mathcal{T}}\}$. Per ogni L_i viene calcolato il raggio massimo r_i entro il quale il pacchetto può essere propagato con

$$r_i = \frac{\overline{\mathcal{R}_i}}{2} \cdot CF \cdot c \tag{2.2}$$

dove $\overline{\mathcal{R}_i}$ rappresenta il RTT medio, CF è il Conversion Factor visto prima, e c rappresenta la velocità della luce nel vuoto ($c \simeq 3 \cdot 10^6 [\frac{km}{s}]$). Ogni landmark stima quindi che il target sia all'interno del cerchio C_{L_i} centrato in L_i.

Viene ordinato l'insieme $\mathcal{L}$ per valori crescenti di $\overline{\mathcal{R}}$ e partendo dal primo landmark viene calcolata l'intersezione geometrica con il successivo:

$$\mathcal{A} = \bigcap_{i=1}^{K} C_{L_i} \tag{2.3}$$

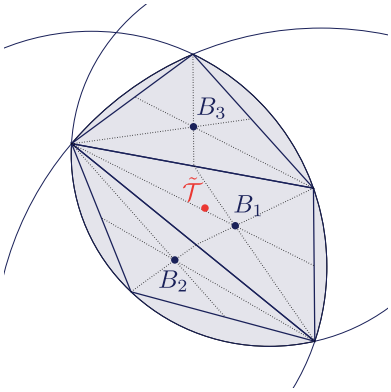


Figura 2.5: Calcolo del baricentro della regione convessa per scomposizione

Il risultato è una regione convessa $\mathcal{A}$ che descrive l'area in cui si trova presumibilmente il target. Prima di ogni intersezione vengono fatti ulteriori controlli che vedremo in dettaglio nelle prossime sezioni. Se al passo i-esimo l'intersezione $C_{L_i} \bigcap C_{L_{i+i}} = \varnothing$, il landmark viene scartato come outlier e l'algoritmo prosegue con il landmark successivo.

La posizione di $\mathcal{T}$ viene stimata come il baricentro di $\mathcal{A}$. Per il calcolo del baricentro la regione viene scomposta in N triangoli e viene eseguita una media pesata del baricentro di ogni triangolo. Le coordinate stimate del target sono quindi date da:

$$\tilde{\mathcal{T}}_{(x,y)} = \frac{\sum_{i=1}^{N} A_i B_{i(x,y)}}{\sum_{i=1}^{N} A_i}, \tag{2.4}$$

$$B_{i(x,y)} = \left(\frac{1}{3}(x_a + x_b + x_c), \;\; \frac{1}{3}(y_a + y_b + y_c) \right) \tag{2.5}$$

dove:

$\tilde{\mathcal{T}}_{(x,y)}$: latitudine e longitudine stimate
A_i : area del triangolo i-esimo
$B_{i(x,y)}$: baricentro del triangolo i-esimo di vertici a, b, c

In Figura 2.5 è illustrato questo procedimento per $N = 3$.

2.4 Validazione dei risultati

Per valutare il miglioramento dell'algoritmo al variare delle condizioni è stata studiata la distribuzione cumulativa dell'errore nella stima dei target, tenendo in considerazione anche il numero di target geolocalizzati con successo. L'errore di stima viene calcolato come la distanza di cerchio massimo tra la posizione reale del target e quella stimata.

Capitolo 3

Analisi dei risultati

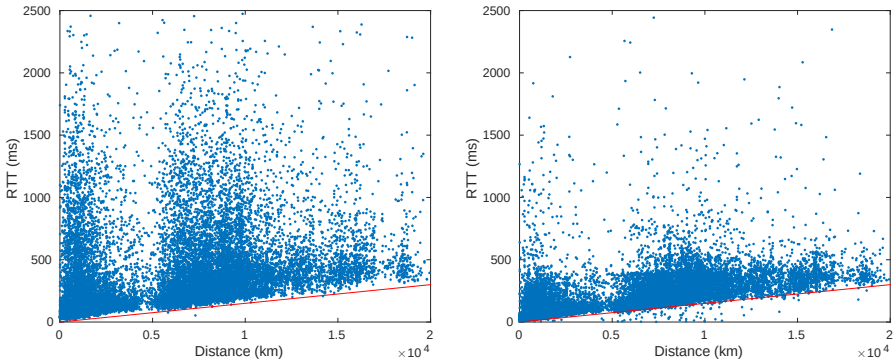
Dopo aver discusso la teoria alla base dell'algoritmo nel Capitolo 2, vedremo adesso le motivazioni empiriche delle scelte fatte. Verranno analizzati i test condotti al variare dei parametri quali CF, RTT ammissibili e tipo di landmark, valutandone i miglioramenti e le possibili cause.

3.1 Utilizzo dei RTT

Durante le campagne di ping ogni landmark ha collezionato in media 30 valori di RTT in un singolo *burst* verso un determinato target. Questi valori presentano una forte dispersione dovuta a:

- *ritardo variabile reti wireless* – scarsa ricezione del segnale o picchi di traffico possono alterare notevolmente singole misure di RTT;
- *approssimazioni hardware* – il calcolo esatto del RTT richiede un clock preciso, e su dispositivi mobili interferiscono vari fattori come il risparmio energetico o il multitasking.

In letteratura il RTT minimo è solitamente scelto per minimizzare gli effetti delle code nei router [4, 10]: l'ipotesi ha senso in un ambiente dove le misurazioni vengono condotte da server dedicati allo scopo e connessi a reti cablate ad alte prestazioni, ma è difficilmente applicabile ad un ambito mobile.



(a) RTT avg. (b) RTT min.

Figura 3.1: Confronto tra RTT medio e RTT minimo

In Figura 3.1 vediamo un confronto tra RTT medio e RTT minimo per tutto il set di dati. Il RTT minimo ha una distribuzione più uniforme e aderente al modello lineare, ma presenta qualche *outlier* (campioni sotto la linea) che influiscono negativamente sulle performance.

Negli esperimenti condotti è stato constatato che risultati migliori si ottengono seguendo un approccio misto: viene utilizzato il valore minimo di un *burst* di RTT per calcolare il CF, mentre viene utilizzato il valore medio per tracciare la regione di plausibilità del landmark. La ragione è da ricercare nella varianza delle misure di RTT minime, che possono portare a vincoli di distanza troppo stretti per ottenere una regione di intersezione. In fase di calibrazione del CF invece viene utilizzata una funzione di fitting che attutisce l'influenza di questi outlier.

3.2 Relazione RTT–distanza e selezione dei landmark

Dai test effettuati è stato rilevato che spesso solo una parte dei landmark disponibili contribuisce al risultato finale. In molti casi i RTT sono così elevati che l'area del landmark copre un intero emisfero e oltre: è il caso ad esempio di landmark in Europa che inviano ping a target in Sud America, e in generale per landmark troppo lontani geograficamente dal target.

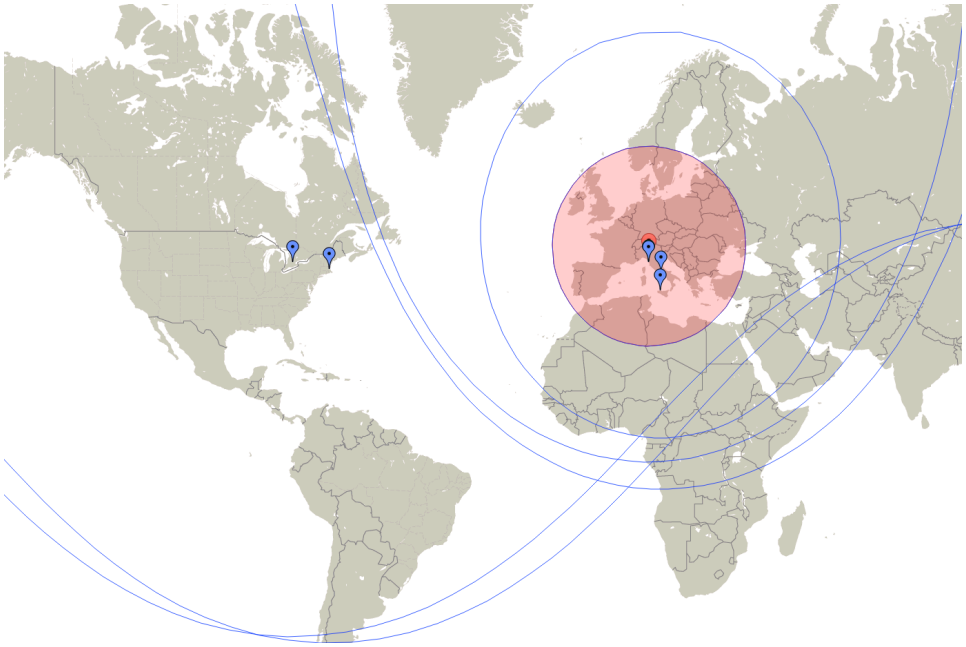


Figura 3.2: Esempio di intersezione tra landmark intercontinentali: i landmark in Nord America non influiscono sul risultato

Nell'esempio in Figura 3.2 vediamo come la regione finale (in rosso) sia interamente contenuta all'interno delle regioni tracciate dai landmark in Nord America.

Le dorsali oceaniche di collegamento possono anche creare problemi in alcune situazioni, poiché i CF sono calibrati per percorsi mediamente più lenti. In Figura 3.3 vediamo un esempio di questa situazione: il target in Nord America (in verde) viene erroneamente localizzato in mezzo all'Oceano Atlantico come risultato dell'intersezione dei due landmark (in blu). In questo caso la regione del landmark europeo è notevolmente sottostimata a causa di un collegamento particolarmente veloce (e diretto) rispetto agli altri con i quali è stato calcolato il CF.

In generale i landmark con RTT elevati sono meno affidabili di quelli con RTT più piccoli. È un risultato atteso, perché con l'aumento della distanza aumentano le probabilità di attraversare percorsi indiretti o di trovare nodi congestionati [12].

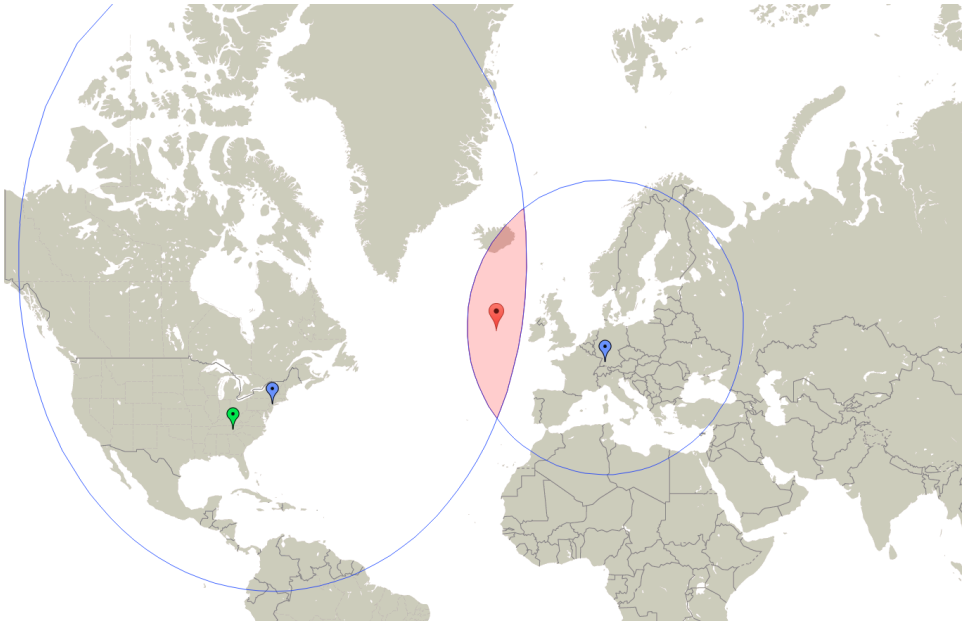


Figura 3.3: Esempio di intersezione tra landmark intercontinentali: risultato errato a causa di un collegamento a bassa latenza

Pertanto questi landmark devono essere scartati in fase di esecuzione.

Osservando meglio il grafico a dispersione dei RTT notiamo anche un evidente diradamento intorno ai 5.000 km. La distanza lascia suggerire un vuoto intorno ai confini continentali; a conferma di questa ipotesi è stato analizzato il continente di provenienza di landmark e target per ogni RTT: in Figura 3.4 si vede come la quasi totalità dei collegamenti oltre i 5.000 km appartenga a continenti diversi.

L'entità di questa separazione è evidente osservando la funzione di ripartizione per le due categorie di RTT in Figura 3.5: il 99 % delle distanze landmark-target che appartengono allo stesso continente sono sotto i 5.900 km, mentre entro questa distanza risiede appena il 12 % dei collegamenti inter-continentali.

Limitando il raggio massimo dei landmark si ottiene una precisione media maggiore, a discapito però del numero di target localizzati con successo. In Figura 3.6 vediamo l'andamento dell'errore nei casi di limite a 3.500 km (95°percentile RTT intra-continentali), 4.000 km (97°percentile),

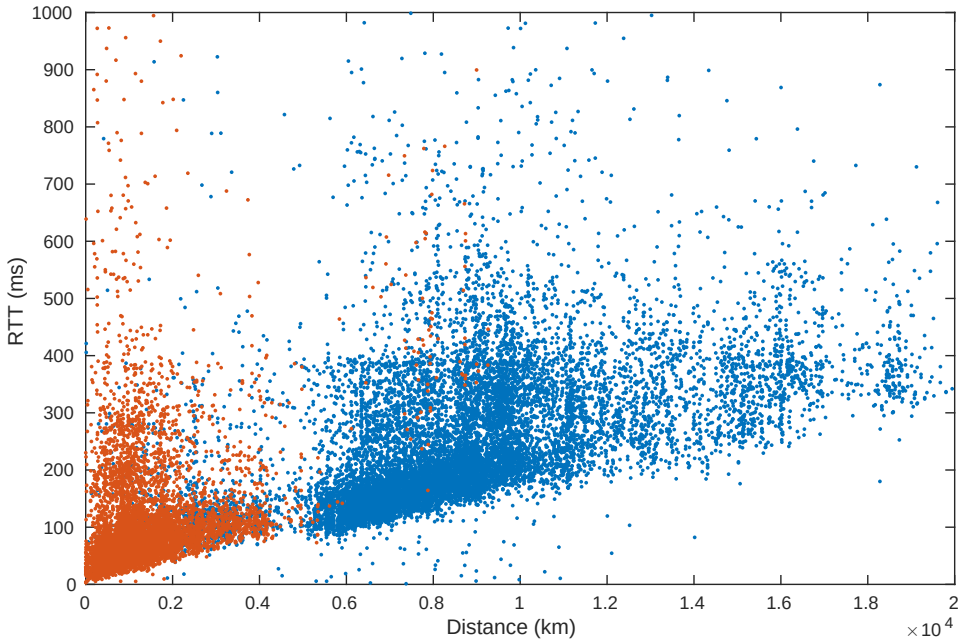


Figura 3.4: Grafico a dispersione dei RTT divisi per collegamenti intra e inter continentali (in rosso e blu rispettivamente)

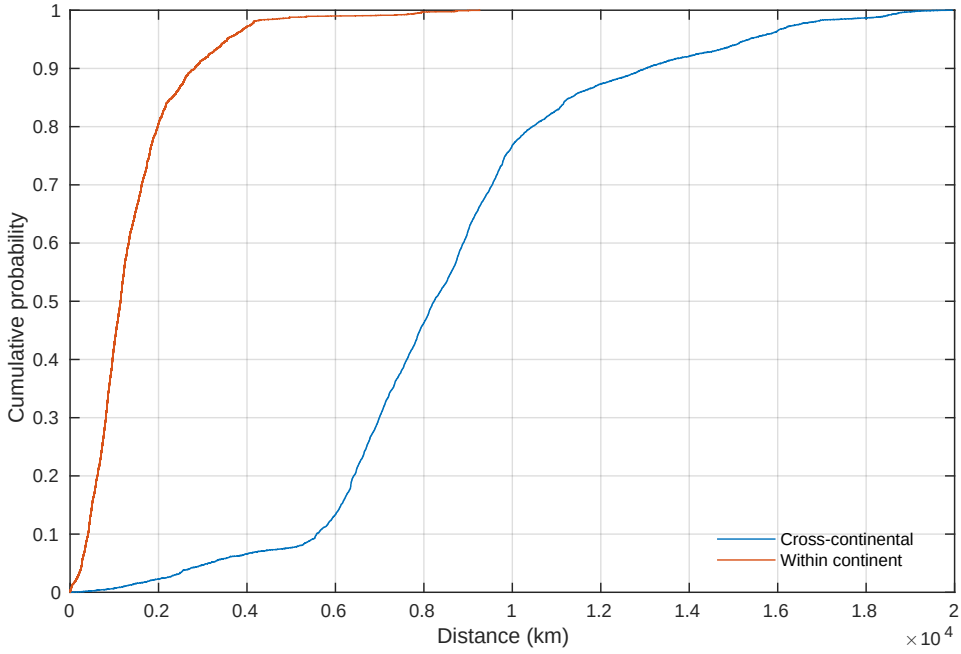


Figura 3.5: Funzione di ripartizione per i RTT di Figura 3.4

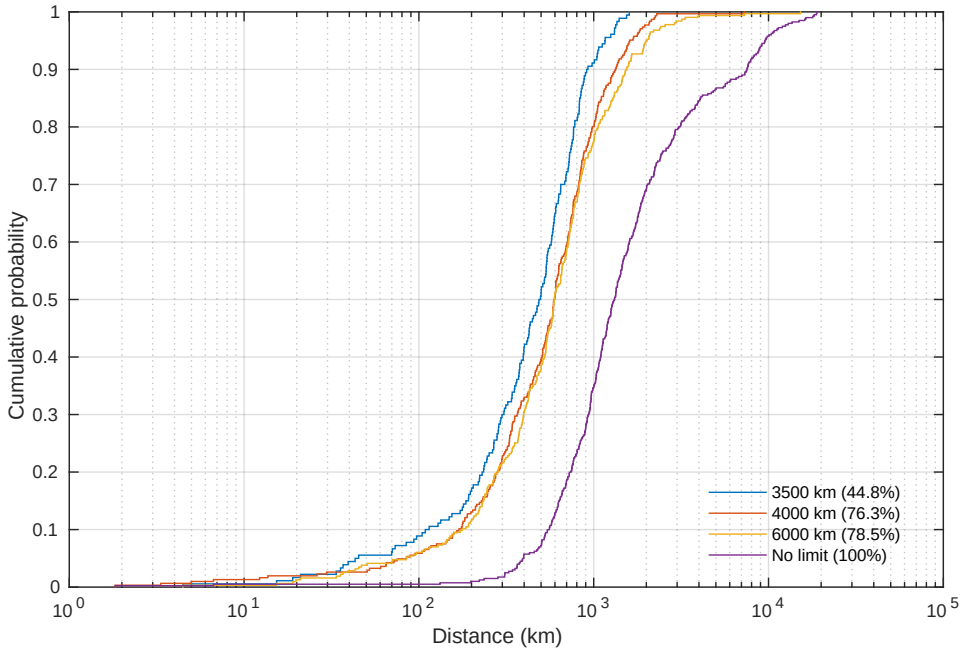


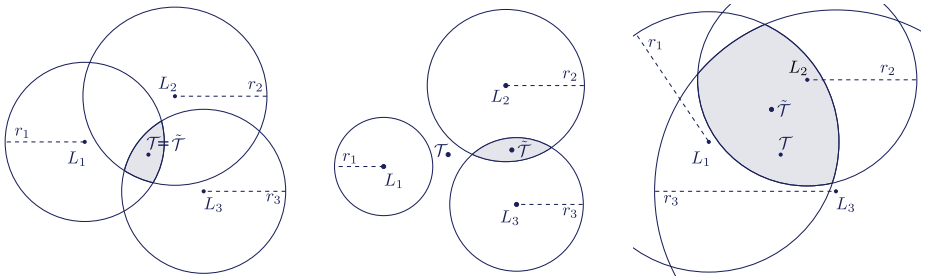
Figura 3.6: Andamento dell'errore al variare del raggio massimo ammissibile in scala logaritmica

6.000 km (99°percentile) e nessun limite, con indicata la percentuale di target localizzati con successo. In tutti i test il CF è arbitrariamente settato a $\frac{4}{9}$.

Nel caso senza limiti sul raggio l'algoritmo riesce a localizzare tutti i 401 target ma con errori che possono arrivare fino a quasi 18.000 km, circa la metà della circonferenza della terra. Risultati così negativi sono ad esempio dovuti a target in Cina raggiunti solo da landmark negli USA. In casi come questi è preferibile non avere un risultato piuttosto che averne uno completamente sbagliato.

3.3 Calibrazione del Conversion Factor

L'utilizzo di un CF appropriato è fondamentale per ottenere risultati accurati. Un CF eccessivamente sovrastimato genererà aree più vaste del necessario, rendendo la posizione troppo imprecisa; un CF sottostimato pro-



(a) Caso ottimale (b) Caso sottostimato (c) Caso sovrastimato

Figura 3.7: Effetti della variazione del CF nel calcolo delle intersezioni

durrà aree troppo piccole, con il rischio di non arrivare a coprire l'area del target, e in casi estremi ottenere un'intersezione nulla.

Con un CF troppo preciso si rischia di ricadere nel caso sottostimato, quindi il CF ottimale deve essere una maggiorazione del caso limite. In Figura 3.7 vediamo i tre casi illustrati. Si confronti con il caso di trilaterazione ideale in Figura 2.3, in cui i tre cerchi si intersecano in un unico punto.

Mentre in CBG viene calcolato il CF tramite la *bestline* di ogni landmark, su smartphone un approccio di questo tipo è impraticabile perché:

- i landmark durante l'esecuzione possono non essere i medesimi con cui è stata fatta la calibrazione (non tutti i terminali sono sempre raggiungibili)
- anche se riuscissimo a calcolare una bestline per ogni landmark, le calibrazioni sarebbero valide solo per un breve lasso temporale in quanto il dispositivo può cambiare posizione e rete di accesso nel tempo.

Per la calibrazione è stata perciò eseguita un'analisi statistica *offline* dei dati raccolti. Utilizzando il grafico a dispersione RTT–distanza (Figura 2.4) è stata calcolata la regressione lineare utilizzando il metodo dei minimi quadrati. Il CF risultante equivale al reciproco del termine di primo grado della retta $p(x) = p_1 x + p_0$:

$$CF = \frac{2}{p_1 \cdot c} = 0{,}337 \tag{3.1}$$

Network Type		Max Down. speed	Max Up. speed	Latency
2.5G	GPRS	114 Kbps	20 Kbps	600 ms
	EDGE	384 Kbps	60 Kbps	
3G	UMTS	384 Kbps	64 Kbps	150 ms
	CDMA - EvDo rev. A	3.1 Mbps	1.8 Mbps	
	HSPA, HSDPA, HDPAP	14.4 Mbps	5.76 Mbps	100 ms
	DC-HSDPA	28.8 Mbps	11.52 Mbps	
pre-4G	HSPA+	168 Mbps	22 Mbps	50 ms
	LTE	150+ Mbps	75 Mbps	10 ms
WiFi		300+ Mbps	300+ Mbps	

Tabella 3.1: Tipologie di reti wireless trattate

3.3.1 Reti di accesso

Ogni dispositivo invia assieme al RTT una serie di informazioni addizionali, tra cui l'indicazione del tipo di connessione attiva nel momento della misurazione. Utilizzando questa informazione sono stati separati i ping in quattro macro categorie secondo la Tabella 3.1, ed è stato calcolato il CF per ognuna.

Le categorie sono:

- *2.5G* – 2% dei burst, $CF = 0{,}479$
- *3G* – 17% dei burst, $CF = 0{,}387$
- *pre-4G* – 18% dei burst, $CF = 0{,}350$
- *WiFi* – 61% dei burst, $CF = 0{,}317$

A questi si affiancano 479 burst (2%) per il quale il tipo di rete non è pervenuto. In Figura 3.8 troviamo i grafici a dispersione per ognuna delle quattro categorie.

Il CF calibrato secondo Equazione 3.1 migliora sensibilmente le performance dell'algoritmo rispetto al valore arbitrario di $\frac{4}{9}$, mentre è quasi impercettibile la differenza tra questo e il CF calibrato per tipo di rete, come possiamo vedere in Figura 3.9. Bisogna però aggiungere che con il CF

calibrato sul tipo di rete i target localizzati con successo sono il 78%, contro il 73% del CF unico.

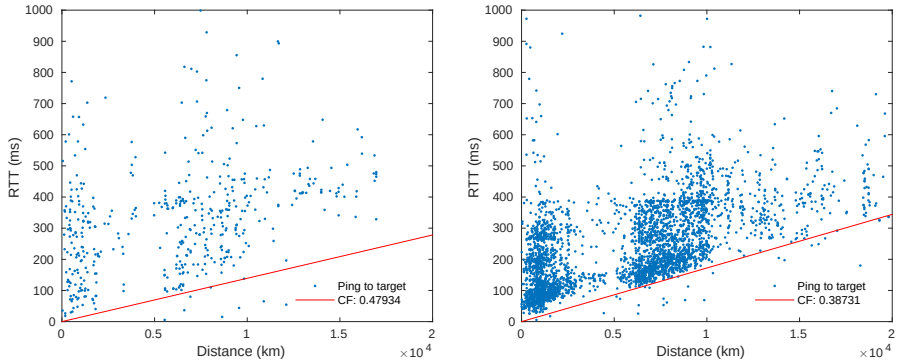
3.4 Integrazione dati di Traceroute

Come abbiamo visto in sezione 3.2 una prima selezione dei landmark viene effettuata controllando che il raggio del landmark rientri entro una determinata soglia. L'obiettivo è evitare percorsi del ping troppo lunghi, poiché potrebbero essere eccessivamente tortuosi e quindi portare ad una sovrastima della distanza, oppure ad una sottostima nel caso di lunghi collegamenti rettilinei.

Landmark con percorsi indiretti si possono evitare controllando il numero di hop attraversati. In Figura 3.10 possiamo vedere per ogni valore di hop massimo raggiunto il numero di occorrenze dei ping: tralasciando la colonna in 0, la maggior concentrazione si trova intorno a 16–17 hop. Gli oltre 9.000 campioni con numero di hop uguale a 0 sono dovuti ad una mancata corrispondenza tra i dati di traceroute e i dati di ping, dovuta forse a campagne di misurazione incomplete.

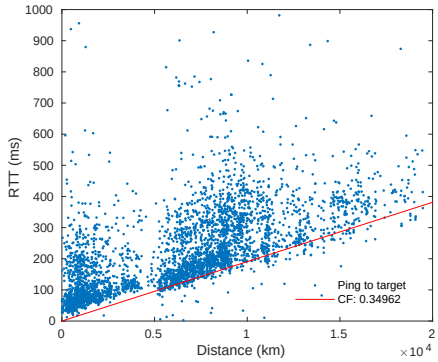
Intuitivamente ci aspetteremmo una relazione crescente tra il numero di hop massimo e la distanza percorsa; allo stesso modo la distanza percorsa dovrebbe crescere al crescere degli Autonomous System attraversati. In Figura 3.11 vediamo le distanze landmark–target raggruppate per numero di hop e per AS attraversati; per ogni gruppo di distanze è rappresentato il valore medio e il minimo.

Hop Analizziamo le relazioni con il numero di hop (Figura 3.11a e Figura 3.11c). Il numero di hop è il massimo TTL raggiunto dai campioni di traceroute. Nel grafico (a) è presente un massimo locale di 7'826 km per soli 3 hop, probabilmente dovuto a collegamenti di lunga percorrenza (tratti oceanici o satellitari). Passati i 9 hop osserviamo una relazione proporzionale tra hop e distanza media. La distanza minima per numero di hop si mantiene pressoché costante sotto i 200 km fino a 20 hop, e successivamente inizia a incrementare. Per 25 hop la distanza minima percorsa è di appe-

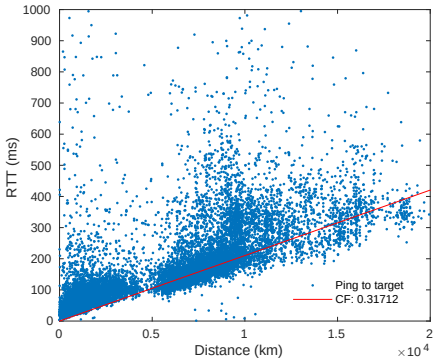


(a) 2.5G

(b) 3G



(c) pre-4G



(d) WiFi

Figura 3.8: Confronto tra CF per differenti tipi di rete

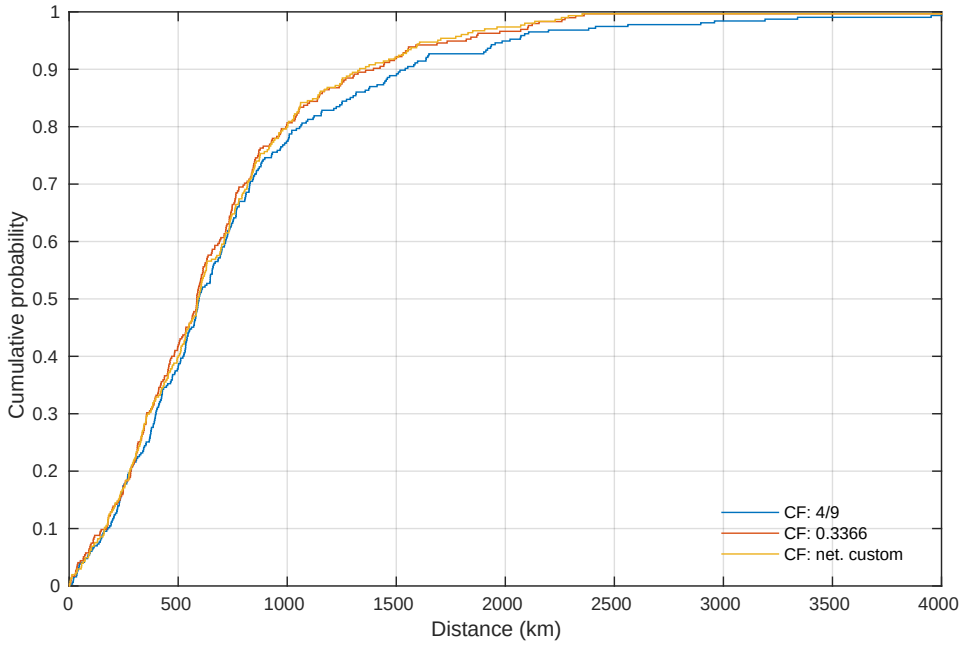


Figura 3.9: Andamento dell'errore al variare del CF

na 12 km, a riprova del fatto che i collegamenti tra host possono essere imprevedibilmente indiretti.

Analogo il rapporto tra RTT (media e minimo dei burst landmark–target) e gli hop in (c). Per il primo hop troviamo un massimo locale di 604.2 ms, mentre i RTT minimi rimangono costanti fino a 20 hop.

Autonomous System Analizziamo le relazioni con il numero di AS attraversati (Figura 3.11b e Figura 3.11d), calcolato contando le occorrenze uniche del numero di AS per ogni trace. L'andamento crescente della distanza media è evidente in (b), ma le distanze minime si mantengono basse fino a 9 AS per 177.7 km.

Il RTT medio si mantiene costante tra i 300 e i 500 ms, dimostrando che il livello di tortuosità del percorso non è direttamente collegato al tempo di propagazione.

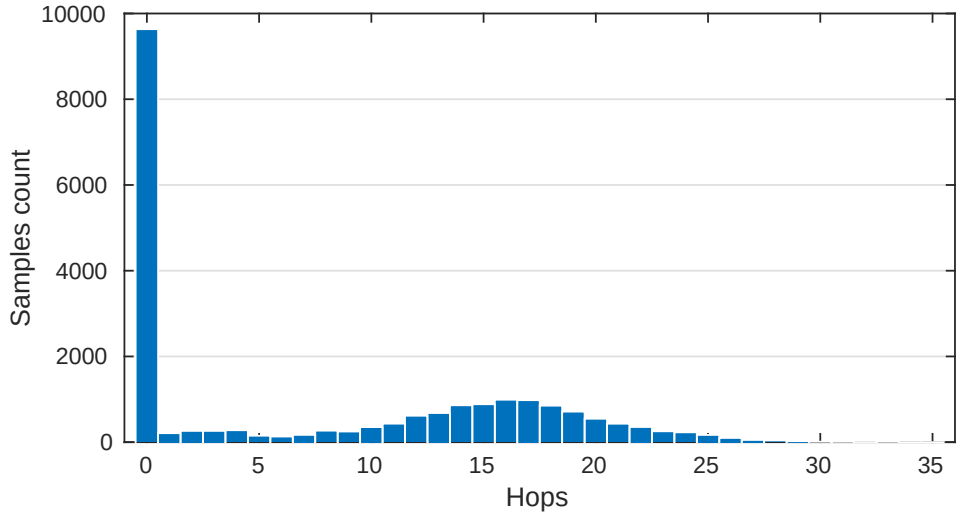


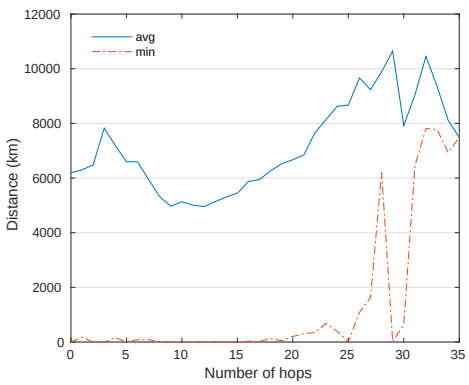
Figura 3.10: Numero di campioni per numero di hop massimo

3.4.1 Selezione dei landmark per hop e AS

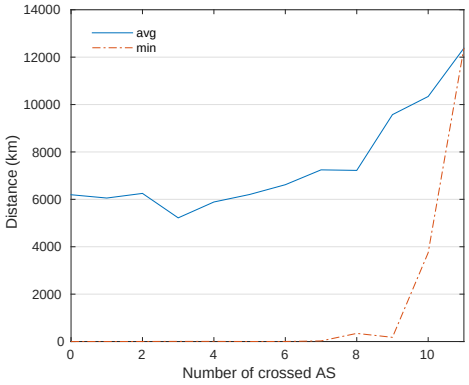
Per ogni target correttamente localizzato dall'algoritmo è stato misurato il numero di hop e di Autonomous System attraversati in media dai ping. Andando a calcolare la retta di regressione vediamo un aumento dell'errore al crescere di hop e Autonomous System (Figura 3.12), aumento che è maggiormente evidente nel caso degli hop (a). La relazione ci suggerisce un possibile filtraggio dei landmark in base al numero di hop o di Autonomous System attraversati; ad esempio scartando tutti i landmark che impiegano più di 10 hop o attraversano più di 3 Autonomous System per raggiungere il target.

L'algoritmo modificato tiene conto del numero di hop e di Autonomous System attraversati da ogni landmark: se un un landmark richiede più di N hop o AS per raggiungere un target viene rimosso in fase di intersezione. Sono state eseguite una serie di prove isolando i parametri di filtraggio per valutarne i miglioramenti confrontandoli con i risultati senza filtri.

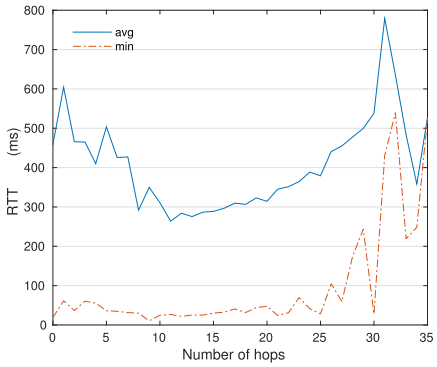
Per gli hop sono state eseguite tre prove con un limite di 5, 10 e 15 hop; il risultato è visibile in Figura 3.13. Il limite di 5 hop è risultato fin troppo restrittivo ed infatti le prestazioni sono decisamente inferiori; con il



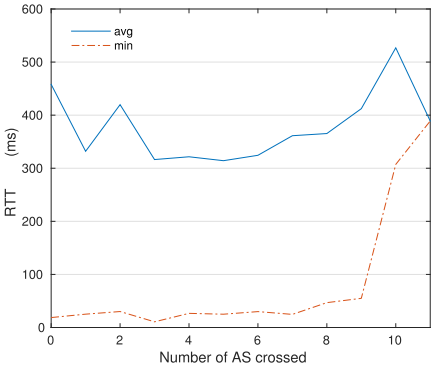
(a) distanze per hop massimi



(b) distanze per AS attraversati

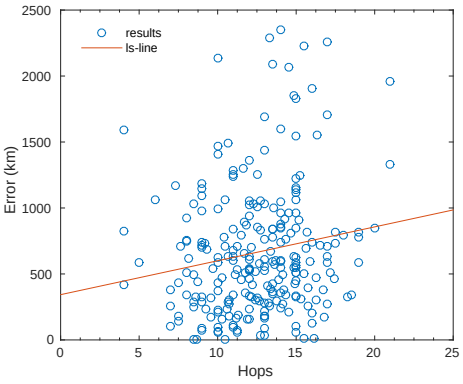


(c) RTT per hop massimi

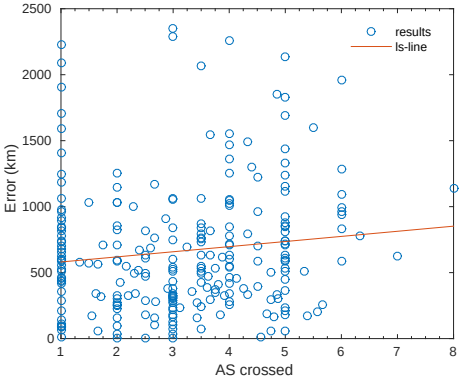


(d) RTT per AS attraversati

Figura 3.11: Distanze e RTT raggruppati per hop e AS attraversati



(a) Errore per hop



(b) Errore per AS

Figura 3.12: Errore nei risultati raggruppato per hop e AS attraversati

limite a 10 hop vediamo un andamento migliore per all'inizio e alla fine della distribuzione, ma il caso senza filtri rimane comunque il più performante.

Del tutto inesistente il miglioramento nel caso di filtraggio per Autonomous System. Nel caso di landmark all'interno dello stesso Autonomous System (limite a 1 AS) la ripartizione dell'errore rimane quasi sempre sotto il caso senza filtri, tranne entro i primi chilometri. La ripartizione con limite a 5 Autonomous System invece ricalca il caso senza filtri.

Purtroppo la scarsità dei dati di traceroute non permette di esprimere un giudizio certo sul beneficio o meno di questo filtraggio: meno del 50 % delle misure di `traceroute` disponibili sono risultate associabili a delle informazioni di `ping`, probabilmente perché eseguite a distanza di troppo tempo l'una dall'altra.

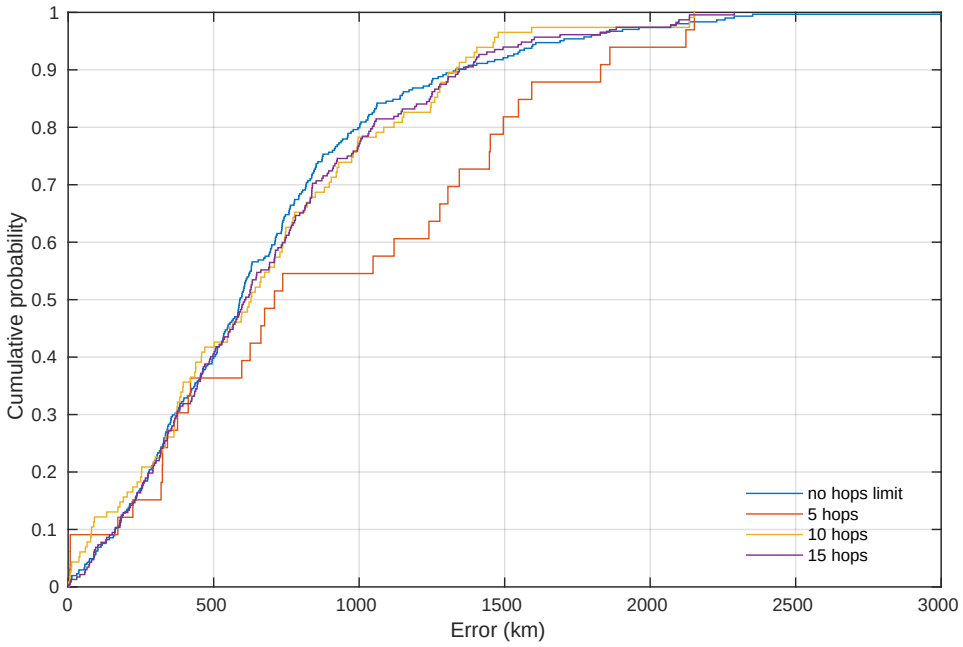


Figura 3.13: Distribuzione dell'errore nel caso di filtro per 5, 10 e 15 hop

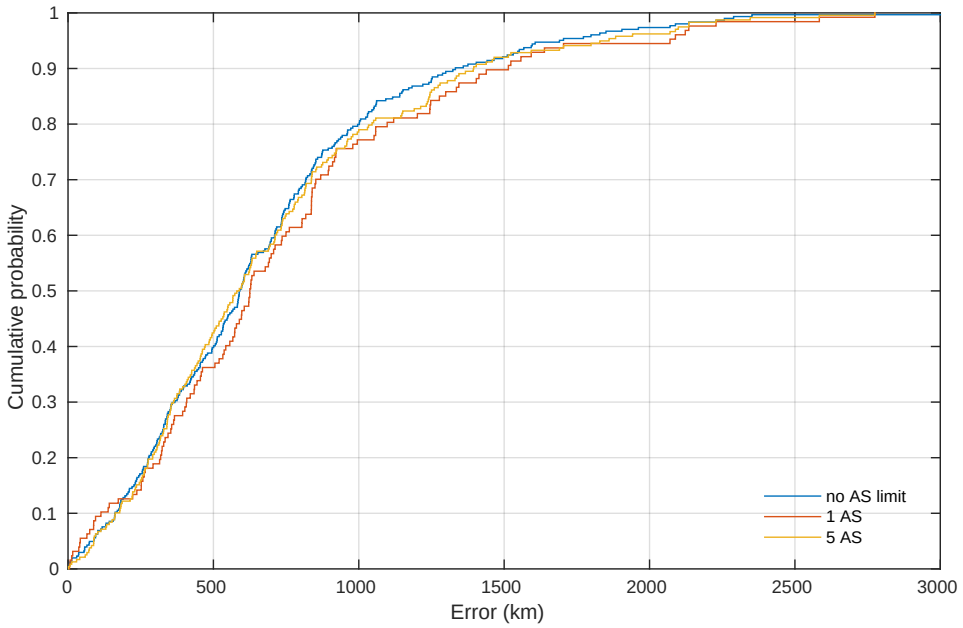
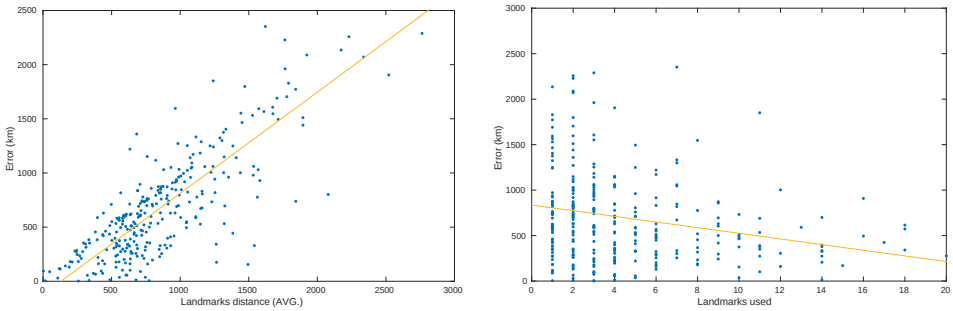


Figura 3.14: Distribuzione dell'errore nel caso di filtro per 1 e 5 AS



(a) Errore in relazione alla distanza media dei landmark utilizzati

(b) Errore in relazione al numero di landmark utilizzati

Figura 3.15: Andamento dell'errore di stima in relazione ai landmark utilizzati

3.5 Performance

Nel caso ottimale l'algoritmo riesce a localizzare 304 target con un errore medio di 697,23 km. L'errore aumenta significativamente con l'aumentare della distanza dei landmark dal target (Figura 3.15a), a conferma delle ipotesi fatte in sezione 3.2. Il numero di landmark usati influisce sull'errore in maniera inversamente proporzionale: maggiore il numero di landmark, minore l'errore medio (Figura 3.15b).

Capitolo 4

Implementazione

Nel Capitolo 2 abbiamo visto il funzionamento teorico dell'algoritmo e discusso i risultati ottenuti nel Capitolo 3. In questo capitolo verranno illustrati nel dettaglio i componenti che lo realizzano e gli strumenti utilizzati.

4.1 Panoramica dei componenti

L'applicazione è stata divisa in tre componenti, in modo da incapsulare la *business logic* in un componente comune, ed implementare l'interfaccia utente separatamente. I componenti sono:

- *Smartgeo Core* – modulo comune contenente le librerie che implementano l'algoritmo;
- *Smartgeo CLI – Command Line Interface* per avviare *Smartgeo Core* ed eseguire batch di operazioni, importazione e esportazione dati;
- *Smartgeo Visualizer – Graphical User Interface* web per interagire con *Smartgeo Core* e visualizzare i risultati su una mappa interattiva.

È stato scelto il linguaggio Ruby [23] per implementare tutti i componenti per la sua espressività e il ricco ecosistema di librerie. Avere un

linguaggio ad alto livello, facile da comprendere e modificare, è stato fondamentale per modellare l'algoritmo in assenza di specifiche senza dedicare troppo tempo al refactoring del codice.

I dati sono gestiti da un database relazionale PostgreSQL [24] dotato di estensione spaziale PostGIS [25]. PostGIS permette la gestione di dati geografici tramite query SQL native in maniera del tutto trasparente.

I calcoli geometrici sono effettuati dalla libreria `GEOS` [26], assieme alla libreria `Proj.4` [27] utilizzata per convertire le proiezioni cartografiche come vedremo più avanti. PostGIS, GEOS e Proj.4 sono software open-source supportati dalla *Open Source Geospatial Foundation* (OSGeo), organizzazione non-profit per lo sviluppo di tecnologie geospaziali.

4.2 Smartgeo Core

Il componente comune è costituito da una libreria Ruby (chiamata *gem* in gergo[1]) che implementa l'algoritmo. Qui troviamo la classe `Smartgeo::Base`, che contiene il codice esecutivo e implementa i metodi di interfaccia. L'accesso ai dati avviene mediante classi dedicate (dette *modelli*), secondo il pattern *active record* [29, p. 160]:

- `Smartgeo::Target` – contiene le informazioni dei target; ogni istanza possiede due relazioni *uno a molti* verso `Smartgeo::Ping` e `Smartgeo::Trace`;
- `Smartgeo::Ping` – informazioni e validazioni relative ai dati di ping;
- `Smartgeo::Trace` – informazioni e validazioni relative ai dati di traceroute.

A questi si aggiungono i modelli:

- `Smartgeo::Landmark` – rappresenta il dispositivo da cui hanno origine le misure; aggrega le informazioni di `Smartgeo::Ping` e `Smartgeo::Trace` verso uno o più `Smartgeo::Target`;

[1] RubyGems è un sistema di pacchettizzazione Ruby progettato per facilitare la creazione, la condivisione e l'installazione di librerie [28]

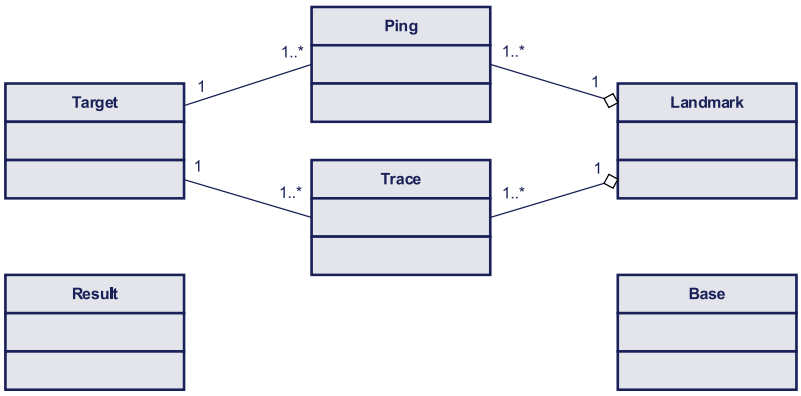


Figura 4.1: Diagramma UML dei modelli di `smartgeo`

- `Smartgeo::Result` – classe che incapsula le informazioni di output dell'algoritmo e fornisce dei metodi per la loro formattazione.

In Figura 4.1 è possibile vedere lo schema UML dei modelli descritti.

4.2.1 Informazioni di output

L'output del risultato è un'istanza della classe `Result` che contiene la posizione stimata, la posizione reale (se disponibile), l'errore di stima (distanza tra le due posizioni), e una serie di informazioni aggiuntive per l'analisi statistica (estensione dell'area di intersezione, numero di landmark utilizzati, continente di provenienza di landmark e target). La distanza tra punti è risolta con la formula dell'emisenoverso approssimando la terra con una sfera (con un errore massimo dello 0,5 % rispetto al geoide esatto).

4.3 Proiezioni cartografiche

Calcolare efficientemente intersezioni e baricentri di Equazione 2.3 e Equazione 2.4 su poligoni disposti sulla superficie tridimensionale della terra (*poligoni sferici*) non è banale. Il problema è già stato affrontato in modo più o meno approssimato dagli studi precedentemente visti: ad esempio CBG utilizza dei cerchi regolari tracciati su una proiezione di mercatore senza

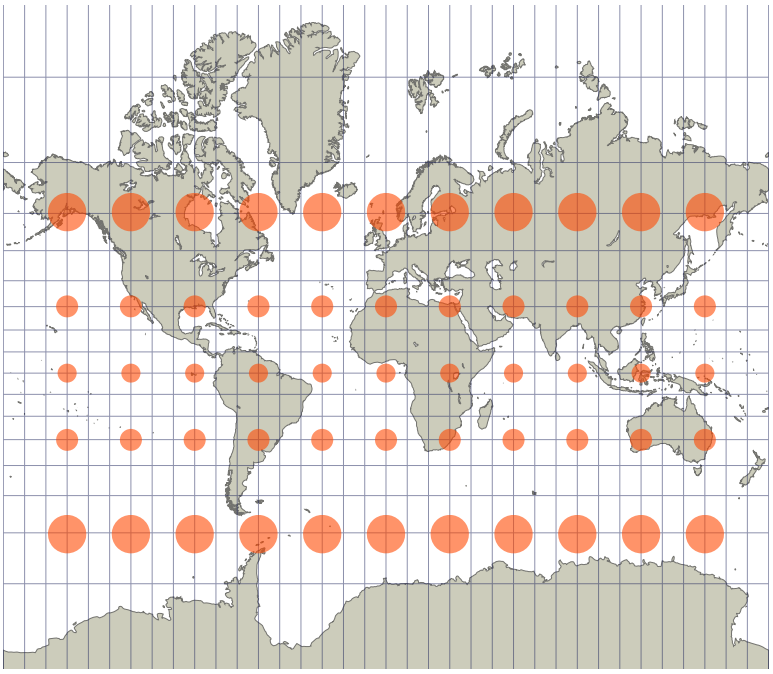


Figura 4.2: Proiezione di Mercatore del mondo con rappresentato l'indicatore di Tissot. Tutti i cerchi rossi hanno un raggio di 500 km

tener conto delle distorsioni [30] (distorsione che è evidente in Figura 4.2). Spotter invece utilizza un algoritmo di triangolamento ricorsivo della sfera terrestre per approssimare i calcoli su poligoni sferici [31, 32].

La tecnica comunemente utilizzata in ambito GIS [2] consiste nel proiettare il poligono su un sistema di coordiante bidimensionali, effettuare i calcoli semplificati di geometria piana, e quindi proiettare nuovamente il risultato in un sistema di coordinate sferiche.

[2]GIS o Geographic Information System, è una tipologia di software per la gestione delle informazioni geografiche

4.3.1 Proiezioni utilizzate

Esistono una moltitudine di sistemi di proiezione bidimensionale, con scopi e utilizzi differenti. In Figura 4.3 vediamo a confronto i sistemi di proiezione utilizzati in questo studio. In tutte le figure le forme sono cerchi esatti sul globo, concentrici e con raggi di 1.500, 3.000, 4.500, 6.000, 7.500 e 9.000 km. La proiezione di Mercatore (Figura 4.3a) mantiene le aree localmente ma presenta forti distorsioni su larga scala; le proiezioni azimutali equivalenti (Figura 4.3b) mantengono le distanze e le aree inalterate, ma provocano distorsioni rilevanti verso gli estremi della mappa. Centrando però la proiezione nell'area di interesse (proiezione *obliqua*, cioè non parallela all'equatore) i cerchi tornano ad essere privi di distorsioni (Figura 4.3c).

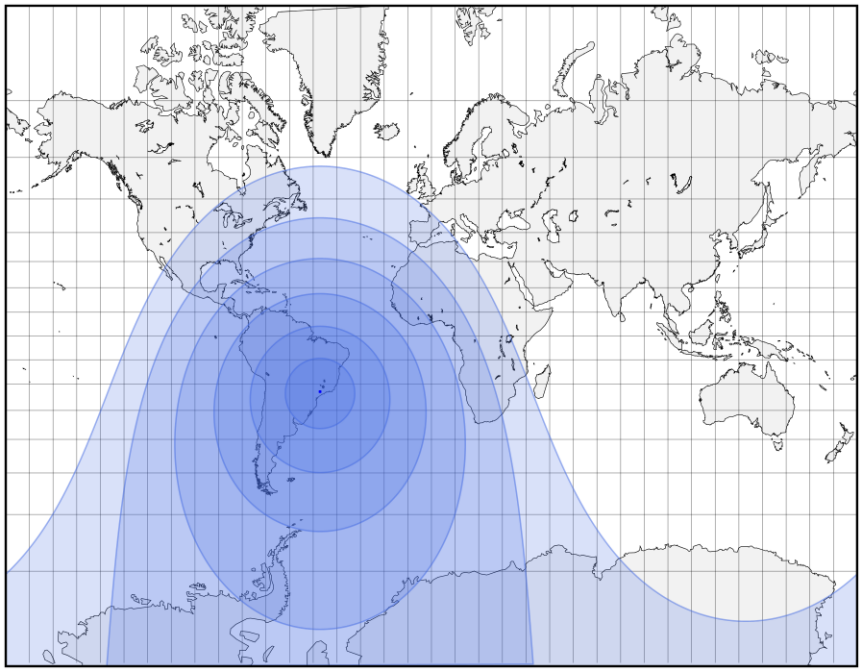
4.3.2 Intersezione dei cerchi

Le coordinate fornitemi sono rappresentate secondo lo standard WGS84 (anche conosciuto come EPSG:4326 [33]), sistema nato nel 1984 per indicare tramite latitudine e longitudine un punto univoco sulla superficie terrestre, che è rappresentata come uno sferoide schiacciato ai poli. L'area circolare centrata nel landmark viene approssimata con un poligono di 64 lati su questo sistema di coordinate.

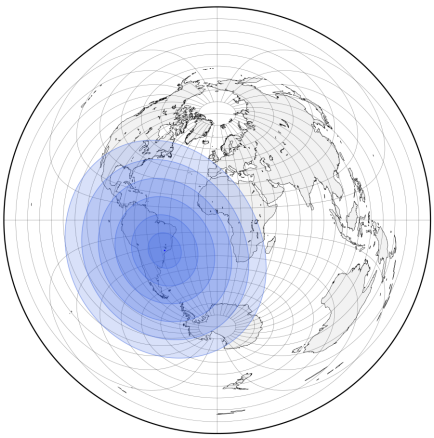
Per calcolare le intersezioni converto le coordinate dei vertici del poligono in coordinate bidimensionali (EPSG:3785, basato su una proiezione di Mercatore [34]). Questo tipo di proiezione preserva gli angoli uniformente su tutta la superficie (*proiezione conforme* o *isogonica*), mantenendo linee rette i segmenti dei poligoni che approssimano i cerchi. Questo permette di eseguire le operazioni di intersezione con precisione su tutta la superficie terrestre, ma non permette il calcolo del baricentro poiché l'area delle figure non è mantenuta.

4.3.3 Calcolo del baricentro

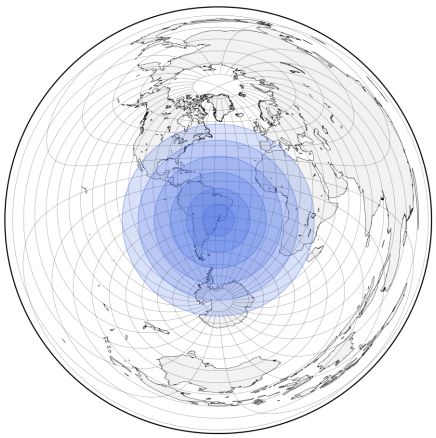
Il calcolo del baricentro dell'area di interesezione avviene in due passaggi: trovo inizialmente le coordinate di un baricentro provvisorio utilizzando la proiezione attuale (EPSG:3785); dopodichè riproietto la figura con una



(a) Proiezione di Mercatore equatoriale



(b) Proiezione azimutale equivalente equatoriale



(c) Proiezione azimutale equivalente obliqua

Figura 4.3: Confronto tra differenti sistemi di proiezione

```
$ smartgeo --help
Usage: smartgeo [options]
  -d, --database-config FILE       Database config in .yml format (default database.yml)
  -t, --target IP                  Target IP
  -c, --conversion-factor FLOAT    RTT/Distance conversion factor
  -b, --batch                      Batch mode: perform algorithmn on all targets found
  -o, --output FILE                Output file (default result.csv, only for batch mode)
  -r, --max-radius NUMBER          Max allowable radius
      --max-hops NUMBER            Max hops from landmark to target
      --max-as NUMBER              Max AS crossed from landmark to target
```

Figura 4.4: Schermata di aiuto per l'esecuzione di `smartgeo` da riga di comando

proiezione azimutale equivalente di Lambert (proiezione LAEA) centrata nel baricentro provvisorio. Questo consente di mantenere la forma localmente e quindi trovare il baricentro corretto, come si vede in Figura 4.3c.

L'Equazione 2.4 è implementata dalla libreria GEOS [35].

4.4 Smartgeo CLI

Smartgeo CLI mette a disposizione i metodi di Smarteo Core tramite tre script eseguibili da riga di comando:

- `dataimport` – effettua parsing, validazione e salvataggio su database dei dati letti da file CSV;
- `delaydistance` – esegue un'analisi statistica dei dati e calcola il CF;
- `smartgeo` – lancia l'algoritmo per singoli target o in batch. Per ogni esecuzione è possibile specificare i parametri quali CF, massimo numero di hop e massimo numero di AS attraversati (Figura 4.4).

4.5 Smartgeo Visualizer

In fase di studio dell'algoritmo è stato necessario poter visualizzare l'esecuzione di ogni passo dell'algoritmo su una mappa interattiva, per verificare

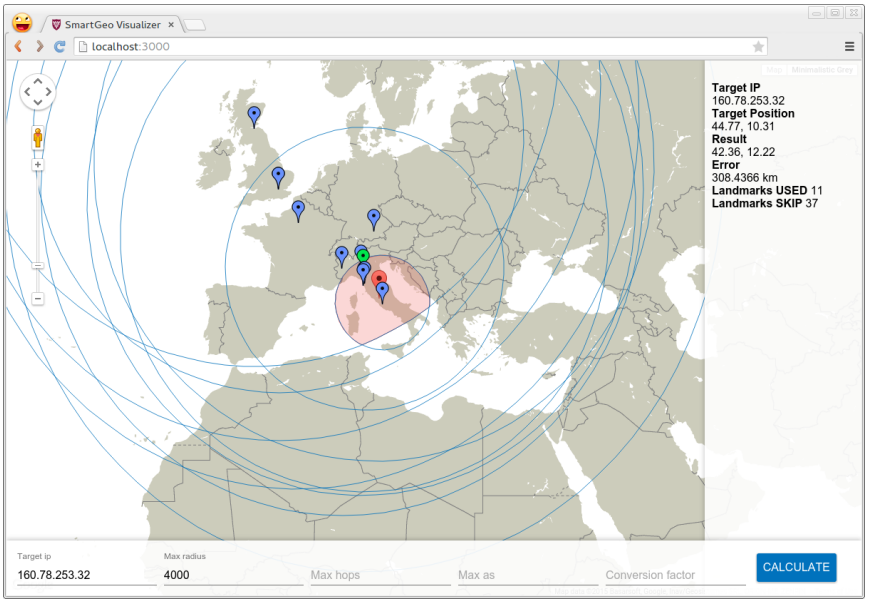


Figura 4.5: Una schermata di Smartgeo Visualizer

e correggere particolari situazioni di errore. Sono state utilizzate le API di Google Maps [36] per rappresentare i landmark e le relative regioni su una mappa della terra. L'interfaccia web permette di lanciare un'esecuzione dell'algoritmo impostando i parametri necessari (target IP, CF, filtro per hop e AS) e di interagire con i landmark per visualizzarne i parametri.

Smartgeo Visualizer è una web application costruita con il framework Ruby on Rails [37], pertanto utilizzabile da qualsiasi tipo di piattaforma dotata di un browser web.

Capitolo 5

Conclusioni

Determinare la posizione geografica di un host Internet è un tassello fondamentale per numerose applicazioni di rete. Ad oggi non esiste un protocollo standard che permetta la diffusione delle informazioni geografiche, dobbiamo quindi basarci sull'analisi di altri dati per cercare di inferire la posizione. Abbiamo visto come i metodi basati sulle misurazioni di rete siano i più performanti, ed abbiamo proposto un nuovo metodo per condurre le misurazioni utilizzando degli smartphone grazie al *crowdsourcing*, invece che tramite costosi sistemi dedicati.

I vantaggi principali di questa scelta sono:

- *facilità di aggiunta di nuovi landmark* – basta installare l'applicazione Portolan per utilizzare un dispositivo come sonda di misurazione;
- *diffusione globale in continua crescita* – il mercato degli smartphone è cresciuto del 28 % nell'ultimo anno [13].

Lo svantaggio è la qualità inferiore delle misurazioni rispetto a quelle effettuate da sistemi dedicati. Oltre alle difficoltà già incontrate dai precedenti metodi di geolocalizzazione (percorsi di rete indiretti, code variabili sui router, etc.), bisogna aggiungere gli errori introdotti dall'uso di dispositivi mobili:

- *connessioni wireless* – gli smartphone accedono ad Internet tramite reti wireless, che presentano una latenza maggiore e meno predicibile rispetto alle connessioni cablate;

- *limitazioni software* – i sistemi operativi mobili non offrono il controllo completo delle interfacce di rete per motivi di sicurezza [38], limitando l'implementazione degli strumenti di misurazione a `ping` e `traceroute`;

- *limitazioni hardware* – le performance dei dispositivi mobili sono tipicamente limitate per estendere la durata della batteria, influendo negativamente sulla precisione delle misure di RTT e della posizione GPS. Inoltre gli smartphone presentano hardware molto differenti tra di loro, e i modelli di fascia bassa spesso non hanno livelli di performance adeguati;

- *limitazioni di uso* – occorre preservare il consumo di traffico dati e di batteria del dispositivo per salvaguardare gli utenti, pertanto le campagne di misurazione devono utilizzare il numero minimo di dispositivi necessari e non possono essere eccessivamente intensive.

5.1 Sviluppi futuri

Il metodo proposto in questa tesi riesce a stimare la posizione dei target con semplici misurazioni (ping), e in tempi ragionevoli (2–3 secondi per target). Abbiamo visto una possibile strada per migliorare i risultati utilizzando le informazioni di `traceroute` inviateci dagli smartphone, ma occorrono prove su set di dati più estesi per poter confermare le ipotesi. Integrando anche altre misurazioni quali il *throughput* e la potenza del segnale wireless ricevuto si potrebbe migliorare l'algoritmo per assegnare dei pesi ai landmark in base alla loro affidabilità.

Nel set di dati analizzato l'errore medio risulta essere di 697,23 km, e nel 40 % dei risultati inferiore a 500 km. Come abbiamo visto la grandezza dell'errore dipende fortemente dal numero e dalla distanza dei landmark dal

target (soprattutto per effetto dei percorsi indiretti). I risultati ottenuti sono il frutto di un piccolo campione di utilizzatori (532 dispositivi), per migliorare occorre:

1. aumentare la capillarità e l'estensione della rete di dispositivi che utilizzano Portolan, cercando di coprire soprattutto le aree dove il mercato degli smartphone è in rapido sviluppo, come Asia e Sud America;

2. disponendo di un parco maggiore di landmark, utilizzare vincoli di selezione più stretti per eliminare le misurazioni meno accurate.

Con oltre 1.200.000 dispositivi venduti nel 2014, lo geolocalizzazione basata su smartphone potrebbe divenire un servizio efficiente ed affidabile.

Bibliografia

[1] "Google checks for suspicious account activity." `https://support.google.com/accounts/answer/3067630`.

[2] C. Davis, P. Vixie, T. Goowin, and I. Dickinson, "A means for expressing location information in the domain name system," tech. rep., Internet RFC 1876, January 1996.

[3] D. Moore, R. Periakaruppan, J. Donohoe, and k. claffy, "Where in the world is netgeo.caida.org?," in *International Networking Conference (INET) '00*, (Yokohama, Japan), The Internet Society, Jul 2000.

[4] S. Laki, P. Mátray, P. Hága, T. Sebők, I. Csabai, and G. Vattay, "Spotter: A model based active geolocation service," *INFOCOM, 2011 Proceedings IEEE*, pp. 3173–3181, April 2011.

[5] B. Gueye, S. Uhlig, and S. Fdida, "Investigating the imprecision of ip block-based geolocation.," in *PAM* (S. Uhlig, K. Papagiannaki, and O. Bonaventure, eds.), vol. 4427 of *Lecture Notes in Computer Science*, pp. 237–240, Springer, 2007.

[6] "Geourl." `http://www.geourl.org/`.

[7] MaxMind Inc, "Geoip2." `https://www.maxmind.com/en/geoip2-services-and-databases`.

[8] Geobytes, "Geonetmap." `http://www.geobytes.com/GeoNetMap/`.

[9] V. Padmanabhan and L. Subramanian, "An investigation of geographic mapping techniques for internet hosts," *Proceedings of ACN SIGCOMM*, pp. 173–185, August 2001.

[10] B. Gueye, A. Ziviani, M. Crovella, and S. Fdida, "Constraint based geolocation of internet hosts," *IEEE/ACM Transactions on Networking*, 2004.

[11] E. Katz-Bassett, J. P. John, A. Krishnamurthy, T. Anderson, and Y. Chawathe, "Towards ip geolocation using delay and topology measurements," *Proceedings of the 6th ACM SIGCOMM conference on Internet measurement*, pp. 71–84, 2006.

[12] B. Wong, I. Stoyanov, and E. G. Sirer, "Octant: A comprehensive framework for the geolocalization of internet hosts," *NSDI 2007 Symposium and Cambridge and Massachusetts*, pp. 71–84, April 2007.

[13] "Gartner says smartphone sales surpassed one billion units in 2014." `http://www.gartner.com/newsroom/id/2996817`.

[14] "Itu statistics: Global ict developments, 2001-2014." `http://www.itu.int/ict/statistics`.

[15] "Portolan: Mapping the internet." `http://portolan.iet.unipi.it/`.

[16] A. Faggiani, E. Gregori, L. Lenzini, S. Mainardi, and A. Vecchio, "On the feasibility of measuring the internet through smartphone-based crowdsourcing," in *WiOpt*, pp. 318–323, IEEE, 2012.

[17] E. Gregori, L. Lenzini, V. Luconi, and A. Vecchio, "Sensing the Internet through crowdsourcing," in *Proceedings of the Second IEEE PerCom Workshop on the Impact of Human Mobility in Pervasive Systems and Applications (PerMoby)*, May 2013.

[18] B. Augustin, X. Cuvellier, B. Orgogozo, F. Viger, T. Friedman, M. Latapy, C. Magnien, and R. Teixeira, "Avoiding traceroute anomalies with paris traceroute," in *Proceedings of the 6th ACM SIGCOMM Con-*

ference on Internet Measurement, IMC '06, (New York, NY, USA), pp. 153–158, ACM, 2006.

[19] F. S. Solheim, J. Vivekanandan, R. H. Ware, and C. Rocken, "Propagation delays induced in gps signals by dry air, water vapor, hydrometeors, and other particulates," *Journal Of Geophysical Research*, vol. 104, pp. 9663–9670, April 27 1999.

[20] S. Kedar, G. A. Hajj, B. D. Wilson, and M. B. Heflin, "The effect of the second order gps ionospheric correction on receiver positions," *Geophysical Research Letters*, vol. 30, no. 16, pp. n/a–n/a, 2003. 1829.

[21] Y. He, M. Faloutsos, S. Krishnamurthy, and B. Huffaker, "On routing asymmetry in the internet," *Global Telecommunications Conference, GLOBECOM '05 IEEE*, 28 Nov.-2 Dec. 2005.

[22] S. Savage, A. Collins, E. Hoffman, J. Snell, and T. Anderson, "The end-to-end effects of internet path selection," *SIGCOMM Comput. Commun. Rev.*, vol. 29, pp. 289–299, Aug. 1999.

[23] "Ruby language." `https://www.ruby-lang.org/`.

[24] "Postgresql." `http://www.postgresql.org/`.

[25] "Postgis." `http://postgis.net/`.

[26] "Geos." `http://trac.osgeo.org/geos/`.

[27] "Proj.4." `https://trac.osgeo.org/proj/`.

[28] "Rubygems." `https://rubygems.org/`.

[29] M. Fowler, *Patterns of enterprise application architecture*. Addison-Wesley, 2003. ISBN 978-0-321-12742-6.

[30] "Homepage of Mark E. Crovella, author of CBG." `http://www.cs.bu.edu/~crovella/links.html`.

[31] A. Szalay, J. Gray, G. Fekete, P. Kunszt, P. Kukol, and A. Thakar, "Indexing the sphere with the hierarchical triangular mesh," tech. rep., MSR-TR-2005-123, Microsoft Research, 2005.

[32] Sloan Digital Sky Survey, "Hierarchical triangular mesh library." `http://skyserver.org/HTM/`.

[33] "Epsg:4326 reference." `http://spatialreference.org/ref/epsg/wgs-84/`.

[34] "Epsg:3785 reference." `http://spatialreference.org/ref/epsg/popular-visualisation-crs-mercator/`.

[35] "Geos centroid algorihmn." `http://geos.osgeo.org/doxygen/classgeos_1_1algorithm_1_1Centroid.html`.

[36] "Google maps javascript api v3." `https://developers.google.com/maps/documentation/javascript/maptypes`.

[37] "Ruby on rails web development framework." `http://rubyonrails.org/`.

[38] Android issue report n. 4039, "Allow binding privileged ports or creating raw sockets." `https://code.google.com/p/android/issues/detail?id=4039`.