Universidad Autónoma de Madrid
Escuela politécnica superior

Proyecto fin de carrera

CARACTERIZACIÓN Y CUANTIFICACIÓN DE LOS EFECTOS DEL PATH INFLATION EN UNA RED COMERCIAL EN PRODUCCIÓN

Ingeniería de Telecomunicación

Roberto González Rey
OCTUBRE 2011

CARACTERIZACIÓN Y CUANTIFICACIÓN DE LOS EFECTOS DEL PATH INFLATION EN UNA RED COMERCIAL EN PRODUCCIÓN

AUTOR: Roberto González Rey
TUTOR: Felipe Mata Marcos

High Performance Computing and Networking group
Dpto. de Ingeniería de Telecomunicación
Escuela Politécnica Superior
Universidad Autónoma de Madrid

OCTUBRE 2011

Resumen

Resumen

La distancia de encaminamiento entre dos emplazamientos físicos en una red de comunicaciones es, por lo general, superior a la distancia geográfica entre ambos lugares. Este hecho de que las rutas extremo a extremo sean superiores a lo necesario es conocido como "Path Inflation". Cuando la diferencia entre estas dos distancias es significativa, repercute directamente en el rendimiento de las comunicaciones, produciéndose retardos mayores de los necesarios que impiden que se alcancen los límites deseados de velocidad de transmisión.

El presente proyecto de fin de carrera tiene como objetivo caracterizar y cuantificar el Path Inflation de una red comercial en producción, como es la red académica española RedIRIS, y sus principales destinos de tráfico internacionales. RedIRIS pone a disposición de las instituciones afiliadas una red de comunicaciones de alta capacidad, que permite a la comunidad académica y científica española el acceso a Internet, facilitando la participación en proyectos de investigación nacionales e internacionales.

Para caracterizar y cuantificar el Path Inflation, se han realizado mediciones de las rutas de encaminamiento entre Madrid, nodo central de RedIRIS, y localizaciones significativas internacionales con la herramienta `traceroute`. Las medidas obtenidas se han comparado con las distancias geográficas entre dichos puntos, permitiendo así realizar una caracterización del Path Inflation entre RedIRIS y dichas localizaciones internacionales.

Posteriormente, se ha cuantificado el impacto real que tiene la medida de Path Inflation obtenida. Para ello se ha utilizado una caracterización de los destinos más populares a nivel de país desde RedIRIS, obteniendo una medida de Path Inflation ponderada, denominada Path Inflation Traffic, que tiene en cuenta no sólo la diferencia entre las distancias geográfica y de encaminamiento, sino también cuanto tráfico es transmitido entre ambos extremos.

El objetivo principal ha sido determinar si los destinos más populares están bien comunicados con la RedIRIS, y por tanto el impacto del Path Inflation no es determinante en la experiencia de los usuarios. Se incluye dicha clasificación de los países más críticos en cuanto a Path Inflation Traffic a nivel continental y nivel mundial.

La conclusión más importante de este estudio es que RedIRIS no está bien conectada con los países europeos que mayor tráfico reciben (Alemania, Reino Unido, Francia, Italia, Portugal y Suiza). Respecto a los demás países, la longitud de las distancias de enrutamiento no es alarmante teniendo en cuenta el tráfico que reciben.

Como objetivo secundario se estudian los retardos medios y como el Path Inflation influye directamente en estas latencias. La métrica para el cálculo de retardos usada es el Round-Trip Time, tiempo de ida y vuelta de un paquete entre dos extremos en una comunicación, y el análisis indica que los países más populares de la RedIRIS (los que más tráfico reciben) comprenden un Round-Trip Time normalmente aceptable.

Palabras Clave

Distancia de Encaminamiento, Distancia Geográfica, Path Inflation, Path Inflation Traffic, RedIRIS, Retardos, Round-Trip Time, Tráfico de Red

Abstract

The routing distance between two physical emplacements in a communications network is, generally, larger than the geographical distance between such locations. This fact of end-to-end routes larger than necessary is referred in the literature as "Path Inflation". When the difference between these two distances is significant, reverberates directly in the performance of the communications, producing delays larger than necessary which prevent reaching expected limits of transmission speed.

The present study aims to characterize and quantify the Path Inflation in a commercial network under production, such as the Spanish academy network RedIRIS, and its main international traffic destinations. RedIRIS offers a high-capacity communications network, enabling the spanish academic and scientific community Internet access and fostering the participation in national and international research projects.

To characterize and quantify the Path Inflation, we have made measurements of the routing distances between Madrid and RedIRIS significant international locations by means of the `traceroute` tool. The measurements obtained were compared with geographic distances between such locations, thus allowing a characterization of the Path Inflation within RedIRIS.

Subsequently, we quantified the real impact of the Path Inflation so far obtained. To do this we used a characterization of the most popular foreign destinations from RedIRIS, obtaining a weighted measure of Path Inflation, namely Path Inflation Traffic, which takes into account not only the difference between geographic and routing distance, but also how much traffic is transmitted between both ends.

The main objective was to determine whether the most popular destinations are well connected to RedIRIS, and therefore the impact of the Path Inflation is not decisive in the user experience. We include such critical countries classification in Path Inflation Traffic at continental and worldwide level.

The most important finding of this study is that RedIRIS is not well connected with the European countries that receive most traffic (Germany, United Kingdom, France, Italy, Portugal and Switzerland). In the other countries, the length of the routing distance is not alarming taking into account the traffic they receive.

As a secondary objective we studied the delays and how the Path Inflation directly influences in these latencies. The metric for the delays calculation is Round-Trip Time, time of return of a packet between two ends in a communication, and the analysis indicates that the countries most popular in RedIRIS (which receive more traffic) typically comprise an acceptable Round-Trip Time.

Key words

Geographical Distance, Latencies, Network Traffic, Path Inflation, Path Inflation Traffic, Routed Distance, RedIRIS, Round-Trip Time

Agradecimientos

Gracias a todos aquellos que me han acompañado en esta etapa universitaria, tanto dentro como fuera de las aulas.

Gracias a mi familia, padres y hermano, especialmente a este último, por enseñarme que siempre se puede mejorar.

Gracias a mis amigos de toda la vida, y a los fantásticos ingenieros de la broma (Pablo, Javi, Chema, José Rubén, De Sevilla, Santi y Manzano) por su inestimable compañía.

Muchas gracias a Felipe Mata, tutor de este proyecto, por cederme parte de su conocimiento y valioso tiempo, por el cuál he llegado a conocer bastante más las redes de comunicaciones.

¡Gracias a todos!

Índice general

Índice de figuras

Índice de cuadros

Glosario de acrónimos

- **AS**: Anonimous System
- **BGP**: Border Gateway Protocol
- **BBDD**: Bases de Datos
- **CBG**: Constraint Based Geolocation
- **CDF**: Cumulative Distribution Function
- **GeoIP**: IP Geolocation
- **GPS**: Global Position System
- **HTTP**: Hypertext Transfer Protocol
- **ISP**: Internet Service Provider
- **IP**: Internet Protocol
- **ICMP**: Internet Control Message Protocol
- **MPLS**: Multiprotocol Label Switching
- **NAP**: Network Access Point
- **PI**: Path Inflation
- **PIC**: Path Inflation Control
- **PIT**: Path Inflation Traffic
- **PoP**: Point of Presence
- **RTT**: Round Trip Time
- **TCP**: Transmission Control Protocol
- **UDP**: User Datagram Protocol
- **WDM**: Wavelength-Division Multiplexing
- **WAN**: Wide Area Network
- **WGS84**: World Geodetic System 84

1 Introducción

1.1. Motivación del proyecto

El "Path Inflation (PI)" en una red de comunicaciones es el hecho de que la longitud de los caminos enrutados sea superior a lo necesario e influye severamente en el rendimiento de la red, ya que las latencias de los paquetes son directamente proporcionales a la distancia de enrutamiento. Es por esto que existe un interés en la identificación del PI con el objetivo de reducir sus efectos en las redes de comunicaciones.

La comunidad científica es consciente de la existencia del PI (GW02; TGSE01), y ha realizado numerosos estudios para identificar sus causas (SMA03; GW02; TGSE01) y consecuencias (PZMH07; ZLPG05). Sin embargo, estos estudios han obviado en sus análisis las demandas de tráfico entre los destinos analizados (FGL+01; SPK01). Al igual que sucede con el tráfico aéreo, hay demarcaciones geográficas que están mejor conectadas entre sí (conocidas como destinos populares), motivado este hecho por el alto número de usuarios que demandan esas rutas. De esta forma, aparece un compromiso entre la calidad de conexión entre dos destinos y la cantidad de tráfico que existe entre ellos.

El presente proyecto de fin de carrera tratará de cuantificar el PI en conexiones internacionales tomando como origen del tráfico la red académica española RedIRIS, en concreto desde Madrid. Partiendo de esta información, se estudiará la calidad de conexión de RedIRIS con sus principales destinos internacionales, dando una caracterización de la severidad de los efectos que pueda tener el PI sobre las comunicaciones con dichos destinos.

Para cuantificar los efectos del PI utilizaremos una métrica que comprenda la distancia enrutada, la distancia geográfica y la demanda de tráfico para cada destino, realizando una clasificación de los efectos de este fenómeno teniendo en cuenta la popularidad del destino.

1.2. Objetivos y enfoque

El presente proyecto de fin de carrera tiene como objetivo caracterizar y cuantificar el PI de una red comercial en producción como es RedIRIS. La topología de RedIRIS se observa en la Figura 1.1.

Para ello, se harán mediciones de las distancias de encaminamiento entre RedIRIS, en concreto desde Madrid, y localizaciones significativas internacionales con la ayuda de la herramienta `traceroute` (GALM08). Las medidas obtenidas se compararán con las distancias geográficas

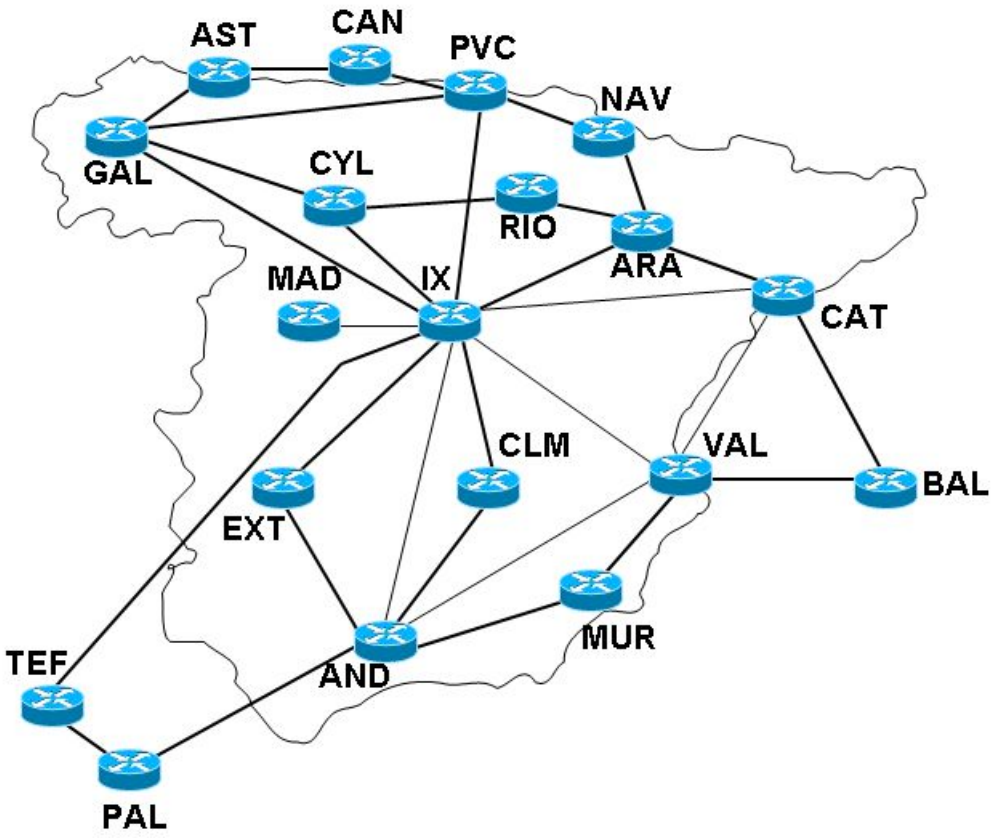


Figura 1.1: Mapa de la topología de RedIRIS. Se observa que el nodo de Madrid es cercano al nodo central, por lo que será ideal como origen del tráfico.
`Fuente: www.redIRIS.es`

entre dichos puntos, permitiendo así realizar una caracterización del PI. La elección de Madrid como origen de las trazas viene determinado por su localización central en RedIRIS (ver Figura 1.1).

Finalmente, se hará una cuantificación del impacto real que tiene la medida de PI obtenida. Para ello se utilizará una caracterización de los destinos más populares a nivel de país (MGDALdVon) desde RedIRIS, de forma que se obtenga una medida de PI ponderada, es decir, una medida de PI que tenga en cuenta no sólo la diferencia entre las distancias geográfica y de encaminamiento, sino también la cantidad de tráfico que es transmitido entre ambos emplazamientos. El objetivo final será determinar si los destinos más populares están bien comunicados entre sí y por tanto el impacto del PI no es determinante.

La consecución de dichos objetivos conllevará por tanto la realización de los siguientes puntos:

1. Estudio del estado del arte en análisis y caracterizaciones del PI.
2. Estudio del estado del arte de efectos adversos conocidos del PI.
3. Estudio del estado del arte en herramientas de geolocalización de direcciones IP, seleccionando la más adecuada para la realización de los objetivos del presente proyecto.
4. Implementación de un software sonda que sea capaz de identificar los diferentes nodos intermedios que atraviesan los paquetes en una comunicación entre dos extremos, calculando además la distancia de encaminamiento.
5. Identificación de emplazamientos representativos dentro de cada país, que formarán parte del conjunto de destinos analizados en la evaluación del PI.

6. Propuesta de diferentes métricas para estudiar el compromiso entre calidad de conexión y demandas de ancho de banda entre los destinos más populares. Selección justificada de la métrica que mejor represente este compromiso.

1.3. Organización de la memoria

La memoria consta de los siguientes capítulos:

Capítulo 1: Introducción. Este primer capítulo presenta el proyecto fin de carrera, los objetivos a conseguir y el enfoque para alcanzarlos.

Capítulo 2: Revisión del estado del arte. En este capítulo se estudia el estado del arte sobre PI, analizando sus causas y consecuencias. Se comentan algunas soluciones propuestas en la literatura que disminuyen este efecto y la motivación para estudiar el PI en función del tráfico. También se estudia el estado del arte sobre geolocalización IP, técnica necesaria para el desarrollo de este PFC. Se analizan las técnicas más populares de geolocalización IP para elegir aquella que será utilizada en nuestro estudio.

Capítulo 3: Diseño y desarrollo. Este capítulo presenta la metodología empleada para el estudio del PI en función del tráfico. La metodología contiene las siguientes actividades:

- Identificación de los nodos intermedios que atraviesan los paquetes de comunicación. Conocer los nodos intermedios de un ruta a nivel IP permite, mediante técnicas de geolocalización IP, localizarlos geográficamente.
- Identificación de emplazamientos representativos dentro de cada país. El estudio es sobre RedIRIS, por lo que se obtendrán los destinos de cada país identificando las direcciones IPs que proporcionan los exportadores de tráfico en un período determinado.
- Cálculo de la distancia de encaminamiento entre dos extremos. Para esta tarea se utilizan algoritmos que calculan la distancia geográfica de dos emplazamientos a partir de sus coordenadas geodésicas (latitud y longitud). Para cada tramo de una ruta, es decir, camino entre cada nodo intermedio, se calculará la distancia geográfica. La distancia de encaminamiento entre dos extremos será la suma de las distancias geográficas de todos los tramos.
- Propuesta de diferentes métricas para estudiar el compromiso entre calidad de conexión y demandas de ancho de banda entre los destinos más populares.

Capítulo 4: Integración, pruebas y resultados. En este capítulo se calculan las métricas propuestas en el capítulo anterior. Finalmente se identifican aquellos países cuya métrica es crítica y se elabora una clasificación a nivel continental y a nivel mundial de la severidad del PI.

Capítulo 5: Análisis de retardos y saltos. Existe una relación de proporcionalidad entre el PI y el tiempo necesario para completar un camino en Internet. En este capítulo se identifica esta relación y se estudian los retardos en los países más populares de RedIRIS. También se calcula el número de saltos medio para alcanzar estos países.

Capítulo 6: Conclusiones y trabajo futuro. En este capítulo final se comentan las conclusiones obtenidas durante el desarrollo del PFC y el trabajo futuro como continuación de este PFC.

2

Revisión del estado del arte

2.1. Revisión del estado del arte sobre Path Inflation

2.1.1. Introducción

La distancia de encaminamiento entre dos emplazamientos físicos en una red de comunicaciones comúnmente es superior a la distancia geográfica entre ambos lugares. Este efecto se conoce como "Path Inflation (PI)" y repercute directamente en el rendimiento de las comunicaciones, produciéndose retardos mayores de los necesarios que impiden que se alcancen los límites deseados de velocidad de transmisión.

El estado del arte sobre PI analiza las causas y consecuencias de este efecto, así como el desarrollo de mecanismos para minimizarlo. Sin embargo, estos estudios han obviado en su análisis las demandas de tráfico entre los destinos analizados.

El presente proyecto de fin de carrera tratará de cuantificar el PI en las conexiones internacionales, tomando como origen del tráfico el nodo de Madrid que pertenece a la red académica española RedIRIS . Partiendo de esta información, se estudiará la calidad de conexión de RedIRIS con sus principales destinos internacionales, dando una caracterización de la severidad de los efectos que pueda tener el PI sobre las comunicaciones con dichos destinos.

2.1.2. Causas y cuantificación

La comunidad científica ha realizado numerosos estudios para identificar las causas del PI (SMA03; GW02; TGSE01). Las principales causas de este efecto son la topología de la red (SMA03) y sus cambios (PZMH07), fluctuaciones de tráfico (GALM08; FGL+01; ZZX01), los tamaños de los ISPs (SMA03; GW02; BGW05) y las políticas de enrutado de los ISPs (SMA03; GW02; TGSE01) en tres niveles — el formado por el conjunto de circuitos pertenecientes a un ISP, el que comprende los enlaces de los ISPs entre destinos vecinos, y un nivel superior con la secuencia de ISPs necesarios para alcanzar destinos lejanos.

Se ha observado que el PI varía de un ISP a otro. Es posible clasificar a los ISPs en función de su tamaño (ver Figura 2.1):

1. Tier-1: ISPs grandes que se conectan entre ellos y con los puntos de acceso público (NAP). El tamaño de estos ISPs produce normalmente el mayor Path Inflation (BGW05).

2. Tier-2: ISPs más pequeños (a menudo regionales). Se conectan a 1 o más Tier-1, y posiblemente a otros Tier-2.

3. Tier-3: ISPs locales conectados a los sistemas terminales.

Diversos análisis aproximan el PI medio a un 11 % para Tier-1, un 22 % para Tier-2 y un 33 % para Tier-3 (GW02; BGW05). Estas cuantificaciones son el porcentaje en el que las rutas superan la distancia geográfica entre dos extremos para los diferentes tipos de ISPs según su tamaño.

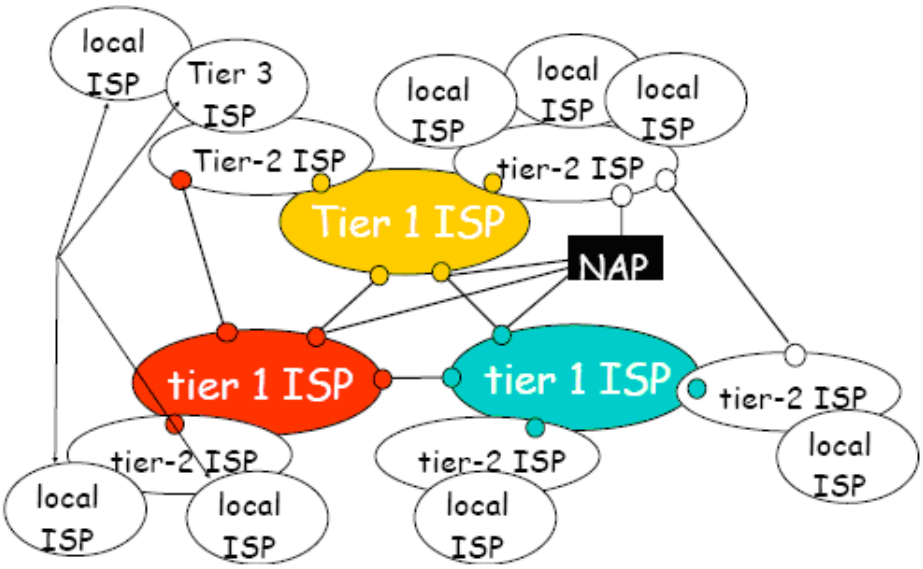


Figura 2.1: Tipos y tamaños de ISPs. Se observa el tamaño relativo y la conectividad entre ellos. `Fuente: http://profesores.elo.utfsm.cl/ agv/elo322/1s05/`

Las políticas de enrutado ("peering policies") no están estandarizadas. Su caracterización está basada en las tendencias que siguen los caminos en Internet, atendiendo a tres comportamientos distinguidos:

1. Estrategia "early-exit" o "hot-potato": esta política utiliza los enlaces entre dos ISPs más cercanos al origen del circuito. Esta estrategia es normalmente la que mayor PI produce.

2. Estrategia "late-exit" o "cold-potato": esta política utiliza los enlaces entre dos ISPs más lejanos al origen del circuito.

3. Estrategia "load-balancing": existe un compromiso entre el circuito y el tráfico.

El uso predominante de la política "early-exit" provoca encontrar mayor PI en un camino que atraviesa varios ISPs que en otro bastante más largo que se mantiene dentro de un ISP (BGW05).

Los niveles que atraviesa un camino enrutado influyen de manera diferente en el aumento de la longitud (ver Figura 2.2). En el primer nivel (formado por el conjunto de caminos pertenecientes a un ISP) se produce PI al no existir conexión directa entre cada par de destinos. Esto es porque normalmente sólo existe conexión directa en localizaciones geográficas con alta densidad de usuarios. Para alcanzar otras localizaciones se utilizan modelos jerárquicos partiendo de aquellas con gran densidad de usuarios. Esto no es tan estricto como parece debido a la redundancia y a acuerdos entre ISPs, pero es útil para empezar a comprender las longitudes extras de los caminos en Internet. En el segundo nivel (formado por los enlaces de los ISPs adyacentes) se produce PI debido a "peering policies" no óptimas, principalmente por la estrategia "early-exit", que es la más utilizada. En el tercer nivel (secuencia de ISPs necesarios para alcanzar un camino) se produce PI debido a que no existe conexión directa entre todos los ISPs (SMA03).

Varias publicaciones proponen la geografía como característica intrínseca de las redes (SPK01; BGW05). La geografía puede proporcionar pistas en la estructura y funcionamiento de Internet,

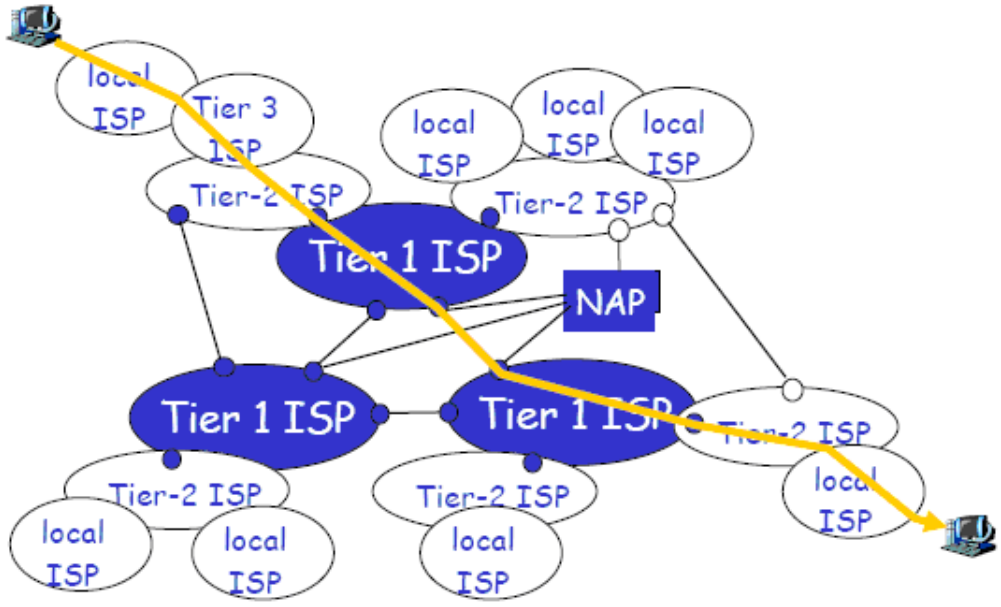


Figura 2.2: Niveles en una red de comunicaciones. El formado por el conjunto de circuitos pertenecientes a un ISP, el que comprende los enlaces de los ISPs entre destinos vecinos, y el superior con la secuencia de ISPs necesarios para alcanzar destinos lejanos.
`Fuente: http://profesores.elo.utfsm.cl/~agv/elo322/1s05/`

incluyendo las interacciones entre diferentes Sistemas Autónomos (AS). Un AS es un grupo de redes IP con políticas de encaminamiento independientes que realizan su propia gestión del tráfico que fluye entre él y los restantes ASs. La agregación de ASs conectados entre sí es lo que forma Internet (DA04).

En particular, la información geográfica puede ser útil para cuantificar la influencia de las políticas de enrutado en la distancia enrutada entre dos emplazamientos. Conocer el camino geográfico nos proporciona pistas para identificar la política de enrutado que utiliza un ISP. Si se observa que el ISP utiliza enlaces cercanos al origen la estrategia es por tanto "early-exit". Si por el contrario utiliza enlaces lejanos al origen la estrategia es "late-exit" (SPK01).

Estas publicaciones cuantifican el PI mediante una métrica que comprende la distancia enrutada y la distancia geográfica entre dos emplazamientos. El cálculo de la distancia enrutada es la suma de las distancias que recorre un paquete entre los distintos nodos intermedios que encuentra en su camino. Utilizan normalmente Geotrack[1] (herramienta de geolocalización IP) para identificar la localización de cada nodo a nivel de coordenadas geodésicas (latitud y longitud). Posteriormente se calcula la distancia enrutada con dichas coordenadas geodésicas mediante trigonometría. En concreto, la métrica utilizada por (SPK01) es la siguiente:

$$PI = \frac{d_{enrutada}}{d_{geografica}} \tag{2.1}$$

Una conclusión interesante es que se encuentra mayor PI en Europa que en Estados Unidos al coexistir muchos ISPs regionales y nacionales cuyos "peer agreements"[2] aumentan el camino (Ver Figura 2.3) (SPK01). Para llegar a esta conclusión se analizó el PI (ver Ecuación 2.1) en Europa con tres hosts origen y un conjunto de destinos pertenecientes a Europa denominado "EuroWeb dataset" (1092 direcciones en 25 países diferentes). Para el cálculo del PI dentro de

[1] GeoTrack utiliza una base de datos que relaciona el nombre DNS del interfaz del router con la localización geógrafica correspondiente

[2] "Peer agreements" son los acuerdos existentes entre los ISPs para utilizar determinadas políticas de enrutado ("peer policies")

Estados Unidos se utilizaron diecisiete hosts origen y un conjunto de destinos pertenecientes a Estados Unidos denominado "LibWeb dataset" (1205 direcciones en 49 estados de Estados Unidos).

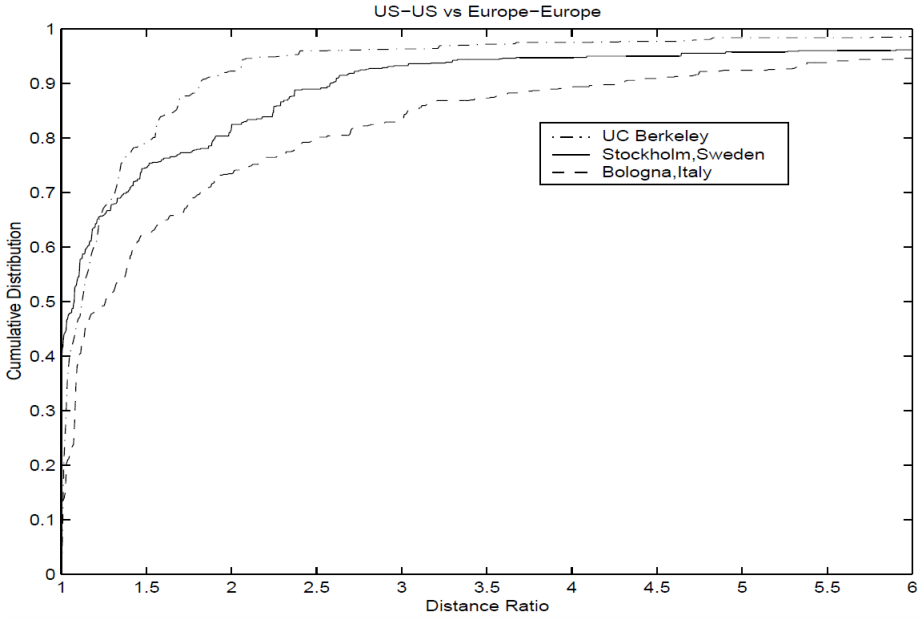


Figura 2.3: CDF de la métrica de PI (Ecuación 2.1) dentro de USA y de Europa. Se observa menor PI en Estados Unidos que en Europa.
`Fuente: Geographic Properties of Internet Routing: Analysis and Implications` (SPK01), artículo pionero en considerar la geografía en el PI.

Los dos eventos principales de la red influyentes en el PI son los cambios en la topología de la red y fluctuaciones de tráfico (PZMH07).

Internet es una compleja red que se soporta sobre una topología de interconexión de redes y computadoras en forma de grafo, que no pertenece a ningún organismo ni existe un control centralizado. Su evolución no esta definida, depende de una serie de factores sociales, económicos y tecnológicos, que provocan que el crecimiento de la demanda influya directamente en la evolución de esta topología (DA04). La evolución de la topología proporciona, en general, nuevos caminos por los que pueden ser enrutados los paquetes. En este punto, es necesario un buen algoritmo de encaminamiento para elegir el camino que mejor se adapte a las necesidades (en nuestro caso, menor distancia) (GD04). Los cambios en la topología de la red son presididos por la tendencia emergente de los grandes suministradores de contenidos a conformar grandes redes WAN (ZZX01).

Las fluctuaciones de tráfico dentro de un ISP son más frecuentes pero con menor duración que entre ISPs, y este hecho delega en un aumento de los caminos enrutados, ya que la saturación de tráfico impide utilizar el enlace predeterminado (que miniminiza la distancia recorrida) en detrimento de enlaces que proporcionan mayor distancia (GALM08; FGL+01; ZZX01).

2.1.3. Soluciones propuestas en la literatura

Los niveles que atraviesa una ruta influyen de manera diferente en el aumento de la longitud del camino enrutado. El aumento más significativo se produce en aquellos circuitos que atraviesan diversos ISPs, debido a la política "early-exit" normalmente utilizada y que mayor PI introduce. Para la comunicación entre routers de diferentes ASs se utiliza normalmente el protocolo de enrutamiento externo BGP (Border Gateway Protocol). BGP es un protocolo mediante el cual se intercambia información de encaminamiento entre ASs. Cada router de un AS conectado a otro AS mantiene una tabla BGP que contiene la ruta para que un paquete de datos llegue hasta

una red cualquiera de Internet. En estas tablas se encuentran para cada prefijo de red destino, el siguiente salto a visitar y el camino de ASs que debe atravesar para llegar a su destino. Por tanto, los ASs intercambian tráfico con sus ASs vecinos a través de estas rutas definidas con BGP de manera totalmente dinámica (DA04).

La comunidad científica propone mejoras en los protocolos de gestión de ASs, encargados de la comunicación entre routers de diferentes ASs, para reducir el PI (SMA03; CKZ05b). Cabe destacar la propuesta de desarrollar mecanismos en el protocolo BGP, ya que permite el enrutamiento obedeciendo una cierta política (coste, velocidad), para controlar el PI (SMA03).

Las relaciones de los ISPs determinan el rendimiento del circuito. Por motivos económicos y de negocio adoptan estas relaciones ignorando el PI. En la literatura se proponen políticas que comprenden este problema, como por ejemplo, la política denominada PIC ("Path Inflation Control") (CKZ05b; CKZ05a), útil para regular el PI. La política PIC limita la propagación de la congestión en IP/MPLS[3] a través de redes WDM. Para esto, PIC proporciona servicios diferenciados dependiendo de la prioridad asignada, encontrando menos bloqueo las solicitudes de alta prioridad que las de inferior prioridad. Las simulaciones indican que PIC funciona bien, incluso bajo cargas de tráfico fluctuando dinámicamente en la red (CKZ05b).

2.1.4. Discusión

La comunidad científica ha identificado el Path Inflation caracterizando sus principales causas como son las relaciones entre los ISPs, topologías de la red, fluctuaciones de tráfico, etc. Posteriormente ha cuantificado el PI con una métrica que comprende la distancia enrutada y la distancia geográfica de una ruta. Esta cuantificación ignora el tráfico envíado por la red de comunicaciones.

El presente proyecto de fin de carrera tratará de cuantificar el PI haciendo uso de una métrica que comprenda la distancia enrutada, la distancia geográfica y la demanda de tráfico para cada destino, realizando una clasificación de los efectos de este fenómeno teniendo en cuenta la popularidad del destino. De esta forma, aparece un compromiso entre la calidad de conexión entre dos destinos y la cantidad de tráfico que existe entre ellos.

2.2. Revisión del estado del arte sobre geolocalización IP

2.2.1. Introducción

En esta sección consideramos el estado del arte en geolocalización IP (GeoIP). La técnica de GeoIP consiste en identificar con precisión la localización correspondiente de una dirección IP. Muchas aplicaciones se benefician de poder conocer la ubicación geográfica de los hosts de Internet. Tales aplicaciones pueden ser, por ejemplo, el pronóstico del tiempo en un área relacionada con la localización de la dirección IP, la elección del idioma inicial para mostrar en las páginas web, publicidad dirigida, registro de visitas de una zona geográfica o la permisión de contenido restringido de acuerdo a políticas locales. Conocer el emplazamiento de los hosts también ayudará a las aplicaciones P2P para llevar una mejor experiencia de usuario (LCG+).

Existen dos métodos principalmente para conseguir este propósito, calcular la geolocalización actívamente o pasívamente (CK). En el primer grupo se utilizan cálculos basados en retardos para estimar la localización IP en función de distancias (ZFdRD05; PS01). Debido a que el retardo no sólo depende de la distancia, existen mejoras que utilizan otras variables como velocidad y

[3]MPLS es un protocolo agnóstico (diseñado para ser utilizado con múltiples protocolos) y altamente escalable, útil para la transmisión de datos. En una red MPLS, los paquetes de datos contienen etiquetas. Las decisiones de reenvío de paquetes se realiza únicamente observando el contenido de esta etiqueta, sin la necesidad de examinar el propio paquete. Esto permite crear circuitos de extremo a extremo a través de cualquier tipo de medio de transporte, utilizando cualquier protocolo

el retardo desde múltiples localizaciones (GZCF04; GZCF06; KBJK+06). En el segundo grupo no se precisa de cálculo para obtener la relación localización-dirección IP, sino que se hace uso de bases de datos ya confeccionadas o se analizan aspectos de la dirección IP que pueden proporcionar valiosa información sobre su localización.

2.2.2. Técnicas de geolocalización IP activas

Las técnicas automáticas que han sido propuestas para identificar la geolocalización IP dependen de medidas de retardos y características topológicas. Los actuales esquemas de asignación de GeoIP (PS01; GZCF04; KBJK+06; WSS06) están principalmente basados en la medición de retardos. En estos sistemas, hay una serie de puntos de referencia (landmarks) cuya localización es conocida. Los retardos entre el objetivo a localizar y los landmarks son calculados, y se mapea su localización con respecto a tales retardos. Sin embargo, la mayoría de estas técnicas están basadas en la suposición de una correlación lineal entre los retardos de la red y la distancia entre el objetivo y los landmarks. La fuerte correlación se ha comprobado en algunas regiones de Internet, tales como América del Norte y el oeste de Europa (PS01; ZFdRD05). El problema, como se señala en la literatura (ZFdRD05), es que la conectividad de Internet en todo el mundo es muy compleja, y la correlación tan fuerte se debilita en algunas zonas de Internet (LCG+), perdiendo precisión en la localización.

La Figura 2.4 muestra un estudio de (ZFdRD05) donde se estudia la correlación del retardo de la red y la distancia geográfica entre un host ubicado en París (Francia) y un conjunto de destinos denominados "LibWeb dataset" (aunque de nombre igual no confundir con el "LibWeb dataset" de (SPK01)) cuya localización es conocida. La distribución geográfica de los hosts de "LibWeb dataset" es: 56 en América del Norte, 44 en Europa del Oeste, 7 en Europa del Este, 7 en América Latina, 4 en el Medio Este, 3 en África y 3 en Oceanía. De los 135 hosts, 109 respondieron a las peticiones `ping`. El retardo considerado es el mínimo de varias medidas, intentando minimizar el aumento del retardo debido a la congestión en routers intermedios. El retardo de la red utilizado es el RTT (Round-Trip Time), que es el tiempo de ida y vuelta de un paquete en una comunicación. El RTT entre dos equipos es fácil de averiguar con el comando `ping`.

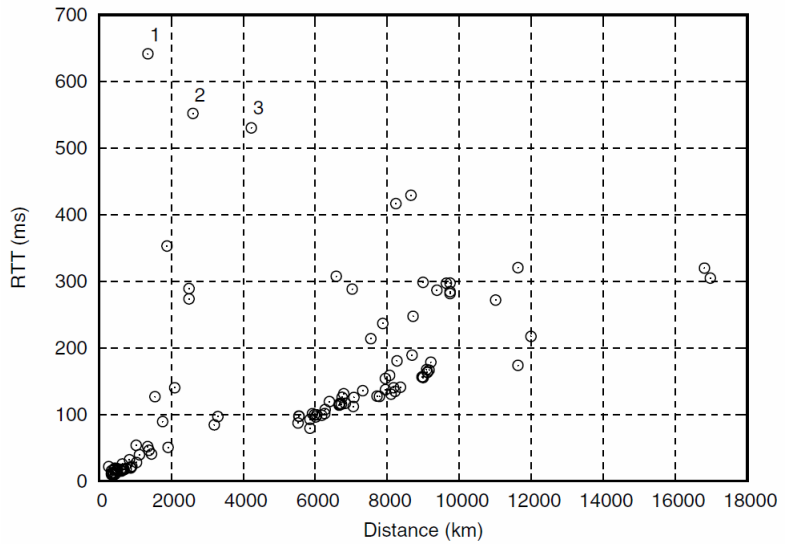


Figura 2.4: RTT vs Distancia en "LibWeb dataset". La correlación entre la distancia geográfica y el retardo de la red es débil, resultando en un coeficiente de correlación de R = 0.2971. Se observa que algunos puntos están muy alejados de otros, por lo que para misma distancia geográfica, el retardo observado puede ser mayor de un orden de magnitud.
`Fuente: Geographic Properties of Internet Routing: Analysis and Implications` (SPK01)

1. Geolocalización IP basada en retardos

"Shortest Ping"

Shortest Ping es una técnica muy simple basada en retardos para estimar la localización de una dirección IP. Se calcula el RTT de la dirección IP a localizar con cada uno de los landmarks de referencia cuya localización conocemos. Recordamos que RTT es el tiempo que tarda un paquete enviado desde un emisor en volver a este mismo habiendo pasado por el receptor de destino, es decir, tiempo de ida y vuelta de un paquete. Entonces el tiempo RTT de cada landmark es computado, y la localización del objetivo es definida como la del landmark que menor RTT ha obtenido (que se supone más cercano). En la Figura 2.5 se puede observar un ejemplo. Este método fue inicialmente investigado con el fin de convertirse en una base para evaluar técnicas más complejas (KBJK+06), aunque proporciona resultados que son comparables en precisión con sistemas avanzados de GeoIP.

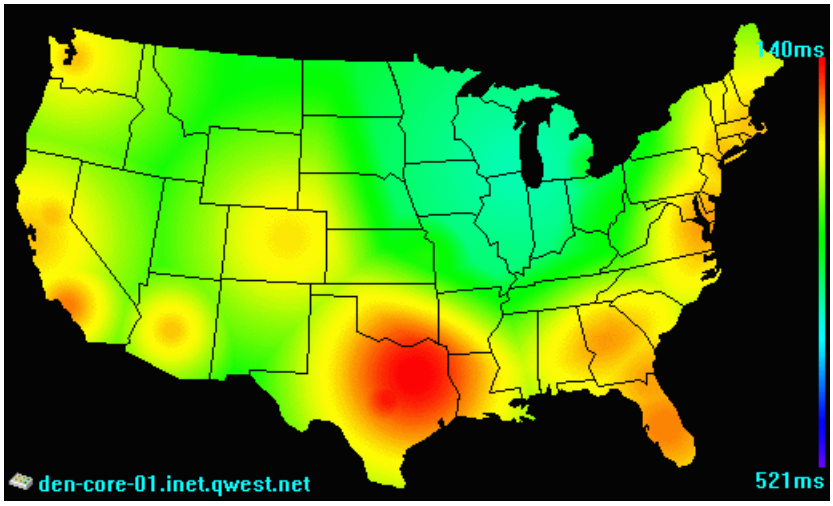


Figura 2.5: Ejemplo de Shortest Ping. Se realizan sucesivos pings a los landkmarks cuya localización es conocida. Es necesario ir computando un conjunto suficiente de RTT para asegurar estabilidad, ya que RTT es un parámetro muy variable como se observa en la figura anterior. En esta imagen se muestra una única medida de RTT, por lo que se observan muchas zonas dónde puede estar ubicado el host objetivo.
`Fuente: http://halley.cc/stuff/geoping.html`

"GeoPing"

GeoPing determina la ubicación geográfica de un host de Internet, aprovechando el retardo de la red y la distancia geográfica (PS01). Está técnica está motivada por la observación de que hosts con similares retardos en la red tienen una localización geográfica próxima.

Geoping necesita construir previamente un mapa de latencias con un conjunto de landmarks de localización conocida para posteriormente determinar la localización de un host desconocido τ. El mapa de latencias consiste en crear un vector DV_i para cada landmark i que contenga los retardos con respecto a los N landmarks localizados. Este mapa de retardos es calculado una única vez para las sucesivas geolocalizaciones IP.

Dada un nuevo host objetivo τ cuya localización debe ser determinada, se calcula un vector de latencias DV' con los retardos del host objetivo y los N landmarks de localización conocida. Este vector DV' es comparado con cada vector DV_i del mapa de retardos para encontrar el vector de retardos DV_i que resulta más cercano a DV'. La localización del landmark al cuál corresponde el vector DV_i es la ubicación que GeoPing estima del host objetivo τ.

2. Geolocalización CBG (Constraint-based Geolocation)

Para comprender esta técnica es necesario definir dos conceptos:

- Multilateration: método que estima la posición de un punto usando un número suficiente de distancias a puntos fijos de localización conocida.
- Distorsión aditiva de la distancia: el retardo se distorsiona progresivamente con respecto al tiempo que necesita la luz para pasar a través de un camino de Internet.

La principal idea de CBG es estimar la distancia entre los landmarks usando retardos, crear restricciones basadas en estas distancias y aplicarlas. CBG transforma con precisión medidas de retardos en restricciones de distancia geográficas para utilizar multilateration. Estas restricciones mejoran la estimación de la localización al compensar la imperfección de las medidas de la distancia utilizando la velocidad de la luz (GZCF04; GZCF06)(ver Figura 2.6).

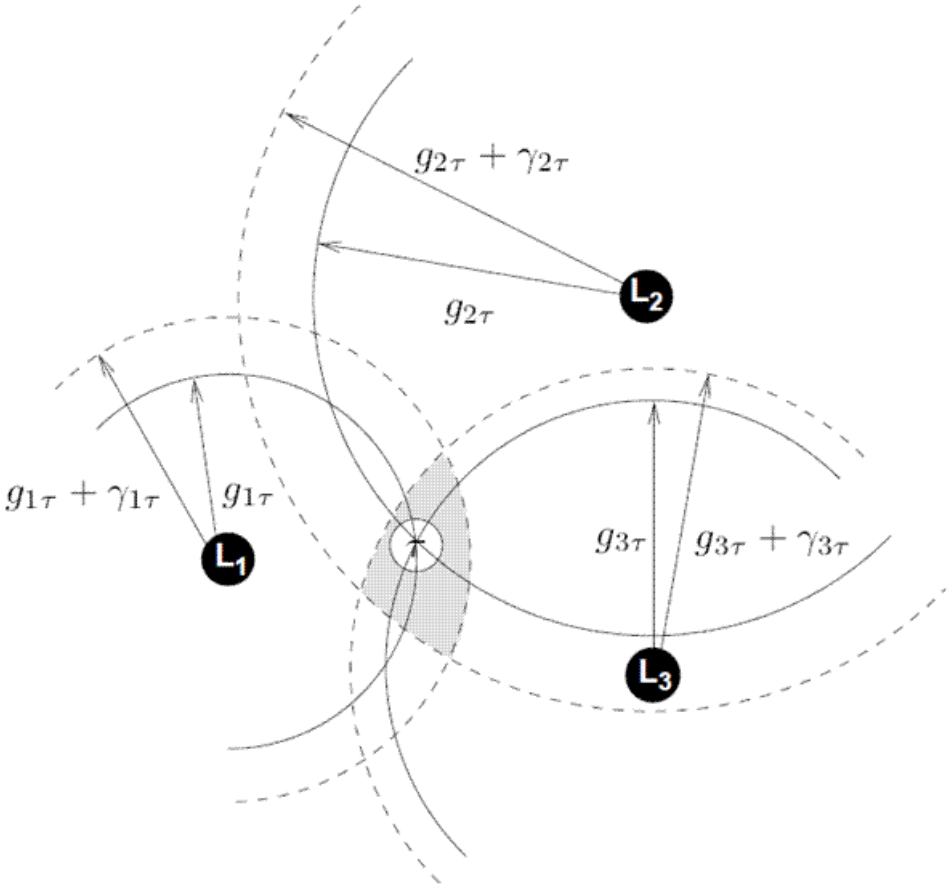


Figura 2.6: Ejemplo de CBG. CBG transforma con precisión medidas de retardos en restricciones de distancia geográficas para utilizar multilateration. En este ejemplo, a partir de tres landmarks (L1,L2,L3) y las restricciones de distancia geográfica se consigue localizar el host τ.
`Fuente: IP Geolocation` (CK)

3. Geolocalización SOI (Speed of Internet Geolocation)

Es una variación de CBG, basado en la observación de que las distancias obtenidas entre localizaciones de Internet son en realidad menores debido a las limitaciones de la velocidad de la luz. Se conoce que la información viaja a través de las fibras ópticas a casi dos tercios de la velocidad de la luz en el vacío. Debido a los circuitos, reducción a paquetes, y otros retardos

de enrutamiento, el tiempo empleado en alcanzar un destino aumenta. Por este motivo en esta técnica se utilizan restricciones ajustadas a cuatro novenos la velocidad de la luz (KBJK$^+$06).

2.2.3. Técnicas de geolocalización IP pasivas

1. Utilización de bases de datos existentes

Debido a la complejidad en los cálculos de geolocalización IP, existen a la disposición del usuario bases de datos (BBDDs) que relacionan una dirección IP con el ISP, país, ciudad, región, latitud y longitud a la que corresponden. Estas BBDDs son mantenidas y actualizadas periódicamente y se ofrecen al usuario a un precio estipulado, y algunas de ellas, gratis (arta; artb; artd). La estructura común de estas BBDDs es la división en bloques de rangos de direcciones IPs asociados, según la precisión de la BBDDs, a un país, región o ciudad. Estos bloques se encuentran en la BBDDs en orden ascendente según su rango de direcciones IP.

Los bloques se identifican por una entrada, que es la dirección IP mínima del rango. De esta manera, un rango de direcciones IP está limitado por la dirección IP de entrada del bloque y por la inmediatamente inferior a la dirección IP de la entrada del bloque siguiente.

Estas direcciones IP se encuentran en formato decimal. La fórmula para convertir una IP expresada de manera tradicional de cuatro octetos separados por puntos de formato A.B.C.D en formato decimal es:

$$IP = ((A \cdot 256 + B) \cdot 256 + C) \cdot 256 + D \tag{2.2}$$

En (PAKD$^+$11) se realiza una comparación de las BBDDs de GeoIP más populares. En la Tabla 2.1 se observa el número de bloques de los que se componen las BBDDs, el número de coordenadas geodésicas (latitud y longitud), el número de países y el número de ciudades de cada BBDDs.

Cuadro 2.1: Características generales de las BBDDs más utilizadas de GeoIp. HostIP registra el mayor número de bloques mientras que Maxmind identifica el mayor número de coordenadas geodésicas, ciudades y países.
`Fuente: IP Geolocation Databases: Unreliable?` (CK)

Database	**Blocks**	**(lat ; long)**	**Countries**	**Cities**
HostIP	8,892,291	33,680	238	23,700
IP2Location	6,709,973	17,183	240	13,600
InfoDB	3,539,029	169,209	237	98,143
Maxmind	3,562,204	203,255	244	175,035
Software77	99,134	227	255	0

Aunque estas BBDDs son ampliamente utilizadas, no son tan fiables como se cree. En primer lugar, la gran mayoría de los bloques en las BBDDs se refieren sólo a unos pocos países populares. Esto crea un desequilibrio en la representación de países en los bloques IP de las BBDDs. En la Figura 2.7 se observa la distribución de paises en las BBDDs de GeoIP más utilizadas. Queda constancia de que gran parte de los bloques pertenecen a direcciones IPs de Estados Unidos, lo que crea un claro desequilibrio en la precisión de identificar un host entre Estados Unidos y otro país o región administrativa.

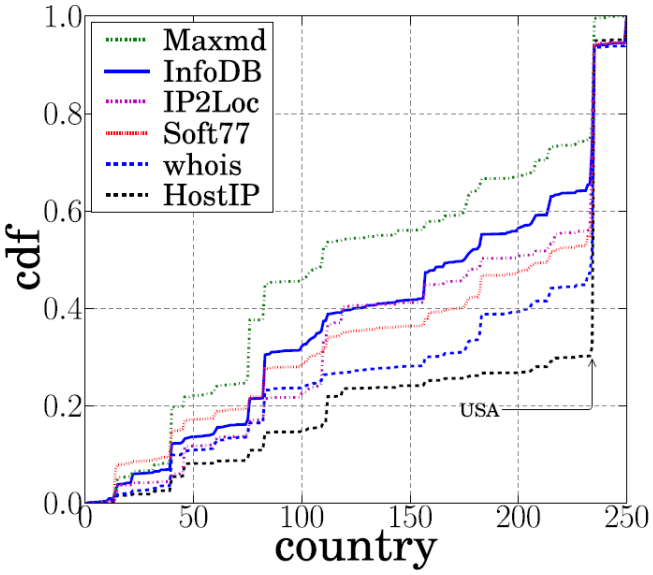


Figura 2.7: Distribución de países en las BBDDs comunes de GeoIP. Una gran parte de los bloques de las BBDDs corresponden a Estados Unidos, como mínimo más del 20 % en la BBDDs de Maxmind y como máximo un 70 % en la BBDDs de HostIP.
`Fuente: IP Geolocation Databases: Unreliable?` (CK)

En segundo lugar, estas entradas no reflejan la localización correcta de los bloques de direcciones IP. Para demostrar esto, (PAKD+11) cuantifica la precisión de geolocalización IP de estas BBDDs comparándolas con la tabla de enrutado de un gran ISP europeo. En la Tabla 2.2 se comparan los bloques de direcciones IPs de la tabla de enrutado de dicho gran ISP Europeo y las BBDDs de GeoIP de IP2Location, Maxmind e InfoDB. Existen cuatro posibilidades: Exacto (el bloque está presente en ambos), Menor (el bloque está presente pero es menor en la BBDDs), Mayor (el bloque está presente pero es mayor en la BBDDs), y Parcial (el bloque de la BBDDs solapa con otro prefijo de la tabla de enrutado del ISP Europeo.)

Cuadro 2.2: Comparación de los bloques de IPs de la tabla de enrutado de un gran ISP Europeo y las BBDDs de GeoIP de IP2Location, Maxmind e InfoDB. IP2Location es la BBDDs que obtiene el mayor número de bloques idénticos a la tabla de enrutado del mencionado gran ISP Europeo.
`Fuente: IP Geolocation Databases: Unreliable?` (CK)

Database	Exacto	Mayor	Menor	Parcial
IP2Location	32,429	70,693	3,531	373
Maxmind	27,917	79,735	4,092	128
InfoDB	9,954	51,399	1,763	104

2. Información de geolocalización IP en el nombre DNS

Es un método intuitivo que concluye que el nombre DNS relacionado con la IP a localizar contiene en algunos casos valiosa información geográfica, como puede ser el ISP, ciudad, estado, etc. Sin embargo, esta técnica no es trivial al no existir una nomenclatura normalizada para los ISPs (PS01).

Ejemplos:

ngcore1-serial-8-0-0-0.**Seattle**.sw.net => **Seattle**

184.atm6-0.xr2.**ewr**1.alter.net => **New York**

dnvr-scrm.abilene.uacaid.edu => **Denver**

GeoTrack es una herramienta con base en esta técnica de GeoIP y es utilizada por (SPK01; BGW05) para estudiar el efecto de la geografía en la cuantificación del PI.

2.2.4. Técnica de geolocalización IP empleada

La técnica de geolocalización IP empleada para el presente proyecto es pasiva. Es una BBDDs basada en la versión gratuita de Maxmind de nombre InfoDB (artc) cuyas características son las siguientes:

Cuadro 2.3: Características de la BBDDs InfoDB de Maxmind. A priori parece interesante la precisión asegurada, ya que el estudio del PI en RedIRIS será a nivel de país.

Tipo de BBDDs	Precisión asegurada	Número de bloques
Completa	95 % a nivel de país, 60 % a nivel de ciudad	3.5 Millones

La BBDDs InfoDB contiene información sobre la ciudad, región, país, latitud y longitud, entre otros. La BBDDs es actualizada durante la primera semana de cada mes. Esta BBDDs relacionará la ubicación de los objetivos con su dirección IP. La información relativa a la longitud y a la latitud nos proporcionará la distancia geográfica entre dos localizaciones mediante técnicas trigonométricas como veremos próximamente. En la Figura 2.8 se observa el error de la BBDDs de InfoDB con la tabla de enrutado de un gran ISP Europeo (PAKD+11).

Observando la descompensación en el número de bloques entre los países, la comparación de las entradas de los bloques con las tablas de enrutado de un gran ISP Europeo y la Figura 2.8, estoy de acuerdo con la conclusión de (PAKD+11) en que estas BBDDs útiles para la GeoIP tienen precisión de país.

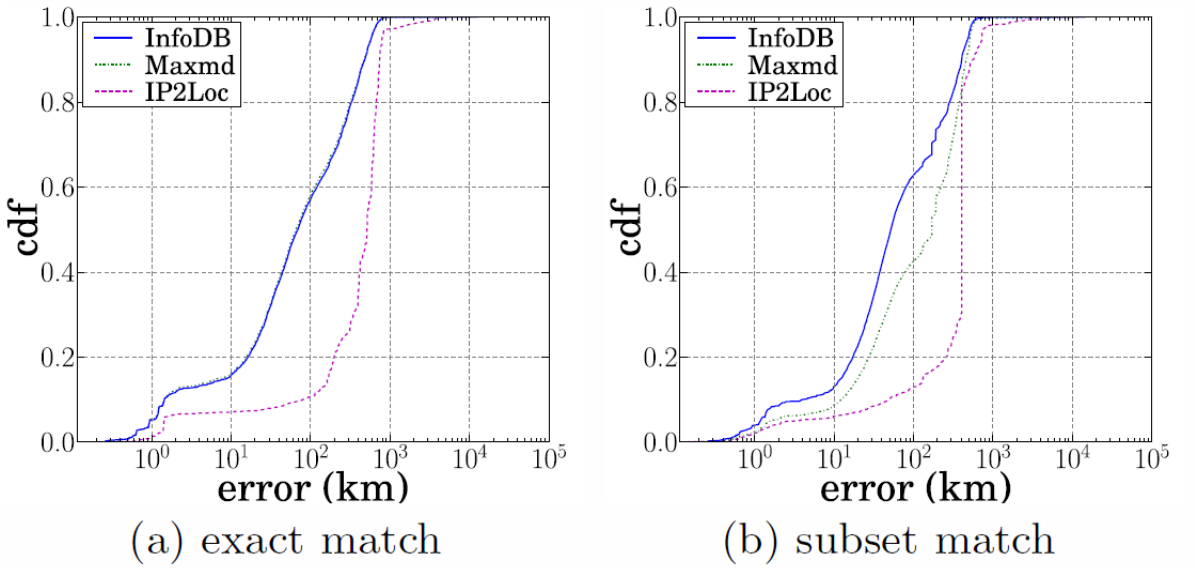


Figura 2.8: Error de GeoIP de InfoDB, Maxmind e IP2Location comparado con un gran ISP Europeo. InfoDB encuentra un error menor en más del 90 % de los bloques comparada con las otras dos BBDDs. InfoDB encuentra un error de unos 100 km en el 60 % de los bloques.
`Fuente: IP Geolocation Databases: Unreliable?` (CK)

En conclusión, InfoDB parece ser una BBDDs de GeoIP válida para identificar direcciones

IPs a nivel de país. Según el estudio pionero en fiabilidad de BBDDs de GeoIP de la literatura (PAKD+11), InfoDb es la tercera en el ranking total, y la primera en el ranking de las BBDDs gratuitas. InfoDB se sitúa tercera en el ranking total debido a que aunque tiene el menor error en las entrada de la BBDDs (según la comparación con un gran ISP Europeo) posee un menor número de bloques.

2.3. Conclusiones sobre la variabilidad de las rutas en Internet

2.3.1. Introducción

En el estudio del estado del arte sobre Path Inflation hemos comprendido las causas y consecuencias de este efecto, y que para su cuantificación es necesario identificar la ruta extremo a extremo entre dos puntos. Existen generalmente caminos por defecto entre dos puntos, pero en ocasiones, se utilizan otras rutas debido a problemas de congestión, indisponibilidad de routers u otras causas. La utilización de rutas alternativas en nuestro estudio comprenderá cambios a nivel de RTT, distancia de enrutamiento y ancho de banda. Es interesante conocer que el uso de rutas alternativas no tiene porqué implicar menor rendimiento que el propio de la ruta por defecto ya que los protocolos y políticas de enrutado no están íntimamente relacionados con el rendimiento de la red (Sav99).

En la Figura 2.9 se observa el hecho comentado previamente. Se observa la diferencia en el RTT medio entre las rutas por defecto y las mejores rutas en diferentes conjuntos de datos (datasets). El dataset UW1 contiene 54034 medidas durante 34 días en América del Norte, el UW3 contiene 94420 medidas durante 7 días en América del Norte, el D2-NA contiene 14892 medidas durante 48 días en América del Norte y el D2 que contiene 35109 medidas durante 48 días en el Mundo.

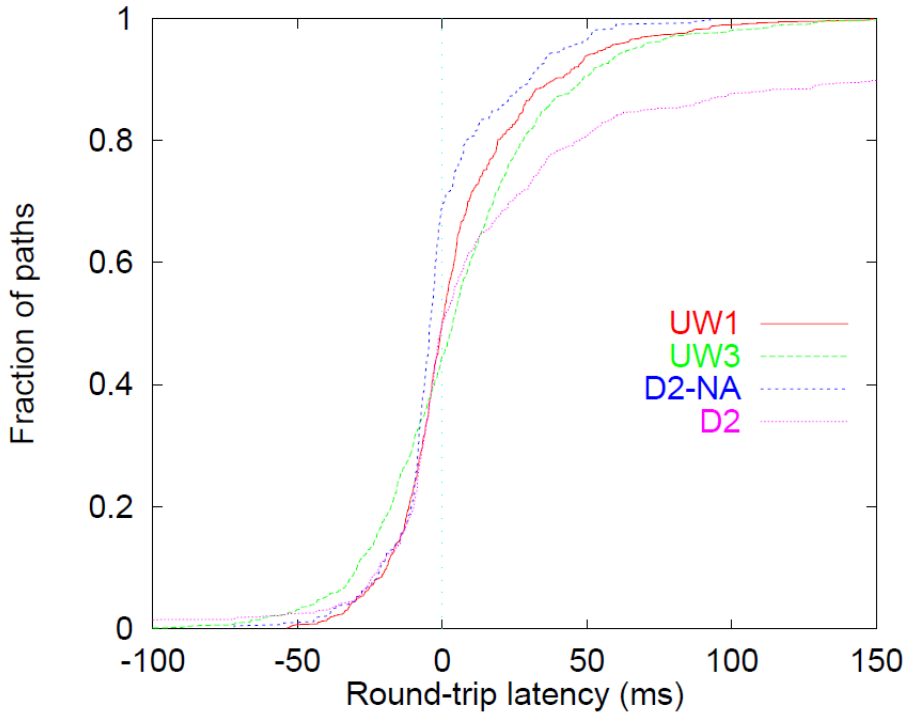


Figura 2.9: Diferencia del RTT medio de la ruta por defecto y de la mejor ruta en diversos conjuntos de datos. Valores positivos indican que la ruta por defecto tiene mayor RTT que la mejor ruta, y valores negativos lo contrario. Se observa que del 30 % al 55 % de los caminos por defecto tienen mayor RTT que la mejor ruta alternativa.

En concreto estamos interesado en los cambios en la distancia de enrutamiento para el cálculo

del PI y cambios en el RTT para el estudio de retardos.

2.3.2. Variabilidad en la distancia enrutada

La conclusión más importante se extrae de (Pax96) y es que normalmente los caminos en Internet están dominados por una única ruta. Esta conclusión nos asegura calcular la distancia enrutada del camino por defecto en la mayoría de medidas que realizaremos. Asímismo nos proveeremos de un conjunto de datos considerable con el fin de obtener medidas fiables. Esta premisa es válida también para el número de saltos, ya que es otra característica intrínseca de una ruta.

2.3.3. Variabilidad en el retardo

A diferencia de la distancia enrutada y del número de saltos, el retardo sufre variaciones en la propia ruta debido principalmente a la congestión. Debido a esta variabilidad será necesario realizar más medidas del retardo para poder caracterizar conjuntos de caminos de Internet.

3

Diseño y desarrollo

3.1. Metodología empleada

La metodología consta de las siguientes fases:

Fase 1. Identificación de emplazamientos representativos dentro de cada país. Formarán parte del conjunto de destinos analizados en la evaluación del Path Inflation.

Fase 2. Identificación de los diferentes nodos intermedios que atraviesan los paquetes en una comunicación entre dos extremos.

Fase 3. Cálculo de la distancia de encaminamiento entre dos extremos.

Fase 4. Estudio de diferentes métricas propuestas para estudiar el compromiso entre calidad de conexión y demandas de ancho de banda entre los destinos más populares. Selección justificada de la métrica que mejor represente este compromiso.

3.2. Identificación de emplazamientos representativos dentro de cada país

3.2.1. Sobre RedIRIS

Como ya hemos comentado, RedIRIS es la red académica y de investigación española y proporciona servicios avanzados de comunicaciones a la comunidad científica y universitaria nacional (ver Figura 3.1). RedIRIS cuenta con más de 350 instituciones afiliadas, principalmente universidades y centros públicos de investigación, que llegan a formar parte de esta comunidad mediante la firma de un acuerdo de afiliación.

3.2.2. Origen de los emplazamientos seleccionados

RedIRIS posee exportadores de tráfico en cada nodo o Punto de Presencia (PoP) (ver Figura 3.1). Cada exportador de tráfico registra parte de las conexiones del nodo en el que se ubica con otras localizaciones y la cantidad de tráfico que pasa a su través. Con esta información se puede obtener la dirección IP de las localizaciones con las que un PoP intercambia tráfico.

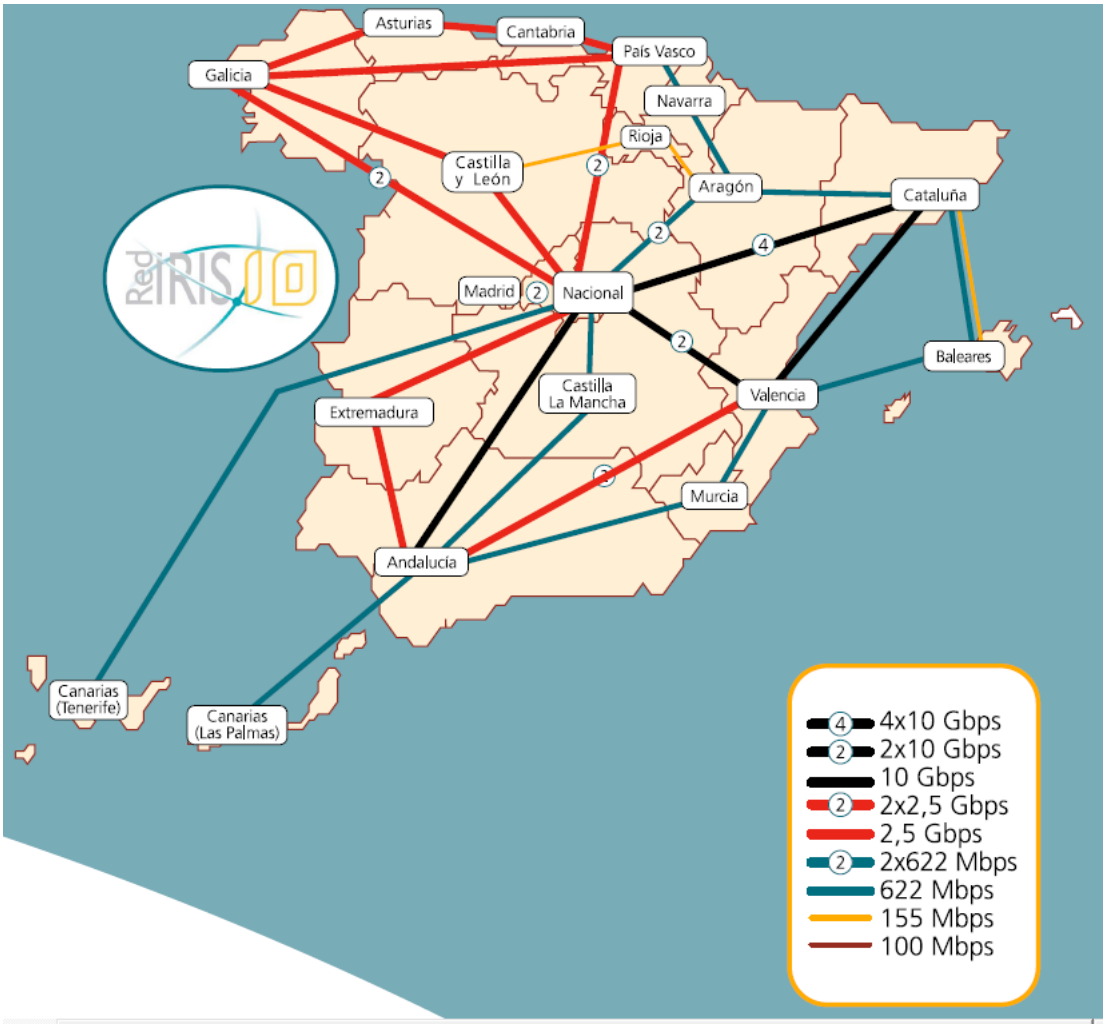


Figura 3.1: Mapa de la topología de RedIRIS.

RedIRIS nos ha donado gentílmente datos del exportador de tráfico del PoP de Madrid, por lo que de dicho exportador obtendremos las direcciones IP para nuestro estudio y el tráfico intercambiado. Utilizaremos los registros de Netflow de este exportador de Madrid relativos a 35 días. Esta cantidad de días es suficiente para obtener un conjunto de datos representativo de las direcciones IP que intercambian tráfico con RedIRIS según un estudio de la estabilidad del tráfico en RedIRIS (MGDALdVon). La cabecera del netflow se observa en la Tabla 3.1.

Start	End	Sif	SrcIPaddress	SrcP	DIf	DstIPaddress	DstP	P	Fl	Pkts	Octets

Cuadro 3.1: Cabecera del netflow proporcionado por RedIRIS.

El contenido de cada campo es:

- Start: contiene la fecha y la hora de comienzo de la comunicación.
- End: contiene la fecha y la hora de la finalización de la comunicación.
- Sif: interfaz de salida del origen.
- SrcIpaddres: contiene la dirección IP del origen.
- SrcP: contiene el número del puerto utilizado para la comunicación con el transmisor.
- DIf: contiene el número de la interfaz de entrada del destino.
- DstIPaddress: contiene la dirección IP del destino.

- DstP: contiene el número del puerto utilizado para la comunicación con el destino.
- P: protocolo de transporte (17 = UDP , 6 = TCP).
- Fl: contiene el número de flags del protocolo de transporte TCP. Obtendrá un valor nulo si se utiliza el protocolo de transporte UDP.
- Pkts: contiene el número de paquetes enviados en la comunicación.
- Octets: contiene el número de Bytes enviados en la comunicación.

Los campos que nos interesan son DstIPaddress, para identificar la dirección IP de los destinos, y Octets, para calcular el tráfico dirigido a cada destino.

Agruparemos las direcciones IP obteniendo el tráfico total que atraviesa el PoP de Madrid de RedIRIS para posteriormente estudiarlo. Así obtendremos una entrada por dirección IP con el tráfico total identificado por el exportador para cada una. También clasificaremos las direcciones IP por país/región administrativa para observar el número de destinos o direcciones IP diferentes por país, ya que el estudio del PI será a este nivel. Debido a que este estudio pretende caracterizar y cuantificar el PI en los destinos internacionales más populares, las direcciones IP pertenecientes a España y por tanto su tráfico serán filtrados.

Es importante comprender que los países presentan diferente número de direcciones IP o destinos diferentes (muestras), por lo que países con mayor número de muestras presentarán unos resultados más fiables. Al haber mucha variabilidad en el número de muestras entre países, existirá también una amplia variación en el tráfico que recibe cada país. Hay países que reciben muy poco tráfico, por lo que el análisis de PI en función del tráfico no será importante en estos casos. Tras estudiar este punto, se llegó a la decisión de estudiar aquellos países cuyo tráfico recibido supere el 0,005 % del tráfico total enviado por RedIRIS durante el período de extracción de datos (superan el mínimo tráfico 146 países). Aplicando este filtro, se observa en la Tabla 3.2 el número de direcciones IP distintas identificadas , el tráfico total recibido y el tráfico medio recibido por dirección IP en cada país que cumple el mínimo tráfico recibido.

3.2.3. Estadísticas de los emplazamientos seleccionados

Estadísticas generales del tráfico

Cuadro 3.2: Características del tráfico enviado por RedIRIS.

País/ Reg. Administrativa	**Continente**	**GB recibidos**	**# IP**	$\overline{MB/IP}$
Aland Islands	Europa	13.35	752	17.75
Albania	Europa	31.61	4561	6.93
Algeria	Africa	285.41	61162	4.67
Andorra	Europa	405.71	8506	47.70
Angola	Africa	27.98	4113	6.80
Antarctica	Antartida	1.59	32	49.58
Antigua and Barbuda	America	4.81	1227	3.92
Argentina	America	15600.72	1767606	8.83
Armenia	Asia	34.65	6107	5.67
Aruba	America	9.06	3555	2.55
Australia	Oceania	2276.57	275989	8.25
Austria	Europa	10897.60	103924	104.86
Azerbaijan	Asia	28.81	13563	2.12
Bahamas	America	10.34	7270	1.42
Bahrain	Africa	18.16	5826	3.12

Cuadro 3.2 — Continúa de la página anterior

País/ Reg. Administrativa	**Continente**	**GB recibidos**	**# IP**	$\overline{MB/IP}$
Bangladesh	Asia	47.12	5748	8.20
Barbados	America	25.37	7447	3.41
Belarus	Europa	58.43	60890	0.96
Belgium	Europa	15234.98	136188	111.87
Bermuda	America	8.36	1972	4.24
Bolivia	America	1610.75	70412	22.88
Bosnia and Herzegovina	Europa	74.40	11941	6.23
Botswana	Africa	7.50	1695	4.43
Brazil	America	8012.03	1854013	4.32
Bulgaria	Europa	587.41	199859	2.94
Cambodia	Asia	10.68	4108	2.60
Cameroon	Africa	9.28	4433	2.09
Canada	America	9586.60	558546	17.16
Chile	America	12343.70	1020621	12.09
China	Asia	8459.04	1170975	7.22
Colombia	America	15646.95	1363386	11.48
Costa Rica	America	1277.78	98666	12.95
Croatia	Europa	643.07	67317	9.55
Cuba	America	374.65	3366	111.30
Cyprus	Asia	384.60	13653	28.17
Czech Republic	Europa	1833.10	97193	18.86
Denmark	Europa	2529.48	74966	33.74
Djibouti	Africa	5.37	411	13.07
Dominica	America	3.33	878	3.79
Dominican Republic	America	1172.09	157686	7.42
Ecuador	America	3420.66	174852	19.56
Egypt	Africa	473.87	111090	4.27
El Salvador	America	881.60	80236	10.99
Equatorial Guinea	Africa	12.70	1047	12.13
Estonia	Europa	573.30	23439	24.46
Ethiopia	Africa	9.58	1428	6.71
Finland	Europa	2187.59	140129	15.61
France	Europa	74800.14	938339	79.72
French Polynesia	Oceania	5.97	1224	4.88
Georgia	Asia	48.55	19214	2.53
Germany	Europa	96202.46	1155339	83.27
Ghana	Africa	21.05	10212	2.06
Gibraltar	Europa	41.11	3076	13.36
Greece	Europa	3355.05	245658	13.66
Guadaloupe	America	36.62	5084	7.20
Guatemala	America	1316.58	94927	13.87
Haiti	America	8.77	1984	4.42
Honduras	America	493.90	18384	26.87
Hong Kong	Asia	515.47	97482	5.29
Hungary	Europa	15041.02	140379	107.15
Iceland	Europa	584.62	18445	31.70
India	Asia	1875.75	844391	2.22
Indionesia	Asia	365.40	104721	3.49
Iran Islamic Republic of	Asia	359.51	60968	5.90
Iraq	Asia	14.34	2257	6.35

Cuadro 3.2 — Continúa de la página anterior

País/ Reg. Administrativa	**Continente**	**GB recibidos**	**# IP**	$\overline{MB/IP}$
Ireland	Europa	2012.13	74956	26.84
Israel	Asia	1055.40	197120	5.35
Italy	Europa	51934.24	1197695	43.36
Jamaica	America	58.69	18421	3.19
Japan	Asia	4144.85	305177	13.58
Jordan	Asia	76.85	11929	6.44
Kazakhstan	Asia	60.69	75878	0.80
Kenya	Africa	46.47	10830	4.29
Korea Republic of	Asia	9860.93	258327	38.17
Kuwait	Asia	89.80	30532	2.94
Latvia	Europa	216.66	46226	4.69
Lebanon	Asia	108.93	9250	11.78
Libyan Arab Jamahiriya	Africa	18.37	10384	1.77
Liechtenstein	Europa	6.37	774	8.23
Lithuania	Europa	334.10	55292	6.04
Luxembourg	Europa	859.84	11952	71.94
Macao	Asia	21.40	6540	3.27
Macedonia	Europa	83.35	35150	2.37
Malaysia	Asia	550.15	130719	4.21
Maldives	Asia	4.89	1767	2.77
Malta	Europa	393.13	14380	90.77
Martinique	America	20.75	4331	4.79
Mauritius	Africa	18.41	9271	1.99
Mexico	America	25944.81	3031203	8.56
Moldova Republic of	Europa	64.50	30174	2.14
Mongolia	Asia	21.31	2991	7.12
Montenegro	Europa	35.75	12288	2.91
Morocco	Africa	1364.47	142814	9.55
Mozambique	Africa	11.28	2241	5.04
Namibia	Africa	8.25	3027	2.73
Netherlands	Europa	13422.98	286127	46.91
Netherlands Antilles	America	22.45	4487	5.00
New Caledonia	Oceania	27.72	3247	8.54
New Zealand	Oceania	366.67	46707	7.85
Nicaragua	America	445.74	24946	17.87
Nigeria	Africa	31.53	10494	3.00
Norway	Europa	3632.19	229957	15.80
Oman	Asia	12.82	8090	1.59
Pakistan	Asia	279.43	72238	3.87
Palestinian Territory	Asia	23.90	18767	1.27
Panama	America	932.32	93280	9.99
Paraguay	America	633.40	52820	11.99
Peru	America	9426.74	475430	19.83
Philippines	Asia	526.68	85428	6.17
Poland	Europa	4941.28	493788	10.01
Portugal	Europa	10255.74	288451	35.55
Qatar	Asia	89.99	14608	6.16
Reunion	Africa	53.67	5936	9.04
Romania	Europa	2348.50	274962	8.54
Russian Federation	Asia	48124.55	707457	68.02

Cuadro 3.2 — Continúa de la página anterior

País/ Reg. Administrativa	**Continente**	**GB recibidos**	**# IP**	$\overline{MB/IP}$
Saint Lucia	America	8.02	2126	3.77
San Marino	Europa	14.84	923	16.07
Saudi Arabia	Asia	762.39	267457	2.85
Senegal	Africa	55.46	10186	5.45
Serbia	Europa	480.72	124254	3.87
Seychelles	Africa	17.97	225	79.88
Singapore	Asia	599.23	94452	6.34
Slovakia	Europa	622.33	43097	14.44
Slovenia	Europa	451.03	41060	10.98
South Africa	Africa	236.11	70872	3.33
Sri Lanka	Asia	41.17	20383	2.02
Sudan	Africa	19.90	7094	2.81
Sweden	Europa	19917.88	713614	27.91
Switzerland	Europa	53804.69	136308	394.73
Syrian Arab Republic	Asia	59.92	4700	12.54
Taiwan	Asia	38323.63	389146	98.48
Tanzania United Republic	Africa	8.25	2553	3.23
Thailand	Asia	418.11	166158	2.52
Trinidad and Tobago	America	86.91	13538	6.42
Tunisia	Africa	150.00	28797	5.21
Turkey	Asia	1423.37	406504	3.50
Uganda	Africa	6.69	1116	5.99
Ukraine	Europa	752.81	303315	2.48
United Arab Emirates	Asia	339.44	59236	5.73
United Kingdom	Europa	94016.98	1333510	70.50
United States	America	254474.11	3134798	81.18
Uruguay	America	2397.80	162872	14.72
Uzbekistan	Asia	8.10	3949	2.05
Venezuela	America	7887.08	830330	9.50
Vietnam	Asia	278.19	399244	0.70
Yemen	Asia	17.05	6957	2.45
MEDIA		6746.16	214059	
TOTAL		984939.23	31252684	

TRÁFICO TOTAL = 984939.23 GigaBytes

Se observa que el número medio por país de direcciones IP es de 214000 y el número total de 31 millones. Estas direcciones IP forman parte de nuestro conjunto de datos para el estudio del PI. Parece un conjunto de datos más que suficiente al superar ampliamente a los conjuntos de datos de (SPK01) "EuroWeb dataset" (1092 direcciones en 25 países europeos) y "LibWeb dataset" (1205 direcciones en 49 estados de Estados Unidos) utilizados para el cálculo del PI dentro de Europa y EEUU.

En la Figura 3.2[1] se observa en función de la tonalidad, aquellos países o regiones administrativas con mayor tráfico recibido desde RedIRIS en escala logarítmica. La presentación de datos en una escala logarítmica es útil ya que los datos cubren una amplia gama de valores y el logaritmo los reduce a un rango más manejable.

América posee el destino más popular, Estados Unidos, con la tonalidad más oscura. En una tonalidad aún superior a la media, se encuentran la mayoría de países de este continente, como Brasil, México, Canada, Argentina, Chile, Peru, Venzeuela, etc.

En Europa existen una serie de países cuya tonalidad es ligeramente menor que Estados Unidos, entre ellos, Reino Unido, Francia, Alemania, Italia, Portugal, Suecia, etc.

En Asia destaca con una tonalidad ligeramente inferior a Estados Unidos, Rusia, que también divide su extensión entre Europa y Asia. En una tonalidad media están China, Japón y Taiwan, entre otros.

En África con una tonalidad que ronda la media aparecen Marruecos, Argelia, Libia y Egipto. El resto de países son más claros ya que este continente no recibe mucho tráfico desde RedIRIS.

En Oceanía, están con una tonalidad media Australia y Nueva Zelanda.

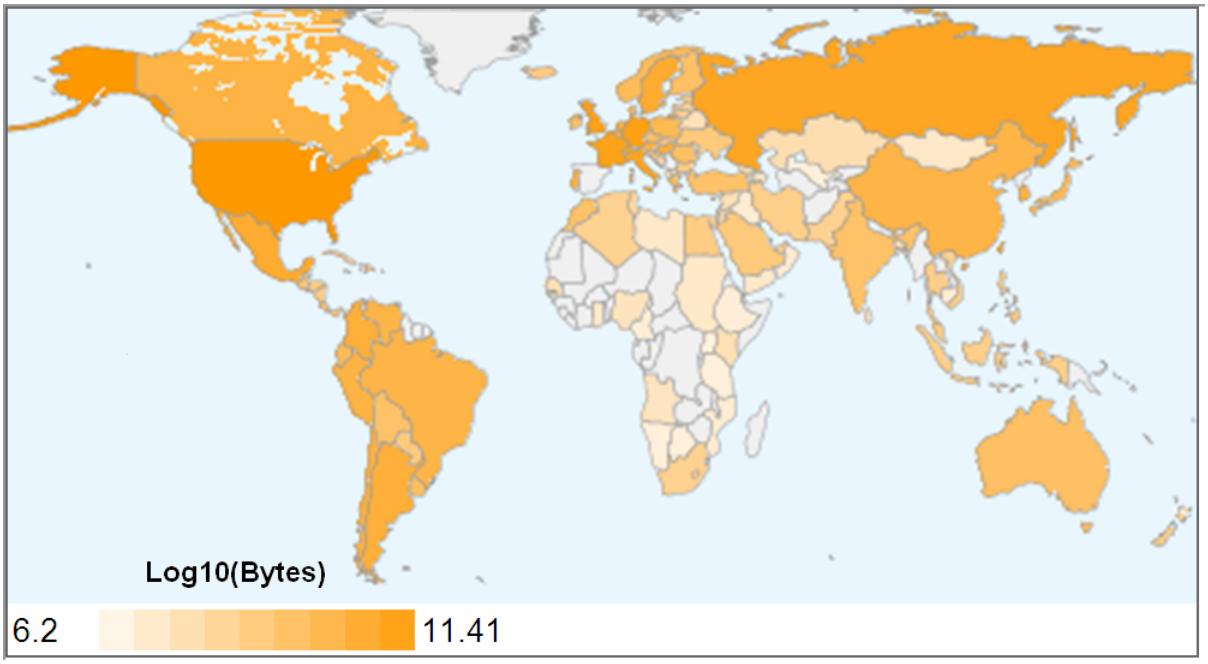


Figura 3.2: Tráfico recibido por país en escala logarítmica.

[1]Imagen generada con el gadget Heat Map de Google Docs. Todos los mapas estilo Heat Map presentes en este documento han sigo generadas con dicho gadget.

Análisis de las estadísticas del tráfico

A nivel continental destaca el continente europeo, recibiendo el mayor porcentaje de tráfico procedente de RedIRIS con casi el 50 % del tráfico enviado. En el segundo puesto se encuentra el continente americano con un 38 % del tráfico enviado. El continente asiático alcanza el 12 % del tráfico y en últimos lugares y sin alcanzar el 1 % se encuentran los continentes de Oceanía y África (ver Figura 3.3 y Figura 3.4).

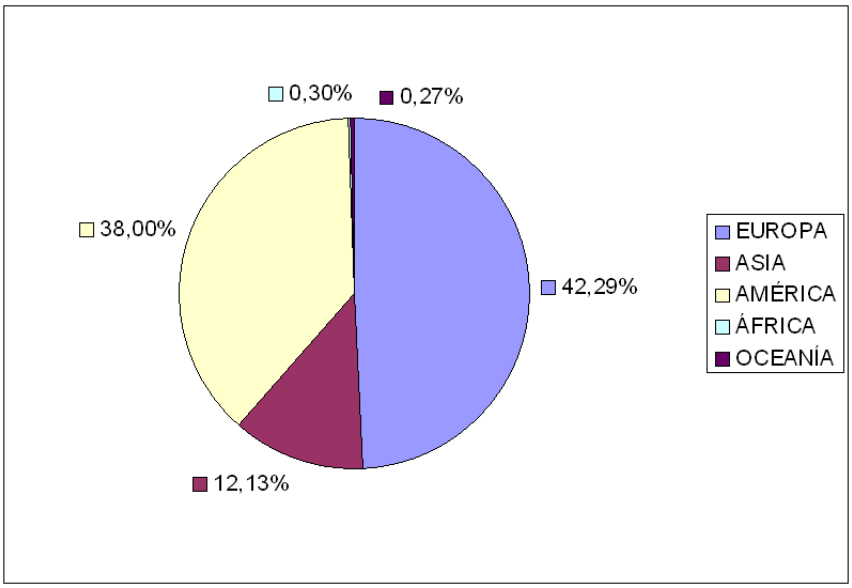


Figura 3.3: Porcentaje de tráfico recibido por continente.

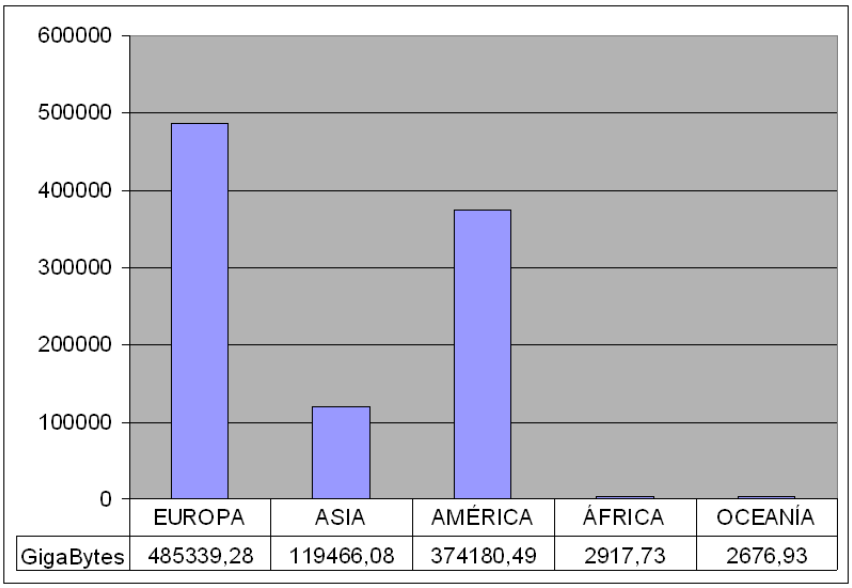


Figura 3.4: Tráfico total recibido por continente.

En la Figura 3.5 y la Figura 3.6 se observa el top15 de países en tráfico recibido. A nivel global ocupa el primer lugar la primera potencia mundial, Estados Unidos, con un alcance del 25 % del tráfico total enviado desde RedIRIS. En una escala menor nos encontramos un segundo bloque de paises mayoritariamente de origen Europeo, como Alemania, Reino Unido, Francia, Suiza, Italia y Rusia (éste último tiene más de dos tercios de su territorio en Europa) y con menor tasa de tráfico se encuentran Suecia, Bélgica, Hungría y Holanda. Un tercer bloque está constituido con países de origen hispanohablante dominados por México y acompañado por Colombia y Argentina. A nivel asiático destaca Taiwan, en el Top8 de países con mayor tráfico recibido desde RedIRIS.

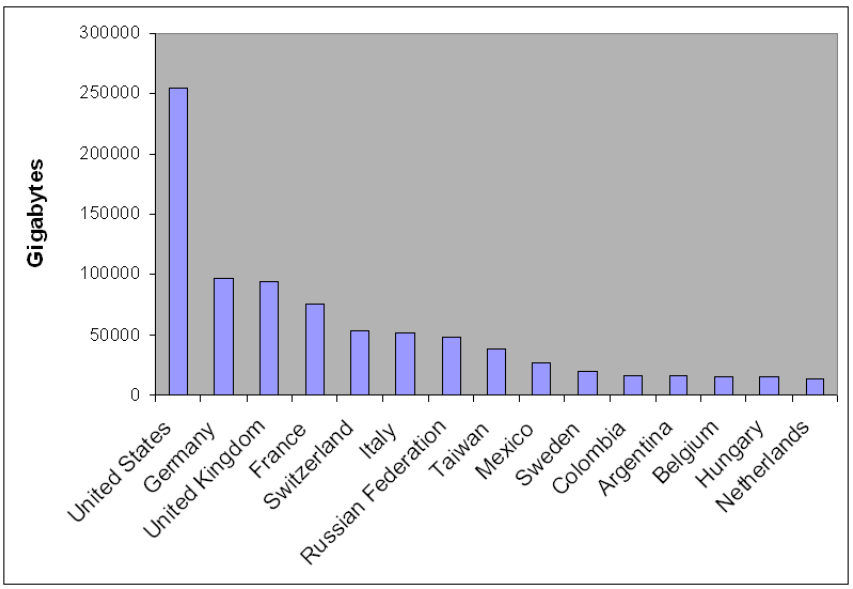


Figura 3.5: Top15 de paises con mayor tráfico recibido.

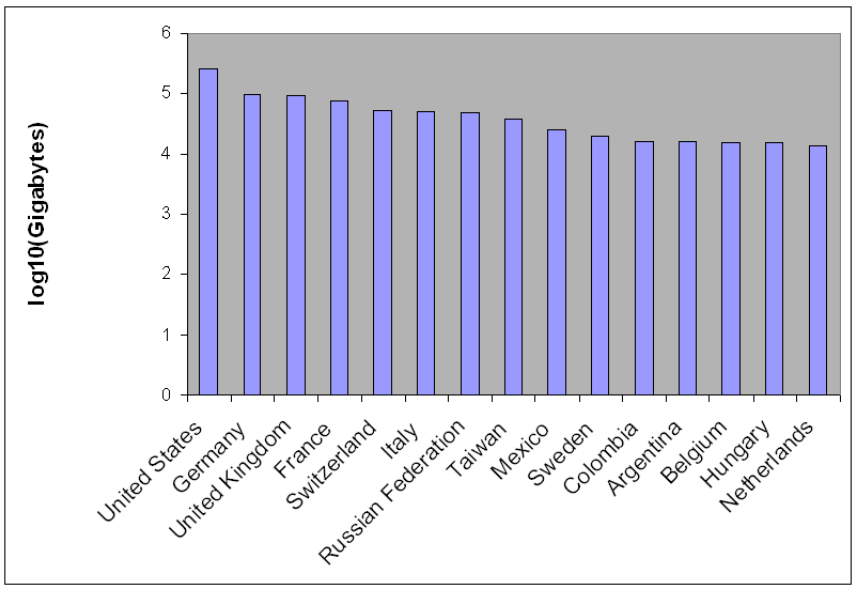


Figura 3.6: Top15 de paises con mayor tráfico recibido en escala logarítmica.

3.3. Identificación de los nodos intermedios que atraviesan los paquetes en una comunicación

3.3.1. Introducción

El conjunto de direcciones IP identificadas en la sección anterior se utilizará como muestra de los destinos más representativos accedidos desde Madrid. Para hacer un estudio del PI, calcularemos las distancias a dichos destinos tomando como origen del tráfico Madrid. Para calcular dicha distancia será necesario conocer los nodos intermedios en estas comunicaciones.

Para esta tarea existen varias herramientas similares, como `traceroute` en UNIX y Linux y `tracert` en Windows. Dichas herramientas obtienen la ruta seguida por un paquete identificando los equipos intermedios que atraviesan. Se obtiene además una estadística del RTT o latencia de red de esos paquetes. Estas herramientas de identificación de nodos intermedios utilizan paquetes UDP o ICMP ECHO.

El problema es que con el masivo uso de cortafuegos (firewalls) en Internet, muchos de los paquetes que utiliza `traceroute` son filtrados, por lo que es imposible rastrear la ruta completa a su destino. Sin embargo, en muchos casos, estos cortafuegos permitirán el acceso a los paquetes TCP entrantes a puertos específicos. Debido a estas limitaciones, existen otras herramientas basadas en paquetes TCP, como `TCPTraceroute`, capaz de eludir los filtros de cortafuegos más comunes.

3.3.2. Análisis de traceroute y TCPTraceroute

En esta sección se analizará el rendimiento de `traceroute` y `TCPTraceroute`[2]. Con este análisis se estudiará el porcentaje de objetivos alcanzados por país/región administrativa de cada herramienta. En la Tabla 3.2 se observa el número de direcciones IP por país identificadas. Para cada una de estas direcciones IP se utilizará `traceroute` y `TCPTraceroute`. Se considera que una ruta es alcanzable por estas herramientas cuando se identifican todos los nodos intermedios presentes en la comunicacón extremo a extremo. Con esta consideración se calcula en la Tabla 3.4 el porcentaje de direcciones IP de cada país/región administrativa que cumplen los requerimientos especificados.

Cuadro 3.4: Porcentaje de destinos alcanzados con `TCPTraceroute` (paquetes TCP) y con `traceroute` (paquetes ICMP).

País/ Región Administrativa	**Continente**	**% Paquetes TCP**	**% Paquetes ICMP**
Aland Islands	Europa	9.84 %	5.59 %
Albania	Europa	23.70 %	13.40 %
Algeria	Africa	11.72 %	10.99 %
Andorra	Europa	7.94 %	4.59 %
Angola	Africa	2.09 %	1.46 %
Antarctica	Antartida	84.38 %	21.88 %
Antigua and Barbuda	America	27.14 %	22.00 %
Argentina	America	18.05 %	10.53 %
Armenia	Asia	8.61 %	5.76 %
Aruba	America	11.50 %	9.62 %
Australia	Oceania	8.06 %	4.72 %
Austria	Europa	11.96 %	6.04 %
Azerbaijan	Asia	12.70 %	5.82 %

[2]Se pueden encontrar los manuales de estas herramientas en el anexo **C Manual de utilización**

Cuadro 3.4 — Continúa de la página anterior

País/ Región Administrativa	**Continente**	**% Paquetes TCP**	**% Paquetes ICMP**
Bahamas	America	9.23 %	6.45 %
Bahrain	Africa	2.44 %	1.10 %
Bangladesh	Asia	19.82 %	15.24 %
Barbados	America	68.38 %	0.63 %
Belarus	Europa	6.40 %	11.09 %
Belgium	Europa	7.77 %	13.65 %
Bermuda	America	11.16 %	10.75 %
Bolivia	America	19.59 %	25.18 %
Bosnia and Herzegovina	Europa	15.93 %	5.25 %
Botswana	Africa	8.08 %	3.78 %
Brazil	America	11.39 %	11.95 %
Bulgaria	Europa	16.41 %	15.96 %
Cambodia	Asia	22.81 %	8.30 %
Cameroon	Africa	19.45 %	6.02 %
Canada	America	8.93 %	4.49 %
Chile	America	28.08 %	10.59 %
China	Asia	15.40 %	12.67 %
Colombia	America	24.54 %	5.44 %
Costa Rica	America	29.42 %	22.36 %
Croatia	Europa	6.21 %	3.50 %
Cuba	America	20.77 %	11.35 %
Cyprus	Asia	12.61 %	5.07 %
Czech Republic	Europa	25.89 %	19.14 %
Denmark	Europa	21.85 %	16.68 %
Djibouti	Africa	25.89 %	19.14 %
Dominica	America	63.37 %	61.96 %
Dominican Republic	America	33.28 %	6.92 %
Ecuador	America	18.60 %	5.52 %
Egypt	Africa	28.16 %	8.76 %
El Salvador	America	42.32 %	3.15 %
Equatorial Guinea	Africa	15.47 %	2.39 %
Estonia	Europa	13.39 %	7.76 %
Ethiopia	Africa	5.53 %	2.80 %
Finland	Europa	7.39 %	4.40 %
France	Europa	11.23 %	6.11 %
French Polynesia	Oceania	5.15 %	2.61 %
Georgia	Asia	35.86 %	17.82 %
Germany	Europa	16.33 %	10.96 %
Ghana	Africa	47.41 %	13.97 %
Gibraltar	Europa	13.43 %	5.33 %
Greece	Europa	6.56 %	4.95 %
Guadaloupe	America	2.77 %	1.51 %
Guatemala	America	10.88 %	1.74 %
Haiti	America	9.38 %	3.88 %
Honduras	America	10.00 %	7.29 %
Hong Kong	Asia	9.86 %	6.33 %
Hungary	Europa	14.33 %	7.44 %
Iceland	Europa	8.36 %	3.70 %
India	Asia	25.54 %	11.97 %
Indonesia	Asia	35.58 %	6.98 %

Cuadro 3.4 — Continúa de la página anterior

País/ Región Administrativa	**Continente**	**% Paquetes TCP**	**% Paquetes ICMP**
Iran Islamic Republic of	Asia	12.41 %	14.59 %
Iraq	Asia	26.50 %	12.98 %
Ireland	Europa	11.15 %	5.96 %
Israel	Asia	11.91 %	14.80 %
Italy	Europa	13.70 %	7.34 %
Jamaica	America	12.42 %	3.20 %
Japan	Asia	14.20 %	9.90 %
Jordan	Asia	9.84 %	3.57 %
Kazakhstan	Asia	34.48 %	15.15 %
Kenya	Africa	10.90 %	10.06 %
Korea Republic of	Asia	14.33 %	26.91 %
Kuwait	Asia	9.19 %	5.05 %
Latvia	Europa	16.19 %	9.46 %
Lebanon	Asia	57.38 %	5.81 %
Libyan Arab Jamahiriya	Africa	2.53 %	16.95 %
Liechtenstein	Europa	28.42 %	12.02 %
Lithuania	Europa	15.36 %	7.06 %
Luxembourg	Europa	11.93 %	6.87 %
Macao	Asia	15.18 %	12.60 %
Macedonia	Europa	15.33 %	3.99 %
Malaysia	Asia	24.25 %	17.80 %
Maldives	Asia	27.16 %	23.20 %
Malta	Europa	8.03 %	6.82 %
Martinique	America	2.68 %	1.15 %
Mauritius	Africa	15.49 %	9.22 %
Mexico	America	32.77 %	22.23 %
Moldova Republic of	Europa	20.89 %	12.25 %
Mongolia	Asia	17.05 %	10.87 %
Montenegro	Europa	7.41 %	3.83 %
Morocco	Africa	13.55 %	9.00 %
Mozambique	Africa	18.65 %	12.58 %
Namibia	Africa	8.62 %	7.23 %
Netherlands	Europa	15.47 %	10.02 %
Netherlands Antilles	America	9.98 %	4.88 %
New Caledonia	Oceania	4.10 %	3.20 %
New Zealand	Oceania	8.68 %	6.29 %
Nicaragua	America	30.08 %	2.63 %
Nigeria	Africa	11.53 %	6.78 %
Norway	Europa	11.55 %	14.40 %
Oman	Asia	11.17 %	1.57 %
Pakistan	Asia	17.72 %	7.04 %
Palestinian Territory	Asia	15.47 %	1.80 %
Panama	America	7.39 %	13.08
Paraguay	America	14.06 %	2.54 %
Peru	America	57.97 %	22.81 %
Philippines	Asia	8.23 %	7.54 %
Poland	Europa	13.86 %	7.10 %
Portugal	Europa	7.14 %	4.37 %
Qatar	Asia	5.09 %	2.13 %
Reunion	Africa	2.75 %	1.01 %

Cuadro 3.4 — Continúa de la página anterior

País/ Región Administrativa	**Continente**	**% Paquetes TCP**	**% Paquetes ICMP**
Romania	Europa	14.50 %	6.76 %
Russian Federation	Asia	17.59 %	13.52 %
Saint Lucia	America	19.99 %	17.50 %
San Marino	Europa	18.96 %	6.39 %
Saudi Arabia	Asia	3.94 %	2.10 %
Senegal	Africa	8.63 %	5.17 %
Serbia	Europa	11.95 %	7.10 %
Seychelles	Africa	22.22 %	20.44 %
Singapore	Asia	32.93 %	31.66
Slovakia	Europa	19.96 %	8.50 %
Slovenia	Europa	9.73 %	6.08 %
South Africa	Africa	9.66 %	5.51 %
Sri Lanka	Asia	11.12 %	6.11 %
Sudan	Africa	9.99 %	3.35 %
Sweden	Europa	8.95 %	4.31 %
Switzerland	Europa	28.52 %	24.75 %
Syrian Arab Republic	Asia	14.02 %	4.89 %
Taiwan	Asia	18.92 %	15.46 %
Tanzania United Republic	Africa	16.06 %	19.66 %
Thailand	Asia	28.72 %	12.97 %
Trinidad and Tobago	America	10.92 %	13.21 %
Tunisia	Africa	24.55 %	8.34 %
Turkey	Asia	12.45 %	9.84 %
Uganda	Africa	27.42 %	19.53 %
Ukraine	Europa	18.88 %	18.67 %
United Arab Emirates	Asia	2.22 %	3.07 %
United Kingdom	Europa	9.01 %	3.99 %
United States	America	34.07 %	19.33 %
Uruguay	America	10.83 %	8.93 %
Uzbekistan	Asia	18.71 %	22.31 %
Venezuela	America	27.38 %	57.53 %
Vietnam	Asia	26.00 %	5.48 %
Yemen	Asia	21.20 %	5.09 %

Como es lógico, el aumento de nodos intermedios que atraviesa un paquete para alcanzar su destino propicia el encuentro de un nodo que impide el paso de paquetes ICMP o TCP. Analizaremos el número de nodos en secciones posteriores para el análisis de retardos, esta sección está orientada únicamente para la elección de la herramienta de identificación de nodos de mayor rendimiento.

En la Figura 3.7 se comparan los rendimientos de `TCPTraceroute` y `traceroute`. El rendimiento de `TCPTraceroute`, que utiliza paquetes TCP para calcular la traza de un objetivo, es del 17,74 % de alcance, mientras que el rendimiento de `traceroute`, que utiliza paquetes ICMP, es de un 10,49 % de alcance. Consigue este alcance, como recordamos, cada dirección IP que logra mediante las herramientas utilizadas identificar todos los nodos intermedios que forman parte de la comunicación.

Se hace patente el efecto de filtrado de los cortafuegos en Internet, con mayor rigurosidad ante paquetes ICMP que paquetes TCP. Este análisis condujo a la determinación de utilizar `TCPTraceroute` para el cálculo de los nodos intermedios que atraviesa un paquete. De esta manera obtendremos más muestras para nuestro estudio que mediante el uso de `traceroute` (ver Figura 3.8).

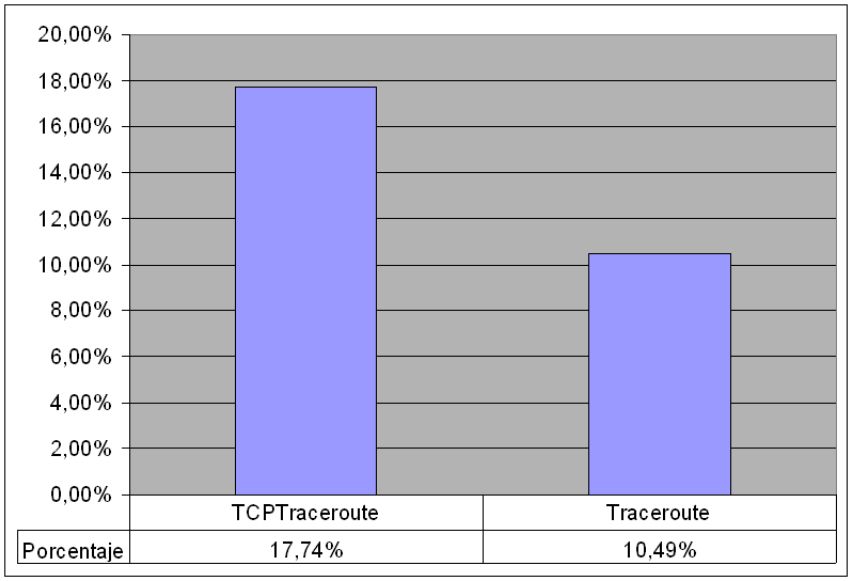


Figura 3.7: Rendimiento de `TCPTraceroute` y `traceroute`.

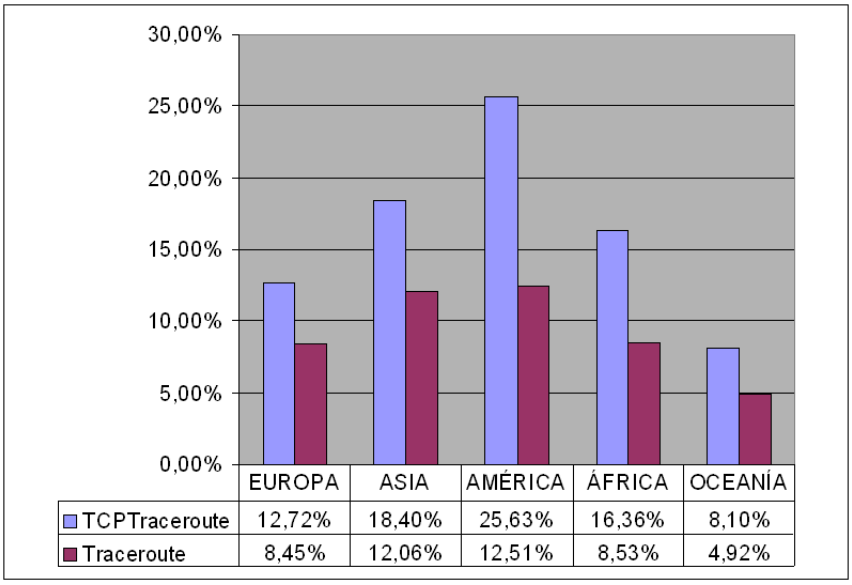


Figura 3.8: Rendimiento de `TCPTraceroute` y `traceroute` por continente.

3.4. Cálculo de la distancia de encaminamiento entre dos extremos

Una vez se tienen identificados los nodos intermedios con su dirección IP, se utilizará la BBDDs InfoDB (ver sección **Técnica de Geolocalización IP empleada** del Capítulo 2, **Revisión del estado del arte sobre Geolocalización IP**) para obtener de cada dirección IP sus coordenadas geográficas (latitud y longitud). Con las coordenadas geográficas de cada nodo se calculará la distancia geográfica de cada tramo, obteniendo como suma de todos los tramos, la distancia enrutada total.

La manera de calcular la distancia geográfica de dos localizaciones a partir de sus coordenadas geográficas es mediante el **modelo de Vincenty** (ver Algoritmo 1) publicado por Thaddeus Vincenty en 1975.

Las fórmulas de Vincenty forman un algoritmo muy eficiente para el cálculo de la distancia entre dos puntos de la superficie de un elipsoide de revolución. Son utilizadas ampliamente

en geodesia para calcular distancias sobre la superficie de la Tierra debido a que requiere un número de operaciones bajo a pesar de dar una precisión de 0.5 mm, mucho mejor que el método tradicional de la fórmula del haverseno usada en trigonometría esférica. Usa un método iterativo.

algorithm 1 Fórmulas de Vincenty

a, b = semiejes mayor y menor del elipsoide
ϕ_1,ϕ_2 = latitud geodética
L = diferencia en longitud
f = achatamiento del elipsoide $(a-b)/a$
s es la distancia (mismas unidades que a, b)
α_1 es el azimuth inicial
α_2 es el azimut final (en dirección p1 $\rightarrow$ p2)

$U_1 = \arctan((1-f)\tan\phi_)$ (U es latitud reducida)
$U_2 = \arctan((1-f)\tan(\phi_2)$
λ = L, λ' = 2π

while $\lambda - \lambda' > 10^{-12}$ (implica un error $<$ 0.06 mm) **do**
$\sin\sigma = \sqrt{(\cos U_2 \sin\lambda)^2 + (\cos U_1 \sin U_2 - \sin U_1 cos U_2 \cos\lambda)^2}$
$\cos\sigma = \sin U_1 \sin U_2 + \cos U_1\ cos U_2 \cos\lambda$
σ = atan2$(\sin\sigma, \cos\sigma)$
$\sin\alpha = \cos U_1 \cos U_2 \sin\lambda / \sin\sigma$
$\cos\alpha^2 = 1 - \sin\alpha^2$
$\cos 2\sigma m = \cos\sigma - 2\sin U_1 \sin U_2 / \cos\alpha^2$
$C = f/16 cos\alpha^2[4 + f(4 - 3\cos\alpha^2)]$
$\lambda = \lambda$'
$\lambda = L + (1-C) f \sin\alpha(\sigma + C\sin\sigma[\cos 2\sigma m + C\cos\sigma(-1 + 2\cos^2 2\sigma m)])$ (úsese $\cos 2\sigma m = 0$ si se está a lo largo del ecuador)
end while

$u^2 = cos\alpha^2(a^2 - b^2)/b^2$
$A = 1 + u^2/16384(4096 + u^2[-768 + u^2(320 - 175u^2)])$
$B = u^2/1024(256 + u^2[-128 + u^2(74 - 47u^2)])$
$\Delta\sigma = B\sin\sigma(\cos 2\sigma m + B/4\cos\sigma(-1 + 2cos^2 2\sigma m) - B/6 cos 2\sigma m(-3 + 4\sin\sigma^2)(-3 + 4\cos 2\sigma m^2)])$
$s = bA(\sigma - \Delta\sigma)$
$\alpha_1 = atan2(\cos U_2 \sin\lambda, \cos U_1 \sin U_2 - \sin U_1 \cos U_2 \cos\lambda)$
$\alpha_2 = atan2(\cos U_1 \sin\lambda, -\sin U_1 \cos U_2 + \cos U_1 \sin U_2 \cos\lambda)$

El elipsoide más extendido es el **WGS84** (World Geodetic System 84) (ver Algoritmo 2). El WGS84 es un sistema de coordenadas cartográficas mundial que permite localizar cualquier punto de la Tierra por medio de tres unidades dadas.

algorithm 2 Unidades del elipse de revolución

a = 6378137 m (± 2 m)
b = 6356752.3142 m
f = 1 / 298.257223563

El WGS84 tuvo varias revisiones (la última en 2004), y se estima un error de cálculo menor a 2 cm. por lo que es utilizado por el **Sistema de Posicionamiento Global (GPS).**

3.5. Propuesta de diferentes métricas para estudiar el compromiso entre calidad de conexión y demandas de ancho de banda entre los destinos más populares

3.5.1. Métrica para el cálculo del Path Inflation

La métrica que utilizaremos para calcular el Path Inflation de una dirección IP comprende la distancia geográfica y la distancia enrutada entre la dirección IP y el origen de las transmisiones, que es Madrid.

$$PI_k = \frac{d_{enrutada,k}}{d_{geografica,k}} \tag{3.1}$$

donde:

- $PI_k \equiv$ PI del destino k
- $d_{enrutada,k} \equiv$ distancia enrutada del destino k
- $d_{geografica,k} \equiv$ distancia geográfica del destino k

Esta métrica es propuesta inicialmente por (SPK01) y es un ratio entre la distancia enrutada y la distancia geográfica. La imagen de esta función está acotada inferiormente por uno (aquellos caminos enrutados de distancia igual a la distancia mínima, que es la distancia geográfica que separa dos localizaciones). Para valores de esta función superiores a uno, tomará por valor el número de veces que la distancia enrutada supera a la distancia geográfica.

3.5.2. Métrica para el cálculo del Path Inflation en función del tráfico

La métrica que utilizaremos para calcular el Path Inflation en función del tráfico de una dirección IP comprende la distancia geográfica y la distancia enrutada entre la dirección IP y el origen de las transmisiones, que es Madrid, y el tráfico recibido por esa dirección IP. Esta métrica es normalizada por el tráfico total envíado por RedIRIS en los días en los que se basa la adquisición de direcciones IP (destinos) y que es cómo se refleja en el pie de la Tabla 3.2 de unos 985 TeraBytes.

$$PIT_k = \frac{PI_k \cdot T_{recibido,k}}{T_{total}} \tag{3.2}$$

donde:

- $PIT_k \equiv$ PIT del destino k
- $PI_k \equiv$ PI del destino k
- $T_{recibido,k} \equiv$ Tráfico recibido en el destino k
- $T_{total} \equiv$ Tráfico total recibido en todos los destinos útiles (985 TeraBytes)

La métrica que se propone es similar a la del PI original, ponderada por el número de Bytes que recibe una localización. La imagen de esta función está acotada superiormente por uno, ya que dicha función está normalizada por el número de Bytes enviados totales. El objetivo de esta métrica es relacionar el PI con la popularidad de los objetivos, aumentando el resultado de la función con aquellos destinos con mayor tasa de tráfico, síntoma de que el PI es más relevante en estas localizaciones. Por tanto, esta métrica será útil para clasificar la mayor o menor relevancia del PI en función del tráfico, y será denominada de aquí en adelante **PIT (Path Inflation Traffic)**.

4

Pruebas y resultados sobre el Path Inflation

4.1. Introducción

El objetivo de este capítulo será caracterizar la severidad del PI teniendo en cuenta el tráfico. Para esto, será calculado en primera instancia el PI según la Ecuación 3.1 y a continuación el PIT según la Ecuación 3.2.

Estudiaremos la relación entre ambas métricas comprendiendo la aportación del tráfico recibido. Para ello se hará un estudio previo del PIT para comprenderlo, y posteriormente se realizará un estudio más preciso para clasificar los países en función del PIT obtenido.

4.2. Análisis del Path Inflation

4.2.1. Introducción

El PI (ver Ecuación 3.1) es calculado para cada destino de RedIRIS. La cuantificación de este efecto será a nivel de país/región administrativa por lo que se obtendrán conjuntos de muestras de PI a dicho nivel. El tamaño muestral de cada país/región administrativa es el propio de los diferentes destinos o direcciones IP identificadas del exportador de Madrid (ver sección **Origen de los emplazamientos seleccionados** del Capítulo 3, **Diseño y desarrollo**).

4.2.2. Estadística del Path Inflation

La estadística del PI medio ($\overline{PI}$) calculado por país/región administrativa se muestra en la tabla 4.1.

Cuadro 4.1: Estadística del PI.

País	**Cont.**	$\overline{PI}$
Aland Islands	Europa	5.64
Albania	Europa	5.03
Algeria	Africa	19.22
Andorra	Europa	44.68
Angola	Africa	4.14
Antarctica	Antartida	1.23

Cuadro 4.1 — Continúa de la página anterior

País	**Cont.**	$\overline{PI}$
Antigua and Barbuda	America	3.79
Argentina	America	2.32
Armenia	Asia	5.13
Aruba	America	3.68
Australia	Oceania	1.66
Austria	Europa	4.78
Azerbaijan	Asia	4.76
Bahamas	America	1.40
Bahrain	Africa	4.65
Bangladesh	Asia	2.54
Barbados	America	3.21
Belarus	Europa	6.68
Belgium	Europa	9.51
Bermuda	America	2.80
Bolivia	America	2.38
Bosnia and Herzegovina	Europa	5.69
Botswana	Africa	2.27
Brazil	America	2.68
Bulgaria	Europa	3.30
Cambodia	Asia	2.21
Cameroon	Africa	4.87
Canada	America	2.40
Chile	America	2.23
China	Asia	1.89
Colombia	America	2.84
Costa Rica	America	1.78
Croatia	Europa	3.46
Cuba	America	3.73
Cyprus	Asia	4.64
Czech Republic	Europa	6.90
Denmark	Europa	6.69
Djibouti	Africa	3.78
Dominica	America	3.93
Dominican Republic	America	1.92
Ecuador	America	2.37
Egypt	Africa	6.61
El Salvador	America	1.30
Equatorial Guinea	Africa	4.62
Estonia	Europa	3.38
Ethiopia	Africa	4.11
Finland	Europa	3.95
France	Europa	9.83
French Polynesia	Oceania	1.25
Georgia	Asia	5.69
Germany	Europa	7.84
Ghana	Africa	4.24
Gibraltar	Europa	2.40
Greece	Europa	5.54
Guadaloupe	America	2.90
Guatemala	America	1.31

Cuadro 4.1 — Continúa de la página anterior

País	**Cont.**	$\overline{PI}$
Haiti	America	1.53
Honduras	America	1.72
Hong Kong	Asia	1.85
Hungary	Europa	2.71
Iceland	Europa	5.28
India	Asia	2.69
Indionesia	Asia	2.05
Iran Islamic Republic of	Asia	4.49
Iraq	Asia	2.62
Ireland	Europa	5.84
Israel	Asia	3.82
Italy	Europa	10.77
Jamaica	America	2.55
Japan	Asia	1.75
Jordan	Asia	3.48
Kazakhstan	Asia	3.73
Kenya	Africa	3.76
Korea Republic of	Asia	2.35
Kuwait	Asia	4.74
Latvia	Europa	4.35
Lebanon	Asia	5.28
Libyan Arab Jamahiriya	Africa	9.61
Liechtenstein	Europa	10.41
Lithuania	Europa	3.67
Luxembourg	Europa	10.79
Macao	Asia	1.94
Macedonia	Europa	1.43
Malaysia	Asia	1.65
Maldives	Asia	2.11
Malta	Europa	7.25
Martinique	America	3.91
Mauritius	Africa	1.94
Mexico	America	1.35
Moldova Republic of	Europa	2.62
Mongolia	Asia	1.48
Montenegro	Europa	8.06
Morocco	Africa	22.27
Mozambique	Africa	1.8
Namibia	Africa	3.19
Netherlands	Europa	7.13
Netherlands Antilles	America	1.87
New Caledonia	Oceania	2.23
New Zealand	Oceania	1.35
Nicaragua	America	1.54
Nigeria	Africa	6.02
Norway	Europa	4.56
Oman	Asia	3.6
Pakistan	Asia	3.34
Palestinian Territory	Asia	5.02
Panama	America	1.78

Cuadro 4.1 — Continúa de la página anterior

País	**Cont.**	$\overline{PI}$
Paraguay	America	3.19
Peru	America	2.54
Philippines	Asia	1.99
Poland	Europa	5.76
Portugal	Europa	23.68
Qatar	Asia	4.57
Reunion	Africa	2.45
Romania	Europa	5.64
Russian Federation	Asia	4.82
Saint Lucia	America	3.75
San Marino	Europa	12.28
Saudi Arabia	Asia	3.59
Senegal	Africa	4.78
Serbia	Europa	4.75
Seychelles	Africa	2.85
Singapore	Asia	2.07
Slovakia	Europa	5.07
Slovenia	Europa	3.15
South Africa	Africa	2.89
Sri Lanka	Asia	2.75
Sudan	Africa	4.62
Sweden	Europa	4.50
Switzerland	Europa	4.49
Syrian Arab Republic	Asia	2.72
Taiwan	Asia	2.30
Tanzania United Republic	Africa	3.62
Thailand	Asia	2.30
Trinidad and Tobago	America	1.96
Tunisia	Africa	7.73
Turkey	Asia	5.88
Uganda	Africa	3.48
Ukraine	Europa	3.89
United Arab Emirates	Asia	3.91
United Kingdom	Europa	7.89
United States	America	1.49
Uruguay	America	2.51
Uzbekistan	Asia	4.25
Venezuela	America	1.94
Vietnam	Asia	1.65
Yemen	Asia	3.75

El PI como se definió en la Ecuación 3.1, es directamente proporcional al camino enrutado e inversamente proporcional a la distancia geográfica que separa dos emplazamientos, en nuestro caso, Madrid y el objetivo destino. Se ha observado que con frecuencia, la ruta que siguen los paquetes desde Madrid, toma como salida un enlace de Estados Unidos para posteriormente ser direccionados a su destino. Siguiendo esta premisa, los países cercanos a Madrid, obtendrán un PI mayor al tener una distancia geográfica pequeña en comparación a países lejanos. En la Figura C.1 se observa este hecho.

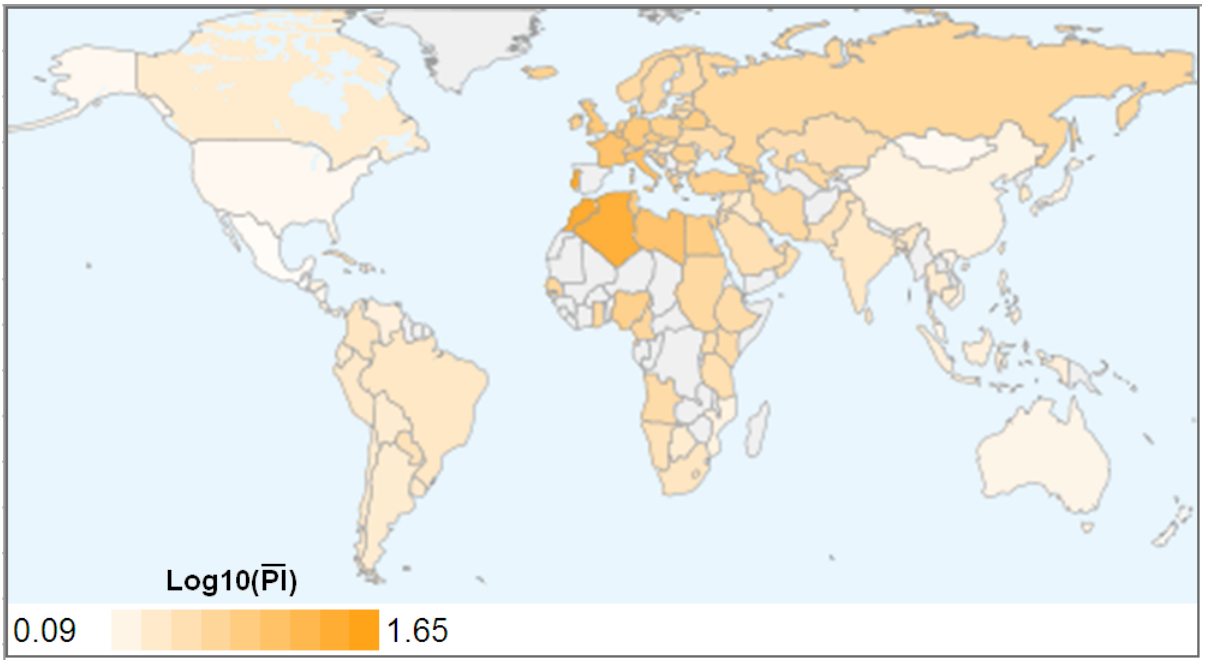


Figura 4.1: $\overline{PI}$ de cada país en escala logarítmica.

4.2.3. Estudio del Path Inflation

En esta sección se analizará con detalle el PI en varios continentes para comprender el efecto comentado en la sección anterior. Estudiaremos las situaciones más destacables, que se encuentran en los continentes de Europa, Sudamérica, África y Asia.

Para observar los caminos enrutados he desarrollado una aplicación en Matlab que visualiza la ruta que siguen los paquetes. La efectividad de dicha aplicación ha sido comparada con herramientas del mercado como `Visual Trace Route Tool` [1] obteniendo resultados similares.

[1] Esta herramienta de visualizado de rutas es propiedad de **You Get Signal** `http://www.yougetsignal.com/tools/visual-tracert/`. Calcula la ruta identificando la localización de los nodos intermedios mediante una BBDDs de GeoIP de MaxMind. Finalmente se representan dichos nodos intermedios en cartografías de Google Maps `http://maps.google.es/`

PI en Europa

En la Figura 4.2 se observa un $\overline{PI}$ alto en Europa, ya que con frecuencia se direcciona el tráfico hacia Estados Unidos para luego volver al contiente europeo aumentando en demasía el camino enrutado. Se puede distiguir un patrón por el cuál los destinos cercanos a España obtienen mayor $\overline{PI}$.

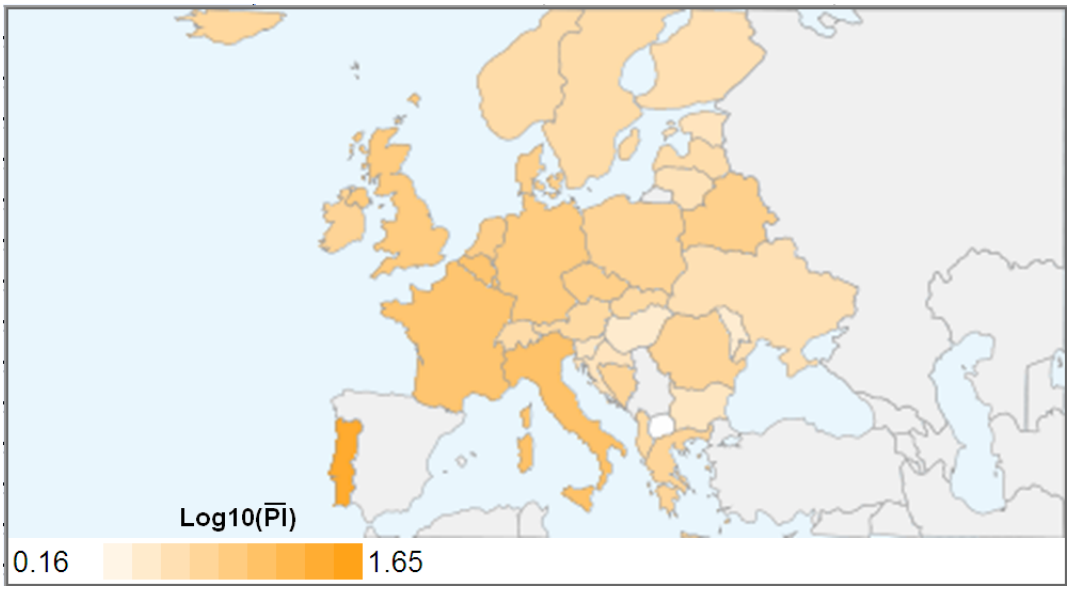


Figura 4.2: $\overline{PI}$ en Europa en escala logarítmica.

He considerado oportuno mostrar dos rutas típicas con mi herramienta de visualizado de rutas. En concreto he elegido una traza a Francia (ver Figura 4.3) y otra a Alemania (ver Figura 4.4). Queda constancia en muchas rutas calculadas con la herramienta desarrollada el uso de Estados Unidos como enlace entre Madrid y el destino objetivo.

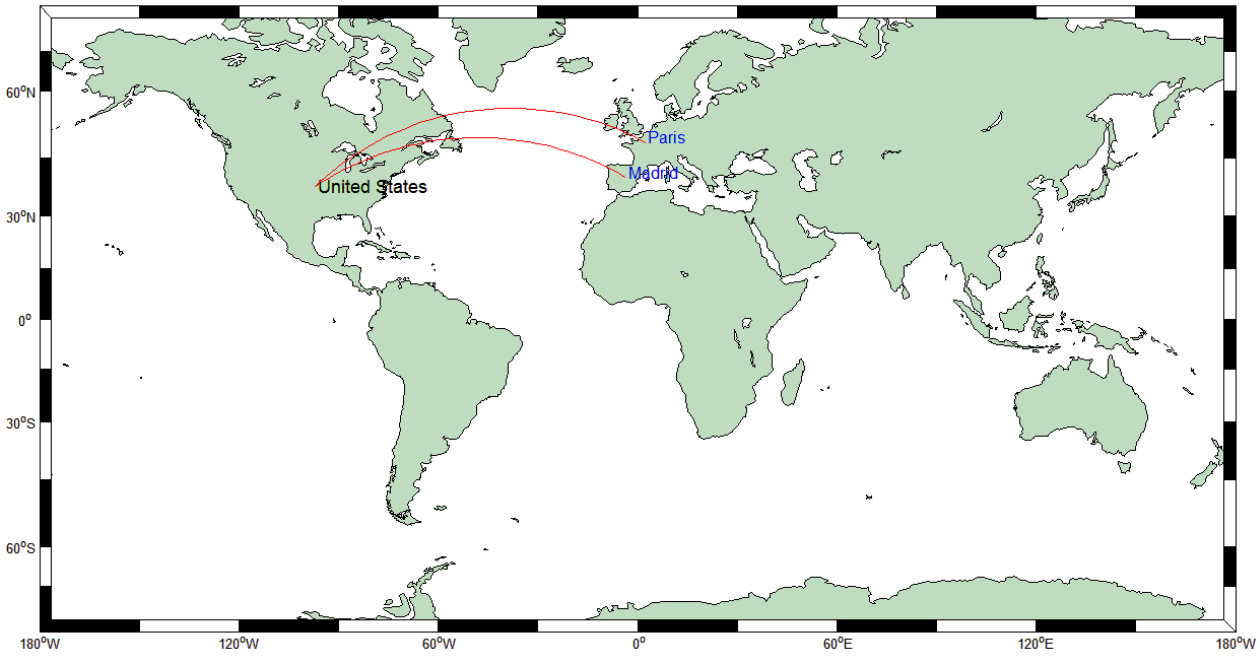


Figura 4.3: Ruta entre Madrid y una IP de París (Francia). Se observa el paso por Estados Unidos desde Madrid para llegar a su destino.

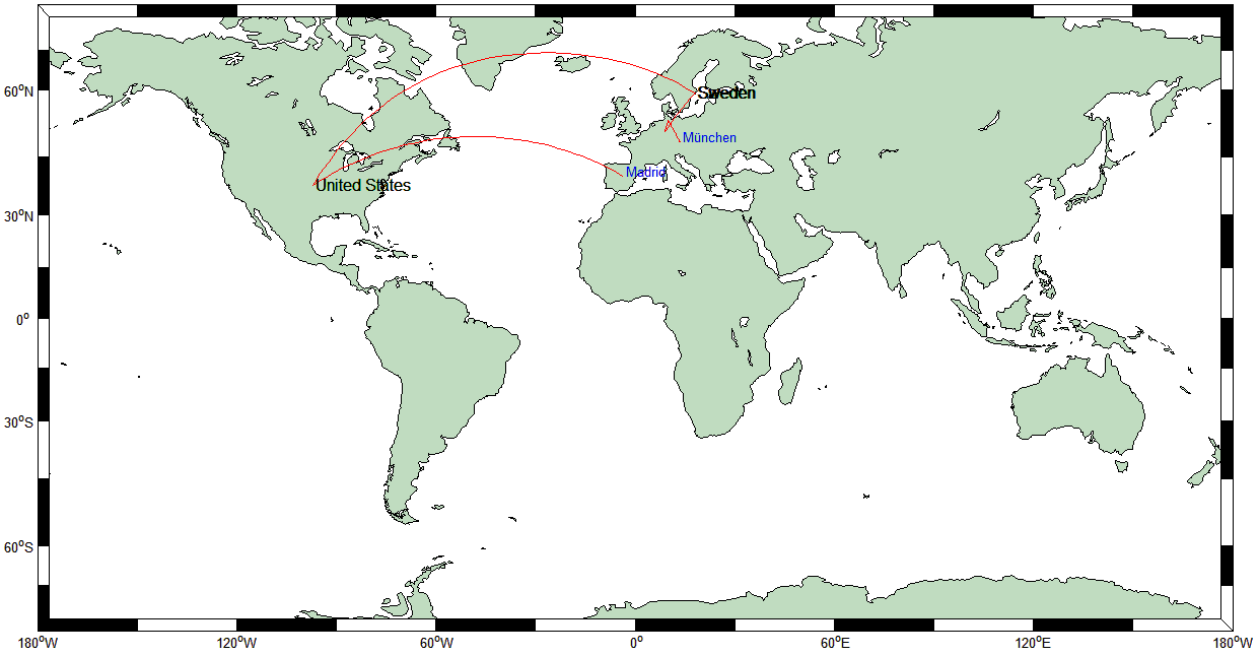


Figura 4.4: Ruta entre Madrid y una IP de Munich (Alemania). Se observa el paso por Estados Unidos desde Madrid para llegar a su destino.

PI en Sudamérica

He considerado suficiente analizar los países de América del Sur, ya que los del Norte no tiene un PI alto, ya que se encuentran próximos a Estados Unidos. El encontrarse cerca de Estados Unidos disminuye el PI, ya que los países americanos toman la ruta desde Madrid a Estados Unidos y desde aquí a su localización. En la Figura 4.5, se observa el $\overline{PI}$ en estos países con un rango en escala natural. La representación en esta escala natural es mejor que en escala logarítmica al exisitir poca variabilidad del $\overline{PI}$. Aún así se incluye la Figura 4.6 con el $\overline{PI}$ en escala logarítmica para poder realizar comparaciones con los demás continentes.

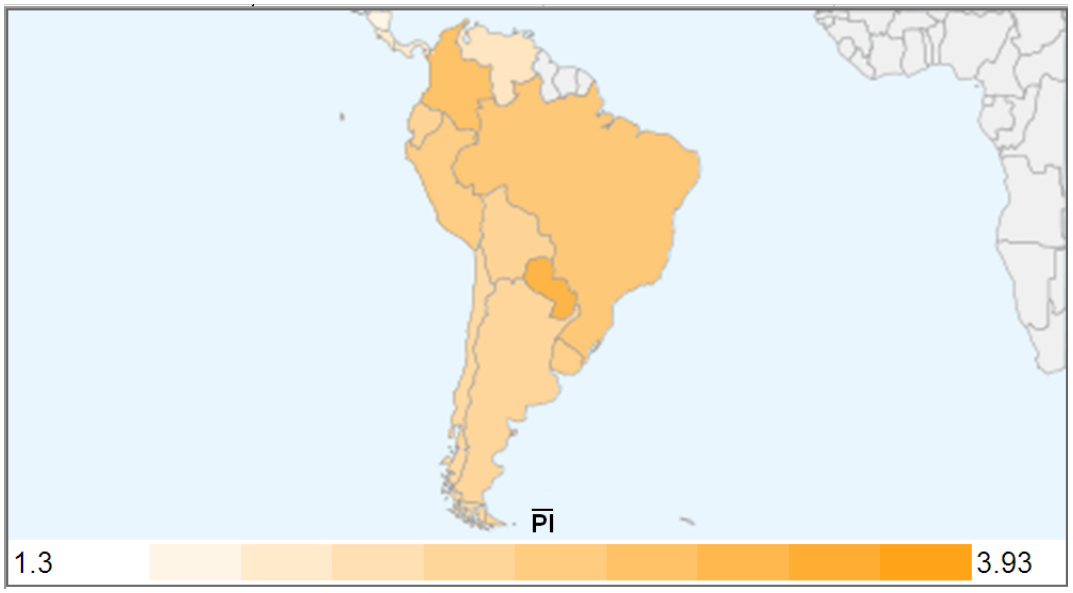


Figura 4.5: $\overline{PI}$ en América del Sur en escala natural

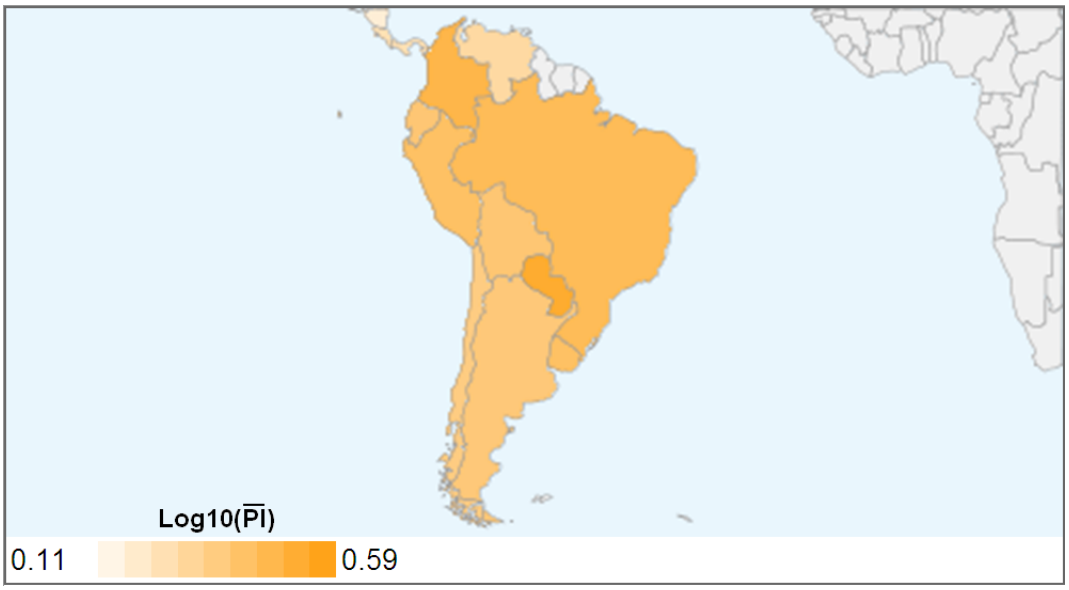


Figura 4.6: $\overline{PI}$ en América del Sur en escala logarítmica

Para comprobar esta característica de las rutas de los países de América del Sur he incluido una traza a Argentina (ver Figura 4.7) y otra a Colombia (ver Figura 4.8). Normalmente se utiliza Estados Unidos como enlace entre Madrid y el destino objetivo. Este hecho no aumenta tan significativamente el PI como en otros continentes.

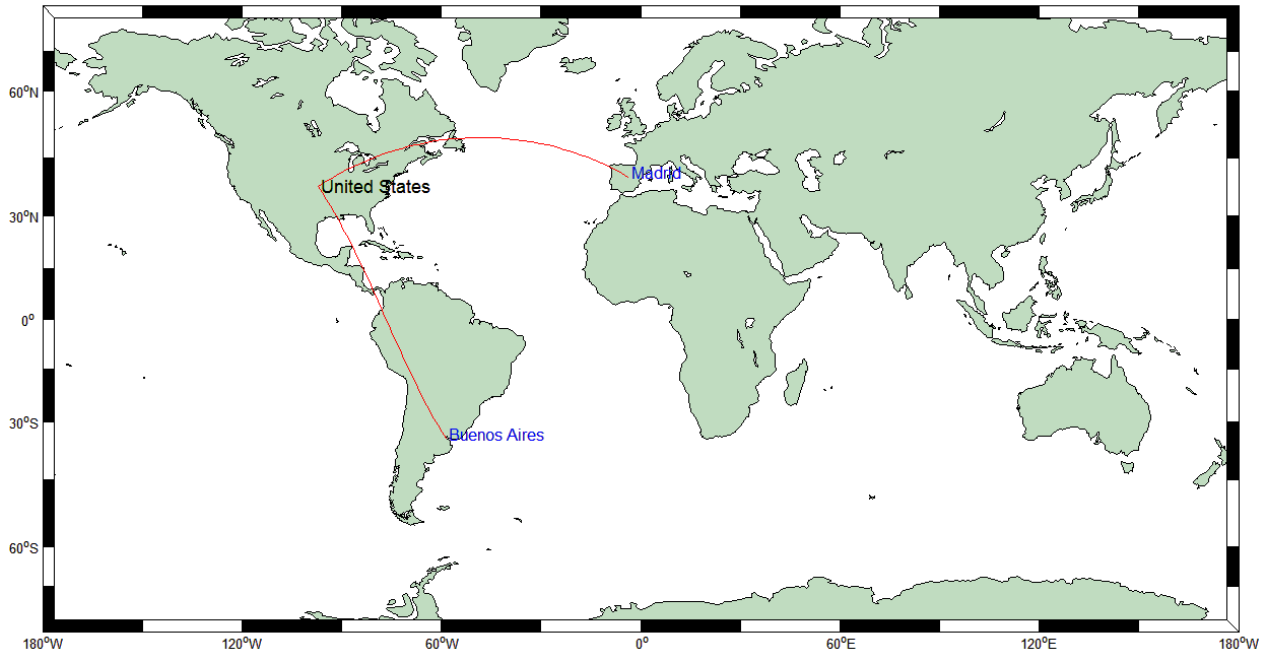


Figura 4.7: Ruta entre Madrid y una IP de Buenos Aires (Argentina). Se observa el paso por Estados Unidos desde Madrid para llegar a su destino.

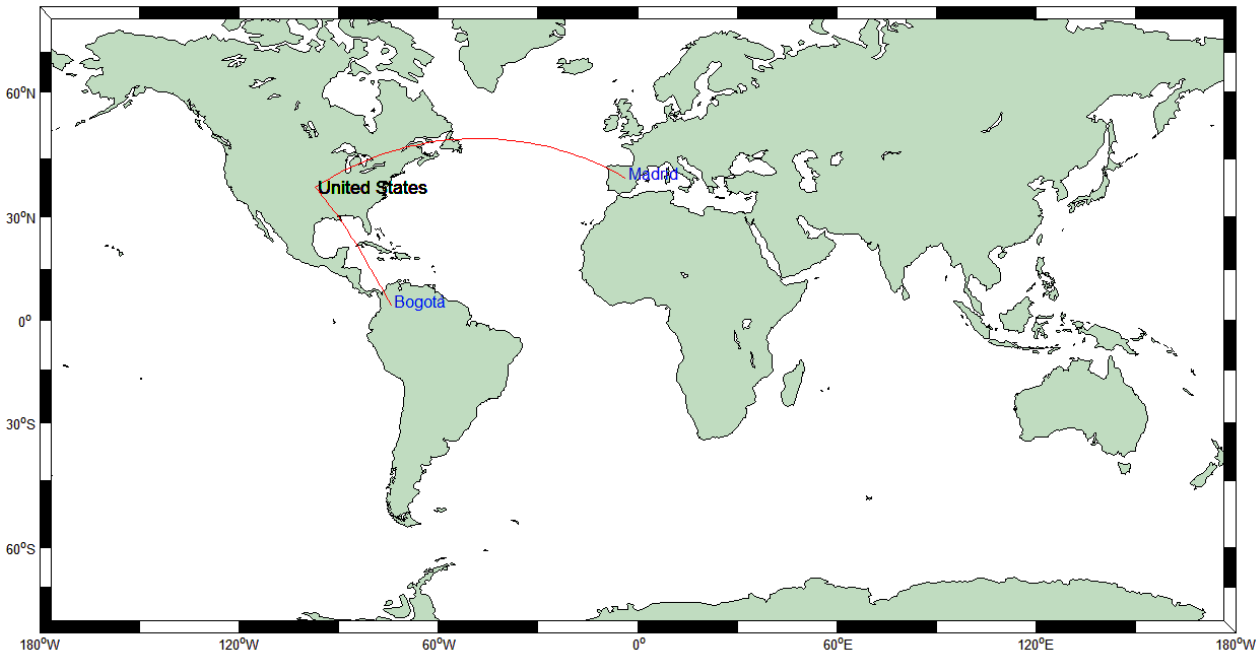


Figura 4.8: Ruta entre Madrid y una IP de Bogotá (Colombia). Se observa el paso por Estados Unidos desde Madrid para llegar a su destino.

PI en África

El comportamiento del PI en el continente africano sigue con frecuencia el siguiente patrón: desde Madrid utiliza el enlace de Estados Unidos para nuevamente alcanzar Europa y desde ahí ser enrutado al continente africano. Por norma general, los países cercanos a Madrid obtienen mayor $\overline{PI}$ que países lejanos (ver Figura 4.9).

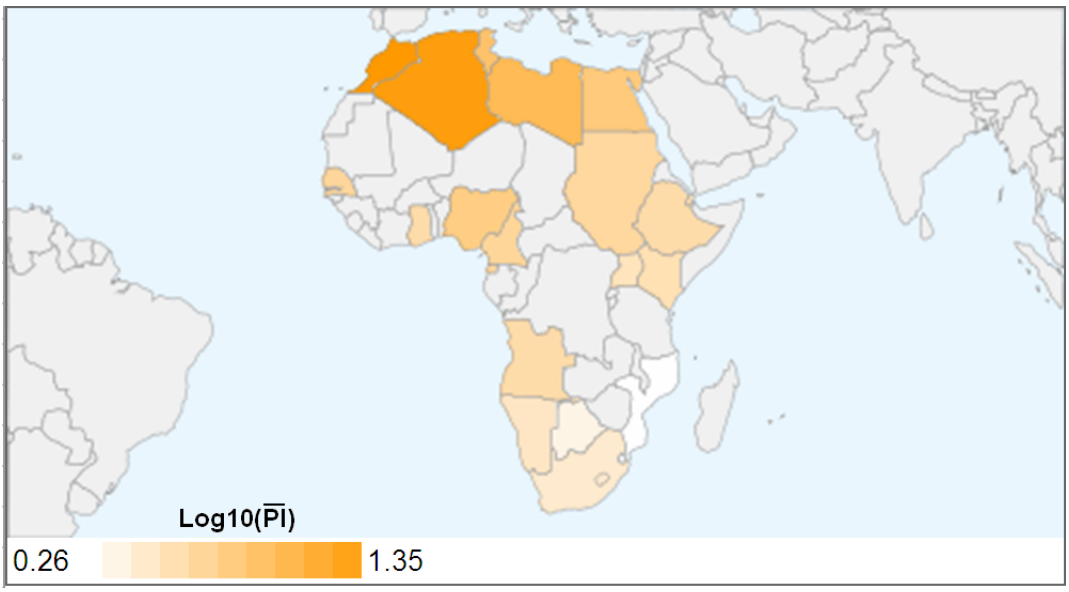


Figura 4.9: $\overline{PI}$ en África en escala logarítmica.

Para observar este patrón he incluido una traza a Angola (ver Figura 4.10) y otra a Sudáfrica (ver Figura 4.11).

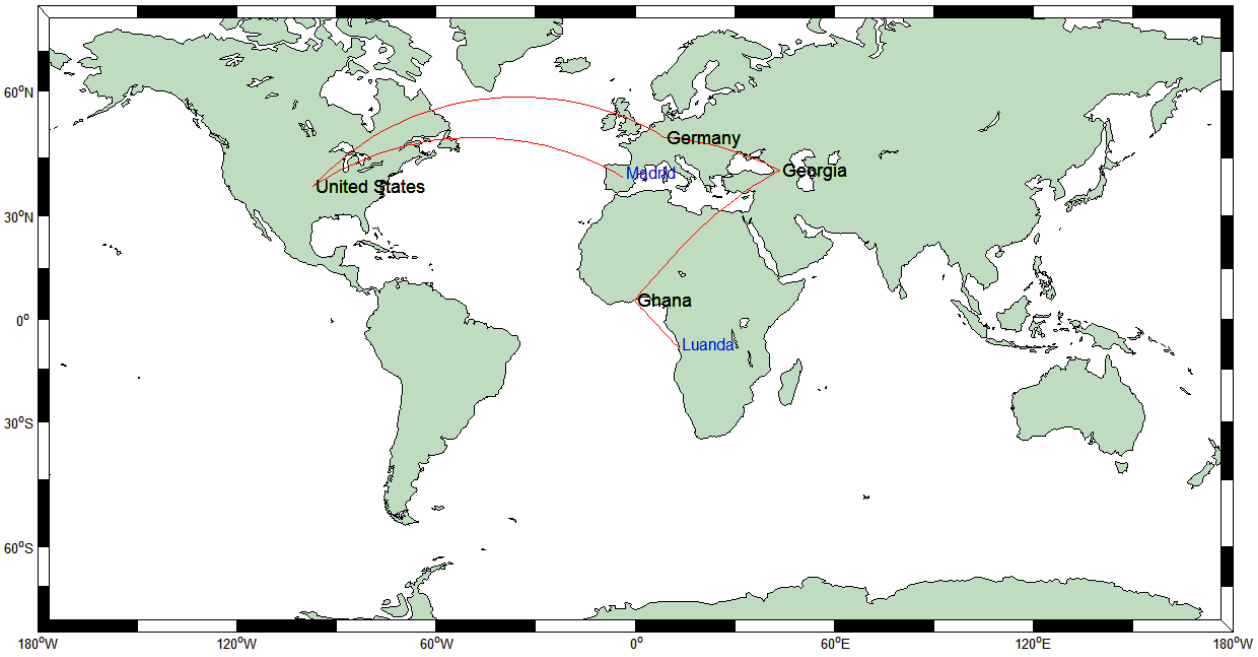


Figura 4.10: Ruta entre Madrid y una IP de Luanda (Angola). Se observa el paso por Estados Unido desde Madrid para regresar a Europa y desde ahí llegar a su destino

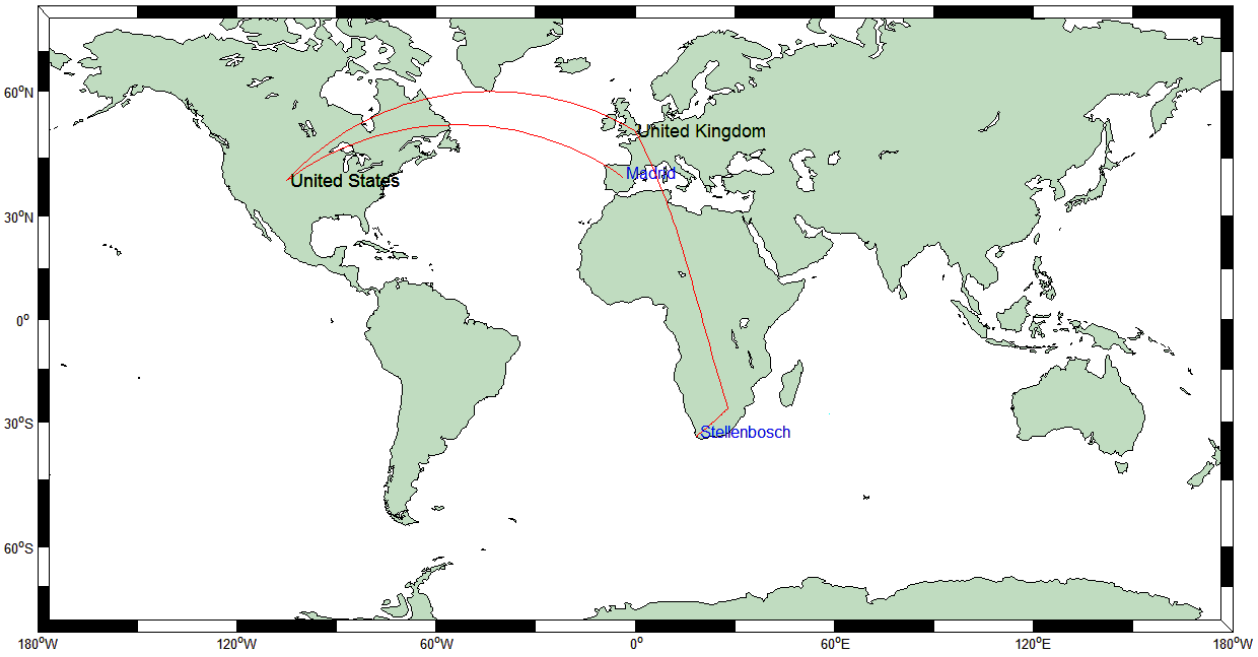


Figura 4.11: Ruta entre Madrid y una IP de StellenBosch (Sudáfrica). Se observa el paso por Estados Unido desde Madrid para regresar a Europa y desde ahí llegar a su destino.

PI en Asia

En este continente, el $\overline{PI}$ no varía tanto entre los países ya que en la ecuación del PI (ver Ecuación 3.1), tanto el numerador como el denominador aumentan en proporción según su distancia con España (ver Figura 4.12).

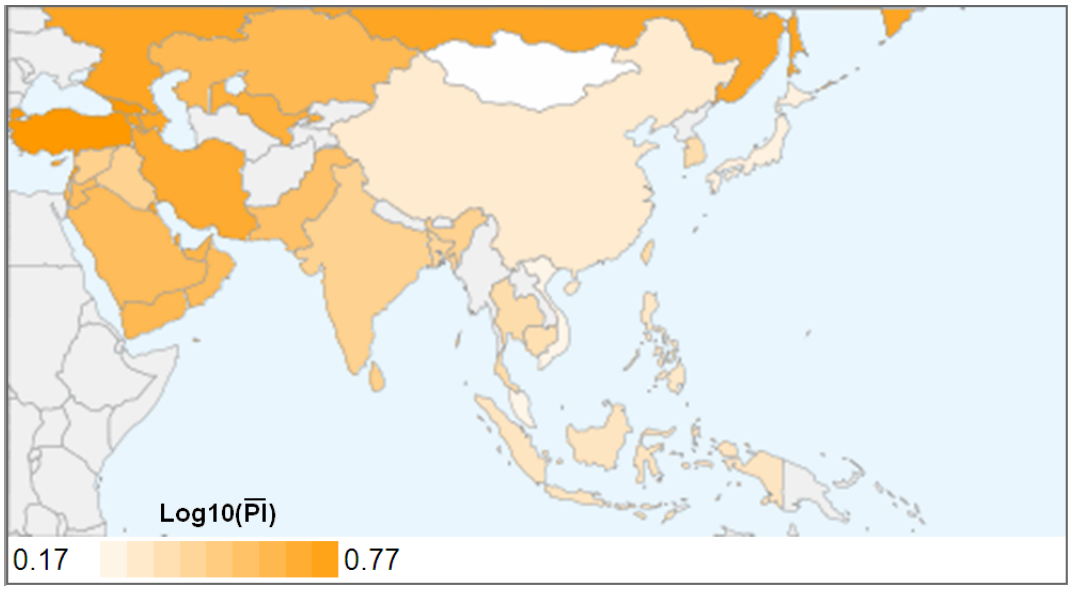


Figura 4.12: $\overline{PI}$ en Asia en escala logarítmica.

Para entender lo expuesto anteriormente con mayor claridad he incluido dos rutas generadas con mi herramienta de visualizado de rutas (ver Figura 4.13 y Figura 4.14).

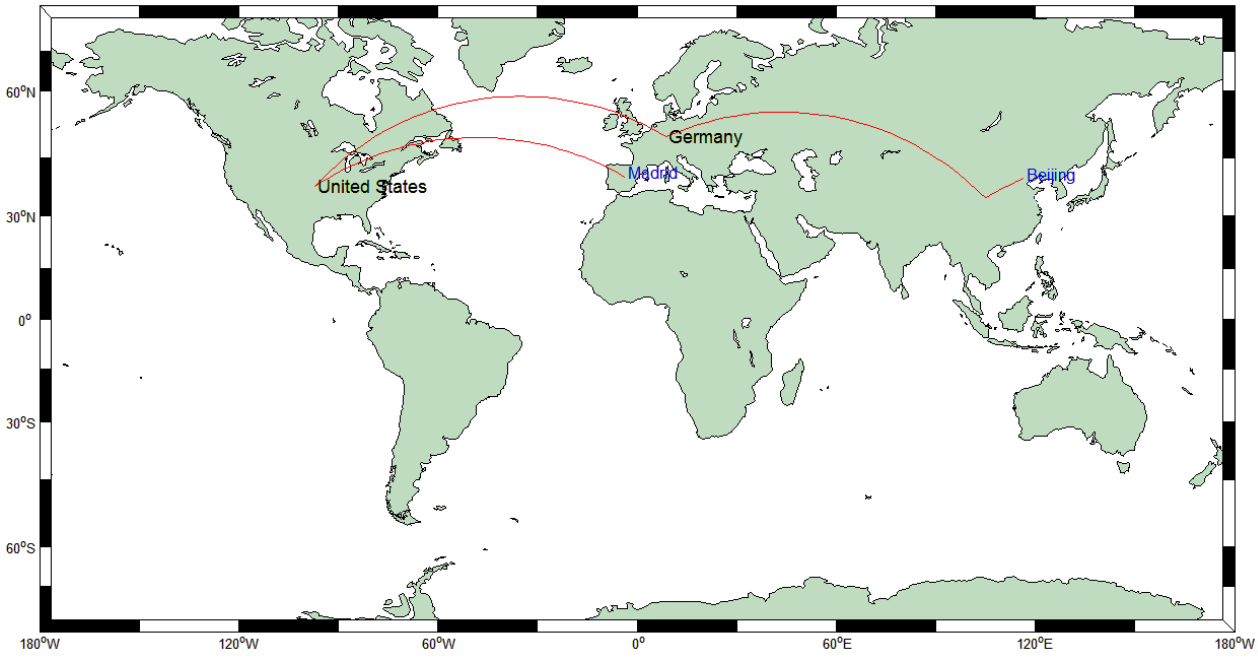


Figura 4.13: Ruta entre Madrid y una IP de Beijing (China).

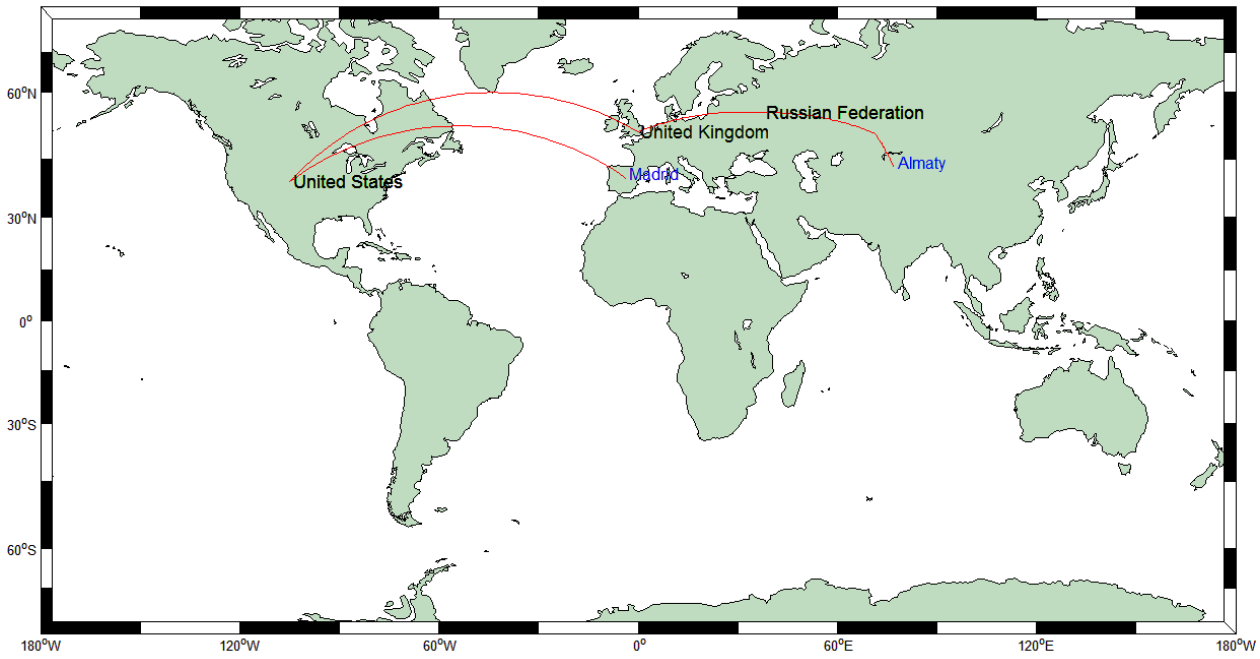


Figura 4.14: Ruta entre Madrid y una IP de Almaty (Kazakhstan).

4.3. Análisis del Path Inflation Traffic

4.3.1. Introducción

El objetivo de esta sección es poder identificar aquellos emplazamientos cuyo PIT es crítico. Para ello se aplicará la métrica de la Ecuación 4.1 para cada país destino de RedIRIS. Se hará un análisis cualitativo del PIT basado en la observación de dicha métrica en mapas. Posteriormente se cuantificará y se clasificará el PIT con mayor precisión.

$$\overline{PIT_k} = \frac{\overline{PI_k} \cdot T_{recibido,k}}{T_{total}} \tag{4.1}$$

donde:

- $\overline{PIT_k} \equiv \overline{PIT}$ del país k
- $\overline{PI_k} \equiv \overline{PI}$ del país k
- $T_{recibido} \equiv$ tráfico recibido por el país k. Se encuentra calculado en la columna **GB recibidos** de la Tabla 3.2.
- $T_{total} \equiv$ Tráfico total recibido en todos los destinos útiles (985 TeraBytes)

4.3.2. Estadística del Path Inflation Traffic

La estadística del Path Inflation Traffic calculado por país/región administrativa se observa en la Tabla 4.2. En dicha tabla se observa el PIT medio ($\overline{PIT}$) multiplicado por 10^6. Hay que comprender que el PIT obtiene un valor siempre menor que uno (ver Ecuación 3.2).

Cuadro 4.2: Estadística del Path Inflation Traffic.

País	**Cont.**	$\overline{PIT}$ x 10^6
Aland Islands	Europa	76.44
Albania	Europa	161.44
Algeria	Africa	5569.45
Andorra	Europa	18404.18
Angola	Africa	117.60
Antarctica	Antartida	1.98
Antigua and Barbuda	America	18.50
Argentina	America	36747.11
Armenia	Asia	180.48
Aruba	America	33.86
Australia	Oceania	3836.89
Austria	Europa	52887.04
Azerbaijan	Asia	139.22
Bahamas	America	14.70
Bahrain	Africa	85.75
Bangladesh	Asia	121.52
Barbados	America	82.69
Belarus	Europa	396.26
Belgium	Europa	147100.05
Bermuda	America	23.76
Bolivia	America	3892.21
Bosnia and Herzegovina	Europa	429.78
Botswana	Africa	17.29
Brazil	America	21800.57
Bulgaria	Europa	1968.11
Cambodia	Asia	23.96
Cameroon	Africa	45.91
Canada	America	23359.65
Chile	America	27947.36
China	Asia	16232.05
Colombia	America	45116.84
Costa Rica	America	2309.22
Croatia	Europa	2259.06
Cuba	America	1418.82
Cyprus	Asia	1811.83
Czech Republic	Europa	12841.81
Denmark	Europa	17180.96
Djibouti	Africa	20.61
Dominica	America	13.28
Dominican Republic	America	2284.82
Ecuador	America	8230.92
Egypt	Africa	3180.17
El Salvador	America	1163.61
Equatorial Guinea	Africa	59.56
Estonia	Europa	1967.38
Ethiopia	Africa	39.97
Finland	Europa	8773.09
France	Europa	746528.69
French Polynesia	Oceania	7.58

Cuadro 4.2 — Continúa de la página anterior

País	**Cont.**	$\overline{PIT}$ x 10^6
Georgia	Asia	280.45
Germany	Europa	765760.25
Ghana	Africa	90.62
Gibraltar	Europa	100.17
Greece	Europa	18871.21
Guadaloupe	America	107.81
Guatemala	America	1751.09
Haiti	America	13.62
Honduras	America	862.50
Hong Kong	Asia	968.20
Hungary	Europa	41384.45
Iceland	Europa	3133.98
India	Asia	5120.05
Indionesia	Asia	760.52
Iran Islamic Republic of	Asia	1638.88
Iraq	Asia	38.15
Ireland	Europa	11930.54
Israel	Asia	4093.27
Italy	Europa	567884.55
Jamaica	America	151.94
Japan	Asia	7364.40
Jordan	Asia	271.54
Kazakhstan	Asia	229.82
Kenya	Africa	177.40
Korea Republic of	Asia	23527.53
Kuwait	Asia	432.16
Latvia	Europa	956.88
Lebanon	Asia	583.96
Libyan Arab Jamahiriya	Africa	179.20
Liechtenstein	Europa	67.30
Lithuania	Europa	1244.88
Luxembourg	Europa	9419.49
Macao	Asia	42.15
Macedonia	Europa	121.01
Malaysia	Asia	921.63
Maldives	Asia	10.48
Malta	Europa	2893.77
Martinique	America	82.35
Mauritius	Africa	36.26
Mexico	America	35455.70
Moldova Republic of	Europa	171.57
Mongolia	Asia	32.02
Montenegro	Europa	292.52
Morocco	Africa	30851.45
Mozambique	Africa	20.62
Namibia	Africa	26.72
Netherlands	Europa	97169.28
Netherlands Antilles	America	42.61
New Caledonia	Oceania	62.75
New Zealand	Oceania	502.57

Cuadro 4.2 — Continúa de la página anterior

País	**Cont.**	$\overline{PIT}$ x 10^6
Nicaragua	America	696.93
Nigeria	Africa	192.72
Norway	Europa	16816.03
Oman	Asia	46.87
Pakistan	Asia	947.57
Palestinian Territory	Asia	121.81
Panama	America	1684.90
Paraguay	America	2051.43
Peru	America	24310.04
Philippines	Asia	1064.13
Poland	Europa	28896.98
Portugal	Europa	246569.34
Qatar	Asia	417.56
Reunion	Africa	133.50
Romania	Europa	13448.08
Russian Federation	Asia	235507.23
Saint Lucia	America	30.55
San Marino	Europa	184.99
Saudi Arabia	Asia	2778.82
Senegal	Africa	269.17
Serbia	Europa	2318.33
Seychelles	Africa	52.01
Singapore	Asia	1259.36
Slovakia	Europa	3203.46
Slovenia	Europa	1442.48
South Africa	Africa	692.78
Sri Lanka	Asia	114.96
Sudan	Africa	93.36
Sweden	Europa	91000.98
Switzerland	Europa	245277.11
Syrian Arab Republic	Asia	162.71
Taiwan	Asia	89492.17
Tanzania United Republic	Africa	30.34
Thailand	Asia	976.35
Trinidad and Tobago	America	172.96
Tunisia	Africa	1177.26
Turkey	Asia	8497.37
Uganda	Africa	23.63
Ukraine	Europa	2973.19
United Arab Emirates	Asia	1347.49
United Kingdom	Europa	753136.83
United States	America	384550.90
Uruguay	America	6110.50
Uzbekistan	Asia	34.95
Venezuela	America	15528.99
Vietnam	Asia	466.03
Yemen	Asia	64.90

En la figura 4.15 se observa el $\overline{PIT}$ en cada país en escala logarítmica. Para este análisis previo, el $\overline{PIT}$ es multiplicado por 10^6 para obtener valores positivos en escalas logarítmicas. Volviendo al análisis previo, por un lado se puede identificar un $\overline{PIT}$ alto en páises tales como Italia, Alemania, Estados Unidos, Francia, Rusia, etc, países con alto tráfico procedente desde Madrid como se puede observar en la Figura 3.5 y en la Figura 3.6 y que obtienen mayor $\overline{PIT}$. Por otro lado, se puede observar el efecto contrario, países con bajo PIT como la mayoría de países procedentes de África y algunos de Asia.

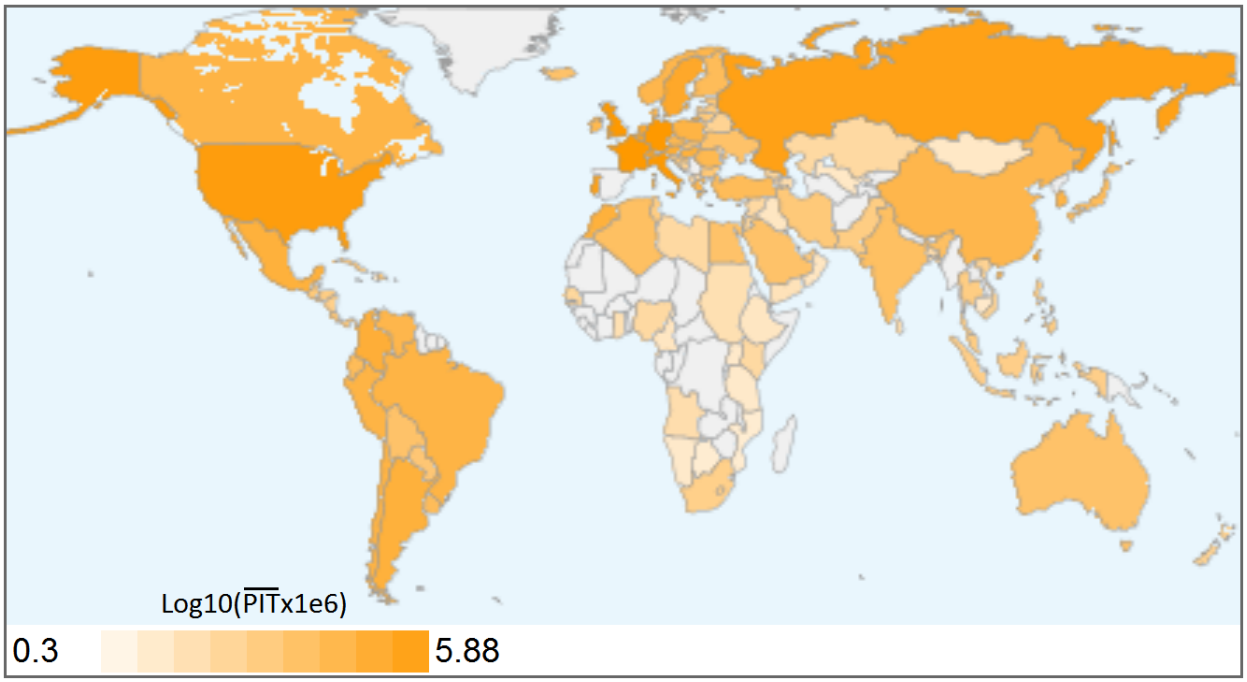


Figura 4.15: $\overline{PIT}$ de cada país en escala logarítmica.

4.3.3. Comprensión del Path Inflation Traffic

En esta sección se realizará un análisis previo del PIT a nivel continental, para ir descubriendo superficialmente aquellos países con PIT crítico. En la Ecuación 4.1 se observa que el PIT es proporcional al tráfico recibido por país. Se presenta en la Figura 3.5 el top15 de países con mayor tráfico recibido, útil para comprender la influencia del tráfico en los valores del $\overline{PIT}$ obtenido.

PIT en Europa

En la figura 4.16 se observa el $\overline{PIT}$ x 10^6 en Europa en escala logarítmica. Con una tonalidad mayor están países como Francia, Alemania, Italia, Reino Unido, por mencionar los más destacables, países que pertenecen al top 15 (ver Figura 3.5) de países con mayor tráfico recibido.

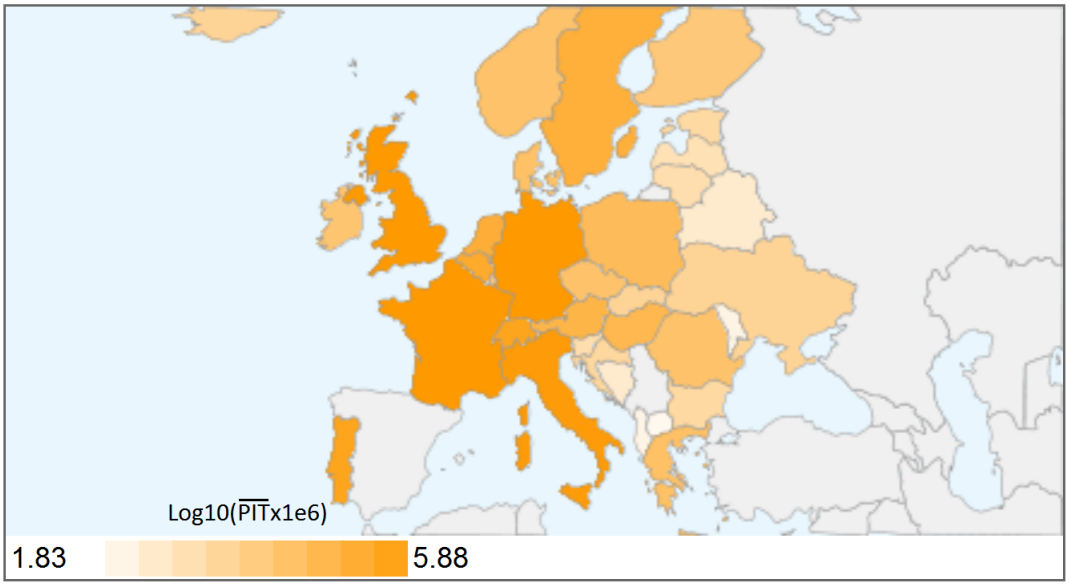


Figura 4.16: $\overline{PIT}$ en Europa en escala logarítmica.

PIT en América del Sur

En la figura 4.17 se observa el $\overline{PIT}$ x 10^6 en América del Sur en escala logarítmica. El $\overline{PI}$ en este contiente tenía poca variabilidad (ver figura 4.5) por lo que se puede observar mejor el efecto del tráfico en el $\overline{PIT}$. En concreto, se observa una penalización mayor en Colombia y Argentina y algo menor en Brasil, Chile, Ecuador y Venezuela.

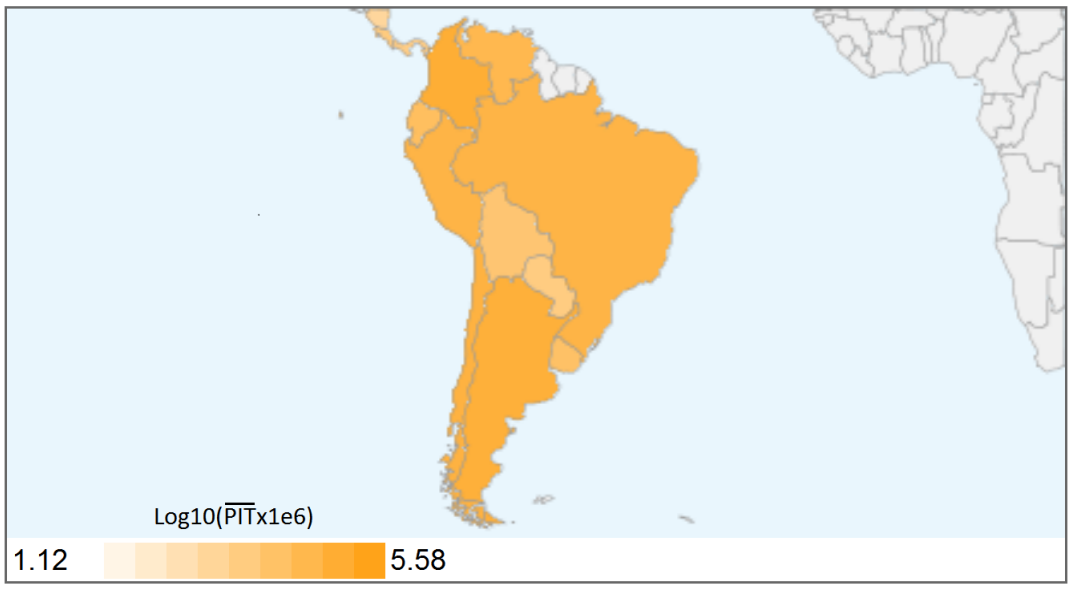


Figura 4.17: $\overline{PIT}$ en América del Sur en escala logarítmica.

PIT en África

En la figura 4.18 se observa el $\overline{PIT}$ x 10^6 en África en escala logarítmica. Destaca Marruecos, Argelia, Egipto y Túnez, que además de tener un PI alto relativo a los demás países africanos, debido al efecto ya comentado del uso del enlace de Estados Unidos, lo que aumentaba el PI en países cercanos a Madrid, son los países que mayor tráfico reciben en África según el exportador de tráfico de RedIRIS en Madrid.

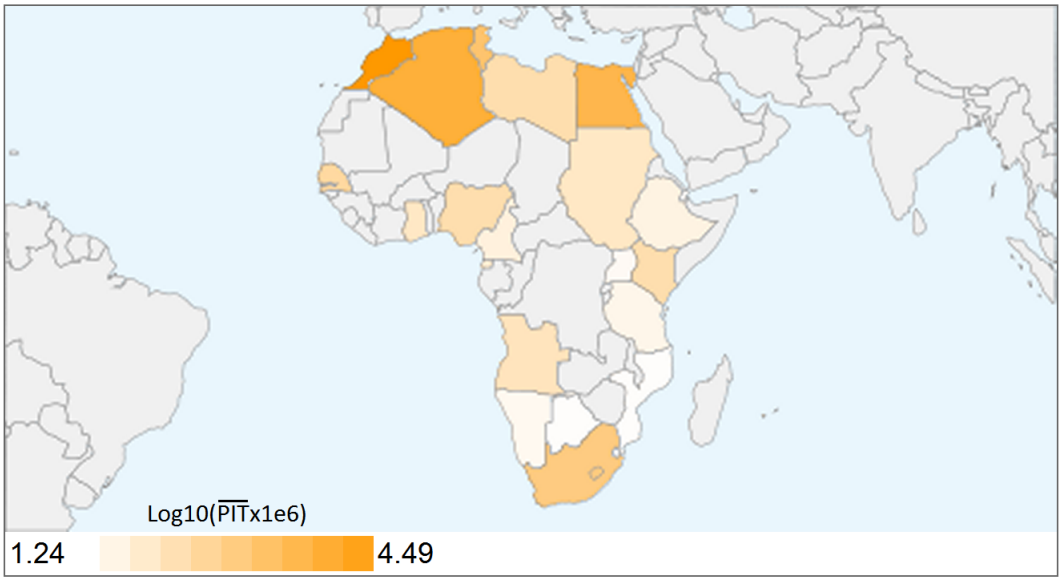


Figura 4.18: $\overline{PIT}$ en África en escala logarítmica

PIT en Asia

En la figura 4.19 se observa el $\overline{PIT}$ x 10^6 en Ásia en escala logarítmica. A primera vista, destacan países con mayor $\overline{PIT}$ como China, Japón, Turquía e India, países con alto $\overline{PI}$, y Taiwan, Rusia y Korea, que además del alto $\overline{PI}$ reciben un tráfico alto como se observa en la Figura 3.5.

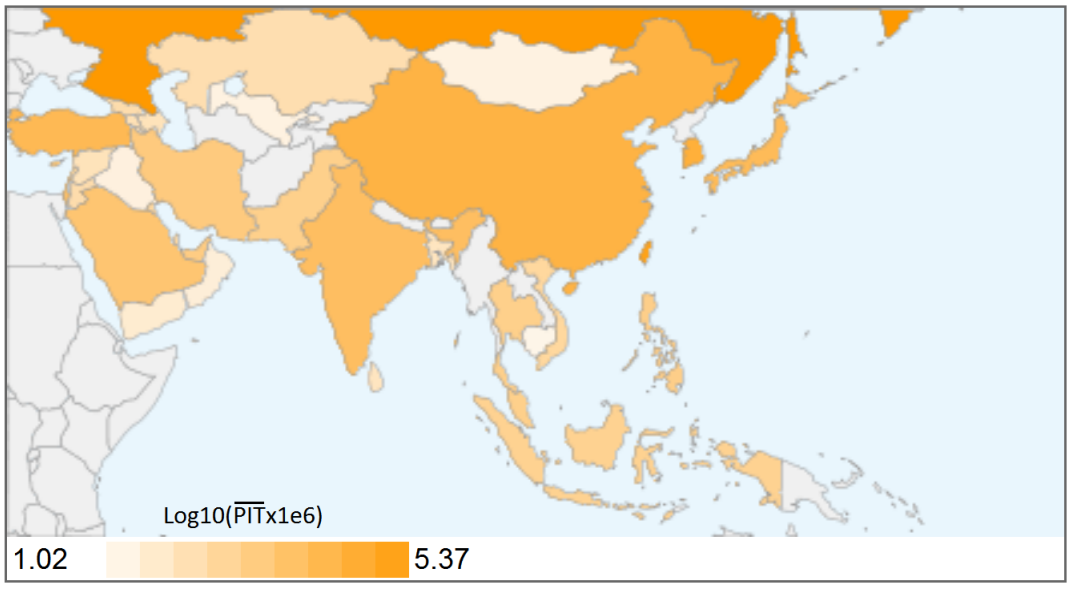


Figura 4.19: $\overline{PIT}$ en Asia en escala logarítmica

4.3.4. Estudio del Path Inflation Traffic

En esta sección se cuantificará el PIT para realizar una clasificación de los países y encontrar un umbral de decisión para estimar si el PIT es crítico o no. Hemos ido observando que el Path Inflation varía significativamente en cada contiente principalmente por el uso frecuente del enlace de Estados Unidos. Debido a esta heterogeneidad del PI a nivel continental, se analizará en detalle los países de cada continente por separado, y se propondrá un umbral de decisión para cada uno, que determinará si un país tiene un PIT crítico o no en su continente. Posteriomente clasificaremos los países de cada continente en una clasificación de países con PIT elevado.

Finalmente integraremos todos los resultados de cada continente, mostrando una clasificación de los países más críticos a efectos de PIT en el mundo.

El umbral propuesto será la métrica PITCON (ver Ecuación 4.2), dónde se calcula la media aritmética del PIT de todos los países de un conjunto. Dicho conjunto será a nivel continental en los estudios característicos de cada continente, y a nivel mundial para integrar todos los resultados.

$$PITCON = \frac{\sum_{i=1}^{n} \overline{PIT_i}}{M_t} \tag{4.2}$$

donde:

$\overline{PIT_i} \equiv \overline{PIT}$ del país i
$M_t \equiv$ número de países del conjunto

Se considerará un PIT aceptable a aquel con un valor menor a dicha media aritmética. Se elaborará la clasificación en función de los valores del PIT de los países que superan el PITCON con la métrica 4.3.

$$CR_i = \frac{\overline{PIT}_i}{PITCON} \tag{4.3}$$

CR_i cuantifica la proporción que el PIT del país i supera al PITCON, y será útil para elaborar clasificaciones de países críticos a nivel continental o mundial.

Estudio del PIT en Europa

Como se ha visto en apartados anteriores, el PI en Europa sigue en general el siguiente patrón: alto PI en países cercanos a Madrid, y no tan alto en países más distantes. El PIT en Europa se observa en la figura 4.20. La línea horizontal representa el PITCON con un valor de 93753.43 x 10^{-6}. Los países que superan dicho umbral se clasifican como críticos y son representados en la Figura 4.21.

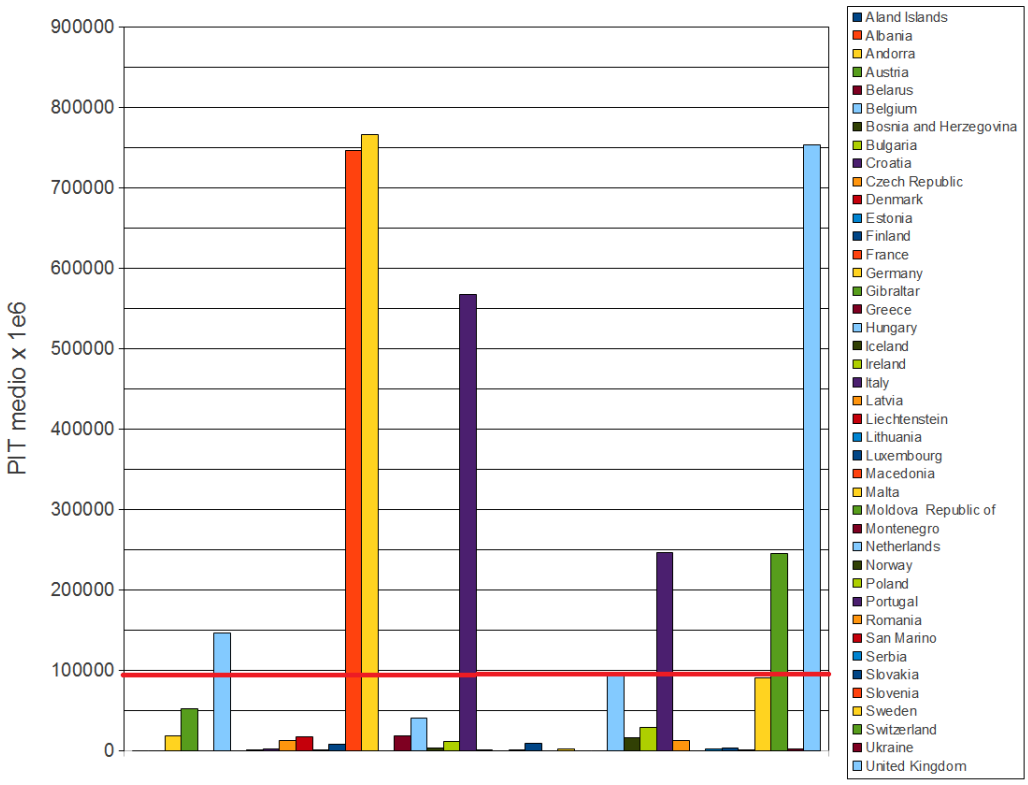


Figura 4.20: $\overline{PIT}$ en Europa

En la Figura 4.21 se observa el CR (Ver ecuación 4.3), que no es más que la proporción que el $\overline{PIT}$ de un país supera el PITCON o media aritmética del PIT en un continente. Los países más críticos son Alemania, Reino Unido y Francia con un CR de 8 aproximádamente. Estos países tienen el CR tan alto debido al alto PI y tráfico recibido. En segundo lugar está Italia con un CR de 6, con un PI similar al grupo de países anterior pero con menor tráfico recibido. En tercer lugar están Portugal y Suiza con un CR entre 2 y 3. Finalmente están Bélgica y Holanda con un CR entre 1 y 2.

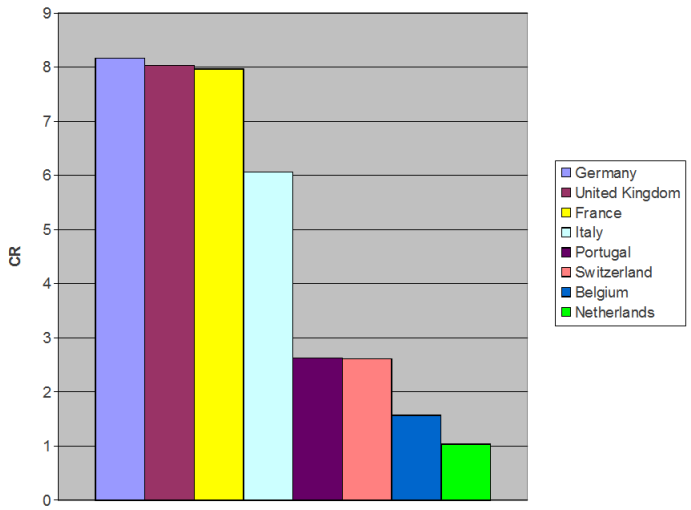


Figura 4.21: CR de países críticos en Europa

Estudio del PIT en América

América es el continente que menor PI obtiene debido a que el tráfico con destino a este continente pasa con frecuencia por Estados Unidos para alcanzar su destino final. Debido a esta característica, hemos observado que el PI en América del Sur no es muy variable comparado con otros continentes (Ver Figura 4.5). Por esto, es más fácil apreciar el efecto del tráfico en el PIT. En la gráfica 4.22 se observa el $\overline{PIT}$ de los países americanos y el PITCON con un valor de 19060.37 x 10^{-6}. Los países que superan dicho umbral se clasifican como críticos y son representados en la Figura 4.23.

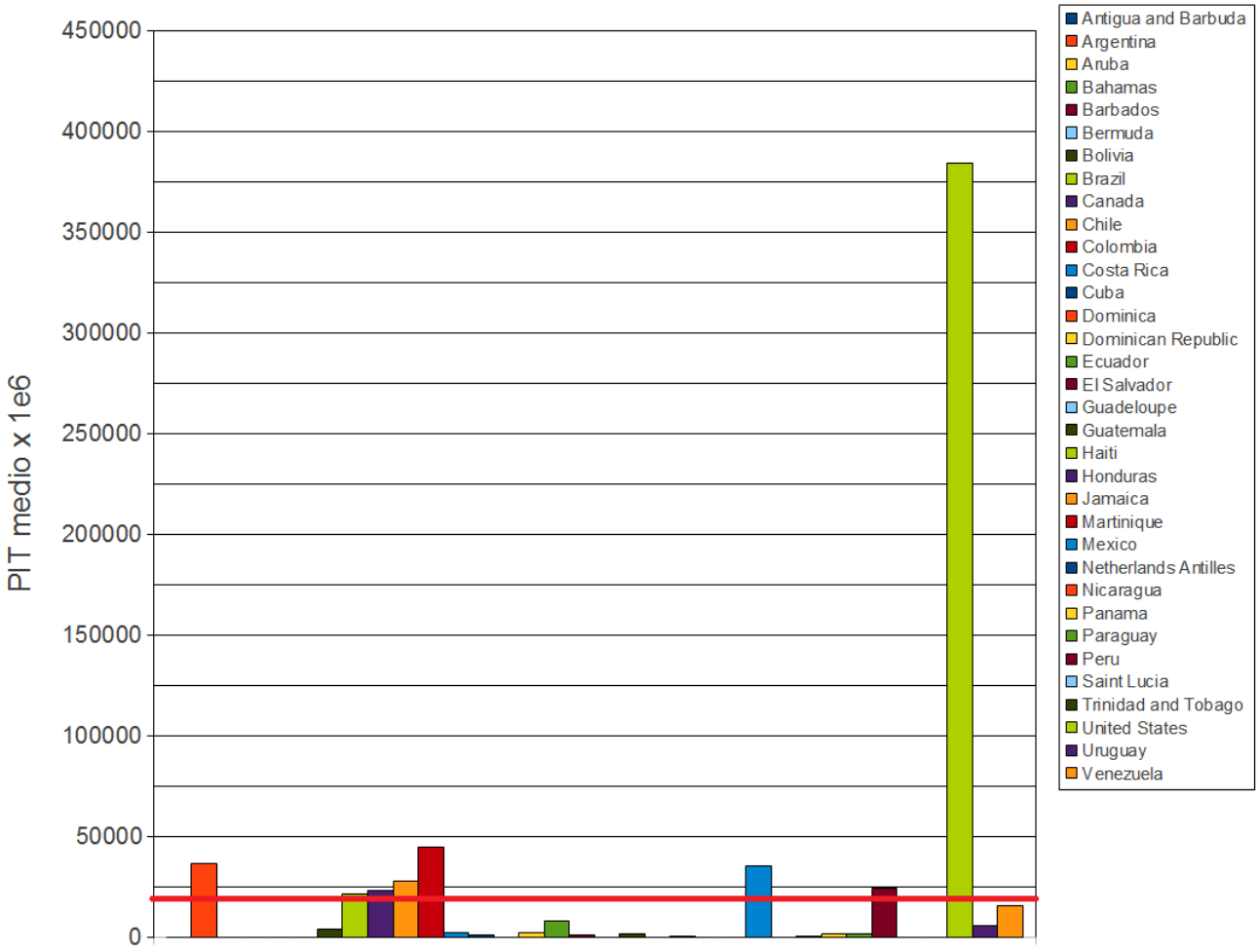


Figura 4.22: $\overline{PIT}$ en América

El país más crítico es Estados Unidos con un CR de 20 puntos. Este valor tan alto es debido al alto tráfico recibido, la cuarta parte del tráfico total analizado en el exportador de Madrid de RedIRIS. Con un CR entre 1 y 2 está un grupo de países formado por Colombia, Argentina, México, Chile, Perú, Canadá y Brasil, que obtienen este valor por ser los que mayor tráfico reciben en este continente.

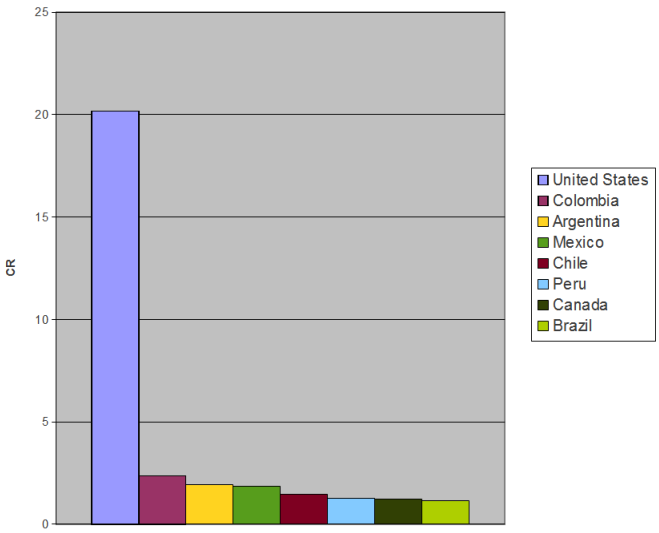


Figura 4.23: CR de países críticos en América

Estudio del PIT en África

El continente de África adopta un patrón de PI similar a Europa. donde países cercanos a Madrid obtienen mayor PI que países lejanos por norma general. En la gráfica 4.24 se observa el PIT de los países africanos y el PITCON con un valor de 1727,32 x 10^{-6}. Los países que superan dicho umbral se clasifican como críticos y son representados en la Figura 4.25.

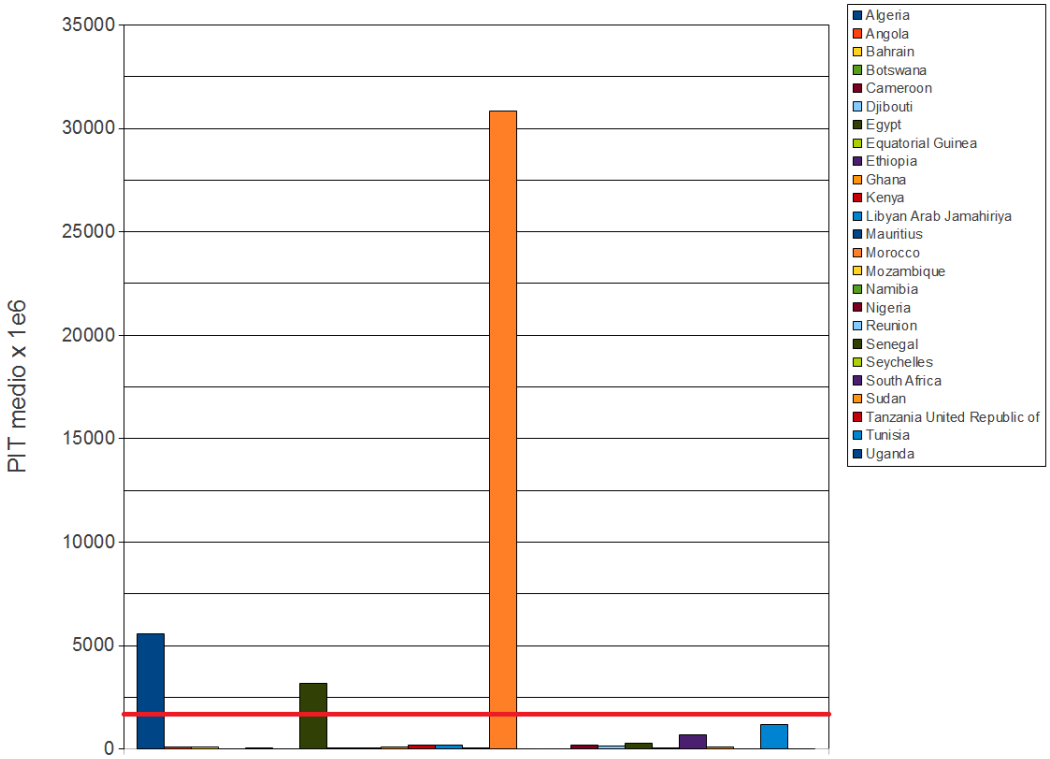


Figura 4.24: $\overline{PIT}$ en África

El país más crítico es Marruecos con un CR de casi 18 puntos, debido al alto PI y al alto tráfico recibido. En segundo lugar, está Argelia con un CR de 3 puntos aproximádamente. En último lugar está Egipto con un CR de casi 2.

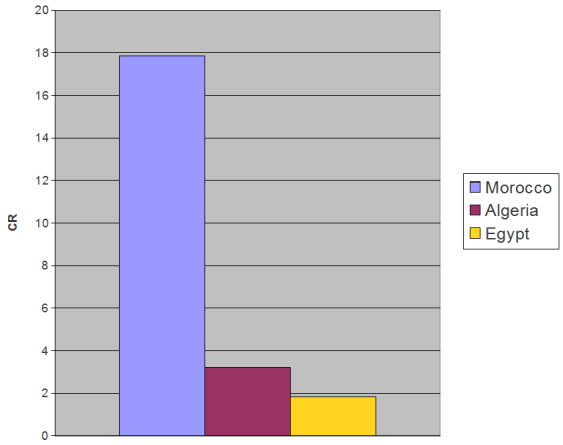


Figura 4.25: CR de países críticos en África

Estudio del PIT en Asia

El continente de Asia no sufre tanto como Europa ni África el efecto de que países más cercanos a Madrid sufran mayor PI al aumentar tanto la distancia enrutada como la distancia geográfica. En la gráfica 4.26 se observa el PIT de los países asiáticos y el PITCON con un valor de 10203.11 x 10^{-6}. Los países que superan dicho umbral se clasifican como críticos y son representados en la Figura 4.27.

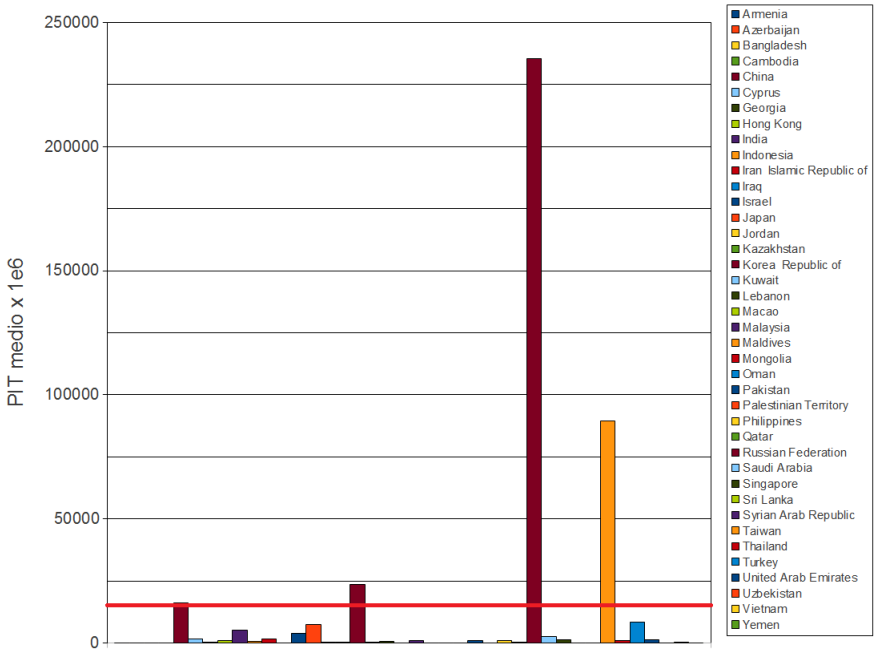


Figura 4.26: $\overline{PIT}$ en Asia

El país más crítico es Rusia con un CR de 23 puntos aproximádamente. Rusia obtiene este valor tan exagerado al ser el país con el PI más alto y ser el país que mayor cantidad de tráfico recibe en el continente asiático. En segundo lugar está Taiwan, con un CR de más de 7, ya que es el segundo país asiático que más tráfico recibe. En tercer lugar están Corea y China con un CR entre 1 y 2, debido a la cantidad de tráfico recibida ya que su PI es de los más bajos del continente asiático.

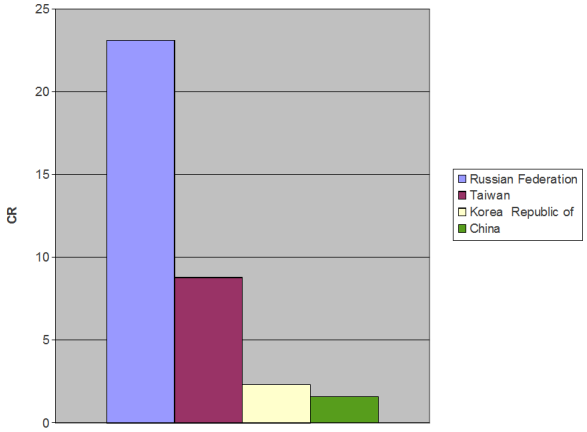


Figura 4.27: CR de países críticos en Asia

Estudio del PIT en Oceanía

En este continente se obervan PIT bajos relativos a los demás países. En la Figura 4.28 se observa el PIT de los países asiáticos y el PITCON con un valor de 1102.45 x 10^{-6}. Los países que superan dicho umbral se clasifican como críticos y son representados en la Figura 4.29.

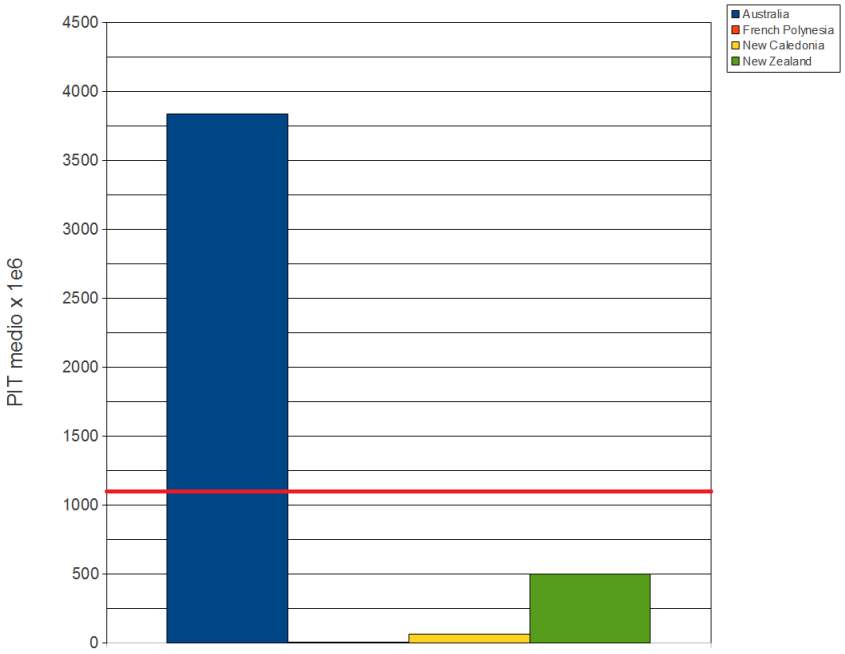


Figura 4.28: $\overline{PIT}$ en Oceanía

El único país crítico, como se observa en la Figura 4.29, es Australia, sin embargo, su $\overline{PIT}$ es muy bajo comparados con el resto de los países del mundo.

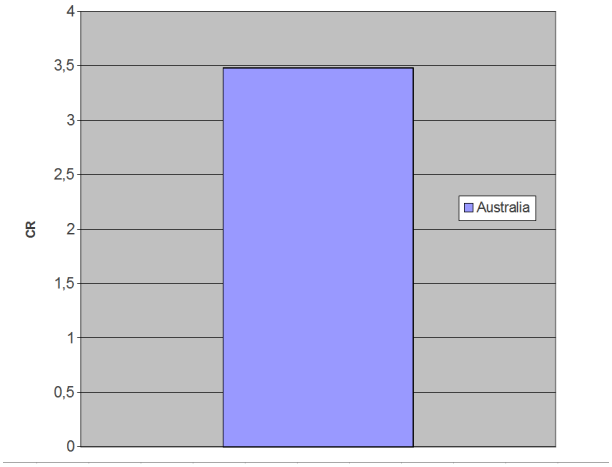


Figura 4.29: CR de países críticos en Oceanía

Estudio del PIT a nivel mundial

A nivel mundial el PITCON obtiene un valor de 34530,32 x 10^{-6}. Los países que superan dicho umbral se clasifican como críticos y son representados en la Figura 4.30.

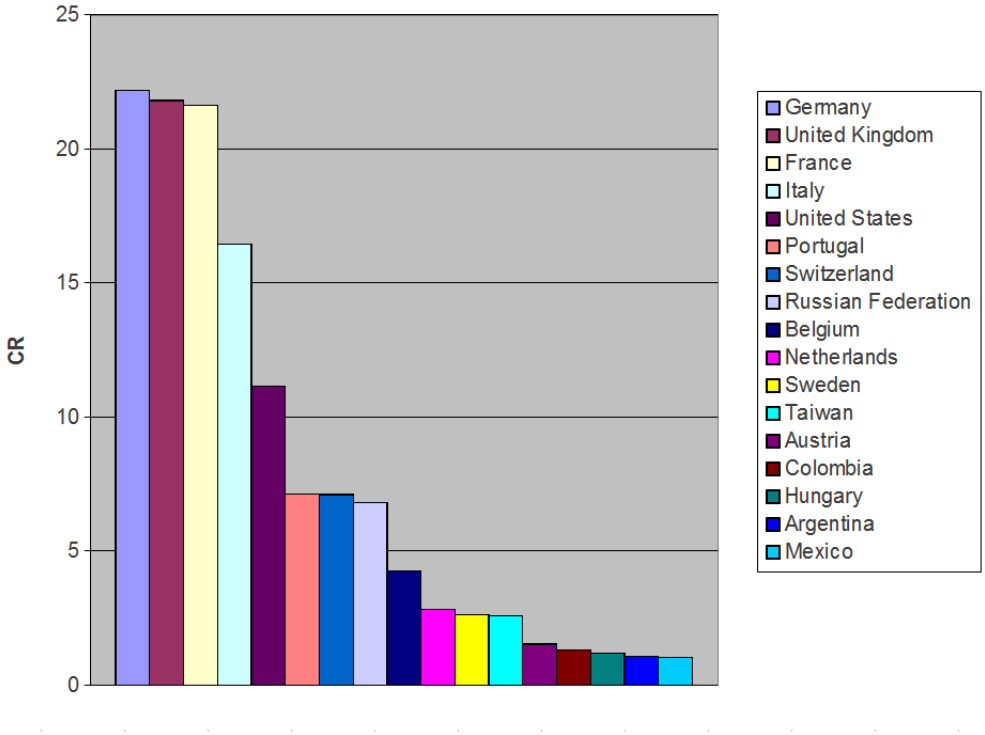


Figura 4.30: CR de países críticos en el Mundo

A nivel mundial, los tres países más críticos son Alemania, Reino Unido y Francia con un CR superior a 20. Estos países de origen europeo obtienen este CR debido al alto PI y a encontrarse entre los 4 países que mayor tráfico reciben. Con un CR superior a 15 está Italia, país con un PI similar a los anteriores, pero que recibe menor cantidad de tráfico. Con un CR superior a 10 se encuentra Estados Unidos, país que recibe la cuarta parte del tráfico total. Esta última característica predomina en el CR ya que su PI es muy bajo. Con un CR entre 5 y 8 puntos se encuentra Portugal, Suiza y Rusia. Con un CR menor de 5 puntos están Bélgica, Holanda, Suecia, Taiwan, Austria, Colombia, Hungría, Argentina y México.

Se concluye que Europa es el continente con el mayor número de países críticos en PI (con 11 países en el ranking). América aporta a Estados Unidos con un CR mayor de 10 y a Colombia, Argentina y México con un CR entre 1 y 2. En Asia, Taiwan tiene un CR de más de 2 puntos. Oceanía y África no poseen ningún país cuyo PIT sea crítico.

5

Análisis de retardos y saltos

5.1. Introducción

Hemos definido el Path Inflation como el hecho de que los caminos en Internet sean mayores de lo necesario. El aumento de una ruta entre dos localizaciones conlleva también un aumento en el tiempo de transmisión, al existir una clara relación entre distancia y tiempo de transmisión, para una velocidad de transmisión constante:

$$t_{enrutado} = \frac{d_{enrutada}}{v_{tx}} + \sum_{i=1}^{n} t_{procesado,i} \tag{5.1}$$

El tiempo enrutado, que es el tiempo que tarda un paquete en alcanzar un destino, depende de dos factores: el primer factor se debe a la distancia que recorre (tiempo de recorrido) en cada tramo hasta alcanzar un router y un segundo factor, que es el tiempo que un router tarda en procesar un paquete (tiempo de procesado). El tiempo de recorrido depende únicamente de la distancia total de todos los tramos, mientras que el tiempo de procesado depende del número de routers o saltos que se encuentre un paquete hasta alcanzar su destino. El tiempo de procesado incluye normalmente la espera del paquete en la cola del router, la desencapsulación del paquete, búsqueda del siguiente salto en la tabla de enrutamiento y la encapsulación del paquete que se reenvía.

Se define el tiempo óptimo de transmisión al propio de una comunicación para la cuál la distancia enrutada es igual a la distancia geográfica entre dos extremos en una comunicación, esto es, la distancia mínima posible:

$$t_{optimo} = \frac{d_{geografica}}{v_{tx}} \tag{5.2}$$

Si se considera la velocidad de transmisión constante en la comunicación típica (en Ecuación 5.1) y en la comunicación ideal (en Ecuación 5.2), y asumimos los $t_{procesado,i}$ iguales y despreciables en comparación con $\frac{d_{enrutada}}{v_{tx}}$ se puede despejar la velocidad de transmisión en ambas ecuaciones e igualarlas:

$$v_{tx} = \frac{d_{enrutada}}{t_{enrutado}} \tag{5.3}$$

$$v_{tx} = \frac{d_{geografica}}{t_{optimo}} \tag{5.4}$$

$$\frac{d_{geografica}}{t_{optimo}} = \frac{d_{enrutada}}{t_{enrutado}} \tag{5.5}$$

El PI según fue definido en la Ecuación 3.1, nos permite expresar el tiempo óptimo en función del PI:

$$t_{optimo} = \frac{d_{geografica} \cdot t_{enrutado}}{d_{enrutada}} = \frac{t_{enrutado}}{PI} \tag{5.6}$$

La Ecuación 5.6 indica que existe una estrecha relación entre el PI y el $t_{enrutado}$ por lo que analizaremos los retardos medios del PoP de Madrid de la RedIRIS con el conjunto de países que más tráfico reciben desde la RedIRIS.

5.2. Estudio de retardos

5.2.1. Metodología

Para el cálculo del tiempo $t_{enrutado}$ usaremos la herramienta `Ping`. `Ping` es una utilidad diagnóstica en redes que comprueba el estado de la conexión del host local con uno o varios equipos remotos por medio del envío de paquetes ICMP de solicitud y de respuesta. El problema de `Ping` es el uso de paquetes ICMP que son frecuentemente filtrados como ocurría con `Traceroute`. Por este motivo se utilizará una herramienta que utiliza paquetes TCP para el cálculo del $t_{enrutado}$ y es `HPing3`[1].

En concreto `HPing3` calcula el RTT, que es el tiempo de ida y vuelta de un paquete, resultando ser el doble del tiempo enrutado. Utilizando esta relación obtenemos en la Ecuación 5.7 el t_{optimo} en función del RTT:

$$t_{optimo} = \frac{RTT}{PI \cdot 2} \tag{5.7}$$

5.2.2. Conjunto de datos del estudio de retardos

Partiendo de los emplazamientos originalmente seleccionados (ver sección **Origen de los emplazamientos seleccionados** del Capítulo 3, **Diseño y desarrollo**) calcularemos el RTT para el porcentaje de las direcciones IP que son alcanzadas mediante `TCPTraceroute` (ver sección **Análisis de traceroute y TCPTraceroute** del Capítulo 3, **Diseño y desarrollo**). Este filtrado previo es útil ya que tanto `TCPTraceroute` como `HPing3` utilizan paquetes TCP, por lo que podemos eliminar las direcciones IP que no permiten el paso de estos paquetes en algún nodo intermedio.

Siguiendo esta consideración, en la Tabla 5.1 se muestra el número de direcciones IP por país/región administrativa útiles para el cálculo del RTT.

Cuadro 5.1: # destinos o direcciones IP por país/región administrativa

País	**Continente**	**# destinos o IP**
Aland Islands	Europa	74

[1]Se pueden encontrar los manuales de estas herramientas en el Anexo **C Manual de utilización**

Cuadro 5.1 — Continúa de la página anterior

País	**Cont.**	**# destinos o IP**
Albania	Europa	1081
Algeria	Africa	7166
Andorra	Europa	675
Angola	Africa	86
Antarctica	Antartida	27
Antigua and Barbuda	America	333
Argentina	America	318169
Armenia	Asia	526
Aruba	America	409
Australia	Oceania	22236
Austria	Europa	12431
Azerbaijan	Asia	1723
Bahamas	America	671
Bahrain	Africa	142
Bangladesh	Asia	1139
Barbados	America	5092
Belarus	Europa	3898
Belgium	Europa	10580
Bermuda	America	220
Bolivia	America	13797
Bosnia and Herzegovina	Europa	1902
Botswana	Africa	137
Brazil	America	211172
Bulgaria	Europa	32804
Cambodia	Asia	937
Cameroon	Africa	862
Canada	America	49869
Chile	America	286590
China	Asia	180330
Colombia	America	334574
Costa Rica	America	29026
Croatia	Europa	4178
Cuba	America	699
Cyprus	Asia	1722
Czech Republic	Europa	25168
Denmark	Europa	16383
Djibouti	Africa	129
Dominica	America	559
Dominican Republic	America	52543
Ecuador	America	32519
Egypt	Africa	31279
El Salvador	America	33954
Equatorial Guinea	Africa	162
Estonia	Europa	3139
Ethiopia	Africa	79
Finland	Europa	10358
France	Europa	105375
French Polynesia	Oceania	63
Georgia	Asia	6890
Germany	Europa	188666

Cuadro 5.1 — Continúa de la página anterior

País	**Cont.**	**# destinos o IP**
Ghana	Africa	4841
Gibraltar	Europa	413
Greece	Europa	16125
Guadaloupe	America	141
Guatemala	America	10324
Haiti	America	186
Honduras	America	1838
Hong Kong	Asia	9608
Hungary	Europa	20110
Iceland	Europa	1542
India	Asia	215631
Indionesia	Asia	37261
Iran Islamic Republic of	Asia	7569
Iraq	Asia	598
Ireland	Europa	8354
Israel	Asia	23486
Italy	Europa	164084
Jamaica	America	2287
Japan	Asia	43335
Jordan	Asia	1174
Kazakhstan	Asia	26159
Kenya	Africa	1181
Korea Republic of	Asia	37025
Kuwait	Asia	2807
Latvia	Europa	7482
Lebanon	Asia	5308
Libyan Arab Jamahiriya	Africa	263
Liechtenstein	Europa	220
Lithuania	Europa	8492
Luxembourg	Europa	1426
Macao	Asia	993
Macedonia	Europa	5387
Malaysia	Asia	31704
Maldives	Asia	480
Malta	Europa	1154
Martinique	America	116
Mauritius	Africa	1436
Mexico	America	993325
Moldova Republic of	Europa	6303
Mongolia	Asia	510
Montenegro	Europa	911
Morocco	Africa	19357
Mozambique	Africa	418
Namibia	Africa	261
Netherlands	Europa	44250
Netherlands Antilles	America	448
New Caledonia	Oceania	133
New Zealand	Oceania	4053
Nicaragua	America	7503
Nigeria	Africa	1210

Cuadro 5.1 — Continúa de la página anterior

País	**Cont.**	**# destinos o IP**
Norway	Europa	26567
Oman	Asia	904
Pakistan	Asia	12799
Palestinian Territory	Asia	2903
Panama	America	6893
Paraguay	America	7424
Peru	America	275597
Philippines	Asia	7027
Poland	Europa	68449
Portugal	Europa	20598
Qatar	Asia	743
Reunion	Africa	163
Romania	Europa	39860
Russian Federation	Asia	124424
Saint Lucia	America	425
San Marino	Europa	175
Saudi Arabia	Asia	10541
Senegal	Africa	879
Serbia	Europa	14843
Seychelles	Africa	50
Singapore	Asia	31106
Slovakia	Europa	8603
Slovenia	Europa	3994
South Africa	Africa	6846
Sri Lanka	Asia	2267
Sudan	Africa	709
Sweden	Europa	63887
Switzerland	Europa	38876
Syrian Arab Republic	Asia	659
Taiwan	Asia	73620
Tanzania United Republic	Africa	410
Thailand	Asia	47718
Trinidad and Tobago	America	1478
Tunisia	Africa	4614
Turkey	Asia	50621
Uganda	Africa	306
Ukraine	Europa	57258
United Arab Emirates	Asia	1316
United Kingdom	Europa	120015
United States	America	1065831
Uruguay	America	17645
Uzbekistan	Asia	739
Venezuela	America	224189
Vietnam	Asia	103805
Yemen	Asia	1475

5.2.3. Configuración de HPing3

Se calcularán tres medidas del RTT por cada dirección IP. La sintaxis del comando utilizado es el siguiente:

```
hping3 -S [IP] -p 80 -c 3
```

Los argumentos son:

- -S: uso de paquetes TCP SYN.
- -p: puerto de destino. Seleccionamos el puerto 80 al ser el referente a aplicaciones HTTP y estar generalmente abierto.
- IP: dirección IP destino.
- -c: número de paquetes de prueba. Seleccionamos 3 al ser el número de medidas que queremos obtener.

5.2.4. Estadística del Round-Trip Time

La estadística del RTT medio ($\overline{RTT}$) calculado por país/región administrativa se observa en la Tabla 5.2:

Cuadro 5.2: Estadística del RTT.

País	**Cont.**	$\overline{RTT}$ **(ms)**
Aland Islands	Europa	149
Albania	Europa	75
Algeria	Africa	177
Andorra	Europa	53
Angola	Africa	557
Antigua and Barbuda	America	193
Argentina	America	335
Armenia	Asia	156
Aruba	America	189
Australia	Oceania	372
Austria	Europa	73
Azerbaijan	Asia	177
Bahamas	America	181
Bahrain	Africa	193
Bangladesh	Asia	323
Barbados	America	224
Belarus	Europa	141
Belgium	Europa	57
Bermuda	America	142
Bolivia	America	379
Bosnia and Herzegovina	Europa	98
Botswana	Africa	346
Brazil	America	304
Bulgaria	Europa	81
Cambodia	Asia	439
Cameroon	Africa	353
Canada	America	157
Chile	America	278
China	Asia	483
Colombia	America	234

Cuadro 5.2 — Continúa de la página anterior

País	**Cont.**	$\overline{RTT}$ **(ms)**
Costa Rica	America	279
Croatia	Europa	104
Cuba	America	822
Cyprus	Asia	82
Czech Republic	Europa	63
Denmark	Europa	72
Djibouti	Africa	251
Dominica	America	220
Dominican Republic	America	221
Ecuador	America	269
Egypt	Africa	222
El Salvador	America	245
Equatorial Guinea	Africa	752
Estonia	Europa	88
Ethiopia	Africa	245
Finland	Europa	147
France	Europa	41
French Polynesia	Oceania	307
Georgia	Asia	163
Germany	Europa	45
Ghana	Africa	324
Gibraltar	Europa	38
Greece	Europa	137
Guadaloupe	America	234
Guatemala	America	247
Haiti	America	279
Honduras	America	177
Hong Kong	Asia	348
Hungary	Europa	80
Iceland	Europa	101
India	Asia	297
Indionesia	Asia	424
Iran Islamic Republic of	Asia	294
Iraq	Asia	211
Ireland	Europa	75
Israel	Asia	164
Italy	Europa	129
Jamaica	America	200
Japan	Asia	303
Jordan	Asia	210
Kazakhstan	Asia	188
Kenya	Africa	296
Korea Republic of	Asia	348
Kuwait	Asia	375
Latvia	Europa	96
Lebanon	Asia	196
Libyan Arab Jamahiriya	Africa	209
Liechtenstein	Europa	48
Lithuania	Europa	98
Luxembourg	Europa	55

Cuadro 5.2 — Continúa de la página anterior

País	**Cont.**	$\overline{RTT}$ **(ms)**
Macao	Asia	349
Macedonia	Europa	126
Malaysia	Asia	439
Maldives	Asia	275
Malta	Europa	122
Martinique	America	215
Mauritius	Africa	423
Mexico	America	245
Moldova Republic of	Europa	170
Mongolia	Asia	317
Montenegro	Europa	317
Morocco	Africa	200
Mozambique	Africa	434
Namibia	Africa	376
Netherlands	Europa	55
Netherlands Antilles	America	204
New Caledonia	Oceania	352
New Zealand	Oceania	337
Nicaragua	America	239
Nigeria	Africa	263
Norway	Europa	84
Oman	Asia	270
Pakistan	Asia	270
Palestinian Territory	Asia	236
Panama	America	199
Paraguay	America	320
Peru	America	261
Philippines	Asia	398
Poland	Europa	81
Portugal	Europa	198
Qatar	Asia	466
Reunion	Africa	252
Romania	Europa	86
Russian Federation	Asia	121
Saint Lucia	America	208
San Marino	Europa	80
Saudi Arabia	Asia	156
Senegal	Africa	194
Serbia	Europa	79
Seychelles	Africa	479
Singapore	Asia	372
Slovakia	Europa	62
Slovenia	Europa	75
South Africa	Africa	266
Sri Lanka	Asia	561
Sudan	Africa	195
Sweden	Europa	98
Switzerland	Europa	55
Syrian Arab Republic	Asia	232
Taiwan	Asia	359

Cuadro 5.2 — Continúa de la página anterior

País	**Cont.**	$\overline{RTT}$ **(ms)**
Tanzania United Republic	Africa	309
Thailand	Asia	402
Trinidad and Tobago	America	206
Tunisia	Africa	166
Turkey	Asia	155
Uganda	Africa	389
Ukraine	Europa	118
United Arab Emirates	Asia	239
United Kingdom	Europa	51
United States	America	414
Uruguay	America	384
Uzbekistan	Asia	215
Venezuela	America	444
Vietnam	Asia	420
Yemen	Asia	363

En la Figura 5.1 se observa el RTT medio ($\overline{RTT}$) de cada páis en ms.

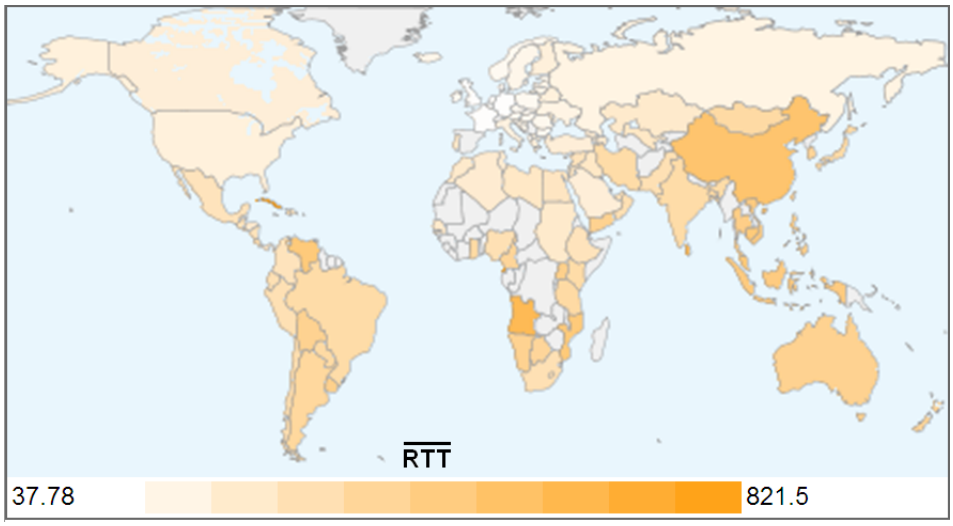


Figura 5.1: Mapa del $\overline{RTT}$ de Madrid a cada país/región administrativa bajo estudio

Se observan los $\overline{RTT}$ menores en algunos países de Europa como Francia, Bélgica, Alemania, Reino Unido, etc. Con una tonalidad todavía clara aparecen países que podemos agrupar en dos clases. En el primer grupo se puede incluir Estados Unidos y Canada, países con $\overline{RTT}$ bajo al estar bien conectados con España (PI bajo). En un segundo grupo aparecen países cercanos a España como Noruega, Suecia, Finlandia, Italia, Ucrania, Rumania, Bulgaria, etc, en Europa y Marruecos, Argelia, Libia, Egipto, etc, en África y Turkía, Iraq, Arabia Saudí en Asia cuya distancia enrutada es similar. Con una tonalidad oscura aparecen casi todos los países de América del Sur al utilizar como enlace Estados Unidos y Oceanía cuya distancia enrutada es considerable alcanzando altos $\overline{RTT}$ aunque tengan un PI aceptable. Los países con mayor $\overline{RTT}$ son India, Indonesia, ya que utilizan Estados Unidos como enlace aumentando considerablemente la distancia enrutada, Angola (ver Figura 4.10) en África y Cuba cuya distancia enrutada incluye con asiduidad los tramos de España a Estados Unidos, de Estados Unidos a África y de África a Cuba (ver Figura 5.2).

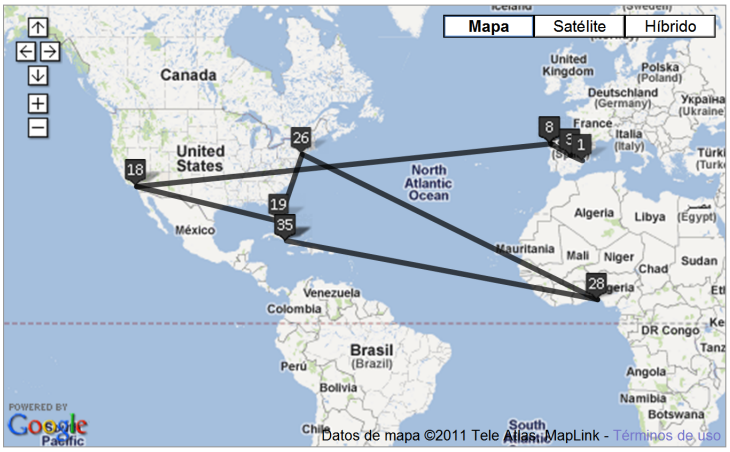


Figura 5.2: Ruta entre Madrid y una dirección IP de Cuba. Estas rutas incluyen con frecuencia los tramos de España a Estados Unidos, de Estados Unidos a África y de África a Cuba aumentando considerablemente el PI y el RTT.

5.2.5. Round-Trip Time vs distancias. Estudio de correlaciones

En esta sección estudiaremos la correlación del $\overline{RTT}$ obtenido con la distancia geográfica media y la distancia enrutada media obtenida en capítulos anteriores para cada país/región administrativa.

Correlación del RTT y la distancia geográfica

En este apartado analizaremos la correlación del RTT medio con la distancia geográfica media. Debido al PI, los caminos en Internet tendrán una longitud mayor que la distancia geográfica que separa dos puntos y en consecuenia mayor RTT que el esperado. El objetivo de este estudio en concreto, es observar si países con distancias geográficas similares obtienen RTTs similares (ver Figura 5.4). La correlación obtenida es debida a los patrones que trazan las rutas desde Madrid hasta el destino objetivo. Este patrón, como estudiamos previamente, es el uso del enlace de Estados Unidos con frecuencia para el enrutamento de los caminos analizados en el estudio del PI. Este patrón es complejo de analizar debido al comportamiento irregular de los diferentes países en la frecuencia de uso de Estados Unidos como enlace.

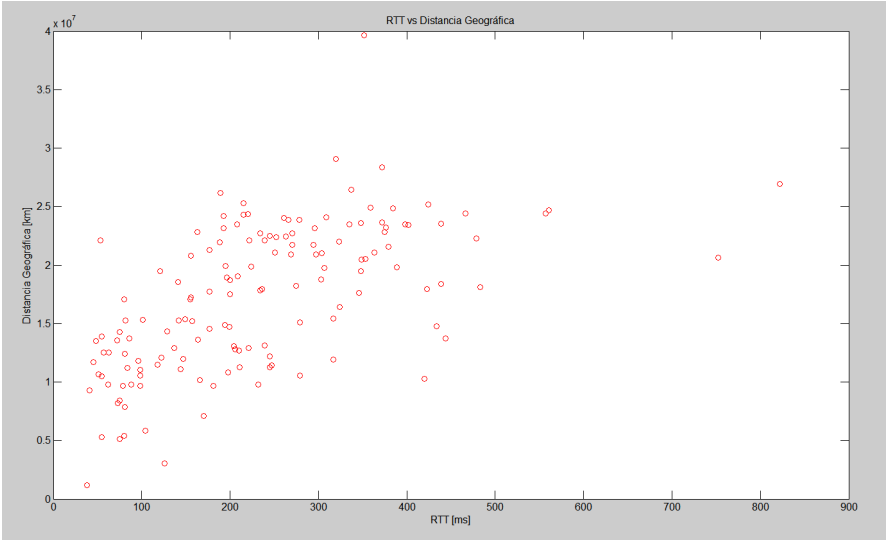


Figura 5.3: $\overline{RTT}$ vs distancia geográfica media. Se calcula un coeficiente de correlación del 60.76 %

Correlación del RTT y la distancia enrutada

En este apartado analizaremos la correlación del RTT medio con la distancia enrutada media. El objetivo es comprender la variabilidad del RTT observando la correlación del RTT con la distancia enrutada o recorrida en las rutas de Internet (ver Figura 5.4). En la Ecuación 5.1 se identifican los dos términos que comprenden el $t_{enrutado}$ y que introducen la alta variabilidad en el RTT, la velocidad de transmisión (v_{tx}) y el número y tiempo de procesado de cada nodo ($\sum_{i=1}^{n} t_{procesado,i}$) intermedio en una ruta de comunicación.

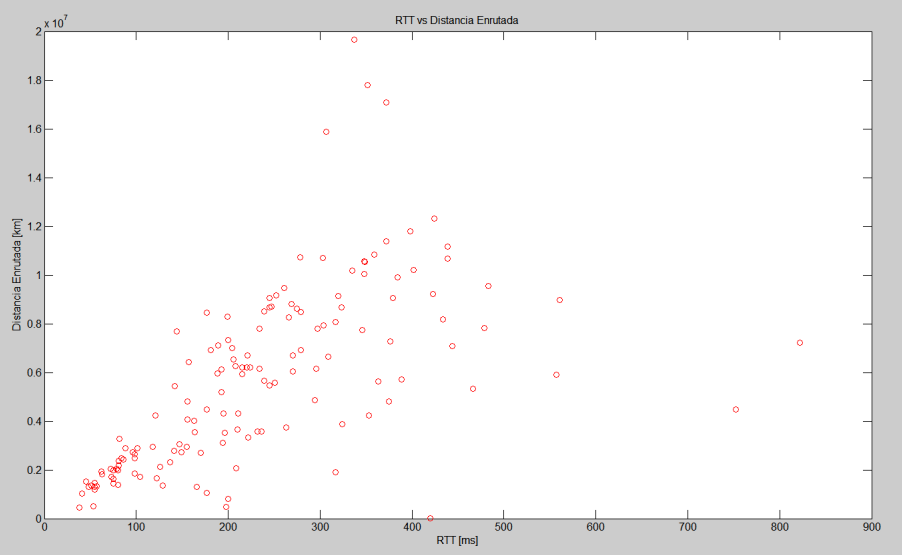


Figura 5.4: $\overline{RTT}$ vs distancia enrutada media. Se calcula un coeficiente de correlación del 59.10 %

5.2.6. Round-Trip Time vs tráfico recibido en los países populares de RedIRIS.

En esta seccíon estudiaremos el efecto del RTT en los destinos más populares de la RedIRIS a nivel de tráfico. De la misma manera que clasificamos la severidad del PI en función del tráfico, trataremos de identificar aquellos países cuyo RTT se considera importante al ser países que reciben una cantidad de tráfico considerable.

Para ello utilizaremos un algoritmo de clusterización para agrupar los países en función de su RTT y su tráfico recibido. En concreto, utilizaremos el algoritmo k-means, creado por MacQueen en 1967, siendo el algoritmo de clustering más conocido y utilizado debido a su eficaz y a su sencilla aplicación. Sigue un procedimiento simple de clasificación de un conjunto de objetos en un determinado número K de clústeres determinado a priori. El nombre de k-means viene porque representa cada uno de los clusters por la media (o media ponderada) de sus puntos, es decir, por su centroide. La representación mediante centroides tiene la ventaja de que tiene un significado gráfico y estadístico inmediato. Cada cluster por tanto, es caracterizado por su centro o centroide que se encuentra en el centro o el medio de los elementos que componen el cluster. Nuestro objetivo está orientado a la distinción de regiones con características similares que recogerán grupos de países más que a la obtención de centroides representativos de las regiones.

El método del k-means agrupa puntos en n-dimensiones utilizando distancias, por lo que será necesario utilizar rangos de RTT y tráfico recibido similares para que no exista una variable predominante. Con esto, en la Figura 5.5 se observa el algoritmo k-means (donde k es 4) aplicado a las variables mencionadas con la normalización del tráfico recibido por 300.

En dicha figura se han utilizado cuatro centroides, que agrupan los países en asociaciones de puntos próximos entre sí. Se pueden apreciar cuatro regiones, la Región 1, cuyo RTT es de 0 a

130 ms, la Región 2, que recoge países cuyo RTT oscila entre 130 y 300 ms, la Región 3, cuyo RTT está en el rango de 300 a 500 ms, y la Región 4, que incluye aquellos países que poseen un RTT mayor que 500 ms. Es interesante observar que los países de mayor tráfico se agrupan en la Región 1, lo que supone un RTT mínimo, mientras que en la Región 2 y 3 se encuentran los países de tráfico medio con un RTT de hasta 500 ms. En la Región 4 se encuentran países con un RTT mayor que 500 ms pero con un tráfico recibido poco importante.

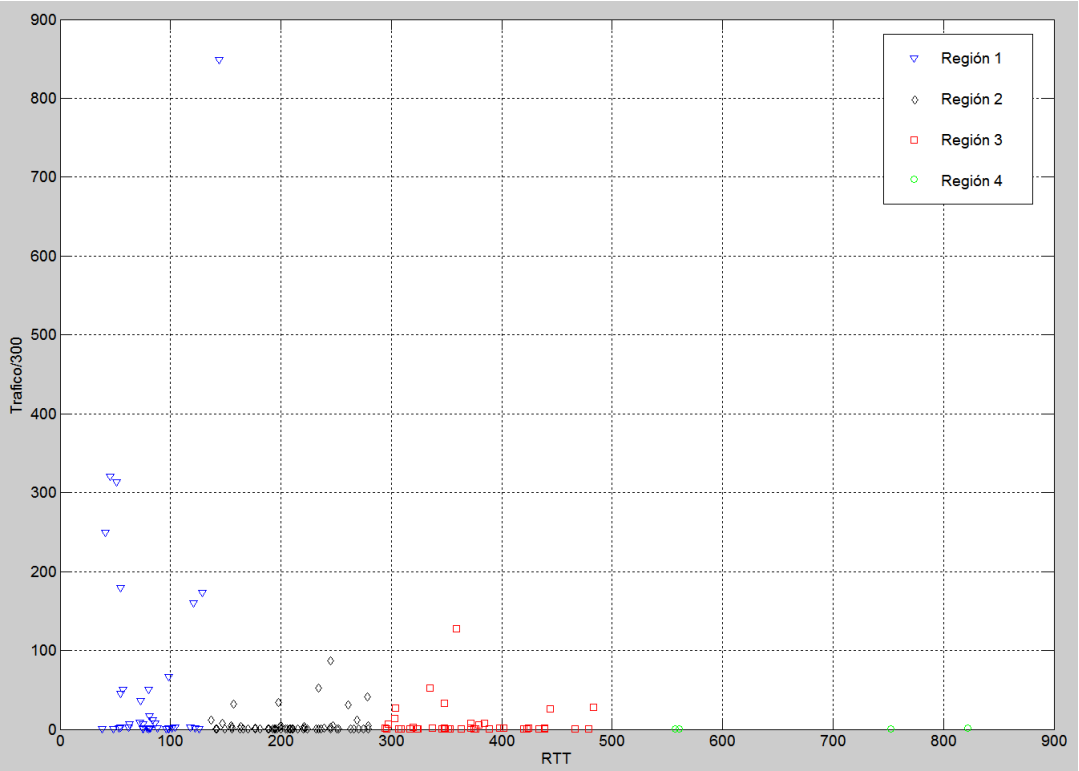


Figura 5.5: $\overline{RTT}$ vs tráfico recibido para los países más populares de RedIRIS. Se utiliza el algoritmo k-means para agrupar los países en regiones con características similares.

5.3. Estudio de saltos

5.3.1. Conjunto de datos del estudio de saltos

Partiendo de los emplazamientos originalmente seleccionados (ver sección **Origen de los emplazamientos seleccionados** del Capítulo 3, **Diseño y desarrollo**) calcularemos el número de saltos para el porcentaje de las direcciones IP que son alcanzadas mediante TCPTraceroute (ver sección **Análisis de traceroute y TCPTraceroute** del Capítulo 3, **Diseño y desarrollo**). El número de saltos lo obtendremos de las trazas de TCPTraceroute que utilizamos para el cálculo del PI.

De esta forma, el número de direcciones IP útiles para el cálculo del número de saltos coinciden con las direcciones IP sobre las que calculamos el RTT (ver Tabla 5.1).

5.3.2. Estadística del número de saltos

El número de saltos medio ($\overline{\#saltos}$) calculado por país/región administrativa se observa en la Tabla 5.3:

Cuadro 5.3: Estadística del # saltos.

País	**Cont.**	$\overline{\#saltos}$
Aland Islands	Europa	11.24
Albania	Europa	17.52
Algeria	Africa	14.29
Andorra	Europa	14.99
Angola	Africa	15.09
Antigua and Barbuda	America	17.18
Argentina	America	15.82
Armenia	Asia	16.06
Aruba	America	16.41
Australia	Oceania	15.90
Austria	Europa	15.46
Azerbaijan	Asia	16.42
Bahamas	America	14.69
Bahrain	Africa	17.69
Bangladesh	Asia	13.62
Barbados	America	17.49
Belarus	Europa	17.05
Belgium	Europa	13.70
Bermuda	America	14.85
Bolivia	America	17.55
Bosnia and Herzegovina	Europa	14.74
Botswana	Africa	16.88
Brazil	America	14.24
Bulgaria	Europa	16.96
Cambodia	Asia	16.32
Cameroon	Africa	11.87
Canada	America	14.71
Chile	America	16.13
China	Asia	16.94
Colombia	America	15.54
Costa Rica	America	15.39
Croatia	Europa	13.93
Cuba	America	16.09
Cyprus	Asia	12.63
Czech Republic	Europa	14.24
Denmark	Europa	13.22
Djibouti	Africa	13.19
Dominica	America	17.72
Dominican Republic	America	16.49
Ecuador	America	15.44
Egypt	Africa	13.80
El Salvador	America	17.24
Equatorial Guinea	Africa	14.38
Estonia	Europa	13.75
Ethiopia	Africa	18.95
Finland	Europa	13.97
France	Europa	12.50
French Polynesia	Oceania	15.49
Georgia	Asia	16.65

Cuadro 5.3 — Continúa de la página anterior

País	**Cont.**	$\overline{\#saltos}$
Germany	Europa	12.98
Ghana	Africa	12.71
Gibraltar	Europa	11.43
Greece	Europa	14.61
Guadaloupe	America	14.84
Guatemala	America	18.62
Haiti	America	17.25
Honduras	America	15.98
Hong Kong	Asia	14.79
Hungary	Europa	15.79
Iceland	Europa	14.42
India	Asia	14.03
Indionesia	Asia	16.33
Iran Islamic Republic of	Asia	18.78
Iraq	Asia	16.81
Ireland	Europa	14.73
Israel	Asia	18.39
Italy	Europa	14.90
Jamaica	America	18.53
Japan	Asia	17.99
Jordan	Asia	16.66
Kazakhstan	Asia	19.91
Kenya	Africa	13.78
Korea Republic of	Asia	19.14
Kuwait	Asia	17.39
Latvia	Europa	13.27
Lebanon	Asia	14.58
Libyan Arab Jamahiriya	Africa	13.92
Liechtenstein	Europa	12.87
Lithuania	Europa	13.35
Luxembourg	Europa	11.38
Macao	Asia	10.23
Macedonia	Europa	20.69
Malaysia	Asia	14.38
Maldives	Asia	15.35
Malta	Europa	16.07
Martinique	America	13.78
Mauritius	Africa	15.27
Mexico	America	14.53
Moldova Republic of	Europa	15.36
Mongolia	Asia	16.16
Montenegro	Europa	18.64
Morocco	Africa	13.91
Mozambique	Africa	15.65
Namibia	Africa	14.61
Netherlands	Europa	12.87
Netherlands Antilles	America	16.26
New Caledonia	Oceania	26.83
New Zealand	Oceania	16.86
Nicaragua	America	18.38

Cuadro 5.3 — Continúa de la página anterior

País	**Cont.**	$\overline{\#saltos}$
Nigeria	Africa	16.38
Norway	Europa	13.72
Oman	Asia	12.29
Pakistan	Asia	17.07
Palestinian Territory	Asia	17.85
Panama	America	12.01
Paraguay	America	16.33
Peru	America	13.04
Philippines	Asia	18.05
Poland	Europa	13.69
Portugal	Europa	13.63
Qatar	Asia	17.66
Reunion	Africa	14.01
Romania	Europa	13.23
Russian Federation	Asia	15.05
Saint Lucia	America	13.80
San Marino	Europa	11.34
Saudi Arabia	Asia	13.39
Senegal	Africa	14.32
Serbia	Europa	16.49
Seychelles	Africa	13.50
Singapore	Asia	17.94
Slovakia	Europa	13.60
Slovenia	Europa	14.62
South Africa	Africa	16.83
Sri Lanka	Asia	13.02
Sudan	Africa	18.39
Sweden	Europa	14.49
Switzerland	Europa	13.86
Syrian Arab Republic	Asia	16.99
Taiwan	Asia	20.00
Tanzania United Republic	Africa	14.88
Thailand	Asia	18.09
Trinidad and Tobago	America	16.79
Tunisia	Africa	14.28
Turkey	Asia	13.01
Uganda	Africa	15.50
Ukraine	Europa	12.61
United Arab Emirates	Asia	18.07
United Kingdom	Europa	14.03
United States	America	13.81
Uruguay	America	15.07
Uzbekistan	Asia	21.50
Venezuela	America	15.64
Vietnam	Asia	15.28
Yemen	Asia	13.68

El estudio sobre el número de saltos medio no será en detalle sino más bien como curiosidad, ya que con el análisis del PI, RTT y tráfico recibido por país se logra una buena caracterización de las rutas examinadas. Siguiendo esta consideración, se observa en la Figura 5.6 el número de saltos medio ($\overline{\#saltos}$) de Madrid a cada país/región administrativa bajo estudio.

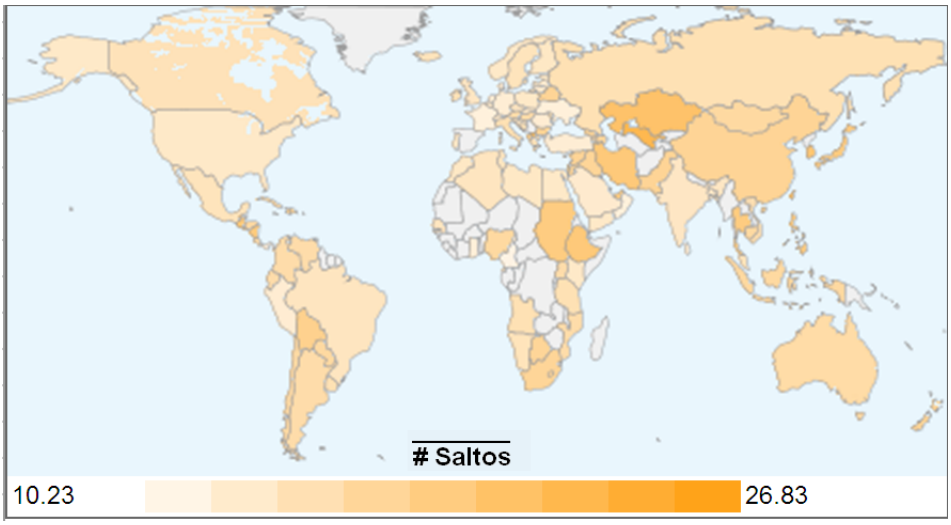


Figura 5.6: Mapa del $\overline{\#saltos}$ de Madrid a cada país/región administrativa bajo estudio

6

Conclusiones y trabajo futuro

6.1. Conclusiones

A lo largo de este proyecto fin de carrera se ha intentado comprender lo que es el Path Inflation y su gran influencia en el rendimiento de las redes de comunicaciones. Se ha identificado la relación directa del Path Inflation en el tiempo de transmisión entre dos enlaces, y cómo dicho Path Inflation es determinante en su aumento.

En el estudio realizado se ha calculado el Path Inflation en función del tráfico recibido (PIT) en 146 países. Estos países son los más populares a nivel de volumen de tráfico desde RedIRIS durante el período de tiempo en el cuál se basa nuestro estudio. El Capítulo 3 explica los procedimiento realizados para conseguir estos cálculos.

En el Capítulo 4 se analiza el Path Inflation sin influencia de tráfico en los países bajo estudio y se extraen resultados interesantes. Se observa que los paquetes que envíamos desde Madrid toman con frecuencia una ruta que incluye Estados Unidos. Este hecho se produce en la mayoría de países bajo estudio sin importar su localización, lo que provoca diferentes patrones de Path Inflation en cada continente debido a su posición respecto a Madrid (origen de las trazas) y a Estados Unidos (enlace usado con frecuencia). A consecuencia de esto, los países más cercanos a Madrid sufren mayor Path Inflation en Europa y África. En Asia y Oceanía este efecto es casi despreciable al tener una localización casi simétrica con Estados Unidos donde Madrid sería el centro de simetría. América es el continente con menor Path Inflation al contener a Estados Unidos.

La segunda parte del Capítulo 4 trata de cuantificar el PIT. Para ello se aplica la métrica (4.15) en los países bajo estudio y se analizan los resultados. Al existir el efecto comentado del uso de Estados Unidos como enlace típico en las comunicaciones, existe gran variabilidad en valores de PIT. Se realiza un primer estudio a nivel continental para descubrir países críticos por continente y finalmente a nivel mundial.

A nivel mundial, los tres países más críticos son Alemania, Reino Unido y Francia con un CR superior a 20. Estos países de origen europeo obtienen este CR debido al alto PI y a encontrarse entre los 4 países que mayor tráfico reciben. Con un CR superior a 15 está Italia, país con un PI similar a los anteriores, pero que recibe menor cantidad de tráfico. Con un CR superior a 10 se encuentra Estados Unidos, país que recibe la cuarta parte del tráfico total. Esta última característica predomina en el CR ya que su PI es muy bajo comparado con otros países. Con un CR entre 5 y 8 puntos se encuentra Portugal, Suiza y Rusia. Con un CR menor de 5 puntos están Bélgica, Holanda, Suecia, Taiwan, Austria, Colombia, Hungría, Argentina y México.

Se concluye que Europa es el continente con el mayor número de países críticos en PIT (con 11 países críticos). América aporta a Estados Unidos, Colombia, Argentina y México. En Asia, Taiwan y Rusia. Oceanía y África no poseen ningún país cuyo PIT sea crítico. En este estudio Oceanía y África son ejemplos de compromiso entre PI y tráfico.

Profundizando más en el efecto del uso del enlace de Estados Unidos para alcanzar un destino, he comprobado mediante mi herramienta de visualizado de rutas, que existen direcciones IP independientemente de su ciudad región/país que utilizan el comentado enlace y otras que no. Esto me lleva a deducir que las tablas de enrutado de BGP están construídas sin tener en cuenta el Path Inflation, ya que no parece lógico que dos direcciones IP de la misma ciudad sigan rutas tan diferentes consiguiendo por tanto un Path Inflation respectivamente muy variable. Esto es sencillo de ver en la BBDDs de GeoIp utilizada, buscando un ciudad y viendo la cantidad de rangos distintos de direcciones IP que pertenecen a una ciudad, y también que los rangos adyacentes pueden pertenecer a ciudades muy distantes a la original.

La reducción del Path Inflation por tanto se llevaría a cabo en un sistema más optimizado de tablas de enrutado, conclusión a la que ha llegado gran parte del sector de investigación acerca de este tema, proponiendo mecanismos dinámicos para la construcción de estas tablas que comprendan el tráfico presente en la comunicación. Esto comprendería también los acuerdos de los ISPs al intercambiar tráfico, mediante el uso de estrategias que tengan en cuenta el Path Inflation y el tráfico.

En el Capítulo 6 se estudian el retardo y el número de saltos medio desde Madrid a cada uno de los páises más populares de RedIRIS. Respecto al retardo, se realiza un estudio incluyendo el RTT y el tráfico recibido por los países más populares de RedIRIS. Para esto se utiliza el algoritmo de clusterización k-means, obteniendo 4 regiones que consiguen agrupar los países con características similares. Afortunadamente, se encuentran los países con mayor tráfico recibido en la Región 1 cuyo RTT medio es menor, síntoma de que el RTT es aceptable en los países más populares de RedIRIS. Los países cuyo tráfico recibido es medio se encuentran en la Región 2 y en la Región 3 alcanzando un RTT máximo menor que 500 ms. El estudio de saltos no es detallado ya que considero al RTT y el PI caracterísiticas suficientes para analizar las rutas en Internet.

La conclusión más importante de este estudio es que RedIRIS no está bien conectada con los países europeos que mayor tráfico reciben (Alemania, Reino Unido, Francia, Italia, Portugal y Suiza). Respecto a los demás países, no se considera que sea alarmante la longitud de las distancias de enrutamiento teniendo en cuenta el tráfico que reciben.

6.2. Trabajo futuro

Hemos visto que la métrica propuesta permite clasificar los destinos de tráfico más populares de una red comercial en producción, por lo que es interesante aplicar el estudio en otras redes. El método de identificación de localizaciones críticas en el intercambio de tráfico con una red, necesita del volumen de tráfico envíado desde la red en un período suficientemente amplio para asegurar estabilidad y la localización a nivel de dirección IP. Estos datos se pueden extraer del exportador de tráfico de los PoP de la red.

Los procedimientos para conseguir resultados en este proyecto, contienen una gran programación que se comparte en el anexo del manual del programador. Se facilita el código de cada paso realizado para el estudio de una red comercial cualquiera de la que se dispone un Netflow suficiente. Se comparte también las trazas de cada dirección IP bajo estudio, para el uso que se crea oportuno.

Se proporcionan funciones que pueden ser útiles para identificar el volumen de tráfico de cada país, dibujar la ruta que sigue una traza de traceroute, calcular distancias de enrutamiento, geográficas y tiempos de transmisión, extraer estadísticas de carpetas con logs de pings, que

pueden ser empleadas para facilitar estudios de muchos tipos en redes comerciales en producción.

Con miras al futuro, sería interesante el estudio del Path Inflation a nivel de AS. El estudio a estos niveles comprendería conclusiones importantes en el rendimiento de un AS que es normalmente gestionado por un único ISP. Esta cuestión facilitaría la toma de decisiones para la reducción del Path Inflation.

Bibliografía

[arta] http://www.hostip.info/.

[artb] http://www.ip2country.net/.

[artc] http://www.ipinfodb.com/.

[artd] http://www.maxmind.com/.

[BGW05] S. Bar, M. Gonen, and A. Wool. A geographic directed preferencial internet topology model. *IEEE*, 2005.

[CK] D. Chatzopoulou and M. Kokkodis. Ip geolocation. *Comput Science and Engineering Dept, UC Riverside*.

[CKZ05a] B. Chen, S. Kumar, and W. Zhong. A path inflation control strategy for dynamic traffic grooming in ip/mpls over wdm network. *IEEE*, 2005.

[CKZ05b] B. Chen, S. Kumar, and W. Zhong. Priority enabled dynamic traffic grooming. *IEEE*, 2005.

[DA04] David Domingo Alegre. Estudio de la evolución de la topología de internet a partir de tablas bgp. 2004.

[FGL+01] A. Feldmann, A. Greenberg, C. Lund, N. Reingold, J. Rexford, and F. True. Deriving traffic demands for operational ip networks: methodology and experience. *IEEE/ACM Trans. Netw., IEEE Press*, pages 265–280, 2001.

[GALM08] P. Gill, M. Arlitt, Z. Li, and A. Mahanti. The flattening internet topology: natural evolution, unsightly barnacles or contrived collapse? *PAM'08: Proceedings of the 9th international conference on Passive and active network measurement, Springer-Verlag*, pages 1–10, 2008.

[GD04] José M. García and José Duato. Estudio de la reconfiguración dinámica de la topología de interconexión en una red de transputers. 2004.

[GW02] L. Gao and F. Wang. The extent of as path inflation by routing policies. *Global Telecommunications Conference, 2002. GLOBECOM '02. IEEE*, pages 2180 – 2184 vol.3, 2002.

[GZCF04] B. Gueye, A. Ziviani, M. Crovella, and S. Fdida. Constraint-based geolocation of internet hosts. *In IMC 04: Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*, pages 288–293, 2004.

[GZCF06] B. Gueye, A. Ziviani, M. Crovella, and S. Fdida. Constraint-based geolocation of internet hosts. *IEEE/ACM Trans. Netw., 14(6)*, pages 1219–1232, 2006.

[KBJK+06] E. Katz-Bassett, J.P. John, A. Krishnamurthy, D. Wetherall, T. Anderson, and Y. Chawathe. Towards ip geolocation using delay and topology measurements. *In IMC 06: Proceedings of the 6th ACM SIGCOMM conference on Internet measurement*, pages 71–84, 2006.

[LCG+] Dan Li, Jiong Chen, Chuanxiong Guo, Yunxin Liu, Jinyu Zhang, Zhili Zhang, and Yongguang Zhang. Ip-geolocation mapping for involving moderately-connected internet regions.

[MGDALdVon] F. Mata, J.L. García-Dorado, J. Aracil, and J.E. López de Vergara. Factor analysis of internet traffic destinations from similar source campus networks. *Internet Research*, Publicación en preparación.

[PAKD+11] Ingmar Poese, Mohamed Ali Kaafar, Benoit Donnet, Bamba Gueye, and Steve Uhlig. Ip geolocation databases: Unreliable? *Technical Report 2011-03*, 2011.

[Pax96] Vern Paxson. End-to-end routing behavior in the internet. *In Proceedings of the ACM SIGCOMM '96*, pages 25–38, 1996.

[PS01] V. N. Padmanabhan and L. Subramanian. An investigation of geographic mapping techniques for internet hosts. *In ACM SIGCOMM*, pages 173–185, 2001.

[PZMH07] Himabindu Pucha, Ying Zhang, Z. Morley Mao, and Y. Charlie Hu. Understanding network delay changes caused by routing events. pages 73–84, 2007. ISBN 978-1-59593-639-4. URL `http://doi.acm.org/10.1145/1254882.1254891`.

[Sav99] S. Savage. The end-to-end effects of internet path selection. *ACM SIGCOMM*, 1999.

[SMA03] Neil Spring, Ratul Mahajan, and Thomas Anderson. The causes of path inflation. pages 113–124, 2003. ISBN 1-58113-735-4. URL `http://doi.acm.org/10.1145/863955.863970`.

[SPK01] L. Subramanian, V. N. Padmanabhan, and R. H. Katz. Geographic properties of internet routing: Analysis and implications. *Microsoft Research*, 2001.

[TGSE01] H. Tangmunarunkit, R. Govindan, S. Shenker, and D. Estrin. Internet path inflation due to policy routing. *Proc. SPIE International Symposium on Convergence of IT and Communication (ITCom)*, 2001.

[WSS06] B. Wong, I. Stoyanov, and E. Sirer. Ip-geolocation mapping for involving moderately-connected internet regions. *In Proceedings of ACM IMC 06*, 2006.

[ZFdRD05] A. Ziviani, S. Fdida, J. F. de Rezende, and O.C.M.B. Duarte. Improving the accuracy of measurement-based geographic location of internet hosts. *Comput. Netw. ISDN Syst., 47*, pages 503–523, 2005.

[ZLPG05] H. Zheng, E. K. Lua, M. Pias, and Griffin. Internet routing policies and round-trip-times. *Passive and Active Network Measurement, Springer Berlin / Heidelberg*, pages 236–250, 2005.

[ZZX01] L. Zheng, L. Zhang, and D. Xu. Characteristics of network delay and delay jitter and its effect on voice over ip (voip). *communications, 2001. ICC 2001. IEEE International Conference on*, pages 122–126, 2001.

A

Manual de utilización

A.1. Manual de Traceroute

TRACEROUTE(8) TRACEROUTE(8)

NAME
traceroute - print the route packets take to network host

SYNOPSIS
traceroute [-Sdnrv] [-g gw_host] [-m max_ttl] [-p port]
[-q nqueries] [-s src_addr] [-t tos] [-w waittime] host [packetlen]

DESCRIPTION
The Internet is a large and complex aggregation of network hardware, connected together by gateways. Tracking the route one's packets follow (or finding the miscreant gateway that's discarding your packets) can be difficult. Traceroute utilizes the IP protocol 'time to live' field and attempts to elicit an ICMP TIME_EXCEEDED response from each gateway along the path to some host.

The only mandatory parameter is the destination host name or IP number. The default probe datagram length is 40 bytes, but this may be increased by specifying a packet length (in bytes) after the destination host name.

Other options are:

-S Print a summary of how many probes were not answered for each hop.

-g Specify a loose source route gateway (8 maximum).

-m Set the max time-to-live (max number of hops) used in outgoing probe packets. The default is 30 hops (the same default used for TCP connections).

-n Print hop addresses numerically rather than symbol- ically and numerically (saves a nameserver address- to-name lookup for each gateway found on the path).

-p Set the base UDP port number used in probes (default is 33434). Traceroute hopes that nothing is listening on UDP ports base to base + nhops - 1 at the destination host (so an ICMP PORT_UNREACH- ABLE message will be returned to terminate the route tracing). If something is listening on a port in the default range, this option can be used to pick an unused port range.

-r Bypass the normal routing tables and send directly to a host on an attached

network. If the host is not on a directly-attached network, an error is returned. This option can be used to ping a local host through an interface that has no route through it (e.g., after the interface was dropped by routed(8C)).

-s Use the following IP address (which must be given as an IP number, not a hostname) as the source address in outgoing probe packets. On hosts with more than one IP address, this option can be used to force the source address to be something other than the IP address of the interface the probe packet is sent on. If the IP address is not one of this machine's interface addresses, an error is returned and nothing is sent.

-t Set the type-of-service in probe packets to the following value (default zero). The value must be a decimal integer in the range 0 to 255. This option can be used to see if different types-of- service result in different paths. (If you are not running 4.4bsd, this may be academic since the nor- mal network services like telnet and ftp don't let you control the TOS). Not all values of TOS are legal or meaningful - see the IP spec for defini- tions. Useful values are probably '-t 16' (low delay) and '-t 8' (high throughput).

-v Verbose output. Received ICMP packets other than TIME_EXCEEDED and UNREACHABLEs are listed.

-w Set the time (in seconds) to wait for a response to a probe (default 5 sec.).

This program attempts to trace the route an IP packet would follow to some internet host by launching UDP probe packets with a small ttl (time to live) then listening for an ICMP "time exceeded'' reply from a gateway. We start our probes with a ttl of one and increase by one until we get an ICMP "port unreachable'' (which means we got to "host'') or hit a max (which defaults to 30 hops can be changed with the -m flag). Three probes (change with -q flag) are sent at each ttl setting and a line is printed showing the ttl, address of the gateway and round trip time of each probe. If the probe answers come from dif- ferent gateways, the address of each responding system will be printed. If there is no response within a 5 sec. timeout interval (changed with the -w flag), a "*'' is printed for that probe.

We don't want the destination host to process the UDP probe packets so the destination port is set to an unlikely value (if some clod on the destination is using that value, it can be changed with the -p flag).

SEE ALSO netstat(1), ping(8)

AUTHOR
Implemented by Van Jacobson from a suggestion by Steve Deering. Debugged by a cast of thousands with particu- larly cogent suggestions or fixes from C. Philip Wood, Tim

27 September 1996 4

TRACEROUTE(8) TRACEROUTE(8)

Seaver and Ken Adelman.

The current version is available via anonymous ftp:

ftp://ftp.ee.lbl.gov/traceroute.tar.Z

BUGS Please send bug reports to traceroute@ee.lbl.gov.

27 September 1996

A.2. Manual de TCPTraceroute

TCPTRACEROUTE(8) TCPTRACEROUTE(8)

NAME

tcptraceroute - A traceroute implementation using TCP packets

SYNOPSIS

tcptraceroute [-nNFSAE] [-i interface] [-f first ttl] [-l length] [-q number of queries] [-t tos] [-m max ttl] [-p source port] [-s source address] [-w wait time] host [destination port] [length]

DESCRIPTION

tcptraceroute is a traceroute implementation using TCP packets. The more traditional traceroute(8) sends out either UDP or ICMP ECHO packets with a TTL of one, and increments the TTL until the destination has been reached. By printing the gateways that generate ICMP time exceeded messages along the way, it is able to determine the path packets are taking to reach the destination.

The problem is that with the widespread use of firewalls on the modern Internet, many of the packets that traceroute(8) sends out end up being filtered, making it impossible to completely trace the path to the destination. However, in many cases, these firewalls will permit inbound TCP packets to specific ports that hosts sitting behind the firewall are listening for connections on. By sending out TCP SYN packets instead of UDP or ICMP ECHO packets, tcptraceroute is able to bypass the most common firewall filters.

It is worth noting that tcptraceroute never completely establishes a TCP connection with the destination host. If the host is not listening for incoming connections, it will respond with an RST indicating that the port is closed. If the host instead responds with a SYN|ACK, the port is known to be open, and an RST is sent by the kernel tcptraceroute is running on to tear down the connection without completing three-way handshake. This is the same half-open scanning technique that nmap(1) uses when passed the -sS flag.

OPTIONS

-n Display numeric output, rather than doing a reverse DNS lookup for each hop. By default, reverse lookups are never attempted on RFC1918 address space, regardless of the -n flag.

-N Perform a reverse DNS lookup for each hop, including RFC1918 addresses.

-f Set the initial TTL used in the first outgoing packet. The default is 1.

-m Set the maximum TTL used in outgoing packets. The default is 30.

-p Use the specified local TCP port in outgoing packets. The default is to obtain a free port from the kernel using bind(2) Unlike with traditional traceroute(8), this number will not increase with each hop.

-s Set the source address for outgoing packets. See also the -i flag.

-i Use the specified interface for outgoing packets.

-q Set the number of probes to be sent to each hop. The default is 3.

-w Set the timeout, in seconds, to wait for a response for each probe. The default is 3.

-S Set the TCP SYN flag in outgoing packets. This is the default, if neither -S or -A is specified.

-A Set the TCP ACK flag in outgoing packets. By doing so, it is possible to trace through stateless firewalls which permit outgoing TCP connections.

-E Send ECN SYN packets, as described in RFC2481.

-t Set the IP TOS (type of service) to be used in outgoing packets. The default is not to set any TOS.

-F Set the IP "don't fragment'' bit in outgoing packets.

-l Set the total packet length to be used in outgoing packets. If the length is greater than the minimum size required to assemble the necessary probe packet headers, this value is automatically increased.

-d Enable debugging, which may or may not be useful.

AUTHOR
Michael C. Toren <mct@toren.net> For updates, please see:

http://michael.toren.net/code/tcptraceroute/

SEE ALSO traceroute(8), ping(8), nmap(1)

A.3. Manual de Ping

NAME
ping, ping6 - send ICMP ECHO_REQUEST to network hosts EXAMPLES SYNOPSIS
ping [-LRUbdfnqrvVaAB] [-c count] [-i interval] [-l preload] [-p pattern] [-s packetsize] [-t ttl] [-w deadline] [-F flowlabel] [-I interface] [-M hint] [-Q tos] [-S sndbuf] [-T timestamp option] [-W timeout] [hop ...] destination

DESCRIPTION
ping uses the ICMP protocol's mandatory ECHO_REQUEST datagram to elicit an ICMP ECHO_RESPONSE from a host or gateway. ECHO_REQUEST datagrams (''pings'') have an IP and ICMP header, followed by a struct timeval and then an arbitrary number of ''pad'' bytes used to fill out the packet.
OPTIONS
-a Audible ping.

-A Adaptive ping. Interpacket interval adapts to round-trip time, so that effectively not more than one (or more, if preload is set) unanswered probes present in the network. Minimal interval is 200msec for not super-user. On networks with low rtt this mode is essentially equivalent to flood mode.

-b Allow pinging a broadcast address.

-B Do not allow ping to change source address of probes. The address is bound to one selected when ping starts.

-c count Stop after sending count ECHO_REQUEST packets. With deadline option, ping waits for count ECHO_REPLY packets, until the timeout expires.

-d Set the SO_DEBUG option on the socket being used. Essentially, this socket option is not used by Linux kernel.

-F flow label Allocate and set 20 bit flow label on echo request packets. (Only ping6). If value is zero, kernel allocates random flow label.

-f Flood ping. For every ECHO_REQUEST sent a period ''.'' is printed, while for ever ECHO_REPLY received a backspace is printed. This provides a rapid display of how many packets are being dropped. If interval is not given, it sets interval to zero and outputs packets as fast as they come back or one hundred times per second, whichever is more. Only the super-user may use this option with zero interval.

-i interval Wait interval seconds between sending each packet. The default is to wait for one second between each packet normally, or not to wait in flood mode. Only super-user may set interval to values less 0.2 seconds.

-I interface address Set source address to specified interface address. Argument may be numeric IP address or name of device. When pinging IPv6 link-local address this option is required.

-l preload If preload is specified, ping sends that many packets not waiting for reply. Only the super-user may select preload more than 3.

-L Suppress loopback of multicast packets. This flag only applies if the ping destination is a multicast address.

-n Numeric output only. No attempt will be made to lookup symbolic names for host addresses.

-p pattern You may specify up to 16 ‘‘pad’’ bytes to fill out the packet you send. This is useful for diagnosing data-dependent problems in a network. For example, -p ff will cause the sent packet to be filled with all ones.

-Q tos Set Quality of Service -related bits in ICMP datagrams. tos can be either decimal or hex number. Traditionally (RFC1349), these have been interpreted as: 0 for reserved (currently being redefined as congestion control), 1-4 for Type of Service and 5-7 for Precedence. Possible settings for Type of Service are: minimal cost: 0x02, reliability: 0x04, throughput: 0x08, low delay: 0x10. Multiple TOS bits should not be set simultaneously. Possible settings for special Precedence range from priority (0x20) to net control (0xe0). You must be root (CAP_NET_ADMIN capability) to use Critical or higher precedence value. You cannot set bit 0x01 (reserved) unless ECN has been enabled in the kernel. In RFC2474, these fields has been redefined as 8-bit Differentiated Services (DS), consisting of: bits 0-1 of separate data (ECN will be used, here), and bits 2-7 of Differentiated Services Codepoint (DSCP).

-q Quiet output. Nothing is displayed except the summary lines at startup time and when finished.

-R Record route. Includes the RECORD_ROUTE option in the ECHO_REQUEST packet and displays the route buffer on returned packets. Note that the IP header is only large enough for nine such routes. Many hosts ignore or discard this option.

-r Bypass the normal routing tables and send directly to a host on an attached interface. If the host is not on a directly-attached network, an error is returned. This option can be used to ping a local host through an interface that has no route through it provided the option -I is also used.

-s packetsize Specifies the number of data bytes to be sent. The default is 56, which translates into 64 ICMP data bytes when combined with the 8 bytes of ICMP header data.

-S sndbuf Set socket sndbuf. If not specified, it is selected to buffer not more than one packet.

-t ttl Set the IP Time to Live.

-T timestamp option Set special IP timestamp options. timestamp option may be either tsonly (only timestamps), tsandaddr (timestamps and addresses) or tsprespec host1 [host2 [host3 [host4]]] (timestamp prespecified hops).

-M hint Select Path MTU Discovery strategy. hint may be either do (prohibit fragmentation, even local one), want (do PMTU discovery, fragment locally when packet size is large), or dont (do not set DF flag).

-U Print full user-to-user latency (the old behaviour). Normally ping prints network round trip time, which can be different f.e. due to DNS failures.

-v Verbose output.

-V Show version and exit.

-w deadline Specify a timeout, in seconds, before ping exits regardless of how many packets have been sent or received. In this case ping does not stop after count packet are sent, it waits either for deadline expire or until count probes are answered or for some error notification from network.

-W timeout Time to wait for a response, in seconds. The option affects only timeout in absense of any responses, otherwise ping waits for two RTTs. When using ping for fault isolation, it should first be run on the local host, to verify that the local network interface is up and running. Then, hosts and gateways further and further away should be ‘‘pinged’’. Round-trip times and packet loss statistics are computed. If duplicate packets are received, they are not included in the packet loss calculation, although the round trip time of these packets is used in calculating the minimum/average/maxi round-trip time numbers. When the specified number of packets have been sent (and received) or if the program is terminated with a SIGINT, a brief summary is displayed. Shorter current statistics can be obtained without termination of process with signal SIGQUIT.

If ping does not receive any reply packets at all it will exit with code 1. If a packet count and deadline are both specified, and fewer than count packets are received by the time the deadline has arrived, it will also exit with code 1. On other error it exits with code 2. Otherwise it exits with code 0. This makes it possible to use the exit code to see if a host is alive or not.

This program is intended for use in network testing, measurement and management. Because of the load it can impose on the network, it is unwise to use ping during normal operations or from automated scripts. ICMP PACKET DETAILS An IP header without options is 20 bytes. An ICMP ECHO_REQUEST packet contains an additional 8 bytes worth of ICMP header followed by an arbitrary amount of data. When a packetsize is given, this indicated the size of this extra piece of data (the default is 56). Thus the amount of data received inside of an IP packet of type ICMP ECHO_REPLY will always be 8 bytes more than the requested data space (the ICMP header).

If the data space is at least of size of struct timeval ping uses the beginning bytes of this space to include a timestamp which it uses in the computation of round trip times. If the data space is shorter, no round trip times are given. DUPLICATE AND DAMAGED PACKETS ping will report duplicate and damaged packets. Duplicate packets should never occur, and seem to be caused by inappropriate link-level retransmissions. Duplicates may occur in many situations and are rarely (if ever) a good sign, although the presence of low levels of duplicates may not always be cause for alarm.

SEE ALSO netstat(1), ifconfig(8).

A.4. Manual de HPing3

NAME
hping3 - send (almost) arbitrary TCP/IP packets to network hosts

SYNOPSIS
hping3 [-hvnqVDzZ012WrfxykQbFSRPAUXYjJBuTG] [-c count] [-i wait] [-fast] [-I interface] [-9 signature] [-a host] [-t ttl] [-N ip id] [-H ip protocol] [-g fragoff] [-m mtu] [-o tos] [-C icmp type] [-K icmp code] [-s source port] [-p[+][+] dest port] [-w tcp window] [-O tcp offset] [-M tcp sequence number] [-L tcp ack] [-d data size] [-E filename] [-e signature] [-icmp-ipver version] [-icmp-iphlen length] [-icmp-iplen length] [-icmp-ipid id] [-icmp-ipproto protocol] [-icmp-cksum checksum] [-icmp-ts] [-icmp-addr] [-tcpexitcode] [-tcp-timestamp] [-tr-stop] [-tr-keep-ttl] [-tr-no-rtt] [-rand-dest] [-rand-source] [-beep] hostname

DESCRIPTION
hping3 is a network tool able to send custom TCP/IP packets and to display target replies like ping program does with ICMP replies. hping3 handle fragmentation, arbitrary packets body and size and can be used in order to transfer files encapsulated under supported protocols. Using hping3 you are able to perform at least the following stuff:

- Test firewall rules
- Advanced port scanning
- Test net performance using different protocols, packet size, TOS (type of service) and fragmentation.
- Path MTU discovery
- Transferring files between even really fascist firewall rules.
- Traceroute-like under different protocols.
- Firewalk-like usage.
- Remote OS fingerprinting.
- TCP/IP stack auditing.
- A lot of others.

It is also a good didactic tool to learn TCP/IP. hping3 is developed and maintained by antirez@invece.org and is licensed under GPL version 2. Development is open so you can send me patches, suggestion and affronts without inhibitions.

HPING SITE
primary site at http://www.hping.org. You can found both the stable release and the instruction to download the latest source code at http://www.hping.org/download.html

BASE OPTIONS
-h -help Show an help screen on standard output, so you can pipe to less.

-v -version Show version information and API used to access to data link layer, linux sock packet or libpcap.

-c -count count Stop after sending (and receiving) count response packets. After last packet was send hping3 wait COUNTREACHED_TIMEOUT seconds target host replies. You are able to tune COUNTREACHED_TIMEOUT editing hping3.h

-i -interval Wait the specified number of seconds or micro seconds between sending each packet. -interval X set wait to X seconds, -interval uX set wait to X micro seconds. The default is to wait one second between each packet. Using hping3 to transfer files tune this option is really important in order to increase transfer rate. Even using hping3 to perform idle/spoofing scanning you should tune this option, see HPING3-HOWTO for more information.

-fast Alias for -i u10000. Hping will send 10 packets for second.

-faster Alias for -i u1. Faster then -fast ;) (but not as fast as your computer can send packets due to the signal-driven design).

-flood Sent packets as fast as possible, without taking care to show incoming replies. This is ways faster than to specify the -i u0 option.

-n -numeric Numeric output only, No attempt will be made to lookup symbolic names for host addresses.

-q -quiet Quiet output. Nothing is displayed except the summary lines at startup time and when finished.

-I -interface interface name By default on linux and BSD systems hping3 uses default routing interface. In other systems or when there is no default route hping3 uses the first non-loopback interface. However you are able to force hping3 to use the interface you need using this option. Note: you do not need to specify the whole name, for example -I et will match eth0 ethernet0 myet1 et cetera. If no interfaces match hping3 will try to use lo.

-V -verbose Enable verbose output. TCP replies will be shown as follows: len=46 ip=192.168.1.1 flags=RA DF seq=0 ttl=255 id=0 win=0 rtt=0.4 ms tos=0 iplen=40 seq=0 ack=1380893504 sum=2010 urp=0

-D -debug Enable debug mode, it is useful when you experience some problem with hping3. When debug mode is enabled you will get more information about interface detection, data link layer access, interface settings, options parsing, fragmentation, HCMP protocol and other stuff.

-z -bind Bind CTRL+Z to time to live (TTL) so you will able to increment/decrement ttl of outgoing packets pressing CTRL+Z once or twice.

-Z -unbind Unbind CTRL+Z so you will able to stop hping3.

-beep Beep for every matching received packet (but not for ICMP errors).

PROTOCOL SELECTION
Default protocol is TCP, by default hping3 will send tcp headers to target hosts port 0 with a winsize of 64 without any tcp flag on. Often this is the best way to do an hide ping, useful when target is behind a firewall that drop ICMP. Moreover a tcp null-flag to port 0 has a good probability of not being logged.

-0 -rawip RAW IP mode, in this mode hping3 will send IP header with data appended with -signature and/or -file, see also -ipproto that allows you to set the ip protocol field.

-1 -icmp ICMP mode, by default hping3 will send ICMP echo-request, you can set other ICMP type/code using -icmptype -icmpcode options.

-2 -udp UDP mode, by default hping3 will send udp to target hostś port 0. UDP header tunable options are the following: -baseport, -destport, -keep.

-8 -scan Scan mode, the option expects an argument that describes groups of ports to scan. port groups are comma separated: a number describes just a single port, so 1,2,3 means port 1, 2 and 3. ranges are specified using a start-end notation, like

1-1000, that tell hping to scan ports between 1 and 1000 (included). the special word all is an alias for 0-65535, while the special word known includes all the ports listed in /etc/services. Groups can be combined, so the following command line will scan ports between 1 and 1000 AND port 8888 AND ports listed in /etc/services: hping -scan 1-1000,8888,known -S target.host.com Groups can be negated (subtracted) using a ! character as prefix, so the following command line will scan all the ports NOT listed in /etc/services in the range 1-1024: hping -scan 1-1024,!known -S target.host.com Keep in mind that while hping seems much more like a port scanner in this mode, most of the hping switches are still honored, so for example to perform a SYN scan you need to specify the -S option, you can change the TCP windows size, TTL, control the IP fragmentation as usually, and so on. The only real difference is that the standard hping behaviors are encapsulated into a scanning algorithm. Tech note: The scan mode uses a two-processes design, with shared memory for synchronization. The scanning algorithm is still not optimal, but already quite fast. Hint: unlike most scanners, hping shows some interesting info about received packets, the IP ID, TCP win, TTL, and so on, don not forget to look at this additional information when you perform a scan! Sometimes they shows interesting details. -9 -listen signature hping3 listen mode, using this option hping3 waits for packet that contain signature and dump from signature end to packets end. For example if hping3 -listen TEST reads a packet that contain 234-09sdflkjs45-TESThello_world it will display hello_world.

IP RELATED OPTIONS
-a -spoof hostname Use this option in order to set a fake IP source address, this option ensures that target will not gain your real address. However replies will be sent to spoofed address, so you will cannot see them. In order to see how it is possible to perform spoofed/idle scanning see the hping3-HOWTO.

-rand-source This option enables the random source mode. hping will send packets with random source address. It is interesting to use this option to stress firewall state tables, and other per-ip basis dynamic tables inside the TCP/IP stacks and firewall software.

-rand-dest This option enables the random destination mode. hping will send the packets to random addresses obtained following the rule you specify as the target host. You need to specify a numerical IP address as target host like 10.0.0.x. All the occurrences of x will be replaced with a random number in the range 0-255. So to obtain Internet IP addresses in the whole IPv4 space use something like hping x.x.x.x -rand-dest. If you are not sure about what kind of addresses your rule is generating try to use the -debug switch to display every new destination address generated. When this option is turned on, matching packets will be accept from all the destinations. Warning: when this option is enabled hping cannot detect the right outgoing interface for the packets, so you should use the -interface option to select the desired outgoing interface.

-t -ttl time to live Using this option you can set TTL (time to live) of outgoing packets, it is likely that you will use this with -traceroute or -bind options. If in doubt try hping3 some.host.com -t 1 -traceroute.

-N -id Set ip->id field. Default id is random but if fragmentation is turned on and id is not specified it will be getpid() 0xFF, to implement a better solution is in TODO list.

-H -ipproto Set the ip protocol in RAW IP mode.

-W -winid id from Windows* systems before Win2k has different byte ordering, if this option is enable hping3 will properly display id replies from those Windows.

-r -rel Display id increments instead of id. See the hping3-HOWTO for more information. Increments arent́ computed as id[N]-id[N-1] but using packet loss compensation. See

relid.c for more information.

-f -frag Split packets in more fragments, this may be useful in order to test IP stacks fragmentation performance and to test if some packet filter is so weak that can be passed using tiny fragments (anachronistic). Default virtual mtu is 16 bytes. see also -mtu option.

-x -morefrag Set more fragments IP flag, use this option if you want that target host send an ICMP time-exceeded during reassembly.

-y -dontfrag Set do not fragment IP flag, this can be used to perform MTU path discovery.

-g -fragoff fragment offset value Set the fragment offset.

-m -mtu mtu value Set different virtual mtu than 16 when fragmentation is enabled. If packets size is greater that virtual mtu fragmentation is automatically turned on.

-o -tos hex_tos Set Type Of Service (TOS), for more information try -tos help.

-G -rroute Record route. Includes the RECORD_ROUTE option in each packet sent and displays the route buffer of returned packets. Note that the IP header is only large enough for nine such routes. Many hosts ignore or discard this option. Also note that using hping you are able to use record route even if target host filter ICMP. Record route is an IP option, not an ICMP option, so you can use record route option even in TCP and UDP mode.

ICMP RELATED OPTIONS
-C -icmptype type Set icmp type, default is ICMP echo request (implies -icmp).

-K -icmpcode code Set icmp code, default is 0 (implies -icmp).

-icmp-ipver Set IP version of IP header contained into ICMP data, default is 4.

-icmp-iphlen Set IP header length of IP header contained into ICMP data, default is 5 (5 words of 32 bits).

-icmp-iplen Set IP packet length of IP header contained into ICMP data, default is the real length.

-icmp-ipid Set IP id of IP header contained into ICMP data, default is random.

-icmp-ipproto Set IP protocol of IP header contained into ICMP data, default is TCP.

-icmp-cksum Set ICMP checksum, for default is the valid checksum.

-icmp-ts Alias for -icmptype 13 (to send ICMP timestamp requests).

-icmp-addr Alias for -icmptype 17 (to send ICMP address mask requests).

TCP/UDP RELATED OPTIONS
-s -baseport source port hping3 uses source port in order to guess replies sequence number. It starts with a base source port number, and increase this number for each packet sent. When packet is received sequence number can be computed as replies.dest.port - base.source.port. Default base source port is random, using this option you are able to set different number. If you need that source port not be increased for each sent packet use the -k -keep option.

-p -destport [+][+]dest port Set destination port, default is 0. If + character precedes dest port number (i.e. +1024) destination port will be increased for each reply received. If double + precedes dest port number (i.e. ++1024), destination port will be increased for each packet sent. By default destination port can be modified interactively using CTRL+z.

-keep keep still source port, see -baseport for more information.

-w -win Set TCP window size. Default is 64.

-O -tcpoff Set fake tcp data offset. Normal data offset is tcphdrlen / 4.

-M -tcpseq Set the TCP sequence number.

-L -tcpack Set the TCP ack.

-Q -seqnum This option can be used in order to collect sequence numbers generated by target host. This can be useful when you need to analyze whether TCP sequence number is predictable.

-b -badcksum Send packets with a bad UDP/TCP checksum.

-tcp-timestamp Enable the TCP timestamp option, and try to guess the timestamp update frequency and the remote system uptime.

-F -fin Set FIN tcp flag.

-S -syn Set SYN tcp flag.

-R -rst Set RST tcp flag.

-P -push Set PUSH tcp flag.

-A -ack Set ACK tcp flag.

-U -urg Set URG tcp flag.

-X -xmas Set Xmas tcp flag.

-Y -ymas Set Ymas tcp flag.

COMMON OPTIONS
-d -data data size Set packet body size. Warning, using -data 40 hping3 will not generate 0 byte packets but protocol_header+40 bytes. hping3 will display packet size information as first line output, like this: HPING www.yahoo.com (ppp0 204.71.200.67): NO FLAGS are set, 40 headers + 40 data bytes

-E -file filename Use filename contents to fill packets data.

-e -sign signature Fill first signature length bytes of data with signature. If the signature length is bigger than data size an error message will be displayed. If you do not specify the data size hping will use the signature size as data size. This option can be used safely with -file filename option, remainder data space will be filled using filename.

-j -dump Dump received packets in hex.

-J -print Dump received packets printable characters.

-B -safe Enable safe protocol, using this option lost packets in file transfers will be resent.

-u -end If you are using -file filename option, tell you when EOF has been reached. Moreover prevent that other end accept more packets. Please, for more information see the hping3-HOWTO.

-T -traceroute Traceroute mode. Using this option hping3 will increase ttl for each ICMP time to live 0 during transit received. Try hping3 host -traceroute. This option implies -bind and -ttl 1. You can override the ttl of 1 using the -ttl option. Since 2.0.0 stable it prints RTT information.

-tr-keep-ttl Keep the TTL fixed in traceroute mode, so you can monitor just one hop in the route. For example, to monitor how the 5th hop changes or how its RTT changes you can try hping3 host -traceroute -ttl 5 -tr-keep-ttl.

-tr-stop If this option is specified hping will exit once the first packet that is not an ICMP time exceeded is received. This better emulates the traceroute behavior.

-tr-no-rtt Do not show RTT information in traceroute mode. The ICMP time exceeded RTT information are not even calculated if this option is set.

-tcpexitcode Exit with last received packet tcp->th_flag as exit code. Useful for scripts that need, for example, to known if the port 999 of some host reply with SYN/ACK or with RST in response to SYN, i.e. the service is up or down.

AUTHOR
Salvatore Sanfilippo <antirez@invece.org>, with the help of the people mentioned in AUTHORS file and at http://www.hping.org/authors.html

BUGS
Even using the -end and -safe options to transfer files the final packet will be padded with 0x00 bytes. Data is read without care about alignment, but alignment is enforced in the data structures. This will not be a problem under i386 but, while usually the TCP/IP headers are naturally aligned, may create problems with different processors and bogus packets if there is some unaligned access around the code (hopefully none).

On solaris hping does not work on the loopback interface. This seems a solaris problem, as stated in the tcpdump-workers mailing list, so the libpcap cant́ do nothing to handle it properly.

SEE ALSO ping(8), traceroute(8), ifconfig(8), nmap(1)

B

Manual del programador

B.1. Funciones de Matlab

B.1.1. ConvertIP

```
%CONVERTIP - convierte IP hexadecimal en decimal para uso en BBDD
%
%function ipd = convertIP(iph)
%
%INPUT-
%iph = direccion IP en formato hexadecimal A.B.C.D
%
%OUTPUT-
%ipd = direccion IP en notacion decimal

function ipd = convertIP(iph)

format compact
i=1;

while(iph(i)~='.')
A(i) = iph(i);
i =i+1;
end

i=i+1;

j=1;
while(iph(i)~='.')
B(j) = iph(i);
i =i+1;
j=j+1;
end

i=i+1;

j=1;
while(iph(i)~='.')
C(j) = iph(i);
i =i+1;
j=j+1;
end

i=i+1;
j=1;
while(i<=length(iph))
D(j) = iph(i);
```

```
i =i+1;
j=j+1;
end

ipd = ((str2num(A)*256+str2num(B))*256+str2num(C))*256 + str2num(D);
```

B.1.2. IP2City

```
%IP2CITY - obtiene la localizacion de la IP (latitud, longitud, ciudad, region, pais)
%
%function [City,Region,Country,Latitude,Longitude] = Ip2City(Ip)
%
%INPUT-
%Ip = Direccion IP en notacion decimal
%
%OUTPUT-
%City = Ciudad
%Region = Region
%Country = Pais
%Latitude = Region
%Longitude = Ciudad

function [City,Region,Country,Latitude,Longitude] = Ip2City(Ip)

if(Ip>=0 & Ip<1070086256)
        if(Ip>=0 & Ip<205848088)

            if(Ip>=0 & Ip<71024640)
            [IpStart,CountryName,RegionName,CityName,Lat,Lon] = textread('IpData/↩
                ip_group_city_1_01_01.csv','%u%s%s%s%f%f','delimiter',';');

            elseif(Ip>=71024640 & Ip<83787776)
            [IpStart,CountryName,RegionName,CityName,Lat,Lon] = textread('IpData/↩
                ip_group_city_1_01_02.csv','%u%s%s%s%f%f','delimiter',';');

            elseif(Ip>=83787776 & Ip<201796608)
            [IpStart,CountryName,RegionName,CityName,Lat,Lon] = textread('IpData/↩
                ip_group_city_1_01_03.csv','%u%s%s%s%f%f','delimiter',';');

            elseif(Ip>=201796608 & Ip<202471112)
            [IpStart,CountryName,RegionName,CityName,Lat,Lon] = textread('IpData/↩
                ip_group_city_1_01_04.csv','%u%s%s%s%f%f','delimiter',';');

            elseif(Ip>=202471112 & Ip<202909600)
            [IpStart,CountryName,RegionName,CityName,Lat,Lon] = textread('IpData/↩
                ip_group_city_1_01_05.csv','%u%s%s%s%f%f','delimiter',';');

            elseif(Ip>=202909600 & Ip<203353984)
            [IpStart,CountryName,RegionName,CityName,Lat,Lon] = textread('IpData/↩
                ip_group_city_1_01_06.csv','%u%s%s%s%f%f','delimiter',';');

            elseif(Ip>=203353984 & Ip<203752000)
            [IpStart,CountryName,RegionName,CityName,Lat,Lon] = textread('IpData/↩
                ip_group_city_1_01_07.csv','%u%s%s%s%f%f','delimiter',';');

            elseif(Ip>=203752000 & Ip<204166240)
            [IpStart,CountryName,RegionName,CityName,Lat,Lon] = textread('IpData/↩
                ip_group_city_1_01_08.csv','%u%s%s%s%f%f','delimiter',';');

            elseif(Ip>=204166240 & Ip<204510176)
            [IpStart,CountryName,RegionName,CityName,Lat,Lon] = textread('IpData/↩
                ip_group_city_1_01_09.csv','%u%s%s%s%f%f','delimiter',';');

            elseif(Ip>=204510176 & Ip<204798040)
            [IpStart,CountryName,RegionName,CityName,Lat,Lon] = textread('IpData/↩
                ip_group_city_1_01_10.csv','%u%s%s%s%f%f','delimiter',';');

            elseif(Ip>=204798040 & Ip<205848088)
            [IpStart,CountryName,RegionName,CityName,Lat,Lon] = textread('IpData/↩
                ip_group_city_1_01_11.csv','%u%s%s%s%f%f','delimiter',';');

            end
```

```
        [...]

        elseif(Ip>=3686793216)

            if(Ip>=3686793216 & Ip<3698238448)
            [IpStart,CountryName,RegionName,CityName,Lat,Lon] = textread('IpData/↩
                ip_group_city_9_10_01.csv','%u%s%s%s%f%f','delimiter',';');

            elseif(Ip>=3698238448 & Ip<3705740288)
            [IpStart,CountryName,RegionName,CityName,Lat,Lon] = textread('IpData/↩
                ip_group_city_9_10_02.csv','%u%s%s%s%f%f','delimiter',';');

            elseif(Ip>=3705740288 & Ip<3707111472)
            [IpStart,CountryName,RegionName,CityName,Lat,Lon] = textread('IpData/↩
                ip_group_city_9_10_03.csv','%u%s%s%s%f%f','delimiter',';');

            elseif(Ip>=3707111472 & Ip<3708151552)
            [IpStart,CountryName,RegionName,CityName,Lat,Lon] = textread('IpData/↩
                ip_group_city_9_10_04.csv','%u%s%s%s%f%f','delimiter',';');

            elseif(Ip>=3708151552 & Ip<3716575096)
            [IpStart,CountryName,RegionName,CityName,Lat,Lon] = textread('IpData/↩
                ip_group_city_9_10_05.csv','%u%s%s%s%f%f','delimiter',';');

            elseif(Ip>=3716575096 & Ip<3722482480)
            [IpStart,CountryName,RegionName,CityName,Lat,Lon] = textread('IpData/↩
                ip_group_city_9_10_06.csv','%u%s%s%s%f%f','delimiter',';');

            elseif(Ip>=3722482480 & Ip<3728918872)
            [IpStart,CountryName,RegionName,CityName,Lat,Lon] = textread('IpData/↩
                ip_group_city_9_10_07.csv','%u%s%s%s%f%f','delimiter',';');

            elseif(Ip>=3728918872 & Ip<3729980304)
            [IpStart,CountryName,RegionName,CityName,Lat,Lon] = textread('IpData/↩
                ip_group_city_9_10_08.csv','%u%s%s%s%f%f','delimiter',';');

            elseif(Ip>=3729980304 & Ip<3733259660)
            [IpStart,CountryName,RegionName,CityName,Lat,Lon] = textread('IpData/↩
                ip_group_city_9_10_09.csv','%u%s%s%s%f%f','delimiter',';');

            elseif(Ip>=3733259660 & Ip<3735579140)
            [IpStart,CountryName,RegionName,CityName,Lat,Lon] = textread('IpData/↩
                ip_group_city_9_10_10.csv','%u%s%s%s%f%f','delimiter',';');

            elseif(Ip>=3735579140 & Ip<=4278190080)
            [IpStart,CountryName,RegionName,CityName,Lat,Lon] = textread('IpData/↩
                ip_group_city_9_10_11.csv','%u%s%s%s%f%f','delimiter',';');

            end

    end

end

         City = '?';
        Region = '?';
        Country = '?';
        Latitude ='?';
        Longitude = '?';

        I = find(Ip<=IpStart,1);

        if (I>2)
        City = CityName(I-1);
        Region = RegionName(I-1);
        Country = CountryName(I-1);
        Latitude = Lat(I-1);
        Longitude = Lon(I-1);
        end
```

B.1.3. IdTraffic

```
%IDTRAFFIC - separa un bloque con direcciones IPs en paises
%
%function IDTraffic(Target1,Target3)
%
%INPUT-
%Target1 = Archivo de texto con las direcciones IPs
%Target3 = ARchivo de texto que contiene el tr\'afico recibido por pa\'is
%
%OUTPUT-
%"Pais.txt" = Archivo de texto con las IPs de cada pais.

function IDTraffic(Target1,Target3)

[Countries] = textread('Country.csv','%','delimiter',';','headerlines',1);

Traffic = zeros(1,size(Countries,1));

[DstIPadress,Octets] = textread(Target1,'%%','delimiter',' ','headerlines',0);

format long
sum(str2num(char(Octets)))
format short

j=0;
Total = 0;

   Dest1 ='Netflow\Paises\Aland Islands_35.txt';
   Dest2 ='Netflow\Paises\Albania_35.txt';
   Dest3 ='Netflow\Paises\Algeria_35.txt';
   Dest4 ='Netflow\Paises\Andorra_35.txt';
   Dest5 ='Netflow\Paises\Angola_35.txt';
   Dest6 ='Netflow\Paises\Anonymous Proxy_35.txt';
   Dest7 ='Netflow\Paises\Antarctica_35.txt';
   Dest8 ='Netflow\Paises\Antigua and Barbuda_35.txt';
   Dest9 ='Netflow\Paises\Argentina_35.txt';
   Dest10 ='Netflow\Paises\Aruba_35.txt';
   Dest11 ='Netflow\Paises\Asia/Pacific Region_35.txt';
   Dest12 ='Netflow\Paises\Australia_35.txt';
   Dest13 ='Netflow\Paises\Austria_35.txt';
   Dest14 = 'Netflow\Paises\Azerbaijan_35.txt';
   Dest15 ='Netflow\Paises\Bahamas_35.txt';
   Dest16 = 'Netflow\Paises\Bahrain_35.txt';
   Dest17 ='Netflow\Paises\Bangladesh_35.txt';
   Dest18 ='Netflow\Paises\Barbados_35.txt';
   Dest19 ='Netflow\Paises\Belarus_35.txt';
   Dest20 ='Netflow\Paises\Belgium_35.txt';
   Dest21 ='Netflow\Paises\Bermuda_35.txt';
   Dest22 ='Netflow\Paises\Bolivia_35.txt';
   Dest23 ='Netflow\Paises\Bosnia and Herzegovina_35.txt';
   Dest24 ='Netflow\Paises\Botswana_35.txt';
   Dest25 ='Netflow\Paises\Brazil_35.txt';
   Dest26 ='Netflow\Paises\Bulgaria_35.txt';
   Dest27 = 'Netflow\Paises\Armenia_35.txt';
   Dest28 = 'Netflow\Paises\United Kingdom_35.txt';
   Dest29 = 'Netflow\Paises\Yemen_35.txt';
   Dest30 ='Netflow\Paises\Cambodia_35.txt';
   Dest31 ='Netflow\Paises\Cameroon_35.txt';
   Dest32 ='Netflow\Paises\Canada_35.txt';
   Dest33 ='Netflow\Paises\Chile_35.txt';
   Dest34 ='Netflow\Paises\China_35.txt';
   Dest35 ='Netflow\Paises\Colombia_35.txt';
   Dest36 ='Netflow\Paises\Costa Rica_35.txt';
   Dest37 ='Netflow\Paises\Cote dIvoire_35.txt';
   Dest38 ='Netflow\Paises\Croatia_35.txt';
   Dest39 ='Netflow\Paises\Cuba_35.txt';
   Dest40 ='Netflow\Paises\Cyprus_35.txt';
   Dest41 ='Netflow\Paises\Czech Republic_35.txt';
   Dest42 ='Netflow\Paises\Denmark_35.txt';
   Dest43 ='Netflow\Paises\Djibouti_35.txt';
   Dest44 ='Netflow\Paises\Dominica_35.txt';
   Dest45 ='Netflow\Paises\Dominican Republic_35.txt';
   Dest46 ='Netflow\Paises\Ecuador_35.txt';
   Dest47 ='Netflow\Paises\Egypt_35.txt';
```

```
Dest48 ='Netflow\Paises\El Salvador_35.txt';
Dest49 ='Netflow\Paises\Equatorial Guinea_35.txt';
Dest50 ='Netflow\Paises\Estonia_35.txt';
Dest51 ='Netflow\Paises\Ethiopia_35.txt';
Dest52 ='Netflow\Paises\Europe_35.txt';
Dest53 ='Netflow\Paises\Finland_35.txt';
Dest54 ='Netflow\Paises\France_35.txt';
Dest55 ='Netflow\Paises\French Polynesia_35.txt';
Dest56 ='Netflow\Paises\Georgia_35.txt';
Dest57 ='Netflow\Paises\Germany_35.txt';
Dest58 ='Netflow\Paises\Ghana_35.txt';
Dest59 ='Netflow\Paises\Gibraltar_35.txt';
Dest60 ='Netflow\Paises\Greece_35.txt';
Dest61 ='Netflow\Paises\Guadeloupe_35.txt';
Dest62 ='Netflow\Paises\Guatemala_35.txt';
Dest63 ='Netflow\Paises\Haiti_35.txt';
Dest64 ='Netflow\Paises\Honduras_35.txt';
Dest65 ='Netflow\Paises\Hong Kong_35.txt';
Dest66 ='Netflow\Paises\Hungary_35.txt';
Dest67 ='Netflow\Paises\Iceland_35.txt';
Dest68 ='Netflow\Paises\India_35.txt';
Dest69 ='Netflow\Paises\Indonesia_35.txt';
Dest70 ='Netflow\Paises\Iran  Islamic Republic of_35.txt';
Dest71 ='Netflow\Paises\Iraq_35.txt';
Dest72 ='Netflow\Paises\Ireland_35.txt';
Dest73 ='Netflow\Paises\Israel_35.txt';
Dest74 ='Netflow\Paises\Italy_35.txt';
Dest75 ='Netflow\Paises\Jamaica_35.txt';
Dest76 ='Netflow\Paises\Japan_35.txt';
Dest77 ='Netflow\Paises\Jordan_35.txt';
Dest78 ='Netflow\Paises\Kazakhstan_35.txt';
Dest79 ='Netflow\Paises\Kenya_35.txt';
Dest80 ='Netflow\Paises\Korea  Republic of_35.txt';
Dest81 ='Netflow\Paises\Kuwait_35.txt';
Dest82 ='Netflow\Paises\Latvia_35.txt';
Dest83 ='Netflow\Paises\Lebanon_35.txt';
Dest84 ='Netflow\Paises\Libyan Arab Jamahiriya_35.txt';
Dest85 ='Netflow\Paises\Liechtenstein_35.txt';
Dest86 ='Netflow\Paises\Lithuania_35.txt';
Dest87 ='Netflow\Paises\Luxembourg_35.txt';
Dest88 ='Netflow\Paises\Macao_35.txt';
Dest89 ='Netflow\Paises\Macedonia_35.txt';
Dest90 ='Netflow\Paises\Malaysia_35.txt';
Dest91 ='Netflow\Paises\Maldives_35.txt';
Dest92 ='Netflow\Paises\Malta_35.txt';
Dest93 ='Netflow\Paises\Martinique_35.txt';
Dest94 ='Netflow\Paises\Mauritius_35.txt';
Dest95 ='Netflow\Paises\Mexico_35.txt';
Dest96 ='Netflow\Paises\Moldova  Republic of_35.txt';
Dest97 ='Netflow\Paises\Mongolia_35.txt';
Dest98 ='Netflow\Paises\Montenegro_35.txt';
Dest99 ='Netflow\Paises\Morocco_35.txt';
Dest100 ='Netflow\Paises\Mozambique_35.txt';
Dest101 ='Netflow\Paises\Namibia_35.txt';
Dest102 ='Netflow\Paises\Netherlands_35.txt';
Dest103 ='Netflow\Paises\Netherlands Antilles_35.txt';
Dest104 ='Netflow\Paises\New Caledonia_35.txt';
Dest105 ='Netflow\Paises\New Zealand_35.txt';
Dest106 ='Netflow\Paises\Nicaragua_35.txt';
Dest107 ='Netflow\Paises\Nigeria_35.txt';
Dest108 ='Netflow\Paises\Norway_35.txt';
Dest109 ='Netflow\Paises\Oman_35.txt';
Dest110 ='Netflow\Paises\Pakistan_35.txt';
Dest111 ='Netflow\Paises\Palestinian Territory_35.txt';
Dest112 ='Netflow\Paises\Panama_35.txt';
Dest113 ='Netflow\Paises\Paraguay_35.txt';
Dest114 ='Netflow\Paises\Peru_35.txt';
Dest115 ='Netflow\Paises\Philippines_35.txt';
Dest116 ='Netflow\Paises\Poland_35.txt';
Dest117 ='Netflow\Paises\Portugal_35.txt';
Dest118 ='Netflow\Paises\Qatar_35.txt';
Dest119 ='Netflow\Paises\Reunion_35.txt';
Dest120 ='Netflow\Paises\Romania_35.txt';
Dest121 ='Netflow\Paises\Russian Federation_35.txt';
Dest122 ='Netflow\Paises\Saint Lucia_35.txt';
Dest123 ='Netflow\Paises\San Marino_35.txt';
Dest124 ='Netflow\Paises\Satellite Provider_35.txt';
```

```
    Dest125 ='Netflow\Paises\Saudi Arabia_35.txt';
    Dest126 ='Netflow\Paises\Senegal_35.txt';
    Dest127 ='Netflow\Paises\Serbia_35.txt';
    Dest128 ='Netflow\Paises\Seychelles_35.txt';
    Dest129 ='Netflow\Paises\Singapore_35.txt';
    Dest130 ='Netflow\Paises\Slovakia_35.txt';
    Dest131 ='Netflow\Paises\Slovenia_35.txt';
    Dest132 ='Netflow\Paises\South Africa_35.txt';
    Dest133 ='Netflow\Paises\Sri Lanka_35.txt';
    Dest134 ='Netflow\Paises\Sudan_35.txt';
    Dest135 ='Netflow\Paises\Sweden_35.txt';
    Dest136 ='Netflow\Paises\Switzerland_35.txt';
    Dest137 ='Netflow\Paises\Syrian Arab Republic_35.txt';
    Dest138 ='Netflow\Paises\Taiwan_35.txt';
    Dest139 ='Netflow\Paises\Tanzania  United Republic of_35.txt';
    Dest140 ='Netflow\Paises\Thailand_35.txt';
    Dest141 ='Netflow\Paises\Trinidad and Tobago_35.txt';
    Dest142 ='Netflow\Paises\Tunisia_35.txt';
    Dest143 ='Netflow\Paises\Turkey_35.txt';
    Dest144 ='Netflow\Paises\Uganda_35.txt';
    Dest145 ='Netflow\Paises\Ukraine_35.txt';
    Dest146 ='Netflow\Paises\United Arab Emirates_35.txt';
    Dest147 ='Netflow\Paises\Uruguay_35.txt';
    Dest148 ='Netflow\Paises\Uzbekistan_35.txt';
    Dest149 ='Netflow\Paises\Venezuela_35.txt';
    Dest150 ='Netflow\Paises\Vietnam_35.txt';

dc1 = fopen(Dest1,'w+');
dc2 = fopen(Dest2,'w+');
dc3 = fopen(Dest3,'w+');
dc4 = fopen(Dest4,'w+');
dc5 = fopen(Dest5,'w+');
dc6 = fopen(Dest6,'w+');
dc7 = fopen(Dest7,'w+');
dc8 = fopen(Dest8,'w+');
dc9 = fopen(Dest9,'w+');
dc10 = fopen(Dest10,'w+');
dc11 = fopen(Dest11,'w+');
dc12 = fopen(Dest12,'w+');
dc13 = fopen(Dest13,'w+');
dc14 = fopen(Dest14,'w+');
dc15 = fopen(Dest15,'w+');
dc16 = fopen(Dest16,'w+');
dc17 = fopen(Dest17,'w+');
dc18 = fopen(Dest18,'w+');
dc19 = fopen(Dest19,'w+');
dc20 = fopen(Dest20,'w+');
dc21 = fopen(Dest21,'w+');
dc22 = fopen(Dest22,'w+');
dc23 = fopen(Dest23,'w+');
dc24 = fopen(Dest24,'w+');
dc25 = fopen(Dest25,'w+');
dc26 = fopen(Dest26,'w+');
dc27 = fopen(Dest27,'w+');
dc28 = fopen(Dest28,'w+');
dc29 = fopen(Dest29,'w+');
dc30 = fopen(Dest30,'w+');
dc31 = fopen(Dest31,'w+');
dc32 = fopen(Dest32,'w+');
dc33 = fopen(Dest33,'w+');
dc34 = fopen(Dest34,'w+');
dc35 = fopen(Dest35,'w+');
dc36 = fopen(Dest36,'w+');
dc37 = fopen(Dest37,'w+');
dc38 = fopen(Dest38,'w+');
dc39 = fopen(Dest39,'w+');
dc40 = fopen(Dest40,'w+');
dc41 = fopen(Dest41,'w+');
dc42 = fopen(Dest42,'w+');
dc43 = fopen(Dest43,'w+');
dc44 = fopen(Dest44,'w+');
dc45 = fopen(Dest45,'w+');
dc46 = fopen(Dest46,'w+');
dc47 = fopen(Dest47,'w+');
dc48 = fopen(Dest48,'w+');
dc49 = fopen(Dest49,'w+');
dc50 = fopen(Dest50,'w+');
```

```
dc51 = fopen(Dest51,'w+');
dc52 = fopen(Dest52,'w+');
dc53 = fopen(Dest53,'w+');
dc54 = fopen(Dest54,'w+');
dc55 = fopen(Dest55,'w+');
dc56 = fopen(Dest56,'w+');
dc57 = fopen(Dest57,'w+');
dc58 = fopen(Dest58,'w+');
dc59 = fopen(Dest59,'w+');
dc60 = fopen(Dest60,'w+');
dc61 = fopen(Dest61,'w+');
dc62 = fopen(Dest62,'w+');
dc63 = fopen(Dest63,'w+');
dc64 = fopen(Dest64,'w+');
dc65 = fopen(Dest65,'w+');
dc66 = fopen(Dest66,'w+');
dc67 = fopen(Dest67,'w+');
dc68 = fopen(Dest68,'w+');
dc69 = fopen(Dest69,'w+');
dc70 = fopen(Dest70,'w+');
dc71 = fopen(Dest71,'w+');
dc72 = fopen(Dest72,'w+');
dc73 = fopen(Dest73,'w+');
dc74 = fopen(Dest74,'w+');
dc75 = fopen(Dest75,'w+');
dc76 = fopen(Dest76,'w+');
dc77 = fopen(Dest77,'w+');
dc78 = fopen(Dest78,'w+');
dc79 = fopen(Dest79,'w+');
dc80 = fopen(Dest80,'w+');
dc81 = fopen(Dest81,'w+');
dc82 = fopen(Dest82,'w+');
dc83 = fopen(Dest83,'w+');
dc84 = fopen(Dest84,'w+');
dc85 = fopen(Dest85,'w+');
dc86 = fopen(Dest86,'w+');
dc87 = fopen(Dest87,'w+');
dc88 = fopen(Dest88,'w+');
dc89 = fopen(Dest89,'w+');
dc90 = fopen(Dest90,'w+');
dc91 = fopen(Dest91,'w+');
dc92 = fopen(Dest92,'w+');
dc93 = fopen(Dest93,'w+');
dc94 = fopen(Dest94,'w+');
dc95 = fopen(Dest95,'w+');
dc96 = fopen(Dest96,'w+');
dc97 = fopen(Dest97,'w+');
dc98 = fopen(Dest98,'w+');
dc99 = fopen(Dest99,'w+');
dc100 = fopen(Dest100,'w+');
dc101 = fopen(Dest101,'w+');
dc102 = fopen(Dest102,'w+');
dc103 = fopen(Dest103,'w+');
dc104 = fopen(Dest104,'w+');
dc105 = fopen(Dest105,'w+');
dc106 = fopen(Dest106,'w+');
dc107 = fopen(Dest107,'w+');
dc108 = fopen(Dest108,'w+');
dc109 = fopen(Dest109,'w+');
dc110 = fopen(Dest110,'w+');
dc111 = fopen(Dest111,'w+');
dc112 = fopen(Dest112,'w+');
dc113 = fopen(Dest113,'w+');
dc114 = fopen(Dest114,'w+');
dc115 = fopen(Dest115,'w+');
dc116 = fopen(Dest116,'w+');
dc117 = fopen(Dest117,'w+');
dc118 = fopen(Dest118,'w+');
dc119 = fopen(Dest119,'w+');
dc120 = fopen(Dest120,'w+');
dc121 = fopen(Dest121,'w+');
dc122 = fopen(Dest122,'w+');
dc123 = fopen(Dest123,'w+');
dc124 = fopen(Dest124,'w+');
dc125 = fopen(Dest125,'w+');
dc126 = fopen(Dest126,'w+');
dc127 = fopen(Dest127,'w+');
```

```
dc128 = fopen(Dest128,'w+');
dc129 = fopen(Dest129,'w+');
dc130 = fopen(Dest130,'w+');
dc131 = fopen(Dest131,'w+');
dc132 = fopen(Dest132,'w+');
dc133 = fopen(Dest133,'w+');
dc134 = fopen(Dest134,'w+');
dc135 = fopen(Dest135,'w+');
dc136 = fopen(Dest136,'w+');
dc137 = fopen(Dest137,'w+');
dc138 = fopen(Dest138,'w+');
dc139 = fopen(Dest139,'w+');
dc140 = fopen(Dest140,'w+');
dc141 = fopen(Dest141,'w+');
dc142 = fopen(Dest142,'w+');
dc143 = fopen(Dest143,'w+');
dc144 = fopen(Dest144,'w+');
dc145 = fopen(Dest145,'w+');
dc146 = fopen(Dest146,'w+');
dc147 = fopen(Dest147,'w+');
dc148 = fopen(Dest148,'w+');
dc149 =fopen(Dest149,'w+');
dc150 =fopen(Dest150,'w+');

size(DstIPadress,1)
for i=1:size(DstIPadress,1)
    Ip = convertip(char(DstIPadress(i,:)));

    [City,Region,Country,Latitude,Longitude] = Ip2City(Ip);

if (strcmp(Country,'Spain') == 0 | strcmp(Country,'Europe') == 0)

            I = find(strcmp(Country,Countries));
                    Traffic(I) = Traffic(I) + str2num(char((Octets(i))));
                    Total = Total + str2num(char((Octets(i))));

                    if  strcmp(Country,'Aland Islands') == 1
                    fprintf(dc1,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets(↩
                        i)))));

                    elseif strcmp(Country,'Albania') == 1
                    fprintf(dc2,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets(↩
                        i)))));

                    elseif strcmp(Country,'Algeria') == 1
                    fprintf(dc3,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets(↩
                        i)))));

                    elseif strcmp(Country,'Andorra') == 1
                    fprintf(dc4,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets(↩
                        i)))));

                    elseif strcmp(Country,'Angola') == 1
                    fprintf(dc5,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets(↩
                        i)))));

                    elseif strcmp(Country,'Anonymous Proxy') == 1
                    fprintf(dc6,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets(↩
                        i)))));

                    elseif strcmp(Country,'Antarctica') == 1
                    fprintf(dc7,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets(↩
                        i)))));

                    elseif strcmp(Country,'Antigua and Barbuda') == 1
                    fprintf(dc8,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets(↩
                        i)))));

                    elseif strcmp(Country,'Argentina') == 1
                    fprintf(dc9,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets(↩
                        i)))));

                    elseif strcmp(Country, 'Aruba') == 1
                    fprintf(dc10,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                        (i)))));

                    elseif strcmp(Country, 'Australia') == 1
```

```
                fprintf(dc12,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country, 'Austria') == 1
                fprintf(dc13,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Azerbaijan') == 1
                fprintf(dc14,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Bahamas') == 1
                fprintf(dc15,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Bahrain') == 1
                fprintf(dc16,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Bangladesh') == 1
                fprintf(dc17,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Barbados') == 1
                fprintf(dc18,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Belarus') == 1
                fprintf(dc19,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Belgium') == 1
                fprintf(dc20,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Bermuda') == 1
                fprintf(dc21,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Bolivia') == 1
                fprintf(dc22,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Bosnia and Herzegovina') == 1
                fprintf(dc23,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Botswana') == 1
                fprintf(dc24,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Brazil') == 1
                fprintf(dc25,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Bulgaria') == 1
                fprintf(dc26,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Armenia') == 1
                fprintf(dc27,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'United Kingdom') == 1
                fprintf(dc28,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Yemen') == 1
                fprintf(dc29,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Cambodia') == 1
                fprintf(dc30,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Cameroon') == 1
```

```
                fprintf(dc31,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Canada') == 1
                fprintf(dc32,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Chile') == 1
                fprintf(dc33,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'China') == 1
                fprintf(dc34,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Colombia') == 1
                fprintf(dc35,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Costa Rica') == 1
                fprintf(dc36,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Cote dIvoire') == 1
                fprintf(dc37,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Croatia') == 1
                fprintf(dc38,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Cuba') == 1
                fprintf(dc39,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Cyprus') == 1
                fprintf(dc40,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Czech Republic') == 1
                fprintf(dc41,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Denmark') == 1
                fprintf(dc42,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Djibouti') == 1
                fprintf(dc43,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Dominica') == 1
                fprintf(dc44,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Dominican Republic') == 1
                fprintf(dc45,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Ecuador') == 1
                fprintf(dc46,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Egypt') == 1
                fprintf(dc47,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'El Salvador') == 1
                fprintf(dc48,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Equatorial Guinea') == 1
                fprintf(dc49,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Estonia') == 1
```

```
                fprintf(dc50,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Ethiopia') == 1
                fprintf(dc51,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Europe') == 1
                fprintf(dc52,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Finland') == 1
                fprintf(dc53,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'France') == 1
                fprintf(dc54,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'French Polynesia') == 1
                fprintf(dc55,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Georgia') == 1
                fprintf(dc56,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Germany') == 1
                fprintf(dc57,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Ghana') == 1
                fprintf(dc58,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Gibraltar') == 1
                fprintf(dc59,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Greece') == 1
                fprintf(dc60,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif  strcmp(Country,'Guadeloupe') == 1
                fprintf(dc61,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Guatemala') == 1
                fprintf(dc62,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Haiti') == 1
                fprintf(dc63,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Honduras') == 1
                fprintf(dc64,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Hong Kong') == 1
                fprintf(dc65,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Hungary') == 1
                fprintf(dc66,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Iceland') == 1
                fprintf(dc67,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'India') == 1
                fprintf(dc68,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Indonesia') == 1
```

```
                fprintf(dc69,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country, 'Iran  Islamic Republic of') == 1
                fprintf(dc70,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif  strcmp(Country,'Iraq') == 1
                fprintf(dc71,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Ireland') == 1
                fprintf(dc72,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Israel') == 1
                fprintf(dc73,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Italy') == 1
                fprintf(dc74,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Jamaica') == 1
                fprintf(dc75,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Japan') == 1
                fprintf(dc76,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Jordan') == 1
                fprintf(dc77,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Kazakhstan') == 1
                fprintf(dc78,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Kenya') == 1
                fprintf(dc79,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country, 'Korea  Republic of') == 1
                fprintf(dc80,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif  strcmp(Country,'Kuwait') == 1
                fprintf(dc81,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Latvia') == 1
                fprintf(dc82,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Lebanon') == 1
                fprintf(dc83,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Libyan Arab Jamahiriya') == 1
                fprintf(dc84,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Liechtenstein') == 1
                fprintf(dc85,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Lithuania') == 1
                fprintf(dc86,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Luxembourg') == 1
                fprintf(dc87,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Macao') == 1
```

```
                fprintf(dc88,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Macedonia') == 1
                fprintf(dc89,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country, 'Malaysia') == 1
                fprintf(dc90,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif  strcmp(Country,'Maldives') == 1
                fprintf(dc91,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Malta') == 1
                fprintf(dc92,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Martinique') == 1
                fprintf(dc93,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Mauritius') == 1
                fprintf(dc94,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Mexico') == 1
                fprintf(dc95,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Moldova  Republic of') == 1
                fprintf(dc96,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Mongolia') == 1
                fprintf(dc97,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Montenegro') == 1
                fprintf(dc98,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country,'Morocco') == 1
                fprintf(dc99,'%s %d\n',char(DstIPadress(i,:)),str2num(char((Octets↩
                    (i)))));

                elseif strcmp(Country, 'Mozambique') == 1
                fprintf(dc100,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif  strcmp(Country,'Namibia') == 1
                fprintf(dc101,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif strcmp(Country,'Netherlands') == 1
                fprintf(dc102,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif strcmp(Country,'Netherlands Antilles') == 1
                fprintf(dc103,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif strcmp(Country,'New Caledonia') == 1
                fprintf(dc104,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif strcmp(Country,'New Zealand') == 1
                fprintf(dc105,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif strcmp(Country,'Nicaragua') == 1
                fprintf(dc106,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif strcmp(Country,'Nigeria') == 1
```

```
                fprintf(dc107,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif strcmp(Country,'Norway') == 1
                fprintf(dc108,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif strcmp(Country,'Oman') == 1
                fprintf(dc109,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif strcmp(Country, 'Pakistan') == 1
                fprintf(dc110,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif  strcmp(Country,'Palestinian Territory') == 1
                fprintf(dc111,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif strcmp(Country,'Panama') == 1
                fprintf(dc112,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif strcmp(Country,'Paraguay') == 1
                fprintf(dc113,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif strcmp(Country,'Peru') == 1
                fprintf(dc114,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif strcmp(Country,'Philippines') == 1
                fprintf(dc115,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif strcmp(Country,'Poland') == 1
                fprintf(dc116,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif strcmp(Country,'Portugal') == 1
                fprintf(dc117,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif strcmp(Country,'Qatar') == 1
                fprintf(dc118,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif strcmp(Country,'Reunion') == 1
                fprintf(dc119,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif strcmp(Country, 'Romania') == 1
                fprintf(dc120,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif  strcmp(Country,'Russian Federation') == 1
                fprintf(dc121,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif strcmp(Country,'Saint Lucia') == 1
                fprintf(dc122,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif strcmp(Country,'San Marino') == 1
                fprintf(dc123,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif strcmp(Country,'Satellite Provider') == 1
                fprintf(dc124,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif strcmp(Country,'Saudi Arabia') == 1
                fprintf(dc125,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif strcmp(Country,'Senegal') == 1
```

```
                fprintf(dc126,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif strcmp(Country,'Serbia') == 1
                fprintf(dc127,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif strcmp(Country,'Seychelles') == 1
                fprintf(dc128,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif strcmp(Country,'Singapore') == 1
                fprintf(dc129,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif strcmp(Country, 'Slovakia') == 1
                fprintf(dc130,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif  strcmp(Country,'Slovenia') == 1
                fprintf(dc131,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif strcmp(Country,'South Africa') == 1
                fprintf(dc132,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif strcmp(Country,'Sri Lanka') == 1
                fprintf(dc133,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif strcmp(Country,'Sudan') == 1
                fprintf(dc134,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif strcmp(Country,'Sweden') == 1
                fprintf(dc135,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif strcmp(Country,'Switzerland') == 1
                fprintf(dc136,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif strcmp(Country,'Syrian Arab Republic') == 1
                fprintf(dc137,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif strcmp(Country,'Taiwan') == 1
                fprintf(dc138,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif strcmp(Country,'Tanzania  United Republic of') == 1
                fprintf(dc139,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif strcmp(Country, 'Thailand') == 1
                fprintf(dc140,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif  strcmp(Country,'Trinidad and Tobago') == 1
                fprintf(dc141,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif strcmp(Country,'Tunisia') == 1
                fprintf(dc142,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif strcmp(Country,'Turkey') == 1
                fprintf(dc143,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif strcmp(Country,'Uganda') == 1
                fprintf(dc144,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                    Octets(i)))));

                elseif strcmp(Country,'Ukraine') == 1
```

```
                    fprintf(dc145,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                        Octets(i)))));

                    elseif strcmp(Country,'United Arab Emirates') == 1
                    fprintf(dc146,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                        Octets(i)))));

                    elseif strcmp(Country,'Uruguay') == 1
                    fprintf(dc147,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                        Octets(i)))));

                    elseif strcmp(Country,'Uzbekistan') == 1
                    fprintf(dc148,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                        Octets(i)))));

                    elseif strcmp(Country,'Venezuela') == 1
                    fprintf(dc149,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                        Octets(i)))));

                    elseif strcmp(Country, 'Vietnam') == 1
                    fprintf(dc150,'%s %d\n',char(DstIPadress(i,:)),str2num(char((↩
                        Octets(i)))));

                    end

        end
end


fid = fopen(Target3,'w+');

for i=1:length(Traffic)

fprintf(fid,'%s;  %d\n',char(Countries(i)),Traffic(i));

end

fclose(fid);
```

B.1.4. Vdist

```
function s = dist2(lat1,lon1,lat2,lon2)
%VDIST - compute distance between points on the WGS-84 ellipsoidal Earth
%        to within a few millimeters of accuracy using Vincenty's algorithm
%
%s = vdist(lat1,lon1,lat2,lon2)
%
%s = distance in meters
%lat1 = GEODETIC latitude of first point (degrees)
%lon1 = longitude of first point (degrees)
%lat2, lon2 = second point (degrees)
%
% Original algorithm source:
% T. Vincenty, "Direct and Inverse Solutions of Geodesics on the Ellipsoid
% with Application of Nested Equations", Survey Review, vol. 23, no. 176,
% April 1975, pp 88-93
%
%Notes: (1) Error correcting code, convergence failure traps, antipodal
%corrections,
%           polar error corrections, WGS84 ellipsoid parameters,
%testing, and comments
%           written by Michael Kleder, 2004.
%       (2) Vincenty describes his original algorithm as precise to within
%           0.01 millimeters, subject to the ellipsoidal model.
%       (3) Essentially antipodal points are treated as exactly antipodal,
%           potentially reducing accuracy by a small amount.
%       (4) Failures for points exactly at the poles are eliminated by
%           moving the points by 0.6 millimeters.
%       (5) Vincenty's azimuth formulas are not implemented in this
%           version, but are available as comments in the code.
%       (6) The Vincenty procedure was transcribed verbatim by Peter
```

```
% Cederholm,
%            August 12, 2003. It was modified and translated to English
% by Michael Kleder.
%            Mr. Cederholm's website is http://www.plan.aau.dk/~pce/
%        (7) Code to test the disagreement between this algorithm and the
%            Mapping Toolbox spherical earth distance function is provided
%            as comments in the code. The maximum differences are:
%            Max absolute difference: 38 kilometers
%            Max fractional difference: 0.56 percent

% Input check:
if abs(lat1)>90 | abs(lat2)>90
    error('Input latitudes must be between -90 and 90 degrees, inclusive.')
end
% Supply WGS84 earth ellipsoid axis lengths in meters:
a = 6378137; % definitionally
b = 6356752.31424518; % computed from WGS84 earth flattening coefficient definition
% convert inputs in degrees to radians:
lat1 = lat1 * 0.0174532925199433;
lon1 = lon1 * 0.0174532925199433;
lat2 = lat2 * 0.0174532925199433;
lon2 = lon2 * 0.0174532925199433;
% correct for errors at exact poles by adjusting 0.6 millimeters:
if abs(pi/2-abs(lat1)) < 1e-10;
    lat1 = sign(lat1)*(pi/2-(1e-10));
end
if abs(pi/2-abs(lat2)) < 1e-10;
    lat2 = sign(lat2)*(pi/2-(1e-10));
end
f = (a-b)/a;
U1 = atan((1-f)*tan(lat1));
U2 = atan((1-f)*tan(lat2));
lon1 = mod(lon1,2*pi);
lon2 = mod(lon2,2*pi);
L = abs(lon2-lon1);
if L > pi
    L = 2*pi - L;
end
lambda = L;
lambdaold = 0;
itercount = 0;
while ~itercount | abs(lambda-lambdaold) > 1e-12  % force at least one execution
    itercount = itercount+1;
    if itercount > 50
        warning('Points are essentially antipodal. Precision may be reduced slightly.'↩
            );
        lambda = pi;
        break
    end
    lambdaold = lambda;
    sinsigma = sqrt((cos(U2)*sin(lambda))^2+(cos(U1)*...
        sin(U2)-sin(U1)*cos(U2)*cos(lambda))^2);
    cossigma = sin(U1)*sin(U2)+cos(U1)*cos(U2)*cos(lambda);
    sigma = atan2(sinsigma,cossigma);
    alpha = asin(cos(U1)*cos(U2)*sin(lambda)/sin(sigma));
    cos2sigmam = cos(sigma)-2*sin(U1)*sin(U2)/cos(alpha)^2;
    C = f/16*cos(alpha)^2*(4+f*(4-3*cos(alpha)^2));
    lambda = L+(1-C)*f*sin(alpha)*(sigma+C*sin(sigma)*...
        (cos2sigmam+C*cos(sigma)*(-1+2*cos2sigmam^2)));
    % correct for convergence failure in the case of essentially antipodal points
    if lambda > pi
        warning('Points are essentially antipodal. Precision may be reduced slightly.'↩
            );
        lambda = pi;
        break
    end
end
u2 = cos(alpha)^2*(a^2-b^2)/b^2;
A = 1+u2/16384*(4096+u2*(-768+u2*(320-175*u2)));
B = u2/1024*(256+u2*(-128+u2*(74-47*u2)));
deltasigma = B*sin(sigma)*(cos2sigmam+B/4*(cos(sigma)*(-1+2*cos2sigmam^2)...
    -B/6*cos2sigmam*(-3+4*sin(sigma)^2)*(-3+4*cos2sigmam^2)));
s = b*A*(sigma-deltasigma);
```

B.1.5. EstadisticasPais

```
% EstadisticasPais - calcula la distancia geografica, la distancia enrutada, el numero de saltos y % el RTT para cada traza de TCPtraceroute de una carpeta de nombre 'Directory'.
% Esta funcion  tambien calcula el porcentaje de destinos que se alcanzan con traceroute y
% TCPtraceroute en un pais
%
% function EstadisticasPais(Resultados,Estadisticas,Country,Directory)
%
% INPUT-
% Resultados = archivo de texto con el PI,PIT del pais
% Estadisticas = archivo de texto con el rendimiento de traceroute y
% TCPtraceroute
% Country = pais a esutidar
% Directory = directorio donde se encuentran las trazas

function EstadisticasPais(Resultados,Estadisticas,Country,Directory)

File = strcat('Netflow/Final/',Country);

[IPx,Traffic] = textread(File,'%s %s','delimiter',' ');

Traffic = str2num(char(Traffic));

fid2 = fopen(Resultados,'w+');
fid3 = fopen(Estadisticas,'w+');

TotalICMP = 0;
TotalTCP = 0;

for k=1:size(IPx,1)
    k
    DistanciaTotal = 0;
    Path = strcat('Tracert/',Directory,'/',num2str(k),'.txt');
    Path2 = strcat('Tracert/',Directory,'2/',num2str(k),'.txt');
    [Step,ip,Time,Unit,x] = textread(Path,'%s %s %d %s %s','delimiter',' ','headerlines',1);
    [Step2,ip2,Time2,Unit2,x2] = textread(Path2,'%s %s %d %s %s','delimiter',' ','headerlines',1);
    if (size(ip2,1) > 0) & (size(ip2,1) < 30) & char(Step2(size(ip2,1),:)) ~= '!X' & char(Step2(size(ip2,1),:)) ~= '!H'

        TotalICMP = TotalICMP +1;

    end

    if (size(ip,1) > 0) & (size(ip,1) < 30) & char(Step(size(ip,1),:)) ~= '!X' & char(Step(size(ip,1),:)) ~= '!H'
        TotalTCP = TotalTCP +1 ;
        ElapsedTime = sum(Time);


        [CityI,RegionI,CountryI,LatitudeI,LongitudeI] = Ip2City(convertip('150.244.56.1'));
        [CityF,RegionF,CountryF,LatitudeF,LongitudeF] = Ip2City(convertip(char(ip(end,:))));
        DistanciaLineaRecta = dist2(LatitudeI,LongitudeI,LatitudeF,LongitudeF);

        for j=2:size(Step,1)

            if (char(ip(j,:)) ~= '*')
            [City,Region,Country,Latitude,Longitude] = Ip2City(convertip(char(ip(j,:))));

            if  LatitudeI ~= Latitude && LongitudeI ~= Longitude
                DistanciaRecorrida = dist2(LatitudeI,LongitudeI,Latitude,Longitude);
                DistanciaTotal = DistanciaTotal + DistanciaRecorrida;
            end

            LatitudeI = Latitude;
            LongitudeI = Longitude;
```

```
                end


            end

            fprintf(fid2,'%s %s %s %s %s %s\n',char(IPx(k,:)),num2str(size(ip,1)),↩
                num2str(Traffic(k)),num2str(DistanciaLineaRecta),num2str(DistanciaTotal)↩
                ,num2str(ElapsedTime));

    end

end

fprintf(fid3,'%d %d %d %d %s',sum(Traffic),k,TotalTCP,TotalICMP);

fclose(fid2);
fclose(fid3);
```

B.1.6. Funciones MAP

Este conjunto de funciones dibujan puntos, líneas y texto en mapas. Han sido útiles para dibujar la traza que siguen las rutas estudiadas.

M_proj

```
function m_proj(proj,varargin)
%M_PROJ  Initializes map projections info, putting the result into a structure
%
%         M_PROJ('set') tells you the current state
%         M_PROJ('get') gives you a list of all possibilities
%         M_PROJ('get','proj name') gives info about a projection in the
%                                   'get' list.
%         M_PROJ('proj name','property',value,...) initializes a projection.
%
%
%         see also M_GRID, M_LL2XY, M_XY2LL.

%Rich Pawlowicz (rich@ocgy.ubc.ca) 2/Apr/1997
%
%This software is provided "as is" without warranty of any kind. But
%it's mine, so you can't sell it.
%
%20/Sep/01 - Added support for other coordinate systems.
%25/Feb/07 - Swapped "get" and "set" at lines 34 and 47
%       to make it consistent with the help
%       (and common Matlab style)
%       - Added lines 62-70 & 74
%       to harden against error when no proj is set
%             (fixes thanks to Lars Barring)

global MAP_PROJECTION MAP_VAR_LIST MAP_COORDS

%Get all the projections
projections=m_getproj;

if nargin==0, proj='usage'; end;

proj=lower(proj);

switch proj,

  case 'set',                %Print out their names
    if nargin==1,
      disp(' ');
      disp('Available projections are:');
      for k=1:length(projections),
        disp(['      ' projections(k).name]);
      end;
    else
```

```
      k=m_match(varargin{1},projections(:).name);
      eval(['X=' projections(k).routine '(''set'',projections(k).name);']);
      disp(X);
    end;

  case 'get',              %Get the values of all set parameters
    if nargin==1,
      if isempty(MAP_PROJECTION),
         disp('No map projection initialized');
         m_proj('usage');
      else
         k=m_match(MAP_PROJECTION.name,projections(:).name);
         eval(['X=' projections(k).routine '(''get'');']);
         disp('Current mapping parameters -');
         disp(X);
      end;
    else
      if isempty(MAP_PROJECTION),
        k=m_match(varargin{1},projections(:).name);
        eval(['X=' projections(k).routine '(''set'',projections(k).name);']);
        X=strvcat(X, ...
                  ' ', ...
                  '**** No projection is currently defined      ****', ...
                  ['**** USE "m_proj(''' varargin{1} ''',<see options above>)" ****'])↩
                      ;
        disp(X);
      else
    k=m_match(varargin{1},projections(:).name);
    eval(['X=' projections(k).routine '(''get'');']);
    disp(X);
      end;
    end;

  case 'usage',
    disp(' ');
    disp('Possible calling options are:');
    disp('   ''usage''                    - this list');
    disp('   ''set''                      - list of projections');
    disp('   ''set'',''projection''         - list of properties for projection');
    disp('   ''get''                      - get current mapping parameters (if defined)↩
        ');
    disp('   ''projection'' <,properties> - initialize projection\n');

 otherwise                %If a valid name, give the usage.
    k=m_match(proj,projections(:).name);
    MAP_PROJECTION=projections(k);

    eval([ projections(k).routine '(''initialize'',projections(k).name,varargin{:});'↩
        ]);

    %With the projection store what coordinate system we are using to define it.
    if isempty(MAP_COORDS),
      m_coord('geographic');
    end;
    MAP_PROJECTION.coordsystem=MAP_COORDS;

end;

%---------------------------------------------------------
function projections=m_getproj;
%M_GETPROJ Gets a list of the different projection routines
%          and returns a structure containing both their
%          names and the formal name of the projection.
%          (used by M_PROJ).

%Rich Pawlowicz (rich@ocgy.ubc.ca) 9/May/1997
%
%9/May/97 - fixed paths for Macs (thanks to Dave Enfield)
%
%7/05/98 - VMS pathnames (thanks to Helmut Eggers)

%Get all the projections

lpath=which('m_proj');
fslashes=findstr(lpath,'/');
bslashes=findstr(lpath,'\');
colons=findstr(lpath,':');
```

```
closparantheses=findstr(lpath,']');
if ~isempty(fslashes),
  lpath=[ lpath(1:max(fslashes)) 'private/'];
elseif ~isempty(bslashes),
  lpath=[ lpath(1:max(bslashes)) 'private\'];
elseif ~isempty(closparantheses),       % for VMS computers only, others don't use ']'↩
    in filenames
  lpath=[ lpath(1:max(closparantheses)-1) '.private]'];
else,
  lpath=[ lpath(1:max(colons)) 'private:'];
end;

w=dir([lpath 'mp_*.m']);

if isempty(w), % Not installed correctly
  disp('*********************************************************');
  disp('* ERROR - Can''t find anything in a /private subdirectory *');
  disp('*         m_map probably unzipped incorrectly - please   *');
  disp('*         unpack again, preserving directory structure   *');
  disp('*                                                        *');
  disp('*          ...Abandoning m_proj now.                     *');
  error('*********************************************************');
end;

l=1;
projections=[];
for k=1:length(w),
 funname=w(k).name(1:(findstr(w(k).name,'.'))-1);
 projections(l).routine=funname;
 eval(['names= ' projections(l).routine '(''name'');']);
 for m=1:length(names);
   projections(l).routine=funname;
   projections(l).name=names{m};
   l=l+1;
 end;
end;


%-----------------------------------------------------------
function match=m_match(arg,varargin);
%M_MATCH Tries to match input string with possible options

% Rich Pawlowicz (rich@ocgy.ubc.ca) 2/Apr/1997

match=strmatch(lower(arg),cellstr(lower(char(varargin))));

if length(match)>1,
  error(['Projection ''' arg ''' not a unique specification']);
elseif isempty(match),
  error(['Projection ''' arg ''' not recognized']);
end;
```

M_coast

```
function h=m_coast(varargin);
%M_COAST Add a coastline to a given map.
%        M_COAST draw a coastline as either filled patches (slow) or
%        lines (fast) on a given projection. It uses a coastline database with
%        a resolution of about 1/4 degree.
%
%        M_COAST( (standard line option,...,...) )  or
%        M_COAST('line', (standard line option,...,...) ) draws the coastline
%        as a simple line.
%        M_COAST('patch' ( ,standard patch options,...,...) ) draws the
%        coastline as a number of patches.
%
%
%        See also M_PROJ, M_GRID

% Rich Pawlowicz (rich@ocgy.ubc.ca) 15/June/98
%
%
```

```
%This software is provided "as is" without warranty of any kind. But
%it's mine, so you can't sell it.


%Set current projection to geographic
Currentmap=m_coord('set');
m_coord('geographic');


h=mu_coast('default',varargin{:},'tag','m_coast');

m_coord(Currentmap.name);
```

M_line

```
function h=m_line(long,lat,varargin);
%M_LINE Create a line on a map
%   M_LINE(LONG,LAT) adds the line in vectors LONG and LAT to the
%   current axes. If LONG and LAT are matrices the same size, one
%   line per column is added.
%
%   LINE returns a column vector of handles to LINE objects,
%   one handle per line. LINEs are children of AXES objects.
%
%   The LONG,LAT pair can be followed by
%   parameter/value pairs to specify additional properties of the lines.
%   These are standard 'line' properties.
%
%   See also LINE, M_LL2XY


%Rich Pawlowicz (rich@ocgy.ubc.ca) 17/Jan/1998
%8/Dec/98 - changed switch test to only 3 letters (thus letting
%           "tag" properties through).
%18/july/00 - Fixed m_line so you could do clipping through it.
%6/Nov/00 - eliminate returned stuff if ';' neglected (thx to D Byrne)

%
%This software is provided "as is" without warranty of any kind. But
%it's mine, so you can't sell it.

clp='on';

k=1;
while k<length(varargin),
  switch lower(varargin{k}(1:3)),
    case 'cli',
      clp=varargin{k+1}
      if isempty(findstr(clp,'on')),
        varargin{k+1}='off';
      else
        varargin{k+1}='on';
%        varargin([k k+1])=[];
      end;
      k=k+2;
    otherwise
      k=k+2;
  end;
end;

[X,Y]=m_ll2xy(long,lat,'clip',clp);

if nargout>0,
  h=line(X,Y,'tag','m_line',varargin{:});
else
  line(X,Y,'tag','m_line',varargin{:});
end;
```

M_text

```
function h=m_text(long,lat,varargin);
%M_TEXT Text Annotation
%   M_TEXT(LONG,LAT,'string') adds the text in the quotes to location
%   (LONG,LAT) on the currently defined map projection. If LONG and LAT
%   are vectors, M_TEXT writes the text at all locations given.
%   If 'string' is an array the same number of rows as the
%   length of LONG and LAT, M_TEXT marks each point with the
%   corresponding row of the 'string' array.
%
%   M_TEXT returns a column vector of handles to TEXT objects, one
%   handle per text object. TEXT objects are children of AXES objects.
%
%   M_TEXT(LONG,LAT,'string',property/value pairs) can be used to
%   change fontsize, weight, color, etc using the standard TEXT
%   properties.
%
%   See also TEXT.

%Rich Pawlowicz (rich@ocgy.ubc.ca) 17/Jan/1998
%
%This software is provided "as is" without warranty of any kind. But
%it's mine, so you can't sell it.


%31/Jul/99 - changed to allow for X/Y vectors.
%6/Nov/00 - eliminate returned stuff if ';' neglected (thx to D Byrne)

global MAP_PROJECTION

%Have to have initialized a map first

if isempty(MAP_PROJECTION),
  disp('No Map Projection initialized - call M_PROJ first!');
  return;
end;

[X,Y]=m_ll2xy(long,lat,'clip','off');
%h=text('position',[X(:) Y(:)],'tag','m_text','string',varargin{:});
%Fix to allow vectors of X/Y to work
h=text(X(:),Y(:),varargin{:});
if length(h)>0 & isempty(get(h(1),'tag')),
 set(h,'tag','m_text');
end;

if nargout==0,
 clear h
end;
```

B.1.7. Metric

```
%METRIC - calcula la media del PI, PIT, RTT, y numero de saltos para un pais.
%
%function Metric(Country)
%
%INPUT-
%Country = pais
%
%OUTPUT-
%Muestra por pantalla las medias de las metricas especificadas.

function Metric(Country)

File = strcat('TRACERT/Resultados/Resultados',Country,'.txt');

[Ip,Saltos,Trafico,DistanciaLineaRecta,DistanciaTotalRutada,time] = textread(File,'%s %s %s %s %s','delimiter',' ');
```

```
TraficoTotal = 984939226552;

PathInflation =(str2num(char(DistanciaTotalRutada))./str2num(char(DistanciaLineaRecta)↩
    ));

PathInflationTrafico = PathInflation.*str2num(char(Trafico))/TraficoTotal;

mean(PathInflation)

%var(PathInflation)

mean(PathInflationTrafico)

%var(PathInflationTrafico)

mean(str2num(char(time)))

DistanciaEnrutadoPonderada = str2num(char(Trafico)).*str2num(char(DistanciaTotalRutada↩
    ))./TraficoTotal;

DistanciaRealPonderada = str2num(char(Trafico)).*str2num(char(DistanciaLineaRecta))./↩
    TraficoTotal;

Ratio = (str2num(char(Trafico)).* str2num(char(DistanciaTotalRutada)))./(TraficoTotal↩
    .* str2num(char(DistanciaLineaRecta)));

mean(Ratio);
```

B.1.8. Tracert

```
%TRACERT - dibuja la ruta de una traza de TCPTraceroute y obtiene valiosos datos
%
%function [DistanciaTotalRutada,DistanciaLineaRecta,x,saltos] =
%tracert(TraceFile)
%
%INPUT-
%TraceFile = carpeta con las trazas
%
%OUTPUT-
%DistanciaTotalRutada = distancia enrutada
%DistanciaLineaRecta = distancai fisica
%x = RTT
%saltos = numero de saltos


function [DistanciaTotalRutada,DistanciaLineaRecta,x,saltos] = tracert(TraceFile)

for num=1:100

    num
    ese = strcat('TRACERT/IMAGES/',TraceFile)
    mkdir(ese)
Path = strcat('TRACERT/Datos',TraceFile,'/',num2str(num),'.txt');
[Step,ip,Time,Unit,x] = textread(Path,'%s %s %d %s %s','delimiter',' ','headerlines',1);

total = 0;
x = 0;
DistanciaLineaRecta=0;
DistanciaTotalRutada = 0;

saltos = 0;

if (size(ip,1) < 30)

ip= char(ip);
total = total + 1;
saltos = size(ip,1);

 i=1;
for(j=1:size(ip,1))
```

```
        if (char(Step(j,:)) ~= '!X' | char(Step(j,:)) ~= '!H' )
       if (ip(j,:) ~= '*')
        string = ip(j,:);
        ipnum(i) = convertip(string);
        [City(i,:),Region(i,:),Country(i,:),Latitude(i),Longitude(i)] = Ip2City(ipnum(↩
            i));
        i = i+1;
       end
        end
end

i=i-1;

DistanciaTotal=0;
for(i=1:size(Latitude,2)-1)
    if Latitude(i) == Latitude(i+1) && Longitude(i) == Longitude(i+1)
        Distancia(i) = 0;
    else
    Distancia(i) = dist2(Latitude(i),Longitude(i),Latitude(i+1),Longitude(i+1));
    end


    DistanciaTotal = DistanciaTotal + Distancia(i);
end

DistanciaTotalRutada = DistanciaTotal;
DistanciaLineaRecta = dist2(Latitude(1),Longitude(1),Latitude(i+1),Longitude(i+1));

scrsz = get(0,'ScreenSize')
figure('Position',[1.05 1.05*scrsz(4) 0.95*scrsz(3) 0.85*scrsz(4)])

m_proj('miller','lat',[-75 75]);
set(gca,'color',[.9 .99 1]);
m_coast('patch',[.7 1 .7],'edgecolor',[1 1 1]);
m_grid('box','fancy','linestyle','none');
ranget = 0;

m_text(Longitude(1),Latitude(1),sprintf('%s',char(City(1,:))),'fontsize',11);

    for i=1:size(City,1)-1
      [range,ln,lt]=m_lldist([Longitude(i) Longitude(i+1)],[Latitude(i) Latitude(i+1)↩
          ],40);
      m_line(ln,lt,'color','r','linewi',1.6);
      if range ~= 0

      end

    ranget = ranget + range;
    end
    ranget
    m_text(Longitude(i+1),Latitude(i+1),sprintf('%s',char(City(i+1,:))),'fontsize',11)↩
        ;

x = sum(Time);


Output = strcat('TRACERT/IMAGES/',TraceFile,'/Image-',num2str(num),'.bmp');
saveas(gcf,Output, 'bmp');

close all

end
end
```

B.1.9. Preping

```
%PREPING - obtiene lista de IPs que han alcanzado su destino con TCPtraceroute
%
%function preping(Resultados,Estadisticas,Country,Directory)
%
%Resultados = archivo de texto con el PI,PIT del pais
```

```
% Estadisticas = archivo de texto con el rendimiento de traceroute y
% TCPtraceroute
% Country = pais a esutidar
% Directory = directorio donde se encuentran las trazas

function preping(Resultados,Estadisticas,Country,Directory)

File = strcat('Netflow/Final/',Country);

[IPx,Traffic] = textread(File,'%s %s','delimiter',' ');

Traffic = str2num(char(Traffic));


filex = strcat('Netflow/Final/',Country,'ping.txt')
fid3 = fopen(filex,'w+');

for k=1:size(IPx,1)
    k
    Path2 = strcat('Tracert/',Directory,'2/',num2str(k),'.txt');

    [Step2,ip2,Time2,Unit2,x2] = textread(Path2,'%s %s %d %s %s','delimiter',' ','↩
        headerlines',1);

    if (size(ip2,1) > 0) & (size(ip2,1) < 30) & char(Step2(size(ip2,1),:)) ~= '!X' & ↩
        char(Step2(size(ip2,1),:)) ~= '!H'

            fprintf(fid3,'%s %s\n',char(IPx(k,:)),num2str(Traffic(k)));

    end


end

fclose(fid3)
```

B.1.10. RTTmean

```
%RTTmean - calcula RTT medio de trazas de ping linux
%
% function [DistanciaTotalRutada,DistanciaLineaRecta,x,saltos] =
% tracert(TraceFile)
%
% DistanciaTotalRutada = distancia enrutada
% DistanciaLineaRecta = distancai fisica
% x = RTT
% saltos = numero de saltos
% TraceFile = carpeta con las trazas

clc
close all;
clear all;

[a,b,c,d,e,f,g,h,i,j,k,l] = textread('ton.txt','%s %s %s %s %s %s %s %s %s %s %s %s','delimiter','↩
    '=','headerlines',0);



[TTL] = strtok(c,' ')

TTLn = str2num(char(TTL))

if TTLn(1) > 128 & TTLn(1) < 256
    TTLn = 256 - TTLn
end

if TTLn(1) > 64 & TTLn(1) < 128
    TTLn = 128 - TTLn
end

if TTLn(1) > 32 & TTLn(1) < 64
```

```
    TTLn = 64 - TTLn
end

if TTLn(1) > 16 & TTLn(1) < 32
    TTLn = 32 - TTLn
end

[time] = strtok(d,' ')
timen = str2num(char(time))

[a,b,c,d,e,f,g,h,i,j,k,l] = textread('ton.txt','%s %s %s %s %s %s %s %s %s %s %s %s','delimiter',↩
    ' ','headerlines',0);
b(1)
```

B.1.11. Kmeans

```
function [idx, C, sumD, D] = kmeans(X, k, varargin)
%KMEANS K-means clustering.
%   IDX = KMEANS(X, K) partitions the points in the N-by-P data matrix
%   X into K clusters.  This partition minimizes the sum, over all
%   clusters, of the within-cluster sums of point-to-cluster-centroid
%   distances.  Rows of X correspond to points, columns correspond to
%   variables.  KMEANS returns an N-by-1 vector IDX containing the
%   cluster indices of each point.  By default, KMEANS uses squared
%   Euclidean distances.
%
%   KMEANS treats NaNs as missing data, and removes any rows of X that
%   contain NaNs.
%
%   [IDX, C] = KMEANS(X, K) returns the K cluster centroid locations in
%   the K-by-P matrix C.
%
%   [IDX, C, SUMD] = KMEANS(X, K) returns the within-cluster sums of
%   point-to-centroid distances in the 1-by-K vector sumD.
%
%   [IDX, C, SUMD, D] = KMEANS(X, K) returns distances from each point
%   to every centroid in the N-by-K matrix D.
%
%   [ ... ] = KMEANS(..., 'PARAM1',val1, 'PARAM2',val2, ...) allows you to
%   specify optional parameter name/value pairs to control the iterative
%   algorithm used by KMEANS.  Parameters are:
%
%   'Distance' - Distance measure, in P-dimensional space, that KMEANS
%      should minimize with respect to.  Choices are:
%            {'sqEuclidean'} - Squared Euclidean distance
%             'cityblock'    - Sum of absolute differences, a.k.a. L1
%             'cosine'       - One minus the cosine of the included angle
%                              between points (treated as vectors)
%             'correlation'  - One minus the sample correlation between
%                              points (treated as sequences of values)
%             'Hamming'      - Percentage of bits that differ (only
%                              suitable for binary data)
%
%   'Start' - Method used to choose initial cluster centroid positions,
%      sometimes known as "seeds".  Choices are:
%                {'sample'} - Select K observations from X at random
%                 'uniform' - Select K points uniformly at random from
%                             the range of X.  Not valid for Hamming distance.
%                 'cluster' - Perform preliminary clustering phase on
%                             random 10% subsample of X.  This preliminary
%                             phase is itself initialized using 'sample'.
%                 matrix    - A K-by-P matrix of starting locations.  In
%                             this case, you can pass in [] for K, and
%                            KMEANS infers K from the first dimension of
%                             the matrix.  You can also supply a 3D array,
%                             implying a value for 'Replicates'
%                             from the array's third dimension.
%
%   'Replicates' - Number of times to repeat the clustering, each with a
%      new set of initial centroids [ positive integer | {1}]
%
```

```
%   'Maxiter' - The maximum number of iterations [ positive integer | {100}]
%
%   'EmptyAction' - Action to take if a cluster loses all of its member
%       observations.  Choices are:
%                {'error'}    - Treat an empty cluster as an error
%                 'drop'      - Remove any clusters that become empty, and
%                               set corresponding values in C and D to NaN.
%                 'singleton' - Create a new cluster consisting of the one
%                               observation furthest from its centroid.
%
%   'Display' - Display level [ 'off' | {'notify'} | 'final' | 'iter' ]
%
%   Example:
%
%       X = [randn(20,2)+ones(20,2); randn(20,2)-ones(20,2)];
%       [cidx, ctrs] = kmeans(X, 2, 'dist','city', 'rep',5, 'disp','final');
%       plot(X(cidx==1,1),X(cidx==1,2),'r.', ...
%            X(cidx==2,1),X(cidx==2,2),'b.', ctrs(:,1),ctrs(:,2),'kx');
%
%   See also LINKAGE, CLUSTERDATA, SILHOUETTE.

%   KMEANS uses a two-phase iterative algorithm to minimize the sum of
%   point-to-centroid distances, summed over all K clusters.  The first
%   phase uses what the literature often describes as "batch" updates,
%   where each iteration consists of reassigning points to their nearest
%   cluster centroid, all at once, followed by recalculation of cluster
%   centroids. This phase may be thought of as providing a fast but
%   potentially only approximate solution as a starting point for the
%   second phase.  The second phase uses what the literature often
%   describes as "on-line" updates, where points are individually
%   reassigned if doing so will reduce the sum of distances, and cluster
%   centroids are recomputed after each reassignment.  Each iteration
%   during this second phase consists of one pass though all the points.
%   KMEANS can converge to a local optimum, which in this case is a
%   partition of points in which moving any single point to a different
%   cluster increases the total sum of distances.  This problem can only be
%   solved by a clever (or lucky, or exhaustive) choice of starting points.
%
% References:
%
%   [1] Seber, G.A.F., Multivariate Observations, Wiley, New York, 1984.
%   [2] Spath, H. (1985) Cluster Dissection and Analysis: Theory, FORTRAN
%       Programs, Examples, translated by J. Goldschmidt, Halsted Press,
%       New York, 226 pp.

%   Copyright 1993-2004 The MathWorks, Inc.
%   $Revision: 1.4.4.5 $  $Date: 2004/03/02 21:49:12 $

if nargin < 2
    error('stats:kmeans:TooFewInputs','At least two input arguments required.');
end

if any(isnan(X(:)))
    warning('stats:kmeans:MissingDataRemoved','Removing rows of X with missing data.')↩
        ;
    X = X(~any(isnan(X),2),:);
end

%n points in p dimensional space
[n, p] = size(X);
Xsort = []; Xord = [];

pnames = {   'distance'  'start' 'replicates' 'maxiter' 'emptyaction' 'display'};
dflts =  {'sqeuclidean' 'sample'          []       100        'error'  'notify'};
[eid,errmsg,distance,start,reps,maxit,emptyact,display] ...
                      = statgetargs(pnames, dflts, varargin{:});
if ~isempty(eid)
    error(sprintf('stats:kmeans:%s',eid),errmsg);
end

if ischar(distance)
    distNames = {'sqeuclidean','cityblock','cosine','correlation','hamming'};
    i = strmatch(lower(distance), distNames);
    if length(i) > 1
        error('stats:kmeans:AmbiguousDistance', ...
              'Ambiguous ''distance'' parameter value:  %s.', distance);
    elseif isempty(i)
```

```
        error('stats:kmeans:UnknownDistance', ...
              'Unknown ''distance'' parameter value:  %s.', distance);
    end
    distance = distNames{i};
    switch distance
    case 'cityblock'
        [Xsort,Xord] = sort(X,1);
    case 'cosine'
        Xnorm = sqrt(sum(X.^2, 2));
        if any(min(Xnorm) <= eps(max(Xnorm)))
            error('stats:kmeans:ZeroDataForCos', ...
                  ['Some points have small relative magnitudes, making them ', ...
                   'effectively zero.\nEither remove those points, or choose a ', ...
                   'distance other than ''cosine''.']);
        end
        X = X ./ Xnorm(:,ones(1,p));
    case 'correlation'
        X = X - repmat(mean(X,2),1,p);
        Xnorm = sqrt(sum(X.^2, 2));
        if any(min(Xnorm) <= eps(max(Xnorm)))
            error('stats:kmeans:ConstantDataForCorr', ...
                  ['Some points have small relative standard deviations, making them '↩
                      , ...
                   'effectively constant.\nEither remove those points, or choose a ', ↩
                      ...
                   'distance other than ''correlation''.']);
        end
        X = X ./ Xnorm(:,ones(1,p));
    case 'hamming'
        if ~all(ismember(X(:),[0 1]))
            error('stats:kmeans:NonbinaryDataForHamm', ...
                  'Non-binary data cannot be clustered using Hamming distance.');
        end
    end
else
    error('stats:kmeans:InvalidDistance', ...
          'The ''distance'' parameter value must be a string.');
end

if ischar(start)
    startNames = {'uniform','sample','cluster'};
    i = strmatch(lower(start), startNames);
    if length(i) > 1
        error('stats:kmeans:AmbiguousStart', ...
              'Ambiguous ''start'' parameter value:  %s.', start);
    elseif isempty(i)
        error('stats:kmeans:UnknownStart', ...
              'Unknown ''start'' parameter value:  %s.', start);
    elseif isempty(k)
        error('stats:kmeans:MissingK', ...
              'You must specify the number of clusters, K.');
    end
    start = startNames{i};
    if strcmp(start, 'uniform')
        if strcmp(distance, 'hamming')
            error('stats:kmeans:UniformStartForHamm', ...
                  'Hamming distance cannot be initialized with uniform random values.'↩
                      );
        end
        Xmins = min(X,1);
        Xmaxs = max(X,1);
    end
elseif isnumeric(start)
    CC = start;
    start = 'numeric';
    if isempty(k)
        k = size(CC,1);
    elseif k ~= size(CC,1);
        error('stats:kmeans:MisshapedStart', ...
              'The ''start'' matrix must have K rows.');
    elseif size(CC,2) ~= p
        error('stats:kmeans:MisshapedStart', ...
              'The ''start'' matrix must have the same number of columns as X.');
    end
    if isempty(reps)
        reps = size(CC,3);
    elseif reps ~= size(CC,3);
```

```
        error('stats:kmeans:MisshapedStart', ...
              'The third dimension of the ''start'' array must match the ''replicates''↩
                  ' parameter value.');
    end

    %Need to center explicit starting points for 'correlation'. (Re)normalization
    %for 'cosine'/'correlation' is done at each iteration.
    if isequal(distance, 'correlation')
        CC = CC - repmat(mean(CC,2),[1,p,1]);
    end
else
    error('stats:kmeans:InvalidStart', ...
          'The ''start'' parameter value must be a string or a numeric matrix or array↩
              .');
end

if ischar(emptyact)
    emptyactNames = {'error','drop','singleton'};
    i = strmatch(lower(emptyact), emptyactNames);
    if length(i) > 1
        error('stats:kmeans:AmbiguousEmptyAction', ...
              'Ambiguous ''emptyaction'' parameter value:  %s.', emptyact);
    elseif isempty(i)
        error('stats:kmeans:UnknownEmptyAction', ...
              'Unknown ''emptyaction'' parameter value:  %s.', emptyact);
    end
    emptyact = emptyactNames{i};
else
    error('stats:kmeans:InvalidEmptyAction', ...
          'The ''emptyaction'' parameter value must be a string.');
end

if ischar(display)
    i = strmatch(lower(display), strvcat('off','notify','final','iter'));
    if length(i) > 1
        error('stats:kmeans:AmbiguousDisplay', ...
              'Ambiguous ''display'' parameter value:  %s.', display);
    elseif isempty(i)
        error('stats:kmeans:UnknownDisplay', ...
              'Unknown ''display'' parameter value:  %s.', display);
    end
    display = i-1;
else
    error('stats:kmeans:InvalidDisplay', ...
          'The ''display'' parameter value must be a string.');
end

if k == 1
    error('stats:kmeans:OneCluster', ...
          'The number of clusters must be greater than 1.');
elseif n < k
    error('stats:kmeans:TooManyClusters', ...
          'X must have more rows than the number of clusters.');
end

%Assume one replicate
if isempty(reps)
    reps = 1;
end

%
%Done with input argument processing, begin clustering
%

dispfmt = '%6d\t%6d\t%8d\t%12g';
D = repmat(NaN,n,k);    %point-to-cluster distances
Del = repmat(NaN,n,k); %reassignment criterion
m = zeros(k,1);

totsumDBest = Inf;
for rep = 1:reps
    switch start
    case 'uniform'
        C = unifrnd(Xmins(ones(k,1),:), Xmaxs(ones(k,1),:));
        %For 'cosine' and 'correlation', these are uniform inside a subset
        %of the unit hypersphere.  Still need to center them for
        %'correlation'.  (Re)normalization for 'cosine'/'correlation' is
```

```
        % done at each iteration.
        if isequal(distance, 'correlation')
            C = C - repmat(mean(C,2),1,p);
        end
        if isa(X,'single')
            C = single(C);
        end
    case 'sample'
        C = X(randsample(n,k),:);
        if ~isfloat(C)      %X may be logical
            C = double(C);
        end
    case 'cluster'
        Xsubset = X(randsample(n,floor(.1*n)),:);
        [dum, C] = kmeans(Xsubset, k, varargin{:}, 'start','sample', 'replicates',1);
    case 'numeric'
        C = CC(:,:,rep);
    end
    changed = 1:k; % everything is newly assigned
    idx = zeros(n,1);
    totsumD = Inf;

    if display > 2 % 'iter'
        disp(sprintf('  iter\t phase\t     num\t         sum'));
    end

    %
    % Begin phase one:  batch reassignments
    %

    converged = false;
    iter = 0;
    while true
        % Compute the distance from every point to each cluster centroid
        D(:,changed) = distfun(X, C(changed,:), distance, iter);

        % Compute the total sum of distances for the current configuration.
        % Can't do it first time through, there's no configuration yet.
        if iter > 0
            totsumD = sum(D((idx-1)*n + (1:n)'));
            % Test for a cycle: if objective is not decreased, back out
            % the last step and move on to the single update phase
            if prevtotsumD <= totsumD
                idx = previdx;
                [C(changed,:), m(changed)] = gcentroids(X, idx, changed, distance, ↩
                    Xsort, Xord);
                iter = iter - 1;
                break;
            end
            if display > 2 % 'iter'
                disp(sprintf(dispfmt,iter,1,length(moved),totsumD));
            end
            if iter >= maxit, break; end
        end

        % Determine closest cluster for each point and reassign points to clusters
        previdx = idx;
        prevtotsumD = totsumD;
        [d, nidx] = min(D, [], 2);

        if iter == 0
            % Every point moved, every cluster will need an update
            moved = 1:n;
            idx = nidx;
            changed = 1:k;
        else
            % Determine which points moved
            moved = find(nidx ~= previdx);
            if length(moved) > 0
                % Resolve ties in favor of not moving
                moved = moved(D((previdx(moved)-1)*n + moved) > d(moved));
            end
            if length(moved) == 0
                break;
            end
            idx(moved) = nidx(moved);
```

```
            % Find clusters that gained or lost members
            changed = unique([idx(moved); previdx(moved)]);
        end

        % Calculate the new cluster centroids and counts.
        [C(changed,:), m(changed)] = gcentroids(X, idx, changed, distance, Xsort, Xord↩
            );
        iter = iter + 1;

        % Deal with clusters that have just lost all their members
        empties = changed(m(changed) == 0);
        if ~isempty(empties)
            switch emptyact
            case 'error'
                error('stats:kmeans:EmptyCluster', ...
                      'Empty cluster created at iteration %d.',iter);
            case 'drop'
                % Remove the empty cluster from any further processing
                D(:,empties) = NaN;
                changed = changed(m(changed) > 0);
                if display > 0
                    warning('stats:kmeans:EmptyCluster', ...
                            'Empty cluster created at iteration %d.',iter);
                end
            case 'singleton'
                if display > 0
                    warning('stats:kmeans:EmptyCluster', ...
                            'Empty cluster created at iteration %d.',iter);
                end

                for i = empties
                    % Find the point furthest away from its current cluster.
                    % Take that point out of its cluster and use it to create
                    % a new singleton cluster to replace the empty one.
                    [dlarge, lonely] = max(d);
                    from = idx(lonely); % taking from this cluster
                    C(i,:) = X(lonely,:);
                    m(i) = 1;
                    idx(lonely) = i;
                    d(lonely) = 0;

                    % Update clusters from which points are taken
                    [C(from,:), m(from)] = gcentroids(X, idx, from, distance, Xsort, ↩
                        Xord);
                    changed = unique([changed from]);
                end
            end
        end
    end % phase one

    % Initialize some cluster information prior to phase two
    switch distance
    case 'cityblock'
        Xmid = zeros([k,p,2]);
        for i = 1:k
            if m(i) > 0
                % Separate out sorted coords for points in i'th cluster,
                % and save values above and below median, component-wise
                Xsorted = reshape(Xsort(idx(Xord)==i), m(i), p);
                nn = floor(.5*m(i));
                if mod(m(i),2) == 0
                    Xmid(i,:,1:2) = Xsorted([nn, nn+1],:)';
                elseif m(i) > 1
                    Xmid(i,:,1:2) = Xsorted([nn, nn+2],:)';
                else
                    Xmid(i,:,1:2) = Xsorted([1, 1],:)';
                end
            end
        end
    case 'hamming'
        Xsum = zeros(k,p);
        for i = 1:k
            if m(i) > 0
                % Sum coords for points in i'th cluster, component-wise
                Xsum(i,:) = sum(X(idx==i,:), 1);
            end
        end
```

```
    end

    %
    % Begin phase two:  single reassignments
    %
    changed = find(m' > 0);
    lastmoved = 0;
    nummoved = 0;
    iter1 = iter;
    while iter < maxit
        % Calculate distances to each cluster from each point, and the
        % potential change in total sum of errors for adding or removing
        % each point from each cluster.  Clusters that have not changed
        % membership need not be updated.
        %
        % Singleton clusters are a special case for the sum of dists
        % calculation.  Removing their only point is never best, so the
        % reassignment criterion had better guarantee that a singleton
        % point will stay in its own cluster.  Happily, we get
        % Del(i,idx(i)) == 0 automatically for them.
        switch distance
        case 'sqeuclidean'
            for i = changed
                mbrs = (idx == i);
                sgn = 1 - 2*mbrs; % -1 for members, 1 for nonmembers
                if m(i) == 1
                    sgn(mbrs) = 0; % prevent divide-by-zero for singleton mbrs
                end
                Del(:,i) = (m(i) ./ (m(i) + sgn)) .* sum((X - C(repmat(i,n,1),:)).^2, ↩
                    2);
            end
        case 'cityblock'
            for i = changed
                if mod(m(i),2) == 0 % this will never catch singleton clusters
                    ldist = Xmid(repmat(i,n,1),:,1) - X;
                    rdist = X - Xmid(repmat(i,n,1),:,2);
                    mbrs = (idx == i);
                    sgn = repmat(1-2*mbrs, 1, p); % -1 for members, 1 for nonmembers
                    Del(:,i) = sum(max(0, max(sgn.*rdist, sgn.*ldist)), 2);
                else
                    Del(:,i) = sum(abs(X - C(repmat(i,n,1),:)), 2);
                end
            end
        case {'cosine','correlation'}
            % The points are normalized, centroids are not, so normalize them
            normC(changed) = sqrt(sum(C(changed,:).^2, 2));
            if any(normC < eps(class(normC))) % small relative to unit-length data ↩
                points
                error('stats:kmeans:ZeroCentroid', ...
                      'Zero cluster centroid created at iteration %d.',iter);
            end
            % This can be done without a loop, but the loop saves memory allocations
            for i = changed
                XCi = X * C(i,:)';
                mbrs = (idx == i);
                sgn = 1 - 2*mbrs; % -1 for members, 1 for nonmembers
                Del(:,i) = 1 + sgn .*...
                      (m(i).*normC(i) - sqrt((m(i).*normC(i)).^2 + 2.*sgn.*m(i).*XCi +↩
                           1));
            end
        case 'hamming'
            for i = changed
                if mod(m(i),2) == 0 % this will never catch singleton clusters
                    % coords with an unequal number of 0s and 1s have a
                    % different contribution than coords with an equal
                    % number
                    unequal01 = find(2*Xsum(i,:) ~= m(i));
                    numequal01 = p - length(unequal01);
                    mbrs = (idx == i);
                    Di = abs(X(:,unequal01) - C(repmat(i,n,1),unequal01));
                    Del(:,i) = (sum(Di, 2) + mbrs*numequal01) / p;
                else
                    Del(:,i) = sum(abs(X - C(repmat(i,n,1),:)), 2) / p;
                end
            end
        end
```

```
        % Determine best possible move, if any, for each point.  Next we
        % will pick one from those that actually did move.
        previdx = idx;
        prevtotsumD = totsumD;
        [minDel, nidx] = min(Del, [], 2);
        moved = find(previdx ~= nidx);
        if length(moved) > 0
            % Resolve ties in favor of not moving
            moved = moved(Del((previdx(moved)-1)*n + moved) > minDel(moved));
        end
        if length(moved) == 0
            % Count an iteration if phase 2 did nothing at all, or if we're
            % in the middle of a pass through all the points
            if (iter - iter1) == 0 | nummoved > 0
                iter = iter + 1;
                if display > 2 % 'iter'
                    disp(sprintf(dispfmt,iter,2,nummoved,totsumD));
                end
            end
            converged = true;
            break;
        end

        % Pick the next move in cyclic order
        moved = mod(min(mod(moved - lastmoved - 1, n) + lastmoved), n) + 1;

        % If we've gone once through all the points, that's an iteration
        if moved <= lastmoved
            iter = iter + 1;
            if display > 2 % 'iter'
                disp(sprintf(dispfmt,iter,2,nummoved,totsumD));
            end
            if iter >= maxit, break; end
            nummoved = 0;
        end
        nummoved = nummoved + 1;
        lastmoved = moved;

        oidx = idx(moved);
        nidx = nidx(moved);
        totsumD = totsumD + Del(moved,nidx) - Del(moved,oidx);

        % Update the cluster index vector, and rhe old and new cluster
        % counts and centroids
        idx(moved) = nidx;
        m(nidx) = m(nidx) + 1;
        m(oidx) = m(oidx) - 1;
        switch distance
        case 'sqeuclidean'
            C(nidx,:) = C(nidx,:) + (X(moved,:) - C(nidx,:)) / m(nidx);
            C(oidx,:) = C(oidx,:) - (X(moved,:) - C(oidx,:)) / m(oidx);
        case 'cityblock'
            for i = [oidx nidx]
                % Separate out sorted coords for points in each cluster.
                % New centroid is the coord median, save values above and
                % below median.  All done component-wise.
                Xsorted = reshape(Xsort(idx(Xord)==i), m(i), p);
                nn = floor(.5*m(i));
                if mod(m(i),2) == 0
                    C(i,:) = .5 * (Xsorted(nn,:) + Xsorted(nn+1,:));
                    Xmid(i,:,1:2) = Xsorted([nn, nn+1],:)';
                else
                    C(i,:) = Xsorted(nn+1,:);
                    if m(i) > 1
                        Xmid(i,:,1:2) = Xsorted([nn, nn+2],:)';
                    else
                        Xmid(i,:,1:2) = Xsorted([1, 1],:)';
                    end
                end
            end
        case {'cosine','correlation'}
            C(nidx,:) = C(nidx,:) + (X(moved,:) - C(nidx,:)) / m(nidx);
            C(oidx,:) = C(oidx,:) - (X(moved,:) - C(oidx,:)) / m(oidx);
        case 'hamming'
            % Update summed coords for points in each cluster.  New
            % centroid is the coord median.  All done component-wise.
            Xsum(nidx,:) = Xsum(nidx,:) + X(moved,:);
```

```
            Xsum(oidx,:) = Xsum(oidx,:) - X(moved,:);
            C(nidx,:) = .5*sign(2*Xsum(nidx,:) - m(nidx)) + .5;
            C(oidx,:) = .5*sign(2*Xsum(oidx,:) - m(oidx)) + .5;
        end
        changed = sort([oidx nidx]);
    end % phase two

    if (~converged) & (display > 0)
        warning('stats:kmeans:FailedToConverge', ...
                'Failed to converge in %d iterations.', maxit);
    end

    % Calculate cluster-wise sums of distances
    nonempties = find(m(:)'>0);
    D(:,nonempties) = distfun(X, C(nonempties,:), distance, iter);
    d = D((idx-1)*n + (1:n)');
    sumD = zeros(k,1);
    for i = 1:k
        sumD(i) = sum(d(idx == i));
    end
    if display > 1 % 'final' or 'iter'
        disp(sprintf('%d iterations, total sum of distances = %g',iter,totsumD));
    end

    % Save the best solution so far
    if totsumD < totsumDBest
        totsumDBest = totsumD;
        idxBest = idx;
        Cbest = C;
        sumDBest = sumD;
        if nargout > 3
            Dbest = D;
        end
    end
end

% Return the best solution
idx = idxBest;
C = Cbest;
sumD = sumDBest;
if nargout > 3
    D = Dbest;
end


%------------------------------------------------------------------

function D = distfun(X, C, dist, iter)
%DISTFUN Calculate point to cluster centroid distances.
[n,p] = size(X);
D = zeros(n,size(C,1));
nclusts = size(C,1);

switch dist
case 'sqeuclidean'
    for i = 1:nclusts
        D(:,i) = sum((X - C(repmat(i,n,1),:)).^2, 2);
    end
case 'cityblock'
    for i = 1:nclusts
        D(:,i) = sum(abs(X - C(repmat(i,n,1),:)), 2);
    end
case {'cosine','correlation'}
    % The points are normalized, centroids are not, so normalize them
    normC = sqrt(sum(C.^2, 2));
    if any(normC < eps(class(normC))) % small relative to unit-length data points
        error('stats:kmeans:ZeroCentroid', ...
              'Zero cluster centroid created at iteration %d.',iter);
    end
    % This can be done without a loop, but the loop saves memory allocations
    for i = 1:nclusts
        D(:,i) = 1 - (X * C(i,:)') ./ normC(i);
    end
case 'hamming'
    for i = 1:nclusts
        D(:,i) = sum(abs(X - C(repmat(i,n,1),:)), 2) / p;
    end
```

```
end


%-------------------------------------------------------------------

function [centroids, counts] = gcentroids(X, index, clusts, dist, Xsort, Xord)
%GCENTROIDS Centroids and counts stratified by group.
[n,p] = size(X);
num = length(clusts);
centroids = repmat(NaN, [num p]);
counts = zeros(num,1);
for i = 1:num
    members = find(index == clusts(i));
    if length(members) > 0
        counts(i) = length(members);
        switch dist
        case 'sqeuclidean'
            centroids(i,:) = sum(X(members,:),1) / counts(i);
        case 'cityblock'
            % Separate out sorted coords for points in i'th cluster,
            % and use to compute a fast median, component-wise
            Xsorted = reshape(Xsort(index(Xord)==clusts(i)), counts(i), p);
            nn = floor(.5*counts(i));
            if mod(counts(i),2) == 0
                centroids(i,:) = .5 * (Xsorted(nn,:) + Xsorted(nn+1,:));
            else
                centroids(i,:) = Xsorted(nn+1,:);
            end
        case {'cosine','correlation'}
            centroids(i,:) = sum(X(members,:),1) / counts(i); % unnormalized
        case 'hamming'
            % Compute a fast median for binary data, component-wise
            centroids(i,:) = .5*sign(2*sum(X(members,:), 1) - counts(i)) + .5;
        end
    end
end
```

B.1.12. Corrcoef

```
unction [r,p,rlo,rup] = corrcoef(x,varargin)
%CORRCOEF Correlation coefficients.
%   R=CORRCOEF(X) calculates a matrix R of correlation coefficients for
%   an array X, in which each row is an observation and each column is a
%   variable.
%
%   R=CORRCOEF(X,Y), where X and Y are column vectors, is the same as
%   R=CORRCOEF([X Y]).
%
%   If C is the covariance matrix, C = COV(X), then CORRCOEF(X) is
%   the matrix whose (i,j)'th element is
%
%          C(i,j)/SQRT(C(i,i)*C(j,j)).
%
%   [R,P]=CORRCOEF(...) also returns P, a matrix of p-values for testing
%   the hypothesis of no correlation.  Each p-value is the probability
%   of getting a correlation as large as the observed value by random
%   chance, when the true correlation is zero.  If P(i,j) is small, say
%   less than 0.05, then the correlation R(i,j) is significant.
%
%   [R,P,RLO,RUP]=CORRCOEF(...) also returns matrices RLO and RUP, of
%   the same size as R, containing lower and upper bounds for a 95%
%   confidence interval for each coefficient.
%
%   [...]=CORRCOEF(...,'PARAM1',VAL1,'PARAM2',VAL2,...) specifies additional
%   parameters and their values.  Valid parameters are the following:
%
%       Parameter  Value
%       'alpha'    A number between 0 and 1 to specify a confidence
%                  level of 100*(1-ALPHA)%.  Default is 0.05 for 95%
%                  confidence intervals.
%       'rows'     Either 'all' (default) to use all rows, 'complete' to
```

```
%                 use rows with no NaN values, or 'pairwise' to compute
%                 R(i,j) using rows with no NaN values in column i or j.
%
%   The p-value is computed by transforming the correlation to create a t
%   statistic having N-2 degrees of freedom, where N is the number of rows
%   of X.  The confidence bounds are based on an asymptotic normal
%   distribution of 0.5*log((1+R)/(1-R)), with an approximate variance equal
%   to 1/(N-3).  These bounds are accurate for large samples when X has a
%   multivariate normal distribution.  The 'pairwise' option can produce
%   an R matrix that is not positive definite.
%
%   Example:  Generate random data having correlation between column 4
%             and the other columns.
%       x = randn(30,4);        % uncorrelated data
%       x(:,4) = sum(x,2);      % introduce correlation
%       [r,p] = corrcoef(x)     % compute sample correlation and p-values
%       [i,j] = find(p<0.05);   % find significant correlations
%       [i,j]                   % display their (row,col) indices
%
%   Class support for inputs X,Y:
%      float: double, single
%
%   See also COV, STD.

%   Copyright 1984-2004 The MathWorks, Inc.
%   $Revision: 5.11.4.5 $  $Date: 2004/07/05 17:01:09 $

if nargin<1
   error('MATLAB:corrcoef:NotEnoughInputs', 'Not enough input arguments.');
end
if length(varargin)>0 && isnumeric(varargin{1})
   y = varargin{1};
   varargin(1) = [];

   %Two inputs, convert to equivalent single input
   x = x(:);
   y = y(:);
   if length(x)~=length(y)
      error('MATLAB:corrcoef:XYmismatch', 'The lengths of X and Y must match.');
   end
   x = [x y];
elseif ndims(x)>2
   error('MATLAB:corrcoef:InputDim', 'Inputs must be 2-D.');
end

%Quickly dispose of most common case
if nargout<=1 && isempty(varargin) && (size(x,2)>0 || size(x,1)<=1)
   r = correl(x);
   return
end

if ~isreal(x) && nargout>1
   error('MATLAB:corrcoef:ComplexInputs',...
         'Cannot compute p-values for complex inputs.');
end

%Process parameter name/value inputs
[alpha,userows,emsg] = getparams(varargin{:});
error(emsg);

%Deal with degenerate inputs
[n,m] = size(x);
if m<=1 || n<=1
   if n==1, x=x'; [n,m] = size(x); end
   if n<=1 || m==0
      r = NaN(m,m,class(x));
      p = r; rlo = r; rup = r;
   else
      r = ones(m,class(x));
      p = r; rlo = r; rup = r;
   end
   return
end

%Compute covariance matrix
t = isnan(x);
if isequal(userows,'all')
```

```
   somenan = 0;
else
   somenan = any(any(t));
end
if ~somenan || ~isequal(userows,'pairwise')
   %Remove observations with any missing values
   if somenan && isequal(userows,'complete')
      t = ~any(t,2);
      x = x(t,:);
   end

   %Subfunction computes correlation
   [r,n] = correl(x);
else
   %Compute correlation for each pair
   r = eye(m,class(x));
   n = diag(sum(t,1));
   jk = 1:2;
   for j = 2:m
      jk(1) = j;
      for k=1:j-1
         jk(2) = k;
         tjk = ~any(t(:,jk),2);
         njk = sum(tjk);
         if njk<=1
            rjk = NaN;
         else
            rjk = correl(x(tjk,jk));
            rjk = rjk(1,2);
         end
         r(j,k) = rjk;
         n(j,k) = njk;
      end
   end
   r = r + tril(r,-1)';
   n = n + tril(n,-1)';
end

%Compute p-value if requested
if nargout>=2
   %Operate on half of symmetric matrix
   lowerhalf = (tril(ones(m),-1)>0);
   rv = r(lowerhalf)';
   lhsize = size(rv);
   nv = n;
   if length(n)>1, nv = nv(lowerhalf)'; end

   %Tstat=Inf and p=0 if abs(r)==1
   denom = (1 - rv.^2);
   Tstat = Inf + zeros(size(rv));
   Tstat(rv<0) = -Inf;

   t = denom>0;
   rv = rv(t);
   if length(n)>1
      nvt = nv(t);
   else
      nvt = nv;
   end
   Tstat(t) = rv .* sqrt((nvt-2) ./ denom(t));
   p = zeros(m,class(x));

   p(lowerhalf) = 2*tpvalue(-abs(Tstat),nv-2);
   p = p + p' + eye(m,class(x));
   risnan = isnan(r);
   p(risnan) = NaN;
   if nargout>=3
      %Compute confidence bound if requested
      z = NaN(lhsize,class(x));
      z(t) = 0.5 * log((1+rv)./(1-rv));
      zalpha = NaN(size(nv),class(x));
      if any(nv>3)
         zalpha(nv>3) = (-erfinv(alpha - 1)) .* sqrt(2) ./ sqrt(nv(nv>3)-3);
      end
      rlo = zeros(m,class(x));
      rlo(lowerhalf) = tanh(z-zalpha);
      rlo = rlo + rlo' + eye(m,class(x));
```

```
      rup = zeros(m,class(x));
      rup(lowerhalf) = tanh(z+zalpha);
      rup = rup + rup' + eye(m,class(x));

      rlo(risnan) = NaN;
      rup(risnan) = NaN;
   end
end


%-------------------------------------------------------
function [r,n] = correl(x)
%CORREL Compute correlation matrix without error checking.

n = size(x,1);
c = cov(x);
d = sqrt(diag(c));
r = c./(d*d');

%-------------------------------------------------------
function p = tpvalue(x,v)
%TPVALUE Compute p-value for t statistic

normcutoff = 1e7;
if length(x)~=1 && length(v)==1
   v = repmat(v,size(x));
end

% Initialize P to zero.
p=zeros(size(x));

% use special cases for some specific values of v
k = find(v==1);
if any(k)
    p(k) = .5 + atan(x(k))/pi;
end
k = find(v>=normcutoff);
if any(k)
    p(k) = 0.5 * erfc(-x(k) ./ sqrt(2));
end

% See Abramowitz and Stegun, formulas 26.5.27 and 26.7.1
k = find(x ~= 0 & v ~= 1 & v > 0 & v < normcutoff);
if any(k),                              % first compute F(-|x|)
    xx = v(k) ./ (v(k) + x(k).^2);
    p(k) = betainc(xx, v(k)/2, 0.5)/2;
end

% Adjust for x>0.  Right now p<0.5, so this is numerically safe.
k = find(x > 0 & v ~= 1 & v > 0 & v < normcutoff);
if any(k), p(k) = 1 - p(k); end

p(x == 0 & v ~= 1 & v > 0) = 0.5;

% Return NaN for invalid inputs.
p(v <= 0 | isnan(x) | isnan(v)) = NaN;

%-------------------------------------------------------
function [alpha,userows,emsg] = getparams(varargin)
%GETPARAMS Process input parameters for CORRCOEF
alpha = 0.05;
userows = 'all';
emsg = '';
while(length(varargin)>0)
   if length(varargin)==1
      emsg = 'Unmatched parameter name/value pair.';
      return
   end
   pname = varargin{1};
   if ~ischar(pname)
      emsg = 'Invalid argument name.';
      return
   end
   pval = varargin{2};
   j = strmatch(pname, {'alpha' 'rows'});
   if isempty(j)
      emsg = sprintf('Invalid argument name ''%s''.',pname);
```

```
        return
    end
    if j==1
        alpha = pval;
    else
        userows = pval;
    end
    varargin(1:2) = [];
end

%Check for valid inputs
if ~isnumeric(alpha) || ~isscalar(alpha) || alpha<=0 || alpha>=1
    emsg = 'The ''alpha'' parameter must be a scalar between 0 and 1.';
end
oktypes = {'all' 'complete' 'pairwise'};
if isempty(userows) || ~ischar(userows)
    i = [];
else
    i = strmatch(lower(userows), oktypes);
end
if isempty(i)
    emsg = 'Valid row choices are ''all'', ''complete'', and ''pairwise''.';
end
userows = oktypes{i};
```

C

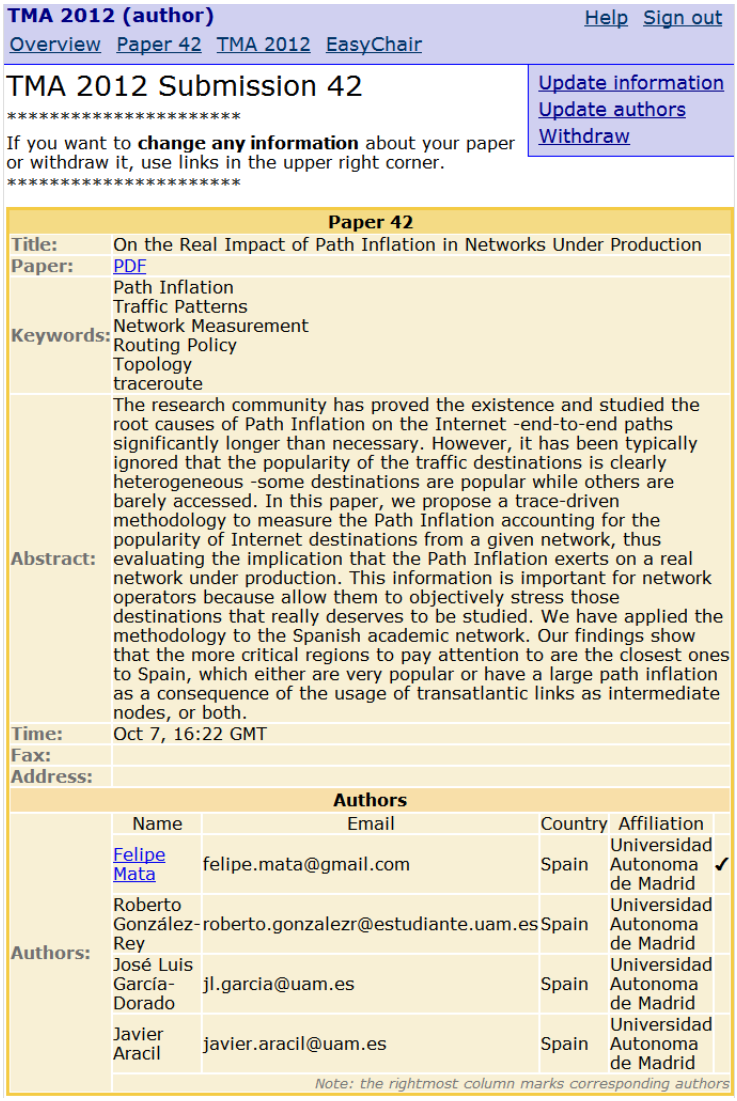
Publicaciones

A consecuencia de este proyecto, se ha enviado un artículo a la cuarta edición del "Traffic Monitoring and Analysis Workshop (TMA)". El nombre de este artículo es "On the Real Impact of Path Inflation in Networks Under Production", describiendo la metodología empleada para incluir el tráfico, además de la distancia enrutada, como factor determinante en la calidad de conexión entre dos emplazamientos.

Las fechas importantes del TMA 2012 son:

- Paper Submission: October 20, 2011
- Notification of Decision: December 20, 2011
- Camera Ready: January 8, 2012
- Workshop: March 12, 2012

El abstract que justifica su envío y el artículo en cuestión se incluyen en las siguientes páginas.



TMA 2012 (author) Help Sign out

Overview Paper 42 TMA 2012 EasyChair

TMA 2012 Submission 42

Update information
Update authors
Withdraw

****

If you want to **change any information** about your paper or withdraw it, use links in the upper right corner.

****

Paper 42	
Title:	On the Real Impact of Path Inflation in Networks Under Production
Paper:	PDF
Keywords:	Path Inflation Traffic Patterns Network Measurement Routing Policy Topology traceroute
Abstract:	The research community has proved the existence and studied the root causes of Path Inflation on the Internet -end-to-end paths significantly longer than necessary. However, it has been typically ignored that the popularity of the traffic destinations is clearly heterogeneous -some destinations are popular while others are barely accessed. In this paper, we propose a trace-driven methodology to measure the Path Inflation accounting for the popularity of Internet destinations from a given network, thus evaluating the implication that the Path Inflation exerts on a real network under production. This information is important for network operators because allow them to objectively stress those destinations that really deserves to be studied. We have applied the methodology to the Spanish academic network. Our findings show that the more critical regions to pay attention to are the closest ones to Spain, which either are very popular or have a large path inflation as a consequence of the usage of transatlantic links as intermediate nodes, or both.
Time:	Oct 7, 16:22 GMT
Fax:	
Address:	

Authors

	Name	Email	Country	Affiliation	
Authors:	Felipe Mata	felipe.mata@gmail.com	Spain	Universidad Autonoma de Madrid	✓
	Roberto González-Rey	roberto.gonzalezr@estudiante.uam.es	Spain	Universidad Autonoma de Madrid	
	José Luis García-Dorado	jl.garcia@uam.es	Spain	Universidad Autonoma de Madrid	
	Javier Aracil	javier.aracil@uam.es	Spain	Universidad Autonoma de Madrid	

Note: the rightmost column marks corresponding authors

Figura C.1: Abstract del artículo enviado al TMA2012.

On the Real Impact of Path Inflation in Networks Under Production

Felipe Mata*, Roberto González-Rey,
José Luis García-Dorado, and Javier Aracil

High Performance Computing and Networking Group,
Escuela Politécnica Superior,
Universidad Autónoma de Madrid
Francisco Tomás y Valiente 11, 28049 Madrid, Spain
{felipe.mata,jl.garcia,javier.aracil}@uam.es
{roberto.gonzalezr}@estudiante.uam.es
http://www.hpcn.es

Abstract. The research community has proved the existence and studied the root causes of Path Inflation on the Internet—end-to-end paths significantly longer than necessary. However, it has been typically ignored that the popularity of the traffic destinations is clearly heterogeneous—some destinations are popular while others are barely accessed. In this paper, we propose a trace-driven methodology to measure the Path Inflation accounting for the popularity of Internet destinations from a given network, thus evaluating the implication that the Path Inflation exerts on a real network under production. This information is important for network operators because allow them to objectively stress those destinations that really deserves to be studied. We have applied the methodology to the Spanish academic network. Our findings show that the more critical regions to pay attention to are the closest ones to Spain, which either are very popular or have a large path inflation as a consequence of the usage of transatlantic links as intermediate nodes, or both.

Keywords: Path Inflation; Traffic Patterns; Network Measurement; Routing Policy; Topology; traceroute

1 Introduction

The existence of Path Inflation (PI) —end-to-end routes significantly larger than necessary— is a well-known fact since almost 15 years ago [6, 10]. The Internet community has largely studied the existence of this phenomenon and its root causes [13, 15], motivated by the impact that this circuitousness could exert on network performance. The network performance parameter that principally is affected, under normal circumstances, is the one way delay, which could be smaller in case a less inflated path between the end-hosts is used. As a result, the optimal throughput in TCP connections is later reached as a consequence

* Corresponding author.

of the slow start congestion control strategy [14], and the error rate increases —the more time a packet spends on the network the higher the chances that any problem affects it. Therefore, not only the existence and the causes of PI has been analyzed, but also some procedures for reducing it have been devised [13]. Among these solutions, we find the proposal of including effective mechanisms to achieve optimal paths directly to BGP, such as appending geographic coordinates to route advertisements. With this information, a trade-off between hop-count metrics and geographic distance could be used in order to improve the latency in the network.

However, these previous studies rely on a small set of pre-defined network hosts —up to several thousands of hosts. This limitation applies to the representativeness of the results they provide. On one hand, it could be possible that no one is interested in connecting between two of the selected hosts, so why to worry about inflation of the route between them? On the other hand, are all the analyzed connections equally likely? Previous work [7] shows that this is far to be the case. As it turns out, these results call for including network traffic analysis into PI metrics, in order to obtain representative information regarding what to measure and how to appropriately weight the obtained measures. Our work fill in this gap, leveraging on a network trace analysis to infer the connections that are in fact realized in the network under study, and which their popularity is among the population of customers of the network.

Each of these informations (inflation of paths and their popularity) is of paramount importance for network operators and service providers. On one hand, knowledge of the inflated paths allow the operators and providers to focus on the locations that are poorly connected, whereas knowledge of the popularity of the traffic destinations serves to focus on the most demanded destinations by their customers, and both tasks eventually result in similar traffic engineering tasks: improving current traffic inter-exchange relationships or establishing new ones. Consequently, merging the PI and remote locations popularity informations allows for setting priorities to these traffic engineering tasks, which the operators and providers should eventually take action on in a short time span.

In this paper, we provide the methodology for merging both metrics into a new one that is able to determine which connections should receive attendance first, and apply this analysis to the Spanish academic network RedIRIS. The measurement of the PI entails the identification of the intermediate nodes in a network path and the geographical mapping of IP addresses to measure the PI in terms of distance. The selection of representative nodes is based on a trace analysis, which should be at least 35 days long in order to obtain stable destination patterns according to the results of [7]. Our findings after applying the proposed metric show that the more critical regions to pay attention are the closest ones to Spain, which either are very popular or have a large PI as a consequence of the usage of transatlantic links as intermediate nodes, or both. Our results show that a byte from our network travels more than 8000 extra kilometers in average on the Internet to reach its destination.

After reviewing the related work in Section 2, the rest of the paper is organized as follows. Section 3 describes the network analyzed in this study, whereas Section 4 and Section 5 are devoted to describe the methodology and present the results, respectively. Then, a discussion of the obtained results and their consequences is presented in Section 6. Finally, Section 7 summarize the achievements and concludes the paper.

2 Related Work

In this work we study the path inflation in the Spanish academic network, estimating the routing distance as the sum of the geographical distances between each router of a given path, by means of traceroute [4] and tcptraceroute tools [18]. Consequently, let us divide this section into these two areas, first *Path Inflation* and then *geolocation*.

2.1 Path Inflation

The path inflation phenomenon has received much attention by the Internet community, since that, almost 15 years ago, the authors in [6, 10] found that the routes in the Internet are clearly longer than necessary. From then, the Internet community has tried to characterize the PI, explain the causes of such phenomenon, and study its correlation with the performance experienced by users.

The authors in [12] explain that there are both technical and economic reasons to expect no optimal Internet routes. Specifically, in that paper it is found that between the 30-80% of the paths are not optimal. On one hand, wide-area routing protocols do not incorporate performance measurements into their decisions. On the other hand, the administrator of a given AS may refuse to carry traffic of another ISP because of competitive reasons or simply because the lack of a contractual agreement.

The authors in [17, 16] focus on the signification of such economic reasons. Those papers show that about 20% of the Internet paths are inflated by more than 50% in terms of number of hops from the source to the destination if the optimal route path would be selected. However, the authors point out the they are assuming that all the routes between each pair of studied hosts are equally likely, and this is not true [7].

Similarly, the authors in [13] wonder why Internet paths are sometimes absurdly long. They analyzed this fact from the intra- and inter-domain ISP points of view as well from the ISP peering relationship. They found that the intra-domain routing is the most significant factor in the path inflation phenomenon and mainly due to the routers typically use minimum AS-hop count ignoring other metrics. They concluded that almost 50% of the paths were longer that the shortest available path because of intra-domain routing. In addition, they remark that according to their measurements the geography is a good indicator of latency for most of the studied ISPs. However, the authors notice that their

study is assuming, not in a totally realistic way, that all nodes are equal important regardless traffic volumes, and point out that would be more interesting to study the fraction of packets that suffer path inflation than the study of the fraction of paths.

The authors in [9] worked further to extend the characterization of the PI phenomenon. In that study, the authors measure the PI as the ratio between the linearized distance, i.e., the sum of distances in kilometers between each of the nodes a path includes, and the end-to-end geographic distance. They evaluated the PI from 20 institutions (placed in the U.S., Sweden, Italy, and Hungary) and two home cable modem networks to an extensive set of pre-defined destination hosts. Such a set included essentially web servers, some of them located on U.S. campuses, and public libraries which were easily geographically placed. They again found that the PI is a common phenomenon in the Internet, and that it strongly depends on the geographic location of the end-hosts. This was explained by the fact that the connectivity of the different parts of the world are far from being homogeneous. That is, this latter paper takes an arbitrary set of end-hosts and no distinctions on its popularity were performed, however, as the authors showed there are significant differences between the PI from some geographic areas to others —specifically, they pointed out that PI in San Francisco bay area was significant smaller than in other places.

Bearing all these previous works in mind, we note that the *real* impact of the PI is not currently well known, that is, it is proven that it exists in an Internet in which all the destinations were equal popular and all the places were equally connected. As this is not the case, in this paper we take a step further and try to fill such a gap, i.e., the relationship between PI and end-host location popularity.

2.2 Geolocation

There are several ways to find the physical location of an IP address, which can be classified in active or passive [1]. The former class includes mechanism based on the delays between a set of reference nodes (named, landmarks) and the objective node, examples of this are [19, 9, 5]. This approach is based on the linear correlation between the delay in the networks and the geographical distance between the objective and the set of landmark nodes. Essentially, it is expected that hosts placed in a similar geographical distance present similar delays measured typically by means of the tool *ping*. Such correlation has been found in some regions of the Internet, essentially north America and west Europe [19], but the precision is limited in the rest of the world. As the target of our work is to span all the possible destinations in the world, this precision depending on the area represents a significant caveat.

On the other hand, the passive mechanism to locate hosts are typically based on i) the identification of some patter in the routers' DNS name —essentially, names and codes of cities or airports— that allows to infer their location or, at least, AS and ii) the use of databases, typically commercial applications, which relate directly IP and location. A example of the use of DNS name patterns is GeoTrack [9]. GeoTrack estimates the location of the objective node as that of

last identifiable router in a given path. Its precision tend to be notable but the number of routers whose name follow some recognizable policy to be named is limited, although according to the authors these are more than 70%; in addition GeoTrack is designed to locate routers but unlikely it can locate final hosts.

According to the literature a more accurate option is the database approach whose implementation is poorly known because they are typically commercial application. The performance of this approach have been studied and reported ([3, 11]), resulting in median error of some 60 kilometers. In this study, we have used the free version of the *GeoIP Country* database of *MaxMind*, i.e., *GeoLite Country*, which has an accuracy of 99.5% according to the company [8] and outperforms other equivalent approaches [11]. Such database has entries for the country code, country name and continent data.

Finally, for a better understanding of geolocation procedures, the reader is referred to [2, Section 5.3.6] and references therein.

3 Description of the Network

This work aims to characterize the PI of the Spanish academic network RedIRIS, paying special attention to the connections that are established from it. RedIRIS network comprises more than 350 institutions, mainly universities and research centers, and kindly provides us with flow traffic measurements for research purposes[1]. Figure 1 graphically describes the network. Our premises are located under the Point of Presence of Madrid, which is at one hop distance from the RedIRIS external gateway that connects to the rest of the Internet through commercial links (TeliaSonera, Global Crossing, Espanix etc.) and with the European research network, GEANT.

4 Methodology

4.1 Selection of Representative Destinations

Based on the results from [7], at least 35 days of traffic measurements from a subnetwork should be aggregated to make the traffic distribution of the geographic destinations stabilize. Consequently, we have gathered 35 days worth of flow measurements from the Point of Presence in Madrid, which allows us to calculate the number of bytes that are destined to each foreign country, and the IP addresses that are requested. The traffic traces used in this analysis partially comprise April-May 2010. From this dataset, we have ruled out those countries that received less than 0.005% of the total sent traffic. Overall, there are more than 31 million different IP addresses receiving almost 1 PB of traffic after the filtering process.

This is a vast dataset compared to others analyzed in the literature that was in the order of thousands IP addresses [15]. The distribution of traffic among the

[1] The data are stored and analyzed in full compilance with Spanish regulation concerning privacy of electronico communications

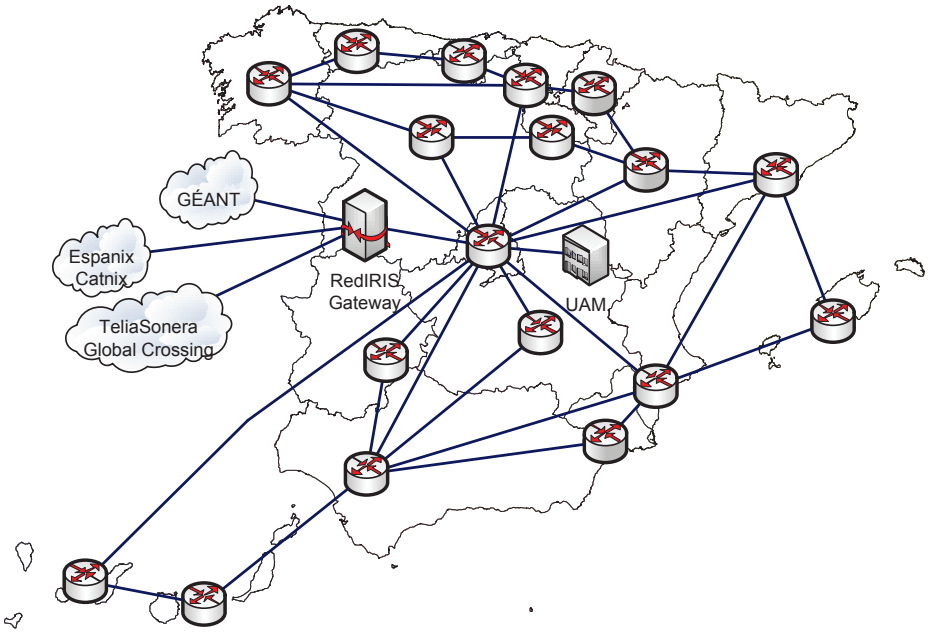


Fig. 1. Map of Spain and RedIRIS Points of Presence, showing the logical connections with the RedIRIS external gateway and UAM premises.

different countries that comprise the dataset is depicted in Fig. 2 in logarithmic scale. A logarithmic scaling is necessary in order to enhance visibility provided the power-law shape of the traffic distribution [7]. To draw the maps, we used the Google's Visualization API [2] that can be used directly as a gadget from Google docs.

4.2 Intermediate Nodes Identification

To identify the intermediate nodes between our premises and the target IP addresses in our dataset, we leverage on `traceroute` and `tcptraceroute` tools. Such tools provide an equivalent approach to identify the intermediate nodes in a path within two IP addresses, but based on different network protocols. `traceroute` sends out either UDP or ICMP echo request packets, whereas `tcptraceroute` uses TCP SYN packets to circumvent the widespread use of firewalls. These tools allow us to identify the IP addresses of the intermediate nodes, which we map to their geographic coordinates by means of the geolocation method described in Section 2.2.

Because our study is trace-driven, our results are limited to the lack of visibility of some Internet hosts that do not reply to `traceroute` or `tcptraceroute` messages. This fact is analyzed in Section 5.1. Further work will be needed to

[2] http://code.google.com/intl/es-ES/apis/visualization/documentation/gallery/intensitymap.html

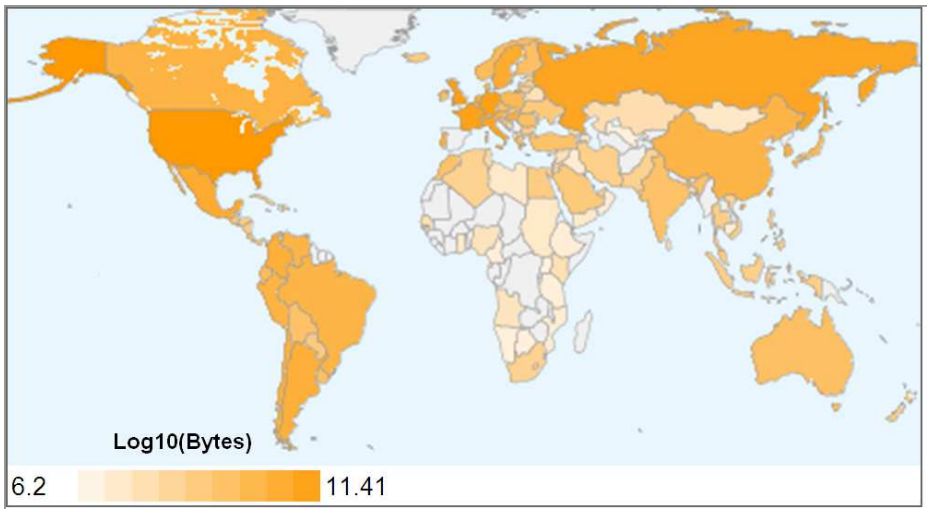


Fig. 2. Received traffic by country in logarithmic scale.

determine the extent to which our results generalize to other periods of time and other set of destinations. In addition, we have ruled out some instances of our dataset that lead to incongruent data, such as network paths traversed at higher speed than the speed-of-light.

4.3 Path Inflation Metric versus Weighted Path Inflation Metric

Other studies present in the literature has faced the Path Inflation analysis leveraging on different metrics, such as distance, time, or number of hops. In this paper, we will focus on a distance metric, such the one used in [15]. The authors of [15] define the PI metric as the ratio between the routing and the geographical distances, where the routing distance is estimated as the sum of the geographical distances between each pair of consecutive intermediate nodes. Defining the unidirectional path $\{X_j\}_{j=0}^{N_{a,b}}$ between the IP addresses a (IP_a) and b (IP_b) as $\mathrm{IP}_a = X_0 \rightarrow X_1 \rightarrow X_2 \rightarrow \cdots \rightarrow X_{N_{a,b}} = \mathrm{IP}_b$, where $N_{a,b}$ equals the number of intermediate nodes plus one, we obtain:

$$PI(\mathrm{IP}_a, \mathrm{IP}_b) = \frac{d_r(\mathrm{IP}_a, \mathrm{IP}_b)}{d_g(\mathrm{IP}_a, \mathrm{IP}_b)} = \frac{\sum_{j=0}^{N_{a,b}-1} d_g(X_j, X_{j+1})}{d_g(\mathrm{IP}_a, \mathrm{IP}_b)}, \tag{1}$$

where $d_g(X, Y)$ is the geographical distance between the locations of IP addresses X and Y, and we have used $d_r(\cdot, \cdot)$ to denote the routing distance.

Consequently, the larger the circuitousness of the path, the larger the PI metric, which is interpreted as the number of times the path is larger than what would be necessary if a straight route would be possible.

The limitation of this metric is that it does not take into account the amount of traffic that is destined to each destination host. To take into account the

amount of traffic, we group the PI metrics by destination country, taking the mean value as a representative, namely $\overline{PI}$. Such mean PI metric by country is then weighted with the base 10 logarithm of traffic that is destined to such country, in order to provide larger weights to the popular destinations, which we define as WPI:

$$WPI(\text{country}_c) = \overline{PI}(\text{country}_c) \log_{10}(\#\text{bytes destined to country}_c). \quad (2)$$

This metric measures the PI taking into account the connection patterns in the network under study. Popular countries $\overline{PI}$ is penalized, whereas the impact of $\overline{PI}$ in those countries which barely receive traffic is reduced.

5 Results

5.1 Traceroute Reachability

This section is devoted to analyze and compare the performance of `traceroute` and `tcptraceroute`. We perform such analysis to select the path-analysis tool that is able to identify the broadest set of path destinations. We define that a path to a given destination is properly identified when the IP addresses of all the intermediate nodes existing in the path can be collected. As one may expect, further destinies typically entail more intermediate nodes, which increase the chances of finding one host that does not reply to `traceroute` or `tcptraceroute` queries. The percentage of path destinations that are fully characterized by means of `tcptraceroute` is 17.74%, whereas `traceroute` only characterizes 10.49% of the destinations. Such results are also confirmed when observing them in a continent basis.

Table 1. `traceroute` and `tcptraceroute` performance analysis by continent.

Continent	`traceroute`	`tcptraceroute`
Europe	8.45%	12.72%
Asia	12.06%	18.40%
America	12.51%	25.63%
Africa	8.53%	16.36%
Australia	4.92%	8.10%

Consequently, given that `tcptraceroute` has better performance when compared to `traceroute` using our set of destinations, and that the set of IP addresses whose paths are fully characterized by `tcptraceroute` approximately comprises the set of `traceroute` fully characterized IP addresses, we decided to use `tcptraceroute` in what follows. Furthermore, we like to note that although there is a reduction in the number of IP addresses we eventually analyze due to the lack of fully characterization, this set is still very large compared to previous ones used in the literature, containing more than 5 million different IP addresses.

5.2 Path Inflation Results

In this section, we present the results of measuring the $\overline{PI}$ in the set of destinations that are fully characterized by `tcptraceroute`. As the PI metric has been deeply analyzed and characterized in previous works, we present such results as a benchmark for comparison with the results obtained when the traffic weights are introduced in the metric, as presented in the next section.

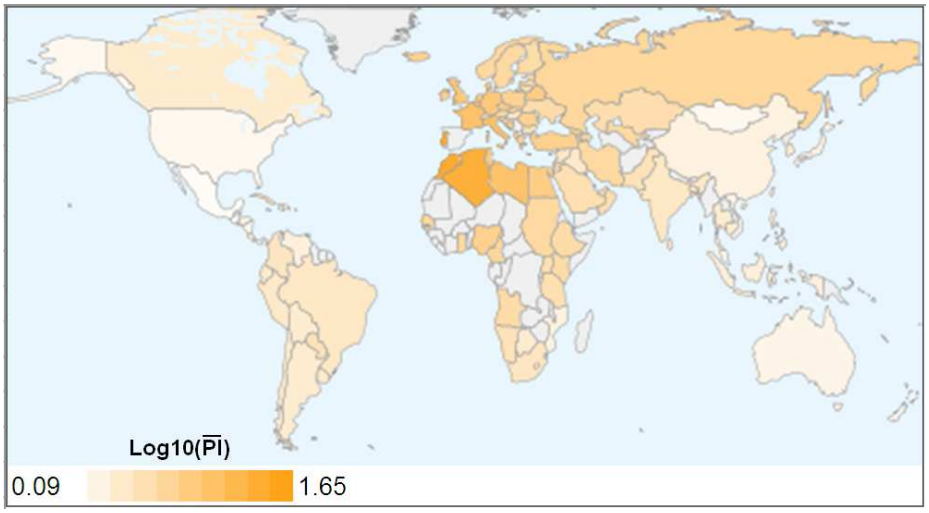
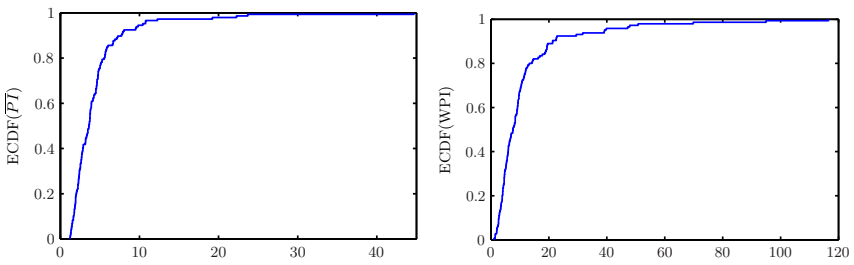


Fig. 3. Mean PI metric by country, in logarithmic scale

In Fig. 3 we present the mean value of the PI metric when grouped by country. As can be observed in the figure, the largest values of $\overline{PI}$ are found in the countries surrounding Spain. Although this may be counterintuitive at first glance —if the distance between two locations is not large, there should be less alternatives to choose a path within them, and consequently the circuitousness should be smaller—, this phenomenon is explained by the common usage of transatlantic routes (via USA), even for connecting pairs of locations within Europe. As a consequence of the popularity of the transatlantic routes, American countries suffer low values of $\overline{PI}$ when measured from our premises. In addition, we observe that Far East and Australian countries also have low values of $\overline{PI}$, which is a consequence of the common usage of a direct link connecting Europe and China.

On the other hand, we present this information summarized in a Cumulative Distribution Function (CDF) plot in Fig. 4(a), where we can observe that approximately 80% of the analyzed countries have paths larger than twice the distance measured in a straight line. This situation has consequences for instance in the minimum one way delay, which is a key performance indicator that

is usually related with quality of service/experience in multimedia services, such as voice conversations.



(a) $\overline{PI}$ (b) WPI

Fig. 4. Empirical Cumulative Distribution Function of the (a): $\overline{PI}$ and (b): WPI by country as observed from the central node of RedIRIS.

5.3 Weighted Path Inflation Results

In this section we present the results of weighting the average PI metric by country with the base 10 logarithm of the number of bytes that are destined to such country, comparing them with the ones previously presented as benchmark in the preceding section. Analogously as in Section 5.2, we present a world map with the WPI results in Fig. 5 and the data summarized in a CDF in Fig. 4(b).

In the world map figure (Fig. 5) we observe many differences with regard to Fig. 3. On one hand, we find that American countries now have negligible values of WPI, which is a consequence of the low average PI of USA (≈1.5) and that most of the connections to America go through USA. Consequently, although the popularity within the users of our network —consider that most of the world popular web services and contents are located in the USA, and our customers share the same language with most of the population in South America—, America can be regarded as well connected to RedIRIS. Similarly, we observe low values of WPI in South Africa, the Far East and Australia. However, the reasons are quite different. On one hand, the popularity of South African countries is low within RedIRIS users, whereas it is the average PI to the Far East and Australian countries what is low on the other hand. Anyhow, the connections to such countries should not require attention from RedIRIS network managers.

On the other hand, we observe that there are countries that have barely experienced variation in the $\overline{PI}$ and WPI maps. Those countries are mainly located in Europe and North Africa. Again, this is due to different reasons. On one hand, Spanish surrounding countries have the largest values of average PI due to the use of transatlantic links. In this group we include North African

countries as well as Andorra and Portugal, which do not have great popularity, but the connections to them are very poor. On the other hand, we found the remaining European countries, which have a mix of great popularity and middle-poor connections given the usage in some cases of the transatlantic links. In both situations, RedIRIS network managers should pay special attention to the connections to such countries, and improve them given that, taken into account the traffic destinations popularity, there is not a good connection between them and RedIRIS.

Finally, in Fig. 4(b) we observe the CDF of the WPI. Compared to the CDF of the $\overline{PI}$, we observe that both distributions are much alike. The major differences cannot be appreciated in the summarizing statistics, because they are in the form of a reordering of the countries. There were some countries with large average PI in Fig. 4(a) that now have small value of WPI, and the same in the opposite way (large WPI value despite a low average PI given the great popularity of the country). In any case, we observe a high clustering of countries in small values of WPI, and a flattening of the distribution for values of WPI larger than 20, which roughly coincides with the 90% percentile. We believe this could be treated as a *knee point*, and the RedIRIS network manager should inspect the countries which have WPI values larger than such knee point.

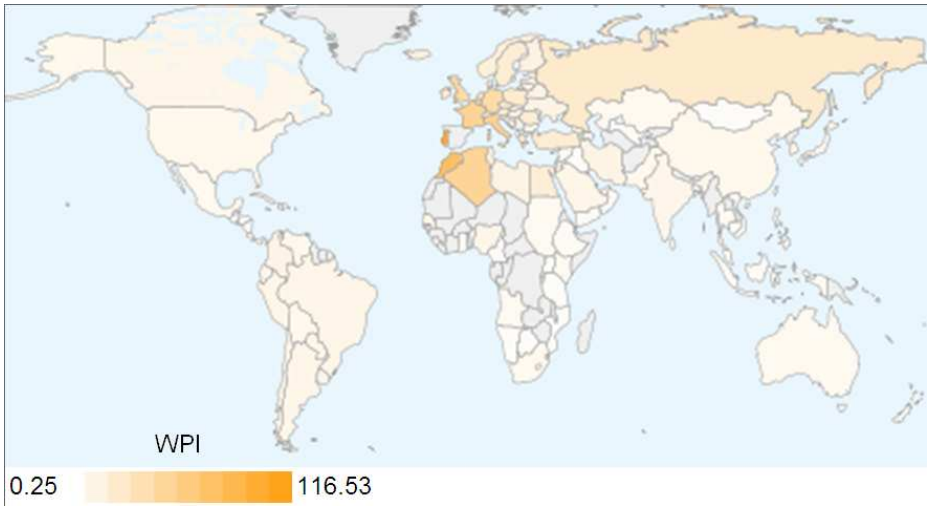


Fig. 5. WPI metric by country, in logarithmic scale after an axis rescaling to enhance visibility.

6 Discussion

When measuring the performance of a network, the PI has been usually used as a measure of the improvement that the network connections could have. Although there is not a direct relationship between network performance and geography [15], the path lengths show good correlation with minimum delays [13]. As a consequence, network operators and service providers should pay special attention to the PI in their networks, and take action to reduce such inflation as much as possible. However, the priority order needs to be established according to some other metrics, such as a description of the network usage by their customers. We have applied this reasoning in the present paper, and shown that the critical countries in terms of connectivity may be different if their popularity is taken into account. As a result, we show in Table 2 the comparison of the critical countries when only the average PI is taken into account, and the critical countries when this average PI is weighted with the amount of traffic that is destined to such country.

Table 2. Comparison of the critical countries with the average PI and WPI metrics.

Rank	Average PI	WPI
1	Andorra	Andorra
2	Portugal	Portugal
3	Morocco	Morocco
4	Algeria	Italy
5	San Marino	France
6	Luxembourg	Algeria
7	Italy	Belgium
8	Liechtenstein	United Kingdom
9	France	Germany
10	Libyan Arab Jamahiriya	Luxembourg
11	Belgium	Netherlands
12	Montenegro	Denmark
13	United Kingdom	Russian Federation
14	Germany	Czech Republic
15	Tunisia	Poland
16	Malta	Switzerland

As can be observed in the table, when only the average PI is taken into account there appear countries that, although have large values of PI, are not of interest from the network operator point of view, such as San Marino, Liechtenstein, Libyan Arab Jamahiriya or Montenegro as they do not reach a 1% of traffic share. However, when the popularity of the countries in terms of received bytes is considered, we can observe that such countries are filtered out. Consequently, leveraging on destinities popularity is of paramount interest for network opera-

tors before deciding which actions regarding improving the network connections take first.

7 Summary and Conclusions

This paper puts on perspective the importance of the PI in the current Internet. Whereas the previous studies detected the existence of such phenomenon in the Internet, we have determined to what extent such inflation is critical, taken into account the amount of the traffic that is destined to each location. We have proposed a new methodology to study the PI, essentially a trade-off between the popularity of the destination and the PI that suffer the traffic volumes sent to a given destination. Such methodology permits network operators and service providers to really identify those paths that deserve to be improved because they suffer PI and, at the same time, much traffic is carried by them.

We present the case study of the Spanish academic network, which have shown that a set of geographically close countries is not as well connected as desired. On the other hand, the PI metric weighted by destination popularity, WPI, has proven to be useful to filter out unpopular destinations, which according solely to the PI would have required special attention to the detriment of popular destinations, which affect a large number of users. This results encourage the Spanish academic network managers to pay attention to the international relationships with ISPs located in these areas. This actions will reduce the amount of extra distance that a byte travels in average, which is larger than 8000 kilometers according to our results.

As future work we plan to extend the work to commercial networks, and also focus on network performance metrics in addition to the popularity of the destinations.

Acknowledgments

The authors would like to thank the support of the Spanish Ministerio de Ciencia e Innovación (*MICINN*) to this work, under project *ANFORA* (TEC2009-13385) and the FPU fellowship program that has funded this research work.

References

1. Chatzopoulou, D., Kokkodis, M.: IP geolocation. Tech. rep., Computer Science and Engineering Dept, UC Riverside (2007)
2. Crovella, M., Krishnamurthy, B.: Internet measurement: infrastructure, traffic and applications. John Wiley and Sons Inc., New York (USA) (2006)
3. Gueye, B., Uhlig, S., Fdida, S.: Investigating the Imprecision of IP Block-Based Geolocation. In: Proceedings of International conference on Passive and Active Network Measurement. pp. 237–240. Louvain-la-neuve, Belgium (2007)
4. Jacobson, V.: traceroute, `ftp://ftp.ee.lbl.gov/traceroute.tar.gz`

5. Katz-Bassett, E., John, J., Krishnamurthy, A., Wetherall, D., Anderson, T., Chawathe, Y.: Towards IP geolocation using delay and topology measurements. In: Proceedings of ACM SIGCOMM Internet Measurement Conference. pp. 71–84. Rio de Janeriro, Brazil (2006)
6. Labovitz, C., Malan, G., Jahanian, F.: Internet routing instability. IEEE/ACM Transactions on Networking 6(5), 515 –528 (oct 1998)
7. Mata, F., García-Dorado, J.L., López de Vergara, J.E., Aracil, J.: Factor analysis of Internet traffic destinations from similar source networks. Accepted for its publication in Internet Research (2011)
8. MaxMind, L.L.C.: GeoLite free country database
9. Padmanabhan, V.N., Subramanian, L.: An investigation of geographic mapping techniques for internet hosts. In: Proceedings of ACM SIGCOMM. pp. 173–185. San Diego, USA (2001)
10. Paxson, V.: Measurements and analysis of end-to-end Internet dynamics. Ph.D. dissertation (1997)
11. Poese, I., Uhlig, S., Kaafar, M.A., Donnet, B., Gueye, B.: Ip geolocation databases: unreliable? SIGCOMM Comput. Commun. Rev. 41, 53–56 (2011)
12. Savage, S., Collins, A., Hoffman, E., Snell, J., Anderson, T.: The end-to-end effects of internet path selection. In: Proceedings of ACM SIGCOMM. pp. 289–299. Cambridge, USA (1999)
13. Spring, N., Mahajan, R., Anderson, T.: Quantifying the causes of path inflation. In: Proceedings of ACM SIGCOMM. pp. 113–124. Karlsruhe, Germany (2003)
14. Stevens, W.: RFC 2001: TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms (1997)
15. Subramanian, L., Padmanabhan, V.N., Katz, R.H.: Geographic properties of Internet routing. In: Proceedings of USENIX Annual Technical Conference. pp. 243–259. Monterey, USA (2002)
16. Tangmunarunkit, H., Govindan, R., Shenker, S.: Internet path inflation due to policy routing. In: Proceedings of SPIE International Symposium on Convergence of IT and Communication. pp. 188–195. Denver, USA (2001)
17. Tangmunarunkit, H., Govindan, R., Shenker, S., Estrin, D.: The impact of routing policy on Internet paths. In: Proceedings of IEEE IFOCOM. vol. 2, pp. 736–742. Anchorage, USA (2001)
18. Toren, M.C.: tcptraceroute, `http://michael.toren.net/code/tcptraceroute`
19. Ziviani, A., Fdida, S., de Rezende, J., Duarte, O.: Improving the accuracy of measurement-based geographic location of Internet hosts. Computer Networks and ISDN Systems 47(4), 503–523 (2005)

D

Presupuesto

1)	**Ejecución Material**	
	▪ Compra de ordenador personal (Software incluido)	2.000 €
	▪ Alquiler de impresora láser durante 1 año	520 €
	▪ Material de oficina	150 €
	▪ Total de ejecución material	2.670 €
2)	**Gastos generales**	
	▪ sobre Ejecución Material	352 €
3)	**Beneficio Industrial**	
	▪ sobre Ejecución Material	132 €
4)	**Honorarios Proyecto**	
	▪ 1000 horas a 15 €/ hora	15000 €
5)	**Material fungible**	
	▪ Gastos de impresión	280 €
	▪ Encuadernación	200 €
6)	**Subtotal del presupuesto**	
	▪ Subtotal Presupuesto	21.034 €
7)	**I.V.A. aplicable**	
	▪ 16 % Subtotal Presupuesto	3365,4 €
8)	**Total presupuesto**	
	▪ Total Presupuesto	24.339,4 €

Madrid, OCTUBRE 2011

El Ingeniero Jefe de Proyecto

Fdo.: Roberto González Rey

Ingeniero Superior de Telecomunicación

E

Pliego de condiciones

Pliego de condiciones

Este documento contiene las condiciones legales que guiarán la realización, en este proyecto, de un *Caracterización y cuantificación de los efectos del Path Inflation en una red comercial en producción*. En lo que sigue, se supondrá que el proyecto ha sido encargado por una empresa cliente a una empresa consultora con la finalidad de realizar dicho sistema. Dicha empresa ha debido desarrollar una línea de investigación con objeto de elaborar el proyecto. Esta línea de investigación, junto con el posterior desarrollo de los programas está amparada por las condiciones particulares del siguiente pliego.

Supuesto que la utilización industrial de los métodos recogidos en el presente proyecto ha sido decidida por parte de la empresa cliente o de otras, la obra a realizar se regulará por las siguientes:

Condiciones generales.

1. La modalidad de contratación será el concurso. La adjudicación se hará, por tanto, a la proposición más favorable sin atender exclusivamente al valor económico, dependiendo de las mayores garantías ofrecidas. La empresa que somete el proyecto a concurso se reserva el derecho a declararlo desierto.

2. El montaje y mecanización completa de los equipos que intervengan será realizado totalmente por la empresa licitadora.

3. En la oferta, se hará constar el precio total por el que se compromete a realizar la obra y el tanto por ciento de baja que supone este precio en relación con un importe límite si este se hubiera fijado.

4. La obra se realizará bajo la dirección técnica de un Ingeniero Superior de Telecomunicación, auxiliado por el número de Ingenieros Técnicos y Programadores que se estime preciso para el desarrollo de la misma.

5. Aparte del Ingeniero Director, el contratista tendrá derecho a contratar al resto del personal, pudiendo ceder esta prerrogativa a favor del Ingeniero Director, quien no estará obligado a aceptarla.

6. El contratista tiene derecho a sacar copias a su costa de los planos, pliego de condiciones y presupuestos. El Ingeniero autor del proyecto autorizará con su firma las copias solicitadas por el contratista después de confrontarlas.

7. Se abonará al contratista la obra que realmente ejecute con sujeción al proyecto que sirvió de base para la contratación, a las modificaciones autorizadas por la superioridad o a las órdenes que con arreglo a sus facultades le hayan comunicado por escrito al Ingeniero Director de obras siempre que dicha obra se haya ajustado a los preceptos de los pliegos de condiciones, con arreglo a los cuales, se harán las modificaciones y la valoración de las diversas unidades sin que el importe total pueda exceder de los presupuestos aprobados. Por consiguiente, el número de unidades que se consignan en el proyecto o en el presupuesto, no podrá servirle de fundamento para entablar reclamaciones de ninguna clase, salvo en los casos de rescisión.

8. Tanto en las certificaciones de obras como en la liquidación final, se abonarán los trabajos realizados por el contratista a los precios de ejecución material que figuran en el presupuesto para cada unidad de la obra.

9. Si excepcionalmente se hubiera ejecutado algún trabajo que no se ajustase a las condiciones de la contrata pero que sin embargo es admisible a juicio del Ingeniero Director de obras, se dará conocimiento a la Dirección, proponiendo a la vez la rebaja de precios que el Ingeniero estime justa y si la Dirección resolviera aceptar la obra, quedará el contratista obligado a conformarse con la rebaja acordada.

10. Cuando se juzgue necesario emplear materiales o ejecutar obras que no figuren en el presupuesto de la contrata, se evaluará su importe a los precios asignados a otras obras o materiales análogos si los hubiere y cuando no, se discutirán entre el Ingeniero Director y el contratista, sometiéndolos a la aprobación de la Dirección. Los nuevos precios convenidos por uno u otro procedimiento, se sujetarán siempre al establecido en el punto anterior.

11. Cuando el contratista, con autorización del Ingeniero Director de obras, emplee materiales de calidad más elevada o de mayores dimensiones de lo estipulado en el proyecto, o sustituya una clase de fabricación por otra que tenga asignado mayor precio o ejecute con mayores dimensiones cualquier otra parte de las obras, o en general, introduzca en ellas cualquier modificación que sea beneficiosa a juicio del Ingeniero Director de obras, no tendrá derecho sin embargo, sino a lo que le correspondería si hubiera realizado la obra con estricta sujeción a lo proyectado y contratado.

12. Las cantidades calculadas para obras accesorias, aunque figuren por partida alzada en el presupuesto final (general), no serán abonadas sino a los precios de la contrata, según las condiciones de la misma y los proyectos particulares que para ellas se formen, o en su defecto, por lo que resulte de su medición final.

13. El contratista queda obligado a abonar al Ingeniero autor del proyecto y director de obras así como a los Ingenieros Técnicos, el importe de sus respectivos honorarios facultativos por formación del proyecto, dirección técnica y administración en su caso, con arreglo a las tarifas y honorarios vigentes.

14. Concluida la ejecución de la obra, será reconocida por el Ingeniero Director que a tal efecto designe la empresa.

15. La garantía definitiva será del 4

16. La forma de pago será por certificaciones mensuales de la obra ejecutada, de acuerdo con los precios del presupuesto, deducida la baja si la hubiera.

17. La fecha de comienzo de las obras será a partir de los 15 días naturales del replanteo oficial de las mismas y la definitiva, al año de haber ejecutado la provisional, procediéndose si no existe reclamación alguna, a la reclamación de la fianza.

18. Si el contratista al efectuar el replanteo, observase algún error en el proyecto, deberá comunicarlo en el plazo de quince días al Ingeniero Director de obras, pues transcurrido ese plazo será responsable de la exactitud del proyecto.

19. El contratista está obligado a designar una persona responsable que se entenderá con el Ingeniero Director de obras, o con el delegado que éste designe, para todo relacionado con ella. Al ser el Ingeniero Director de obras el que interpreta el proyecto, el contratista deberá consultarle cualquier duda que surja en su realización.

20. Durante la realización de la obra, se girarán visitas de inspección por personal facultativo de la empresa cliente, para hacer las comprobaciones que se crean oportunas. Es obligación del contratista, la conservación de la obra ya ejecutada hasta la recepción de la misma, por lo que el deterioro parcial o total de ella, aunque sea por agentes atmosféricos u otras causas, deberá ser reparado o reconstruido por su cuenta.

21. El contratista, deberá realizar la obra en el plazo mencionado a partir de la fecha del contrato, incurriendo en multa, por retraso de la ejecución siempre que éste no sea debido a causas de fuerza mayor. A la terminación de la obra, se hará una recepción provisional previo reconocimiento y examen por la dirección técnica, el depositario de efectos, el interventor y el jefe de servicio o un representante, estampando su conformidad el contratista.

22. Hecha la recepción provisional, se certificará al contratista el resto de la obra, reservándose la administración el importe de los gastos de conservación de la misma hasta su recepción definitiva y la fianza durante el tiempo señalado como plazo de garantía. La recepción definitiva se hará en las mismas condiciones que la provisional, extendiéndose el acta correspondiente. El Director Técnico propondrá a la Junta Económica la devolución de la fianza al contratista de acuerdo con las condiciones económicas legales establecidas.

23. Las tarifas para la determinación de honorarios, reguladas por orden de la Presidencia del Gobierno el 19 de Octubre de 1961, se aplicarán sobre el denominado en la actualidad "Presupuesto de Ejecución de Contratаz anteriormente llamado "Presupuesto de Ejecución Material"que hoy designa otro concepto.

Condiciones particulares.

La empresa consultora, que ha desarrollado el presente proyecto, lo entregará a la empresa cliente bajo las condiciones generales ya formuladas, debiendo añadirse las siguientes condiciones particulares:

1. La propiedad intelectual de los procesos descritos y analizados en el presente trabajo, pertenece por entero a la empresa consultora representada por el Ingeniero Director del Proyecto.

2. La empresa consultora se reserva el derecho a la utilización total o parcial de los resultados de la investigación realizada para desarrollar el siguiente proyecto, bien para su publicación o bien para su uso en trabajos o proyectos posteriores, para la misma empresa cliente o para otra.

3. Cualquier tipo de reproducción aparte de las reseñadas en las condiciones generales, bien sea para uso particular de la empresa cliente, o para cualquier otra aplicación, contará con autorización expresa y por escrito del Ingeniero Director del Proyecto, que actuará en representación de la empresa consultora.

4. En la autorización se ha de hacer constar la aplicación a que se destinan sus reproducciones así como su cantidad.

5. En todas las reproducciones se indicará su procedencia, explicitando el nombre del proyecto, nombre del Ingeniero Director y de la empresa consultora.

6. Si el proyecto pasa la etapa de desarrollo, cualquier modificación que se realice sobre él, deberá ser notificada al Ingeniero Director del Proyecto y a criterio de éste, la empresa consultora decidirá aceptar o no la modificación propuesta.

7. Si la modificación se acepta, la empresa consultora se hará responsable al mismo nivel que el proyecto inicial del que resulta el añadirla.

8. Si la modificación no es aceptada, por el contrario, la empresa consultora declinará toda responsabilidad que se derive de la aplicación o influencia de la misma.

9. Si la empresa cliente decide desarrollar industrialmente uno o varios productos en los que resulte parcial o totalmente aplicable el estudio de este proyecto, deberá comunicarlo a la empresa consultora.

10. La empresa consultora no se responsabiliza de los efectos laterales que se puedan producir en el momento en que se utilice la herramienta objeto del presente proyecto para la realización de otras aplicaciones.

11. La empresa consultora tendrá prioridad respecto a otras en la elaboración de los proyectos auxiliares que fuese necesario desarrollar para dicha aplicación industrial, siempre que no haga explícita renuncia a este hecho. En este caso, deberá autorizar expresamente los proyectos presentados por otros.

12. El Ingeniero Director del presente proyecto, será el responsable de la dirección de la aplicación industrial siempre que la empresa consultora lo estime oportuno. En caso contrario, la persona designada deberá contar con la autorización del mismo, quien delegará en él las responsabilidades que ostente.