Tomografía de Internet

Tesis de Grado de Ingeniería Informática

Septiembre 2007

Alumno: Alejandro D. Zylberberg
Tutor: Dr. Ing. José Ignacio Alvarez-Hamelin

Departamento de Computación
Facultad de Ingeniería
Universidad de Buenos Aires

Índice general

1. INTRODUCCIÓN

El área de conocimiento en la cual se enmarca la presente tesis se conoce con el nombre de "Network tomography" o "Tomografía de red". El término fue utilizado por primera vez por Vardi [Vardi96] y a lo largo de los 11 años que han transcurrido desde entonces se han publicado gran cantidad de artículos científicos proponiendo mejoras a la solución original.

En este capítulo se describe la información básica relativa al área de conocimiento. En la primera sección se explica en qué consiste. En la segunda sección se manifiesta por qué es necesaria. En la tercera sección se presenta el modelo matemático utilizado habitualmente. Finalmente, en la cuarta sección se enumeran algunos de los factores que intervienen.

1.1. Conceptos básicos

Comenzaremos por definir a qué llamamos tomografía de una red:

> "La tomografía de una red es el estudio de sus características internas mediante mediciones externas."

¿Qué significa cada uno de los conceptos que aparecen en esa definición?

- Las *características internas* son determinadas propiedades de una red, por ejemplo:
  - la topología
  - la demora en cada enlace
  - el porcentaje de paquetes que se pierden en cada enlace
  - anomalías e intrusiones

- Las *mediciones externas* reciben ese nombre porque se toman siempre entre dos nodos de la red, y nunca se estudia un paquete mientras el mismo está viajando. En inglés se utiliza el término "end-to-end", que podría traducirse como "de punta a punta" o "entre puntas".

- El nombre *tomografía*, en analogía con el conjunto de técnicas aplicadas en el campo de la medicina, se utiliza porque se trata de un proceso de medición no invasiva. Esta no invasividad de las mediciones puede ser entendida de dos maneras:
  - Como se menciona más arriba, las mediciones son entre nodos, y no se introduce nada en

el medio, para, por ejemplo, monitorear directamente un enlace individualmente. Por eso la llamamos "tomografía", en vez de, por ejemplo, "endoscopia".

- Se trata de que el proceso de medición afecte a la red lo menos posible. Esto nos recuerda en cierta forma al principio de Heisenberg, que básicamente enuncia que no se puede medir sin interactuar. Cuanto mayor sea la precisión que se busca, mayor es cantidad de mediciones que deberán ser desplegadas, y entonces los resultados que se obtendrán no serán los correspondientes a la red original, a la cual se quería evaluar, sino los correspondientes a la red que se ha modificado a causa del proceso mismo de medición.

1.2 Motivación

La administración de grandes redes requiere información sobre parámetros tales como topología, conectividad, ruptura de enlaces, comportamiento anómalo, intrusiones, demora de los enlaces, porcentaje de pérdida de los enlaces, etc.

A modo de ejemplo, podría decirse que obtener las demoras en los enlaces que componen la red es interesante por motivos como:

- permite trazar un mapa de demoras, útil en la administración de la red.
- permite mejorar los protocolos de ruteo.

¿Cuál es entonces la utilidad de la tomografía de la red?

Primero es necesario definir "agregación". Por ejemplo, la ruta entre dos nodos A y B atraviesa varios enlaces. El tiempo que tarda un paquete en llegar de A a B es la suma de los tiempos en cada uno de esos enlaces. Tomar la medición agregada (demora entre A y B) es fácil, pero tomar una medición individual de los enlaces atravesados es difícil, porque requeriría poder controlar los puntos unidos por cada enlace.

La respuesta es entonces que todos los parámetros mencionados anteriormente son difíciles o imposibles de medir en forma directa, pero medianamente fáciles de medir en forma indirecta, mediante el cruce de información agregada. De aquí deriva la utilidad de la tomografía de la red, que justamente consiste en el estudio de las propiedades internas (individuales) mediante mediciones externas (agregadas).

Por ejemplo, la ruta entre A y C posiblemente tendrá una parte en común con la ruta entre A y B. Es de esperar que los paquetes que salen de A sigan un determinado camino hasta llegar a un punto en el que se dividen y siguen caminos distintos hacia sus destinos, como se ve en la figura 1.2.1.

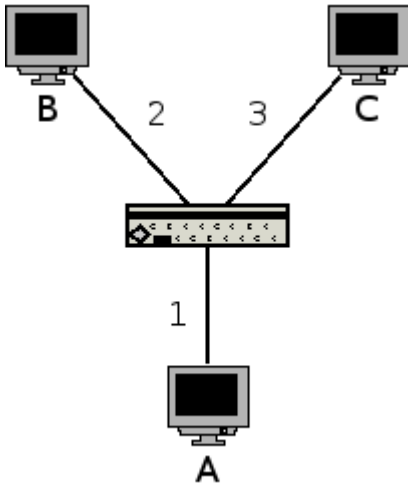


Figura 1.2.1. *Un ejemplo simple*.

La parte que tienen en común (el enlace 1) aparecerá en dos ecuaciones agregadas independientes (1.3.1 y 1.3.2). Esto ilustra la posibilidad de cruzar la información agregada.

1.3. El modelo matemático básico

Continuando con el ejemplo simple de la sección anterior, el objetivo es calcular la demora en los enlaces 1, 2 y 3, a partir de las mediciones de tiempos de las rutas entre A, B y C. Llamando L_1, L_2 y L_3 a las demoras en los enlaces y AB, AC y BC a las demoras en las rutas, se puede plantear:

$$\begin{cases} AB = L_1 + L_2 & (1.3.1) \\ AC = L_1 + L_3 & (1.3.2) \\ BC = L_2 + L_3 & (1.3.3) \end{cases}$$

Empleando notación matricial:

$$\begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} L_1 \\ L_2 \\ L_3 \end{pmatrix} = \begin{pmatrix} AB \\ AC \\ BC \end{pmatrix}$$

De esa forma se pueden apreciar el vector L de las incógnitas (los valores individuales de los enlaces), el vector M de los datos (las mediciones agregadas de las rutas) y una matriz R de ruteo:

$$R \cdot L = M \qquad (1.3.4)$$

La matriz de ruteo indica qué enlaces están incluidos en cada ruta.

En este ejemplo, hay 3 incógnitas y 3 ecuaciones independientes, con lo cual el sistema resulta compatible determinado y las incógnitas se pueden calcular.

Pero en el caso general, R tiene rango deficiente, lo cual es equivalente a decir que hay menos ecuaciones independientes que incógnitas. En ese caso el sistema es indeterminado, y se hace necesario adoptar criterios para llegar a una solución.

La deficiencia del rango es causada por diversos motivos. A continuación se darán algunos ejemplos.

Falta de mediciones

El ejemplo más simple se da cuando no se tienen todas las mediciones posibles. En un grafo no dirigido, como el de la figura 1.2.1, hay $\frac{n\cdot(n-1)}{2}$ mediciones posibles. Si no se contara con todas ellas, la matriz podría resultar deficiente.

Por ejemplo, si no se supiera la demora BC, la matriz quedaría:

$$R=\begin{pmatrix}1 & 1 & 0\\ 1 & 0 & 1\end{pmatrix}$$

En este caso, R tiene 3 columnas y rango 2, presentando una deficiencia de 1.

Enlaces que siempre aparecen juntos

En la figura 1.3.1 se puede apreciar un ejemplo en el cual hay un par de enlaces (1 y 4) que siempre aparecen juntos. Es decir, para cada fila de R, el valor de las columnas 1 y 4 es siempre el mismo. En otras palabras: o aparecen los dos, o no aparece ninguno de los dos. Formalmente:

$$r_{i1}=r_{i4}\quad \forall\, i$$

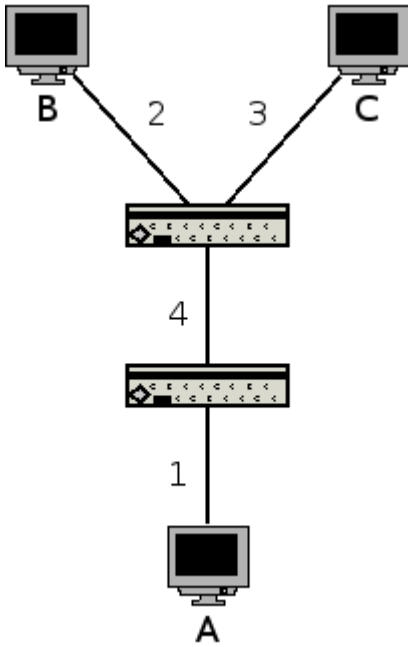


Figura 1.3.1. *Los enlaces 1 y 4 siempre aparecen juntos.*

El sistema queda:

$$\begin{pmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} L_1 \\ L_2 \\ L_3 \\ L_4 \end{pmatrix} = \begin{pmatrix} AB \\ AC \\ BC \end{pmatrix}$$

En este caso, R tiene 4 columnas y rango 3, presentando una deficiencia de 1.
Las incógnitas L_1 y L_4 no se pueden calcular por separado, pero sí se puede calcular su suma:

$$\begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} L_1+L_4 \\ L_2 \\ L_3 \end{pmatrix} = \begin{pmatrix} AB \\ AC \\ BC \end{pmatrix}$$

Los enlaces 1 y 4 se pueden reemplazar por su concatenación, a la que llamaremos 5. El enlace 5 es lo que se denomina un **enlace virtual**, y se ilustra en la figura 1.3.2. El sistema queda:

$$\begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} L_5 \\ L_2 \\ L_3 \end{pmatrix} = \begin{pmatrix} AB \\ AC \\ BC \end{pmatrix}$$

Ahora la matriz de ruteo tiene 3 columnas y rango 3, con lo cual no presenta deficiencia.

Los enlaces virtuales no se utilizan solamente cuando hay enlaces que siempre aparecen juntos. Ese es solamente un caso particular de un problema más general: la *identificabilidad*. Los enlaces virtuales se utilizan cuando hay enlaces no calculables individualmente (o sea, al margen de todos los demás enlaces de la red). A los enlaces no calculables también se los denomina *no identificables*. En otras palabras, un enlace es identificable si y solo si su correspondiente incógnita se puede despejar, es decir, se puede escribir en función de los datos.

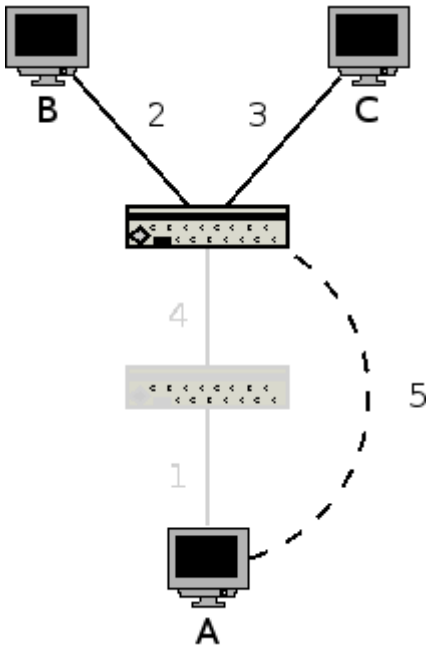


Figura 1.3.2. *Un enlace virtual.*

Grafos dirigidos

Los ejemplos vistos hasta ahora corresponden a grafos no dirigidos. Eso es equivalente a decir que los enlaces son considerados simétricos. Pero si los enlaces tuvieran diferentes propiedades en cada uno de sus sentidos, es necesario modelarlos como asimétricos. Es decir, por cada enlace de los que se vieron hasta ahora, hay 2: el de ida y el de vuelta. En la figura 1.3.3. se puede apreciar el ejemplo original modelado como un grafo dirigido.

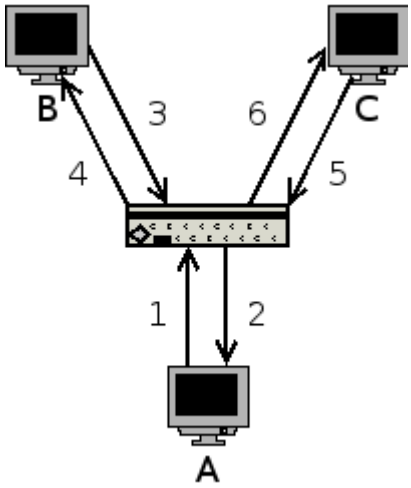


Figura 1.3.3. *Un grafo dirigido.*

El sistema queda:

$$\begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} L_1 \\ L_2 \\ L_3 \\ L_4 \\ L_5 \\ L_6 \end{pmatrix} = \begin{pmatrix} AB \\ BA \\ AC \\ CA \\ BC \\ CB \end{pmatrix}$$

En este caso, R tiene 6 columnas y rango 5, presentando una deficiencia de 1.

En los grafos dirigidos, es un problema bastante común que un determinado enlace no sea identificable. Puntualmente en este caso ninguno de los enlaces es calculable, y los enlaces virtuales quedan como en la figura 1.3.4.

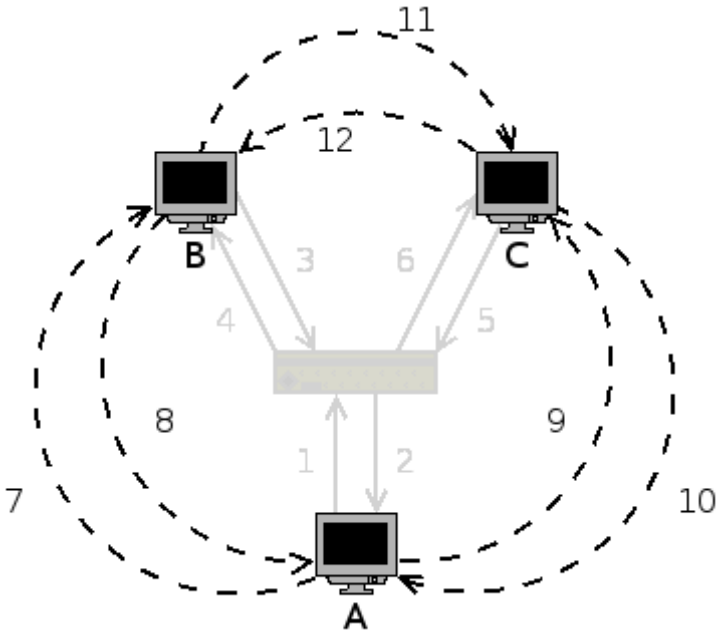


Figura 1.3.4. *Enlaces virtuales en un grafo dirigido.*

1.4. Problemas inherentes

Algunos de los problemas inherentes y decisiones a tomar a la hora de diseñar un método para llevar a cabo la tomografía de una red son:

Escalabilidad

Si una red es muy grande, entonces la cantidad de mediciones que se deben llevar a cabo será muy grande. Por lo tanto, se debe vigilar el orden de la cantidad de mediciones a hacer.

Algo similar ocurre, aunque generalmente en menor medida, con los cálculos necesarios para transformar las mediciones obtenidas en información útil (por ejemplo, transformar los datos de las mediciones externas en información sobre los enlaces internos). Por lo tanto, también se debe vigilar el orden de estos algoritmos.

Precisión

Cuanto mayor sea la precisión que se busca, mayor es la cantidad de mediciones que se deben hacer. Pero esto no es lineal: a veces sacrificando mínimamente a la precisión, se puede hacer un

ahorro enorme en la cantidad de mediciones. E incluso este concepto se puede llevar más allá: con una muy pequeña cantidad de mediciones se puede obtener una medida poco precisa, pero muy rápida.

Distribución de las mediciones

En toda la red, solamente se pueden controlar a unos pocos nodos. Es desde esos pocos nodos que se pueden hacer las medidas. Dados los nodos controlables, debe investigarse qué medidas pueden hacerse desde cada uno de ellos, y luego, mirando la red en su conjunto, debe decidirse qué medidas efectivamente se harán desde cada uno de ellos.

Distribución de los cálculos

Los algoritmos son de elevada complejidad, y hacen un uso extensivo de la memoria. Es necesario decidir si los cálculos se harán en cada nodo o en forma centralizada. Es deseable que los algoritmos responsables de hacer los cálculos sean por lo menos medianamente paralelizables.

Simetría de los enlaces

A la hora de modelizar la red, se debe decidir si los enlaces se consideran simétricos (es decir, que tienen idénticas propiedades en ambos sentidos) o no. Esto trae a colación dos cuestiones:

Por un lado, tratar a los enlaces como simétricos puede parecer evidente, porque, por ejemplo, un cable normalmente tiene las mismas propiedades en un sentido que en otro. Sin embargo, en un determinado momento puede estar fluyendo más información en un sentido que en otro, y entonces la asimetría de la congestión puede causar asimetría en la demora. Consecuentemente, considerar a los enlaces como asimétricos permite desarrollar un modelo más consistente con respecto a la realidad.

Por otro lado, al considerar a los enlaces como asimétricos, se deben usar grafos dirigidos, y tanto la matemática como la diversidad de particularidades que pueden ocurrir y los tiempos de procesamiento se tornan notablemente más complejos.

Aprovechamiento de mediciones existentes

A veces no se puede elegir qué medidas tomar, debido a que las medidas ya fueron tomadas y no se pueden realizar nuevas observaciones. En este caso el problema consiste en aprovechar lo más posible las medidas con las que se cuenta.

Identificación de los enlaces no medibles

Muchos enlaces no pueden ser medidos individualmente, y deben ser agrupados correctamente para posibilitar su medición en conjunto. Juntar en exceso reduce la profundidad del conocimiento pero acelera los procedimientos.

Entonces las cuestiones que deben atenderse son:

- identificar los enlaces que se pueden medir individualmente.
- con respecto al resto, determinar cómo agruparlos para posibilitar su medición en conjunto.
- determinar en qué grado se van a juntar enlaces que de todas maneras son identificables individualmente, con el objeto de acelerar los cálculos (y a costa de perder un poco de granularidad).

Delimitación de la tomografía

La información que se obtiene acerca de la red, siempre es una muy pequeña parte de la que existe realmente. Siempre es posible aprender más acerca de una red. Pero, como en el caso de Internet, el tamaño de la red es tan grande que si tuviéramos la política de aprender todo lo posible entonces el proceso se tornaría en varios sentidos (por ejemplo, computacionalmente) inviable.

Es necesario, entonces, decidir cuál es la información que se desea obtener. Esto no es trivial: significa decidir qué partes de la red se desea investigar, y cuáles son las características a observar en esas partes. En otras palabras, definir un conjunto de políticas que permitan ponerle un límite a los algoritmos.

Tomografía activa o pasiva

Cuando se lleva a cabo una tomografía activa, se hacen pruebas y se analizan los resultados. Esto introduce tráfico adicional en la red, que puede afectarla como se explicó en la sección 1.1.

La alternativa es hacer una tomografía pasiva, en la cual simplemente se analiza la información generada por el funcionamiento habitual de la red, sin introducir tráfico adicional. Esta modalidad es menos invasiva pero se obtienen resultados mucho más lentamente.

Unicast o Multicast

Cuando la tomografía es activa, hay que decidir si implementar un método unicast o multicast.

Un método unicast consiste en enviar cada paquete de prueba a un receptor. La modalidad multicast consiste en enviar copias de cada paquete a varios receptores simultáneamente.

Duración

A lo largo del tiempo, las rutas entre nodos van cambiando. Es necesario que la tomografía sea lo suficientemente rápida como no ser muy afectada por la inestabilidad de las rutas.

Además, y por la misma razón, cabe analizar durante cuánto tiempo se puede esperar que el resultado de una tomografía siga siendo válido. [Zhang00, Zhang01]

2. ESTADO DEL ARTE

En este capítulo se presenta una descripción de la situación actual de la materia, basada en la revisión de la literatura existente hasta el momento (Mayo 2007) y el análisis de los principales proyectos activos actualmente.

En la sección 2.1 se incluyen los trabajos que se ocupan del problema de estudiar las características internas (ver capítulo 1) de una red cualquiera. La mayoría de ellos lleva a cabo experimentos en Internet, pero su principal aporte no es conocimiento sobre Internet en sí, sino sobre las técnicas de estudio de las características internas de una red.

En la sección 2.2 se presentan proyectos actuales de mapeo de Internet, es decir, aquellos que se centran en estudiar sólo la topología (en vez de todas las características internas), y no de una red cualquiera sino de Internet en especial.

En resumen:

Sección	*Trabajos que estudian*	*Red*
2.1. Tomografía de la red	las características internas	cualquiera
2.2. Proyectos de mapeo de Internet	sólo la topología	Internet

En la sección 2.3 se describen brevemente algunos problemas abiertos que surgen del análisis del estado del arte.

2.1. Tomografía de la red

En esta sección se presentan los trabajos más actuales y representativos de varias líneas de investigación activas de la materia.

Selección de las rutas a monitorear

Como se indicó en el capítulo anterior, dados n nodos controlables, la cantidad de rutas entre

ellos es *n(n-1)*, es decir, $O(n^2)$[§], porque se trata de un grafo completo (todos con todos). El resultado *n(n-1)* se deduce fácilmente teniendo en cuenta que cada uno de los *n* nodos se conecta con el resto, que son *n-1*. El grafo completo de 5 nodos se ilustra en la figura 2.1.1, y tiene $5 \cdot 4 = 20$ aristas (en la figura se muestran sólo 10, pero en realidad son 20 porque son bidireccionales). En el peor caso posible, las rutas no comparten enlaces, con lo cual se tienen también $O(n^2)$ incógnitas. Inicialmente, las técnicas de tomografía de la red se basaban en este caso general, es decir, medían todas las $O(n^2)$ rutas.

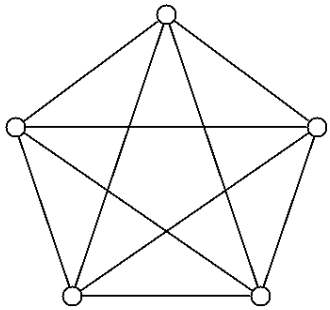


Figura 2.1.1. *Grafo completo de 5 nodos.*

El caso opuesto al de un grafo completo, siempre considerando grafos conexos, es el de un árbol. Un árbol es un grafo conexo y acíclico (no contiene ciclos, o, equivalentemente, hay una sola forma posible de llegar de un determinado nodo a otro). Si la red de *n* nodos controlables fuera un árbol, entonces la cantidad de enlaces sería *n-1*, es decir, *O(n)*. El resultado n-1 se deduce fácilmente por inducción de la siguiente manera: un grafo conexo y acíclico de 2 nodos consiste en los 2 nodos unidos por un enlace (la cantidad de enlaces es *1 = 2-1*); la única forma de agregar un nuevo nodo y que el grafo siga siendo conexo y acíclico es agregando un enlace que una el nuevo nodo al árbol (luego la cantidad de enlaces es *2 = 3-1*); el razonamiento es análogo para agregar un cuarto nodo (quedan *3 = 4-1* enlaces) y así sucesivamente. Un árbol con 5 nodos se ilustra en la figura 2.1.2.

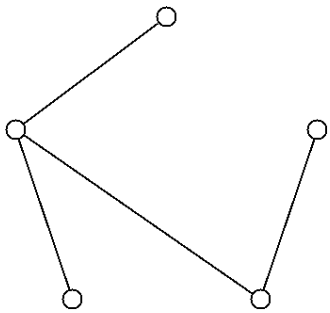


Figura 2.1.2. *Un árbol de 5 nodos.*

§ Esta notación significa que la cantidad de operaciones requeridas para llevar a cabo el procedimiento es a lo sumo kn^2, donde *k* es una constante arbitraria.

Típicamente, muchos enlaces se comparten entre las rutas, y entonces la cantidad real de incógnitas es mucho menor que $O(n^2)$ como en el caso del grafo completo. Chen, Bindel et al. [Chen03, Chen04] proponen medir solamente una cantidad de rutas coincidente con el rango de la matriz de ruteo, es decir, descartar las rutas linealmente dependientes. Los autores conjeturan que de ese modo, la cantidad de rutas a medir es $O(n \log n)$, lo cual es un resultado intermedio entre medir todas las rutas posibles, es decir, $O(n^2)$, y un árbol, que tiene $O(n)$. La verosimilitud de este resultado se pone a prueba mediante análisis de regresión lineal en redes simuladas y reales.

Los autores también desarrollan el algoritmo 2.1.1. El mismo sirve para elegir las rutas a medir, y se basa en técnicas de descomposición reveladoras del rango[¶]. Puntualmente está implementado con una variante de la descomposición *QR* con pivoteo, donde *Q* tiene columnas ortonormales y *R* es triangular superior. En el detalle del algoritmo, la matriz *R* de la descomposición se denota con la letra *A*, para evitar ambigüedad con la notación de la matriz de ruteo.

```
   Entrada:
        R    la matriz de ruteo
   Salida:
        R_S  la matriz con las rutas seleccionadas
1  Para cada fila v de R
2        Â12 = A^T R_S v^T = Q^T v^T
3        Â22 = ||v||^2 − ||Â12||^2
4        Si  Â22 ≠ 0
5             Seleccionar v como ruta a medir
6             A = [ A   Â12 ]
                  [ 0   Â22 ]
7             R_S = [ R_S ]
                    [ v   ]
```

Algoritmo 2.1.1. *Selección de rutas a monitorear.*

En el algoritmo, se analizan las filas de la matriz de ruteo a priori (ciclo de la línea 1), y se van colocando en la matriz de ruteo a posteriori (línea 7) aquellas que son linealmente independientes de las que ya fueron agregadas (líneas 2 a 4). Luego de la ejecución del algoritmo, R_S es una matriz cuyas

¶ Es decir, formas de descomponer una matriz (en este caso factorizarla como el producto Q.R) que permiten inferir en forma directa el rango de la matriz.

filas son un subconjunto de las filas de R y son linealmente independientes. La cantidad de filas de R_S coincide con el rango de R, con lo cual el resultado final podría enunciarse simplemente como: "Se toma la matriz de ruteo y se seleccionan de ella tantas filas linealmente independientes como sea posible".

La complejidad del algoritmo 2.1.1 es $O(rk^2)$, donde r es la cantidad total de rutas y k el rango de la matriz de ruteo R.

Posteriormente, Chua et al. [Chua05] muestran que se puede medir una cantidad de rutas mucho menor, sin sacrificar excesivamente la precisión. Proporcionan un algoritmo para llevarlo a cabo eligiendo una cantidad arbitraria de rutas a medir, y calculan el error en la estimación.

El procedimiento se basa en la descomposición en valores singulares¤ de la matriz de ruteo, que consiste en escribirla como producto de 3 matrices: $R = U\ D\ V^H$, donde U y V son unitarias y tienen columnas ortonormales y D es una matriz diagonal que contiene los valores singulares de R (V^H significa conjugada transpuesta§ de V).

Los valores singulares se ordenan de mayor a menor, conformando así lo que se denomina *espectro* de la matriz de ruteo. La cantidad conveniente de rutas a monitorear se puede obtener del análisis del espectro, en el cual se localizan saltos significativos. Por ejemplo, en la red Abilene de Internet2 [Abilene] (ver figura 2.1.3), los valores singulares ordenados quedan como se ve en la figura 2.1.4, donde se puede apreciar el gran salto luego del segundo valor singular, lo que sugiere que se puede obtener mucha información sobre la red monitoreando solamente 2 rutas.

¤ Los valores singulares de una matriz M son aquellos números reales no negativos s para los cuales existen vectores unitarios u y v tales que $Mv = su$ y $M^H u = sv$.
Los valores singulares guardan estrecha relación con los autovalores, ya que los cuadrados de los valores singulares no nulos de M coinciden con los autovalores no nulos de $M^H M$ ó MM^H.
M^H significa conjugada transpuesta de M (ver nota al pie sobre conjugada transpuesta).

§ Conjugada transpuesta de V: esto significa que se transpone la matriz V y sus elementos se reemplazan por sus conjugados. Es decir: $V^H = \overline{V^T} = \overline{V}^T$ (porque en los espacios de Hilbert no importa el orden en que se realizan las operaciones).
De todas maneras, en lo que respecta a este trabajo, la matriz V siempre está compuesta por coeficientes reales.

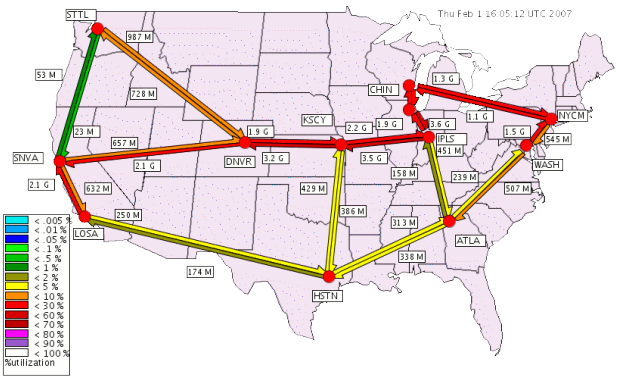


Figura 2.1.3. *La red Abilene.*
Figura tomada de [Abilene].

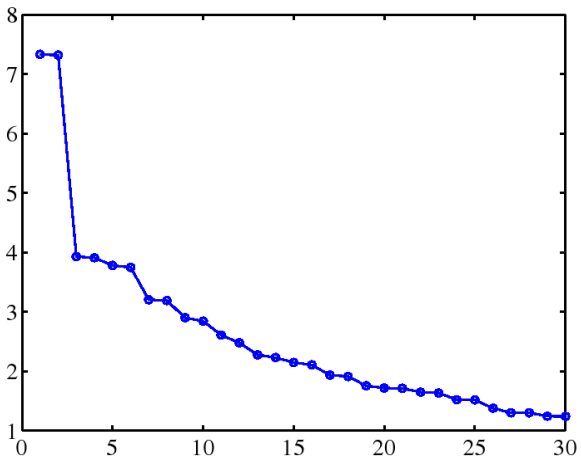


Figura 2.1.4. *Espectro de la red Abilene.*
Figura tomada de [Chua05].

La cantidad de rutas a monitorear se sigue denominando *k* como en los trabajos de Chen et al. La diferencia es que en los trabajos de Chen et al., *k* coincidía con el rango de la matriz de ruteo R, mientras que el trabajo de Chua et al. permite utilizar valores más pequeños de *k* (es decir, monitorear menos rutas).

El valor de *k* a utilizar, es decir, la cantidad de rutas que efectivamente se van a monitorear, se decide normalmente en base a la matriz *D*, que contiene los valores singulares, pero también puede tomarse directamente como parámetro si existen otras restricciones, por ejemplo en el caso de que se decida no medir más de una determinada cantidad de rutas.

Fijado el valor de *k*, hay que elegir las *k* rutas a monitorear. Encontrar el conjunto de las *k* rutas que minimicen en forma exacta el error en la estimación implicaría probar todos los subconjuntos posibles de *k* filas de la matriz *R*, procedimiento que constituye un problema *NP-*

completo[¶]. Debido a esto, los autores utilizan una heurística que se detalla en el algoritmo 2.1.2, y que se basa en intentar aproximar las primeras *k* dimensiones singulares, mediante una matriz que a la vez esté bien condicionada[§]. Las *k* filas seleccionadas se almacenan en la matriz R_S.

La complejidad del algoritmo 2.1.2 es $O(r^2e)$, donde *r* es la cantidad de rutas (es decir, la cantidad de filas de la matriz de ruteo R) y *e* es la cantidad de enlaces (es decir, la cantidad de columnas de la matriz de ruteo R). La complejidad del algoritmo está dominada por la descomposición en valores singulares, ya que la complejidad de la descomposición *QR* con pivoteo es $O(k^2r)$.

```
   Entrada:
        R    la matriz de ruteo
        k    la cantidad de rutas a monitorear (opcional)
   Salida:
        R_S  la matriz con las rutas seleccionadas
1  (U,D,V)  ←  Descomposición de R en valores singulares
2  Si k no se dio como parámetro, decidirlo en base a D
3  U_k  ←  Tomar las k primeras columnas de U
4  U_k^T  ←  Transponer U_k
5  P_k  ←  Obtener matriz de permutación QR de U_k^T
6  P_k^T  ←  Transponer P_k
7  P_kR  ←  P_k · R
8  R_S  ←  Tomar las k primeras filas de P_kR
```

Algoritmo 2.1.2. *Selección de las rutas a monitorear.*

El objetivo es, como siempre, calcular el valor de una determinada métrica (por ejemplo, la demora) en los enlaces, a partir de los valores observados de la métrica en las rutas. En el algoritmo 2.1.3, las variables *x* e *y* representan los valores de la métrica medida en los enlaces y en las rutas respectivamente. Como se detalló en el capítulo 1, las rutas son agregaciones de enlaces (por ejemplo, la demora en una ruta es la suma de las demoras en los enlaces que la componen), y por lo tanto hay una relación lineal entre *x* e *y* dada por lo que se dio en llamar matriz de ruteo, es decir, *R*. La relación lineal es, desde luego, $R\,x = y$.

¶ Los problemas NP (Nondeterministic Polynomial-time) son aquellos en los que la única forma de determinar todas las instancias positivas (es decir aquellas que cumplen alguna propiedad) es verificar todas las instancias posibles. Contrariamente, en los problemas P (deterministic Polynomial-time) se pueden encontrar todas las instancias positivas en tiempo polinomial. Finalmente, un problema NP-completo es aquel que es NP y tal que cualquier problema NP puede reducirse a él [Aaronson].

§ Que la matriz esté bien condicionada garantiza que el sistema lineal resultante podrá resolverse digitalmente con un error numérico razonable.

El algoritmo 2.1.3 sirve para obtener x_m, el vector con las estimaciones de los valores de la métrica en los enlaces a partir de y_S, el vector con los valores observados de la métrica en las rutas monitoreadas. El vector x_m se obtiene resolviendo el sistema lineal $R_S\, x_m = y_S$. Como la matriz R_S no es cuadrada, no es inversible, y entonces se necesita utilizar una pseudo-inversa¤. Los autores proponen usar la pseudo-inversa de Moore-Penrose (que tiene como efecto asignarle un cero a los enlaces que no aparecen en ninguna de las rutas monitoreadas) pero también podrían utilizarse otras.

Una vez obtenidas las estimaciones de los valores de la métrica en los enlaces, se puede predecir el valor de la métrica en las rutas no monitoreadas calculando $y_m = R\, x_m$.

```
   Entrada:
        R    la matriz de ruteo
        R_S  la matriz con las rutas seleccionadas
        y_S  las mediciones correspondientes a las rutas
             seleccionadas
   Salida:
        x_m  las estimaciones de los valores de la métrica en
             los enlaces
        y_m  los valores de la métrica en las rutas, incluyendo
             las estimaciones de las no monitoreadas
1  R_s+  ←  Calcular la pseudo inversa Moore-Penrose de R
2  x_m   ←  R_s+ · y_s
3  y_m   ←  R · x_m
```

Algoritmo 2.1.3. *Estimación de las demoras en los enlaces y predicción para las rutas no monitoreadas.*

La complejidad del algoritmo 2.1.3 está dominada por el cálculo de la pseudo-inversa R_s^+ y es $O(e^2k)$. Por lo tanto, si se aplican en serie los algoritmos 2.1.2 y 2.1.3, la complejidad total es dominada por el primer algoritmo y consecuentemente es $O(r^2e)$.

Continuando con la línea de [Chen03, Chen04], Zhao et al. [Zhao05, Zhao06] desarrollan un método para medir propiedades a nivel enlace con el menor sesgo posible. Definen el concepto de

¤ La inversa de una matriz A es una matriz A^{-1} tal que $A\,A^{-1} = I$. La matriz inversa A^{-1} existe solamente si la matriz A es cuadrada y no tiene deficiencia en el rango (el rango es igual a la cantidad de filas). Esto es lo que que conoce como matriz inversible o "no singular". El sistema lineal $Ax = y$ se puede resolver mediante la inversa de A calculando $x = A^{-1}y$.
Pero cuando la matriz no cumple con las propiedades mencionadas su inversa no existe. En ese caso, el vector incógnita x se puede obtener mediante una versión generalizada de la inversa, llamada pseudo-inversa, calculando $x = A^+y$, donde A^+ es la pseudo-inversa. La pseudo-inversa de Moore-Penrose es la que cumple: $A^+ = (A^TA)^{-1}A^T$.

MILS (minimal identifiable link sequence) consistente en un conjunto consecutivo de enlaces con dos propiedades fundamentales:

- Es identificable: quiere decir que puede ser calculado individualmente.
- Es mínimo: si se le quitara algún enlace dejaría de ser identificable.

¿Por qué utilizar los MILSes reduce el sesgo de la estimación?

Normalmente hay más rutas que enlaces, con lo cual la matriz de ruteo típicamente tiene más filas que columnas, y entonces hay más ecuaciones que incógnitas. Sin embargo, muchas de las ecuaciones son linealmente dependientes, y en general la matriz de ruteo tiene rango deficiente (es decir, que la cantidad de ecuaciones linealmente independientes es menor que la cantidad de incógnitas).

Esto significa que el sistema resulta indeterminado, con lo cual no tiene una única solución. A la hora de hacer el cálculo, se elige una solución en especial, y eso introduce un sesgo [Zhao06] en la estimación.

No se puede hacer una estimación completamente insesgada, porque algunas hipótesis que se utilizan (como la independencia entre las demoras en los enlaces) también sesgan la estimación. Pero esto, al menos en el modelo algebraico, es inevitable.

Sin embargo, el sesgo introducido por el hecho de que el sistema es indeterminado puede ser eliminado transformando el sistema de forma tal que la matriz no presente rango deficiente, es decir, que la cantidad de ecuaciones linealmente independientes sea igual a la cantidad de incógnitas. Y eso es lo que se logra representando las rutas en términos de los MILSes en vez de los enlaces físicos.

El estudio de [Zhao05] y [Zhao06] se centra en grafos no dirigidos, y se proporciona un algoritmo para la determinación de los MILSes. El algoritmo (ver algoritmo 2.1.4) se basa en analizar una por una las rutas de la matriz (ciclo de la línea 1) y verificar exhaustivamente la identificabilidad de cada conjunto consecutivo de enlaces posible (ciclo de la línea 5), de más corto a más largo (ciclo de la línea 4), mediante la comparación de la norma (línea 8) del conjunto con la norma del producto entre el conjunto y una matriz conteniendo una base ortonormal, que debe ser calculada previamente.

El costo de calcular la matriz es $O(rk^2)$, donde r es la cantidad total de rutas y k el rango de la matriz. La complejidad del algoritmo es $O(rkl^3)$, donde l es la cantidad máxima de enlaces en una ruta. Indican que mediante programación dinámica la complejidad del algoritmo se puede bajar a $O(rkl^2)$. Finalmente, manifiestan que la precisión que se pierde buscando los MILSes en las k rutas a monitorear en vez de en las r rutas que hay en la matriz es pequeña, y de esa forma la complejidad es $O(k^2l^2)$.

Como se aclaró más arriba, el estudio se centra en grafos no dirigidos. Al aplicar el algoritmo al caso de los grafos dirigidos, los autores concluyen que en ese caso los MILSes coinciden con las rutas entre los nodos controlables. Es decir, que en un grafo dirigido, los enlaces virtuales identificables mínimos son las rutas completas.

```
   Entrada:
        R    la matriz de ruteo
        Q    la matriz que contiene una base ortonormal
   Salida:
        M    el conjunto de MILSes
 1 Para cada ruta r
 2      Crear un vector booleano m con longitud(r) posiciones
 3      Inicializar en falso todas las posiciones de m
 4      Para i desde 1 hasta longitud(r)
 5           Para cada subruta posible S=r_a..r_b tal que longitud(S)=i
 6                Si m_a es falso
 7                     v es el vector correspondiente a S
 8                     Si ||Q^T v|| = ||v||
 9                          m_a es verdadero
10                          S es un MILS; agregarlo a M
11                     Si no
12                          S no es un MILS
```

Algoritmo 2.1.4. *Búsqueda de todos los MILSes en un grafo no dirigido.*

Puntualmente para el cálculo de las probabilidades de pérdidas en los enlaces, proponen un método que denominan "método de las buenas rutas" y se basa en el hecho de que si una ruta tiene muy poca pérdida entonces la probabilidad de pérdida en los enlaces que la componen es despreciable.

Técnicas que utilizan ICMP¤

Otra gran familia de técnicas es la que hace uso del protocolo *ICMP*. Habitualmente estas técnicas se basan en los mensajes *ICMP timestamp*. El *timestamp* es el mensaje número 13 de *ICMP* y se utiliza para que un nodo pueda averiguar qué hora es según el reloj de otro nodo. Como se verá más adelante, esto es útil a la hora de sincronizar y efectuar mediciones.

¤ ICMP: "Internet control message protocol".
Protocolo de Mensajes de Control para Internet.
El mismo se detalla en la RFC 792 [RFC792].

<table>
<tr><td colspan="6">bits</td></tr>
<tr><td>offset</td><td>0-3</td><td>4-7</td><td>8-15</td><td>16-18</td><td>19-31</td></tr>
<tr><td>0</td><td>versión</td><td>longitud del encabezado</td><td>tipo de servicio (0)</td><td colspan="2">longitud total</td></tr>
<tr><td>32</td><td colspan="3">identificación</td><td>flags</td><td>offset del fragmento</td></tr>
<tr><td>64</td><td colspan="2">TTL (ver sección 2.2)</td><td>protocolo (versión de ICMP)</td><td colspan="2">checksum del encabezado IP</td></tr>
<tr><td>96</td><td colspan="5">dirección de origen</td></tr>
<tr><td>128</td><td colspan="5">dirección de destino</td></tr>
<tr><td>160</td><td colspan="2">tipo (13)</td><td>código (0)</td><td colspan="2">checksum del encabezado</td></tr>
<tr><td>192</td><td colspan="3">identificador</td><td colspan="2">número de secuencia</td></tr>
<tr><td>224</td><td colspan="5">timestamp de origen</td></tr>
</table>

Figura 2.1.5. *Paquete IP (encabezado en gris claro) conteniendo un paquete ICMP (encabezado en gris intermedio) que a su vez lleva un mensaje timestamp (cuyo dato se muestra en gris oscuro).*

El paquete en el cual se envía un mensaje *ICMP timestamp* tiene la estructura que se ve en la figura 2.1.5. En la misma se puede apreciar un paquete *IP* dentro del cual se ubica un paquete *ICMP* que a su vez lleva el mensaje *timestamp* en sí.

La parte indicada en gris claro es el encabezado *IP*.

La parte indicada en gris intermedio es el encabezado *ICMP*. El *tipo* se debe fijar en 13 y el *código* es 0. El identificador y el número de secuencia sirven para poder determinar a qué pedido corresponde cada respuesta.

La parte indicada en gris oscuro es el mensaje *timestamp*. En el mensaje de *pedido*, se envía simplemente el *timestamp* correspondiente al momento en el que se envía el mensaje, expresado en milisegundos transcurridos desde la medianoche del 1ro de enero de 1970.

En la respuesta (ver figura 2.1.6) se debe fijar el *tipo* en 14 y repetir los mismos identificador y número de secuencia. Se envían 3 *timestamps*:

- *timestamp de origen*: el mismo *timestamp* contenido en el pedido. Es decir, el momento en que el emisor envía el pedido, según el reloj del emisor.
- *timestamp de recepción*: el momento en que el receptor recibe el mensaje, según el reloj del receptor.
- *timestamp de respuesta*: el momento en que el receptor envía la respuesta, según el reloj

del receptor.

bits					
offset	0-3	4-7	8-15	16-18	19-31
160	tipo (14)		código (0)	checksum del encabezado	
192	identificador			número de secuencia	
224	timestamp de origen				
256	timestamp de recepción				
288	timestamp de respuesta				

Figura 2.1.6. P*aquete ICMP conteniendo una respuesta timestamp.*

Los trabajos citados hasta ahora en esta sección desestiman el uso de paquetes *ICMP* por considerar que no es confiable depender de que los ruteadores los soporten. Sin embargo, Anagnostakis et al. [Anagnostakis03] presentan evidencia que sugiere que el 94% de los ruteadores soporta *ICMP timestamp*. De acuerdo a sus resultados, el 37% de las rutas está constituido en su totalidad por nodos que soportan *ICMP timestamp*, y el resto tiene nodos intermedios que no lo soportan.

Su metodología se basa en el siguiente hecho: si al trabajar con la ruta *AC* (entre los nodos *A* y *C*) se puede encontrar un ruteador intermedio *B* que soporte *ICMP timestamp*, entonces la subruta *BC* se puede medir como *AC - AB*. El tiempo de espera en cola de los paquetes no constituye un problema porque si se toman varias mediciones entonces el mínimo de esas mediciones tiene una alta probabilidad de corresponder a una medición que no sufrió espera en cola.

Posteriormente, Mahajan et al. [Mahajan03] estudian el problema de un usuario sin privilegios que necesita estudiar el desempeño de la red en una ruta determinada. Se valen del uso de *ICMP timestamp* y de tener en cuenta el *IP-ID* de los ruteadores. Proponen además una serie de mejoras a Internet para hacerla más fácil de diagnosticar.

La primera crítica a estos últimos trabajos es que el soporte de *ICMP* no necesariamente es alto, y de hecho puede bajar a causa de nuevas políticas de seguridad y privacidad. Otra crítica es que están orientados al estudio de una única ruta, y consecuentemente no escalan bien cuando se quiere medir muchas rutas, como en el caso de la tomografía de una red.

Simplificación de la topología y uso de paquetes back-to-back§

Rabbat et al. [Rabbat04] proponen enviar paquetes desde varias fuentes a varios destinos, y descubrir la topología al mismo tiempo que se obtienen las mediciones. Con respecto a la inferencia de la topología, proponen hacer una simplificación y no estudiar todos los nodos intermedios, sino basarse en aquellos en los cuales se juntan los paquetes que van a un mismo destino (*joining points*) o se separan los que salen de un mismo origen (*branching points*).

Explican además que la información sobre toda la red se puede reconstruir a partir del estudio de pequeños problemas con sólo 2 fuentes y 2 destinos. En el problema de 2 fuentes y 2 destinos, la cantidad de nodos intermedios queda reducida a un máximo de 4, como se ve en la figura 2.1.7. En la misma, los nodos 1 y 2 son origen y los nodos 3 y 4 son destino. Se pueden apreciar los *joining points* (J3, J4) de los nodos de destino y los *branching points* (B1, B2) de los nodos de origen. La cantidad de nodos puede ser menor, en el caso de que alguno de los nodos intermedios cumpla más de uno de los roles mencionados.

Se utilizan paquetes *back-to-back*, y se sacan conclusiones en base al orden en que llegan los paquetes a los destinos. Se toma por hipótesis que el orden de los paquetes no sufre alteraciones. Pero el enfoque se ve afectado por el hecho de que a veces los ruteadores alteran el orden de los paquetes que pasan por ellos. La probabilidad de reordenamiento es menor cuanto más separados en el tiempo viajen los paquetes, pero puede ser alta en el caso de que los paquetes viajen *back-to-back*.

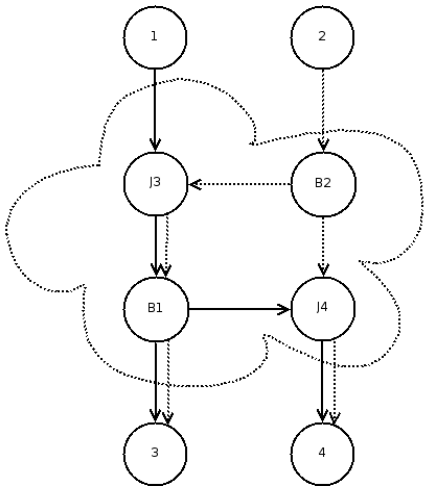


Figura 2.1.7. *Simplificación de la topología intermedia entre 2 nodos origen y 2 nodos destino. Esta figura está basada en la figura 1 de [Rabbat04].*

§ Los paquetes son *back-to-back* (traducción: “espalda con espalda”) cuando se envían uno a continuación del otro (con la mínima demora intermedia posible) de modo tal que haya alta probabilidad de que ambos experimenten los mismos fenómenos, como por ejemplo esperas en colas.

2.2. Proyectos de mapeo de Internet

En esta sección se describen proyectos que se centran en el estudio de la topología de Internet. Como comparación con la sección anterior, cabe destacar que la sección anterior se ocupa de los trabajos orientados a estudiar las características internas de una red cualquiera.

En relación al tema del trazado del mapa de Internet, es necesario mencionar al proyecto NetDimes [Dimes]. Este proyecto está dedicado al estudio de la estructura y la topología de Internet mediante mediciones que se hacen en forma distribuida.

Las herramientas básicas empleadas en las mediciones son *traceroute*¤ y *ping*¶. Ambas permiten medir las demoras, pero *traceroute* consume muchos más recursos en la red porque la cantidad de mensajes que requiere es cuadrática con la longitud de la ruta, mientras que la cantidad de mensajes requeridos por *ping* es lineal.

Por esa razón se utiliza *ping* para medir las demoras, y *traceroute* (en baja frecuencia) para descubrir las rutas [Shavitt04].

Los traceroute se hacen en baja frecuencia principalmente por dos motivos. El primero es evitar la carga excesiva de la red. El segundo es intentar evitar disparar potenciales alarmas de seguridad que pueden activarse ante cantidades significativas de determinados tipos de pedidos.

Los datos son aportados por los clientes instalados en las computadoras de más de 8000 voluntarios ubicados en casi todos los países del mundo. Actualmente [Carmi06] esos datos se están usando para estudiar, entre otras cosas, la estructura de los sistemas autónomos.

Uno de los problemas a la hora de descubrir la topología trazando rutas es que los ruteadores responden con diferentes direcciones *IP* en sus diferentes interfaces. Por ejemplo, en la figura 2.2.1, se ve el caso de un ruteador que tiene asignada la *IP 2* en la interfaz conectada a la red que lo une al nodo *A*, y la *IP 3* en la interfaz conectada a la red que lo une al nodo *B*.

¤ traceroute: aplicación que sirve para descubrir la ruta que sigue un paquete entre dos nodos de una red *IP*. Se basa en un parámetro denominado *TTL* (“*time to live*”, “tiempo a vivir”) el cual es decrementado por cada nodo por el que pasa el paquete. El nodo en el cual el *TTL* se agota lo elimina y envía un mensaje al origen (*ICMP timeexceed*). De esta manera, y enviando paquetes con distintos *TTL*, el origen puede ir recibiendo respuestas del primer nodo de la ruta, del segundo, etc. y así descubrir la ruta. Por cada valor del *TTL* se hacen habitualmente 3 pruebas.

¶ ping: aplicación que sirve para determinar si existe una ruta hasta un determinado nodo, y en caso afirmativo obtener el *RTT* (“*roundtrip time*”, “tiempo de ida y vuelta”) hasta y desde el nodo en cuestión.
Se implementa en base a los mensajes *ICMP* tipos 8 (*echo request*) y 0 (*echo reply*). Ver [RFC792].

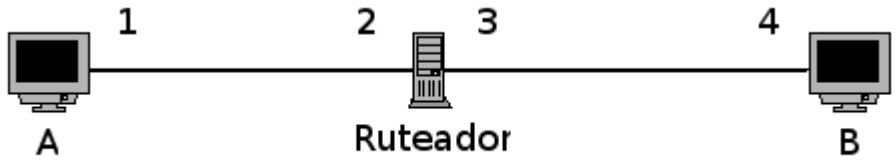


Figura 2.2.1. Ruteador con distintas direcciones IP en sus distintas interfaces.

Cuando se traza la ruta de A a B, se obtiene 1-2-4 y cuando se traza la ruta de B a A, se obtiene 4-3-1. Esto puede causar que la red se modele erróneamente como se ve en la figura 2.2.2.

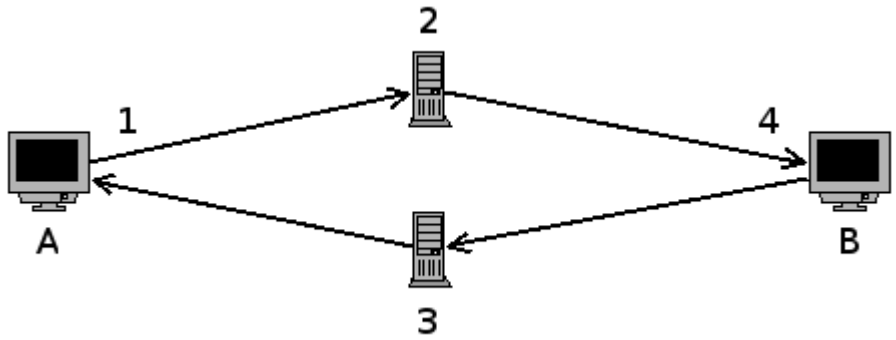


Figura 2.2.2. *Un único ruteador modelado erróneamente como 2 nodos distintos.*

Una posible solución a este problema se plantea en [Govindan00] y consiste en utilizar paquetes *UDP*§ enviados a la dirección *IP* del ruteador para puertos que no existen y comparar la dirección a la que fue enviado el paquete *UDP* con la dirección de origen de la correspondiente respuesta *ICMP destination unreachable*¤.

En el ejemplo de la figura 2.2.2, el nodo A puede enviar el paquete *UDP* al ruteador a la dirección *IP* 2, y cuando el ruteador conteste, lo hará poniendo como origen la dirección *IP* 3, que es la que corresponde a los paquetes que se envían hacia A. De este modo, el nodo A puede saber que las direcciones *IP* 2 y 3 corresponden al mismo ruteador.

En este trabajo también se emplearán datos del proyecto CAIDA [Caida], que promueve la cooperación de sectores comerciales, gubernamentales y educativos para el estudio de las

§ *UDP*: "User datagram protocol"
Protocolo de datagrama de usuario.
Su principal diferencia con *TCP* es que no es orientado a la conexión.

¤ Mensaje *ICMP* número 3, que se usa para indicar que no existe ruta hacia un nodo, o bien, como en este caso, que el puerto no está abierto. [RFC792]

características de Internet con el objetivo de mejorar su funcionamiento y escalabilidad.

Los datos se CAIDA que se emplean en este trabajo se obtuvieron mediante una de sus herramientas, denominada *skitter*, que tiene 3 objetivos fundamentales:

- Estudiar salto por salto las rutas desde el origen hacia una gran cantidad de destinos. Esto se explica en mayor nivel de detalle más adelante.
- Detectar cambios en el ruteo. Esto se hace midiendo el RTT hacia los destinos, porque un cambio muy grande en el RTT puede indicar, por ejemplo, que el camino de vuelta ya no es el mismo. Se intenta descubrir qué cambios en el ruteo son habituales, por ejemplo investigando la correlación entre los cambios y los momentos del día.
- Estudiar la conectividad a Internet desde un determinado origen. Esto lleva al trazado de mapas como el que se ve en la figura 2.2.3.

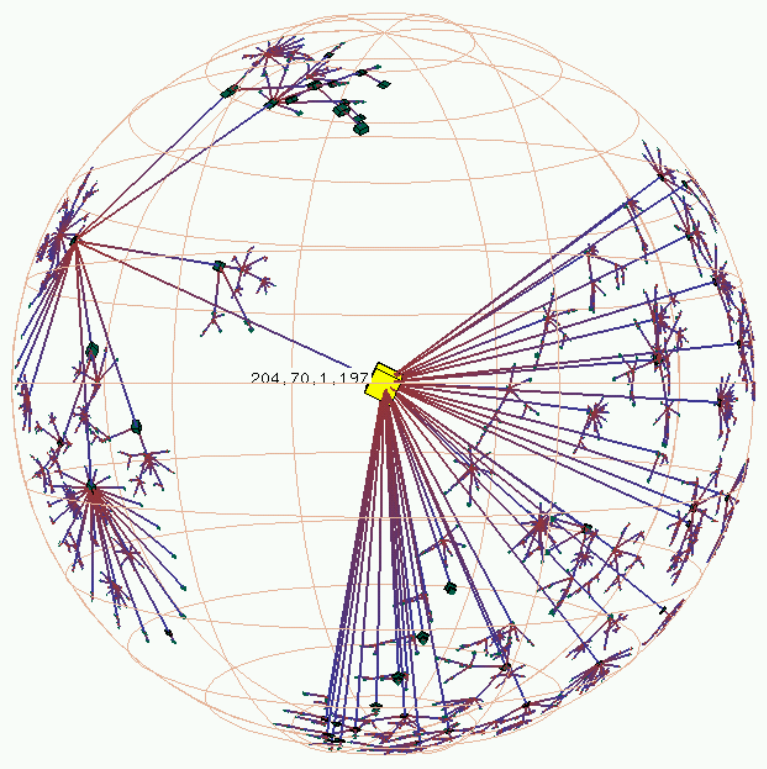


Figura 2.2.3. *Mapa trazado con datos obtenidos mediante skitter. Se puede visualizar la conectividad desde 204.70.1.197. Figura tomada de [Caida].*

La aplicación *skitter* determina las rutas unidireccionales desde su lugar de instalación hacia nodos remotos de la red. Muchos de los ruteadores intermedios descartan paquetes sin enviar el *ICMP timeexceed*. Por eso el criterio que adopta skitter es considerar como el *n*-ésimo nodo de la ruta al ruteador que conteste a causa de los paquetes enviados con *time-to-live* igual a *n*.

Otro de los problemas a los que se enfrenta una herramienta como *skitter* es el conflicto entre dos de sus intereses:

- Por un lado, el tiempo en total que se tarda en estudiar por completo cada ruta no puede ser excesivamente largo, porque de lo contrario el resultado no sería la "fotografía instantánea" que se pretende obtener.
- Por otro lado, para ser verdaderamente útil como herramienta de mapeo de Internet, se necesita poder estudiar muchas rutas en poco tiempo.

En la figura 2.2.4 se ve el formato de los paquetes utilizados por *skitter*.

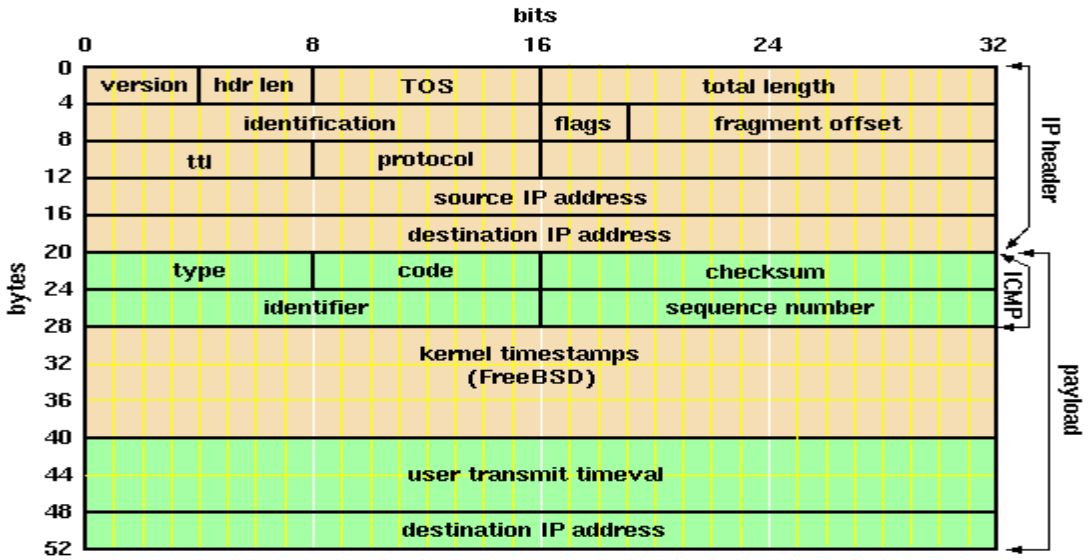


Figura 2.2.4. *Formato de los paquetes enviados por skitter. Figura tomada de [Caida].*

2.3. Problemas abiertos

En las primeras dos secciones de este capítulo se describió la situación actual de la materia.

Se describieron las soluciones existentes hasta el momento para problemas como el descubrimiento de la topología, la minimización del sesgo en la estimación y la determinación de los enlaces medibles.

Se indicó asimismo la complejidad de los algoritmos utilizados, debido a que las técnicas a emplear deben ser escalables. Es decir, deben ser realizables y razonables desde el punto de vista computacional.

De lo expuesto surge un problema que es sumamente interesante, porque su solución implicaría un gran avance en la materia. El problema en cuestión es intentar alcanzar simultáneamente

los siguientes objetivos:

- *Descubrir la topología*. Inicialmente, la topología de la red es desconocida. Sólo se conoce un conjunto de nodos controlables. La red debe ser explorada.
  Este problema se estudia en la sección 3.1.

- *Calcular todo lo posible*. Es decir, aprovechar al máximo los nodos controlables con el objetivo de aprender lo más posible sobre la red a la cual pertenecen.
  Este problema se estudia en la sección 3.1.

- *Minimizar el sesgo*. Llevar a cabo la estimación introduciendo el menor sesgo posible. Como se indicó más arriba, este problema está relacionado con la posibilidad de escribir las rutas no en términos de los enlaces físicos que las componen sino en términos de agregaciones mínimas e identificables de dichos enlaces.
  Este problema se estudia en la sección 3.2.

- *Mantener la escalabilidad*. Los métodos deben ser aplicables a redes grandes, dado que sería de gran utilidad poder aplicarlos a Internet.
  A continuación de cada algoritmo presentado en el capítulo 3 se indica el análisis de su complejidad.

3. INNOVACIONES

Este trabajo contiene dos aportes en el área de la tomografía de Internet.

La primera contribución se relaciona con la determinación de las rutas medibles y se explica en la sección 3.1.

La segunda contribución concierne a la identificabilidad de los enlaces y su aplicación a revolucionar el orden de los algoritmos de minimización del sesgo. Se desarrolla en la sección 3.2.

3.1. Determinación de las rutas medibles

En esta sección se explica la nueva metodología para determinar qué rutas se pueden medir.

La primera etapa consiste en explorar la red (porque se parte de una topología desconocida) y luego clasificar los nodos según el rol que pueden desempeñar en las futuras mediciones. Esta etapa se explica en la sección 3.1.1.

La segunda etapa consiste en determinar las rutas medibles, expresarlas en función de los enlaces físicos, y finalmente crear la matriz de ruteo. Esta etapa se explica en la sección 3.1.2.

3.1.1. Clasificación de nodos

Todo problema de tomografía activa de una red comienza con un conjunto de nodos controlables. El término "controlable" se refiere a que a dichos nodos se les puede instalar software que lleve a cabo mediciones. De ahora en adelante, estos nodos serán denominados **alfa** (α).

Para obtener mayor nivel de detalle y aproximación a la realidad, se utilizará un modelo de enlaces y rutas asimétricos, lo cual permite estudiar el caso general de que los enlaces y rutas tengan distintas propiedades en cada uno de los sentidos. De este modo, el modelo planteado no se verá afectado por el volumen tráfico en un momento determinado ni por las restricciones que a él sean impuestas.

Hecha la aclaración del párrafo anterior, en este trabajo se dice que un par de rutas son "simétricas" cuando tienen los mismos extremos, sentidos opuestos, y atraviesan exactamente los mismos enlaces. Nótese que esto no constituye una contradicción con el párrafo anterior, ya que si bien dos rutas pueden ser físicamente simétricas (atraviesan los mismos enlaces) no necesariamente tienen las mismas propiedades (por ejemplo, no necesariamente se tarda lo mismo en un sentido que en el otro).

Como de cada alfa se pueden enviar paquetes a los demás, si hay *n* alfas entonces la cantidad de rutas que se pueden medir es *n(n-1)*. Dicha cantidad constituye un resultado muy conocido, y es la que se emplea en las más modernas técnicas con las que se cuenta hasta el momento (ver capítulo 2, sección 1).

Si los nodos se la red se pueden sincronizar, se puede obtener mucha más información sobre la red. A continuación se describe la sincronizabilidad y luego se muestran las técnicas que aprovechan la sincronización para obtener información adicional.

Hay básicamente 2 formas de sincronizar un par de alfas. Una es mediante el uso de un soporte externo, como la tecnología GPS. La misma es costosa y difícil de obtener, instalar y/o esperar que los nodos la tengan disponible. Debido a esos problemas, dicha técnica no se emplea en este trabajo, sino que se utiliza una mucho más asequible. De todas maneras, en el improbable caso de que se contara con esa ventaja, puede ser aprovechada.

En este trabajo, la sincronización entre los nodos se lleva a cabo mediante el uso de la técnica de las rutas simétricas: si dos alfas tienen una ruta simétrica, entonces son sincronizables. El método en sí que se usa para sincronizar a los alfas se describe más adelante.

Por simplicidad de la explicación, se asumirá en adelante que, dado el conjunto de alfas con el que se cuenta, existe un árbol que los conecta mediante rutas simétricas. Naturalmente, esto puede no ocurrir, pero en ese caso el problema se soluciona fácilmente separando el conjunto en partes conexas y aplicando la metodología a cada árbol del bosque (aunque eventualmente se pierden algunas mediciones entre los árboles del bosque).

Luego, todos los alfas se pueden sincronizar.

¿Existen otros nodos, además de los alfas, que puedan ser sincronizados? La respuesta es sí.

Los nodos no controlables se pueden sincronizar cuando cumplen con las siguientes dos propiedades:

1) soporte *timestamp*

2) alfa-simetría

La propiedad 1 consiste en soportar *timestamps ICMP* (ver capítulo 2, sección 2). Los trabajos [Zhao05] y [Zhao06] no aprovechan el soporte de *ICMP* por parte de los nodos para no depender de características de la infraestructura interna de la red. Sin embargo en el capítulo 4 se muestra que aún con un nivel bajo de soporte de *ICMP timestamp* los resultados son muchos mejores que si no se tuviera en cuenta a los *timestamps*.

La propiedad 2 es nueva y se define a continuación:
Un nodo es **alfa-simétrico** cuando tiene una ruta simétrica con por lo menos un alfa.

En resumen, los nodos que no son controlables pero son sincronizables porque soportan *timestamps* y son alfa-simétricos, serán denominados **beta** (**β**).

¿Cómo se sincroniza un beta a un alfa?
Para empezar, debe aclararse que sincronizar a un beta no significa que el reloj del beta coincida con el del alfa (lo cual es imposible porque el beta no es controlable) sino conocer la diferencia entre el reloj del beta y el del alfa, de modo tal de poder corregir todos los datos que en el futuro proporcione ese beta.
Lo primero que se hace es medir el t_{RT} (comúnmente conocido como *RTT*, *roundtrip time*, ver capítulo 2, sección 1) entre el beta en cuestión y algún alfa con el cual tenga ruta simétrica (como es beta, tiene que ser alfa-simétrico). Esto debe ser hecho en un momento en el cual la red esté desocupada o bien se puede asumir esa condición tomando el mínimo de un gran número de mediciones. El tiempo entre el alfa y el beta es la mitad del t_{RT}.
Se le envía al beta un pedido de *timestamp* (ver capítulo 2, sección1) en el instante t_0. La respuesta se recibe en el instante t_{REC}, con lo cual el *roundtrip time* se calcula como:

$$t_{RT}=t_{REC}-t_0 \tag{3.1.1}$$

Si el *timestamp* contenido en la respuesta acusa el tiempo t_{TIM} entonces la diferencia entre el reloj del beta y el reloj del alfa se puede calcular como:

$$t_{DIF}=t_{REC}-t_{TIM}-\frac{1}{2}t_{RT} \tag{3.1.2}$$

porque t_{TIM} es la hora que era según el beta, medio *roundtrip* antes de recibir la respuesta.
Por ejemplo, si la respuesta se recibe en el instante 100, el *timestamp* es 80 y la estimación del *RTT* es 50, entonces queda:

$$t_{DIF}=100-80-\frac{1}{2}50=-5$$

Ese resultado significa que el beta está adelantado en 5 unidades de tiempo con respecto al alfa.

La figura 3.1.1 ilustra el procedimiento descripto.

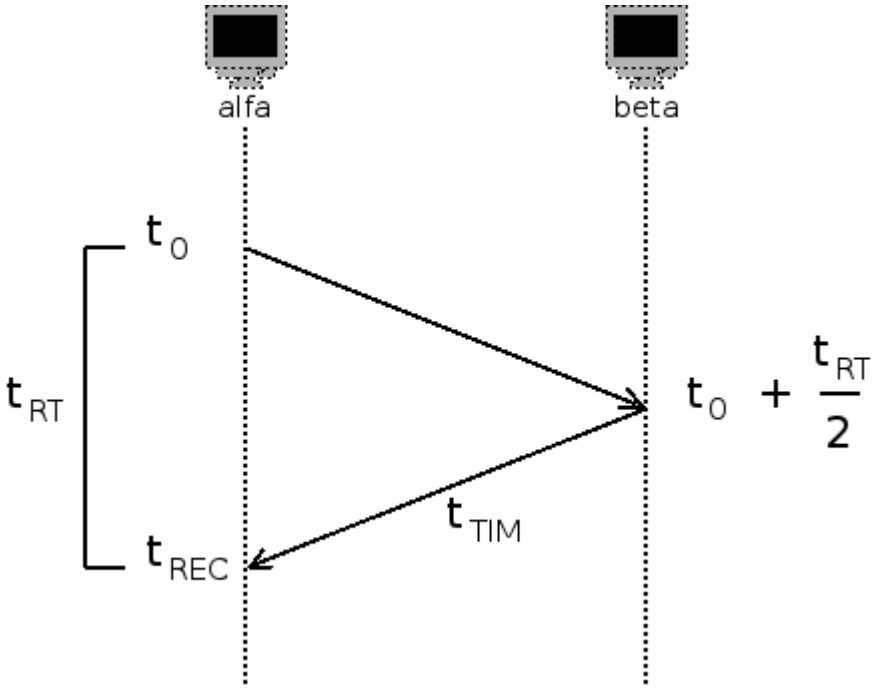


Figura 3.1.1. *Sincronización de un beta a un alfa.*

¿Cómo se sincroniza un par de alfas?

Se puede seguir el mismo procedimiento que se usa para sincronizar un alfa y un beta. De esta manera, un alfa α_1 le envía el paquete *timestamp* al otro alfa α_2 y calcula la diferencia entre los relojes.

Para que α_2 conozca la diferencia entre los relojes, se puede hacer que α_1 se la informe, o bien que α_1 envíe otro timestamp luego de la respuesta de α_2 para que α_2 pueda calcular la diferencia.

Cabe destacar que en los nodos alfa, como son controlables, se puede implementar el soporte para *ICMP timestamp*, a diferencia de los nodos beta, para los cuales se depende de que el soporte de *ICMP timestamp* exista de antemano.

¿Cuáles son las rutas medibles?

Los alfas y los betas son los nodos útiles, porque todas las rutas medibles los tienen en sus extremos. Las rutas medibles siempre son entre dos alfas o entre un alfa y un beta. Se utilizará el término **alfabeto** para referirse a un nodo que puede ser extremo de una ruta, independientemente de si es alfa o beta.

¿Cómo se miden las demoras en una ruta entre un alfa y un beta?

En condiciones normales de operación, las rutas no son simétricas en cuanto a sus retardos, incluso aunque físicamente lo sean. Pero utilizando el valor de t_{DIF} calculado al momento de la sincronización, se pueden calcular las demoras en ambos sentidos. La forma de proceder es la

siguiente:

1) El alfa envía al beta un pedido de *timestamp* en el instante t_1

2) El pedido pasa por el beta en el instante t_2

3) El alfa recibe la respuesta en el instante t_3.

Luego, los tiempos en ambos sentidos son:

$$t_{\alpha\beta}=t_2+t_1 \quad (3.1.3)$$

$$t_{\beta\alpha}=t_3+t_2 \quad (3.1.4)$$

El problema es que t_2 es desconocido para el sistema, que solamente controla al alfa. Pero esto no es un problema grave, ya que la hora indicada en el *timestamp*, t_{IND}, no es otra cosa que t_2 según el reloj del beta. Y como se conoce t_{DIF} (la diferencia entre los relojes) se puede calcular t_2:

$$t_2=t_{IND}+t_{DIF} \quad (3.1.5)$$

La figura 3.1.2 ilustra este procedimiento.

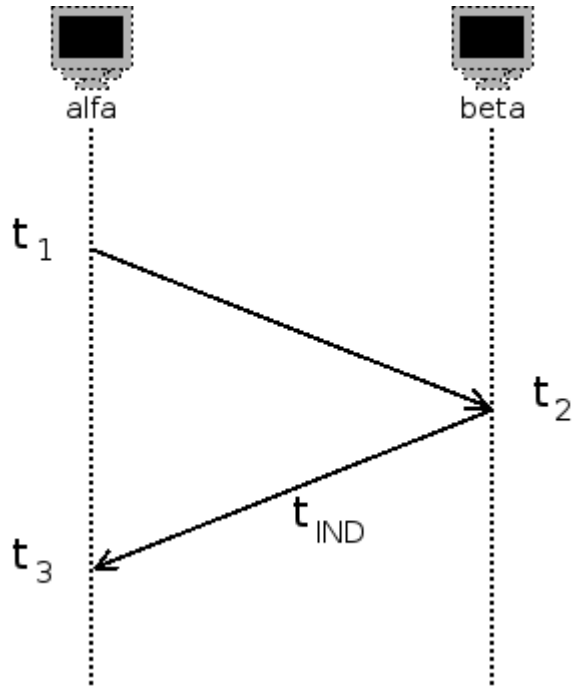


Figura 3.1.2. *Medición en ambos sentidos de la demora entre un alfa y un beta.*

¿Cómo se miden las demoras en una ruta entre dos alfas?

Se sigue el mismo procedimiento que se emplea para medir las demoras en una ruta entre un alfa y un beta.

Volviendo al comienzo del problema, en el principio se conocen únicamente los alfas de la red. Cuando se trazan las rutas entre todo par posible de alfas, se descubren muchos nodos nuevos.

Naturalmente, esos nuevos nodos no son controlables (no son alfas) y todavía no se sabe si son alfa-simétricos y/o soportan *timestamps* (con lo cual todavía no se sabe si pueden ser betas). Solamente se tiene idea de su existencia a partir de la observación de la salida del comando traceroute.

Se denominará **delta** (δ) a los nodos no controlables cuya alfa-simetría y soporte de *timestamps* todavía no han sido investigados. Todos los nuevos nodos que se van descubriendo (es decir, los que se descubren cada vez que se traza una ruta) son inicialmente deltas.

Cuando un delta es investigado y consecuentemente se descubre que es alfa-simétrico, se para a clasificar como **gama** (γ).

Los deltas que son adyacentes (o sea que están conectados directamente) a algún alfa son obviamente alfa-simétricos, porque un par de rutas compuestas de un único enlace atravesado en sus dos sentidos son simétricas (no necesariamente en cuanto a sus propiedades, pero sí físicamente).

Los deltas que no son adyacentes a ningún alfa también pueden ser alfa-simétricos si se les descubre una ruta simétrica con por lo menos un alfa, y esto a veces requiere trazar rutas adicionales.

Por ejemplo, considérese la siguiente ruta va desde el alfa α_1 hasta el alfa α_5:

$$\alpha_1 \rightarrow \delta_2 \rightarrow \delta_3 \rightarrow \delta_4 \rightarrow \alpha_5$$

Los nodos 2 y 4 en realidad gamas porque al estar conectados directamente a alfas son alfa-simétricos. La ruta queda:

$$\alpha_1 \rightarrow \gamma_2 \rightarrow \delta_3 \rightarrow \gamma_4 \rightarrow \alpha_5$$

¿Qué sucede con el nodo 3? Si la ruta del 5 al 1 fuera simétrica con la dada en el ejemplo, entonces el nodo 3 es alfa-simétrico con los alfas 1 y 5. Si no, entonces aún se puede observar que la ruta del nodo 3 al nodo 5 es conocida (porque se descubrió al trazar la ruta del nodo 1 al nodo 5) y se puede hacer un traceroute desde el nodo 5 hasta el nodo 3, para determinar si este último es alfa-simétrico.

Al trazar rutas adicionales, potencialmente se descubren nuevos deltas. El algoritmo se podría aplicar recursivamente a los nuevos deltas, pero dado el tamaño de Internet, es necesario darle un corte al algoritmo. Por eso en este trabajo se usa el criterio de no aplicar el algoritmo a los deltas descubiertos en esta etapa. Este criterio puede cambiarse, tanto en forma estática (es decir, crear un algoritmo con un criterio de corte distinto) como en forma dinámica (aplicar este criterio en la primera pasada del algoritmo, pero luego en pasadas posteriores seguir explorando los nuevos deltas).

Cuando un gama es investigado y consecuentemente se determina que soporta *timestamps*, entonces se lo clasifica como beta, como se describió antes.

Cuando se determina que un nodo no es alfa-simétrico y/o no soporta *timestamps*, entonces no se lo puede sincronizar, y se lo clasifica como **épsilon** (ϵ).

En la tabla 3.1.1 se resume el criterio de clasificación para los nodos no controlables.

		¿El nodo soporta *timestamps*?		
		sí	no se sabe	no
¿El nodo tiene ruta simétrica con por lo menos un alfa?	sí	β	γ	ϵ
	no se sabe	(*)	δ	ϵ
	no	ϵ	ϵ	ϵ

Tabla 3.1.1. *Clasificación de los nodos no controlables.*

(*) este caso no es posible. Nunca se verifica el soporte de timestamps de un nodo cuya alfa-simetría no ha sido determinada aún.

En la tabla 3.1.2 se indican las propiedades de todos los tipos de nodos, resumiendo lo dicho hasta el momento.

Alfa	α	Los nodos controlables.	Estos son los alfabetos. Son los útiles, porque todas las rutas medibles son entre dos alfas o entre un alfa y un beta.
Beta	β	Los nodos que no se pueden controlar pero al menos se pueden sincronizar, porque soportan *timestamps* y son alfa-simétricos.	
Gamma	γ	Los nodos que son alfa-simétricos. Aún no pueden ser considerados betas porque todavía no se relevó si soportan *timestamps*.	Estos nodos están a la espera de que su sincronizabilidad sea relevada.
Delta	δ	Los nodos que aún no fueron relevados.	
Épsilon	ϵ	Los nodos que no son sincronizables, porque no soportan *timestamps* y/o no son alfa-simétricos.	Estos nodos no son sincronizables.

Tabla 3.1.2. *Clasificación de los nodos.*

El ciclo de vida de los nodos se ilustra en la figura 3.1.3. En la misma se puede apreciar cómo algunos deltas pasan a ser clasificados como gamas y otros como épsilons, y luego algunos de los gamas pasan a ser betas y otros épsilons.

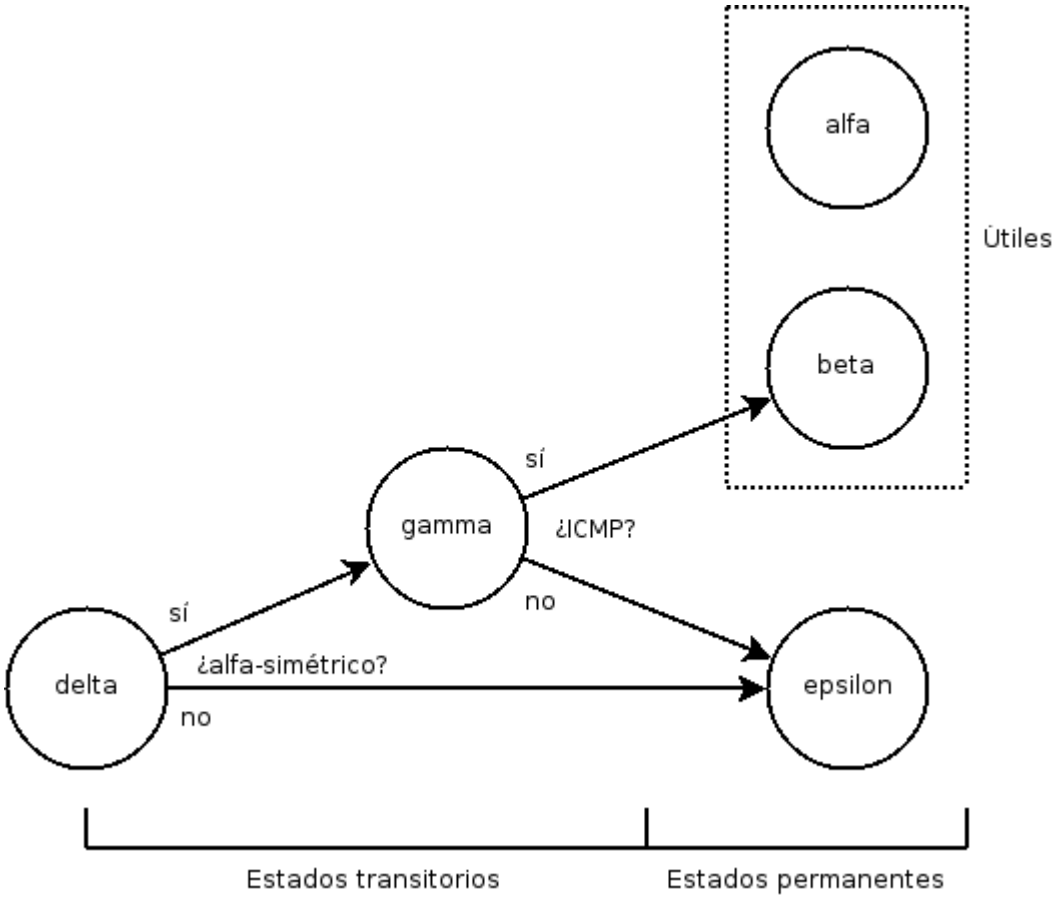


Figura 3.1.3. *Ciclo de vida de los nodos.*

El algoritmo 3.1.1 describe el proceso de clasificación de nodos. La complejidad del algoritmo es $O(n^2D)$, donde n es la cantidad de alfas y D es el diámetro de la red[§]. Esta cantidad es dominada por los ciclos anidados de las líneas 4 y 8. El ciclo de la línea 8 consta de n iteraciones, mientras que el de la línea 4 tiene tantas iteraciones como la cantidad de deltas, y como los deltas existentes hasta ese momento son los nodos descubiertos al trazar las rutas entre los alfas, la cantidad es $O(nD)$. Los ciclos de las líneas 1 y 15 tienen complejidad $O(n^2)$.

Las complejidades enunciadas se refieren a una primera ejecución del algoritmo. Los cambios en Internet son muy lentos en comparación con el total de la red. Por lo tanto, posteriores pasadas del algoritmo, destinadas a detectar actualizaciones, necesitan realizar una cantidad de operaciones mucho menor.

§ El diámetro de la red es la máxima distancia entre dos nodos de la red.

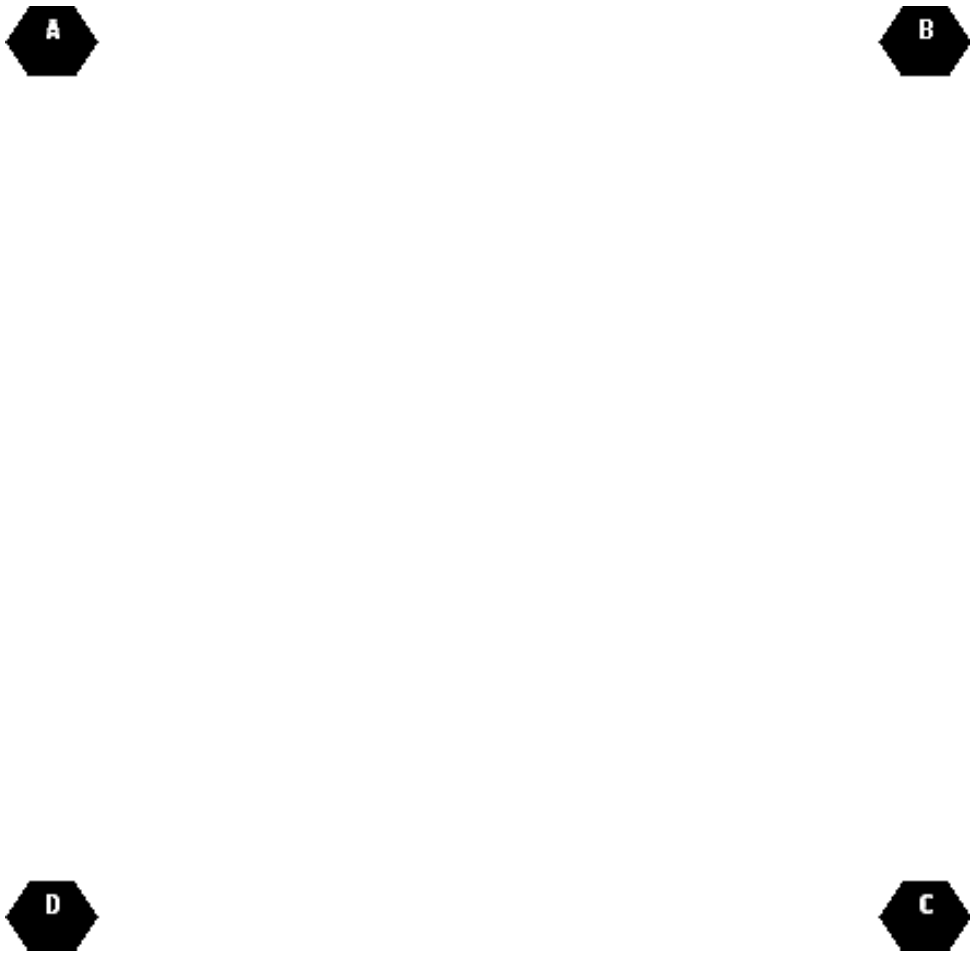
```
   Entrada:
        Lista de alfas
   Salida:
        Lista de betas
        Lista de gamas
        Lista de deltas
        Lista de épsilons
        Lista de rutas
 1 Para cada par posible de alfas
 2      Trazar rutas
 3      Recordar los deltas descubiertos
 4 Para cada delta d
 5      Si es adyacente a un alfa
 6           Convertir a d en gama
 7      Si no
 8           Para cada alfa a
 9                Si se conoce la ruta de d a a
10                     Trazar la ruta de a a d
11                     Si las rutas son simétricas
12                          Convertir a d en gama
13                     Si no
14                          Convertir a d en épsilon
15 Para cada gama
16      Enviar paquete timestamp
17      Si soporta timestamps
18           Convertir en beta
19      Si no
20           Convertir en épsilon
```

Algoritmo 3.1.1. *Clasificación de nodos.*

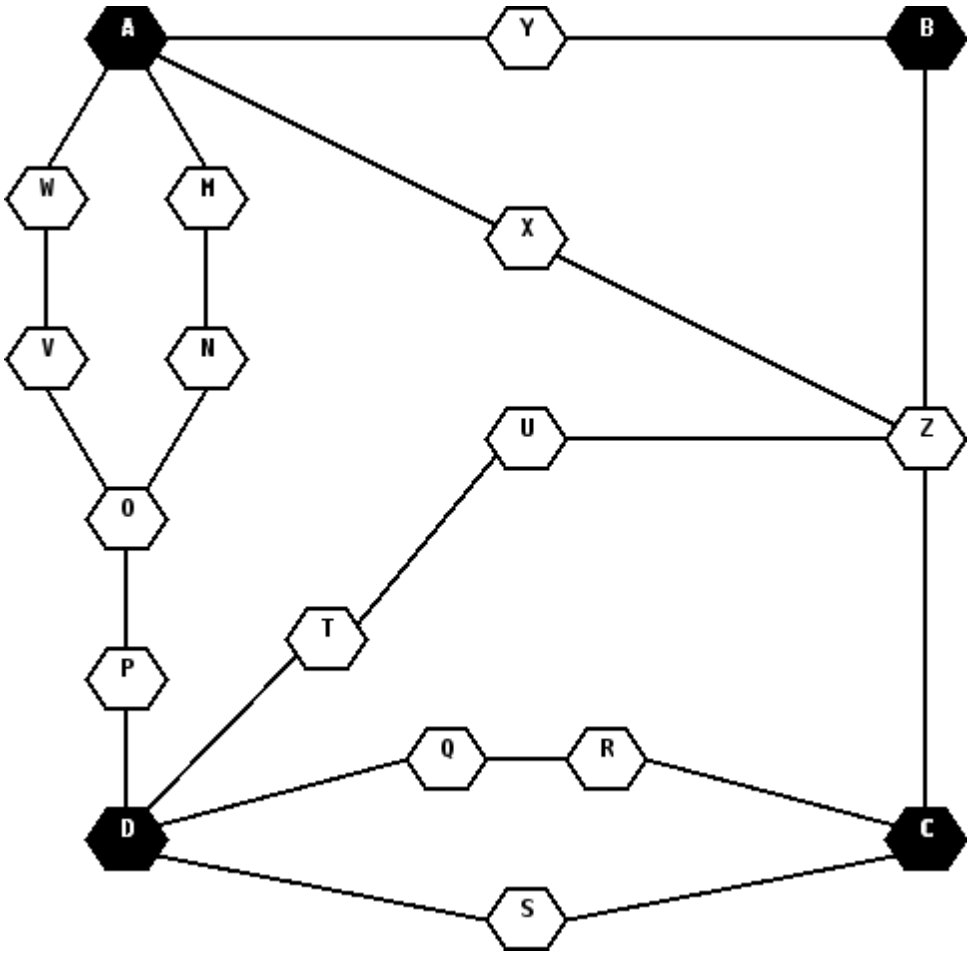
Ejemplo

A partir de la siguiente página se presenta un ejemplo de aplicación del algoritmo de clasificación de nodos. Se comienza con 4 nodos alfa (*A*, *B*, *C* y *D*) y se descubren y clasifican los nodos beta, gama, delta y épsilon.

Al principio sólo se conocen los alfas (nodos controlables).



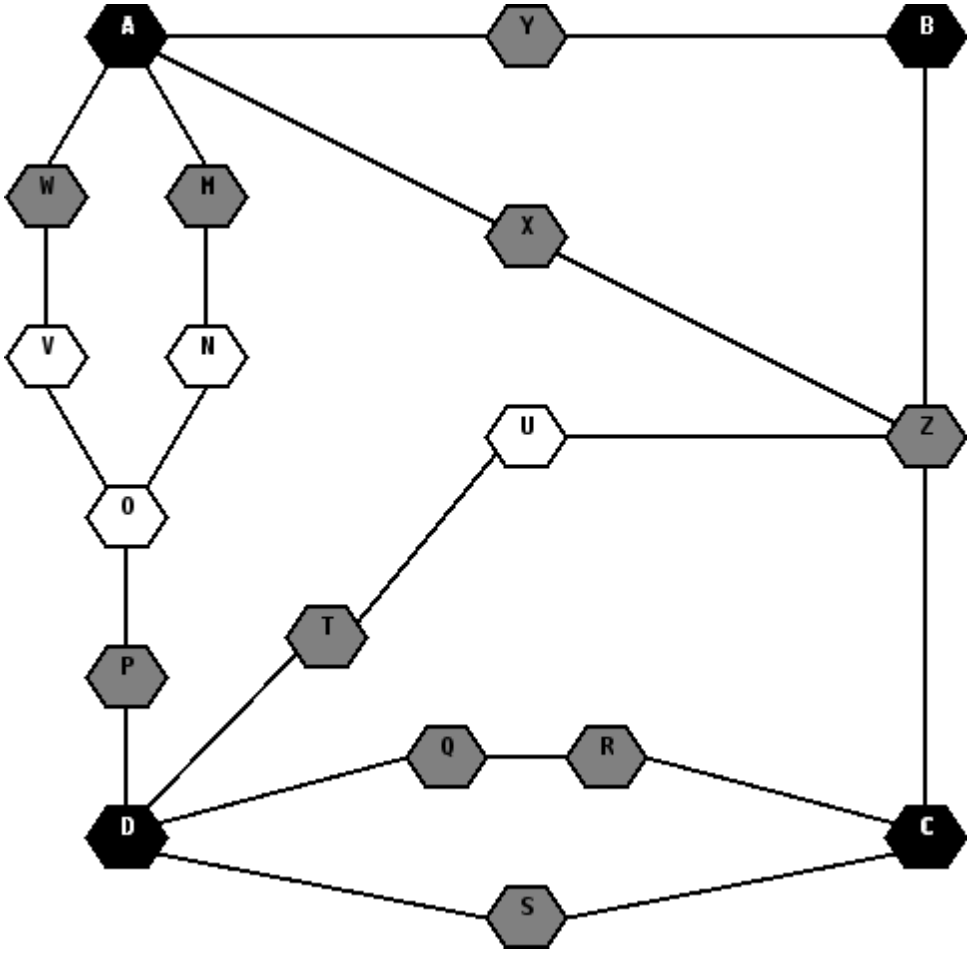
Se traza la ruta entre cada posible par de alfas.
Consecuentemente, se descubren algunos deltas.



Se toman como ejemplo las siguientes rutas, de las cuales se listan los enlaces que las componen:

A->D es AW,WV,VO,OP,PD
D->A es DP,PO,ON,NM,MA

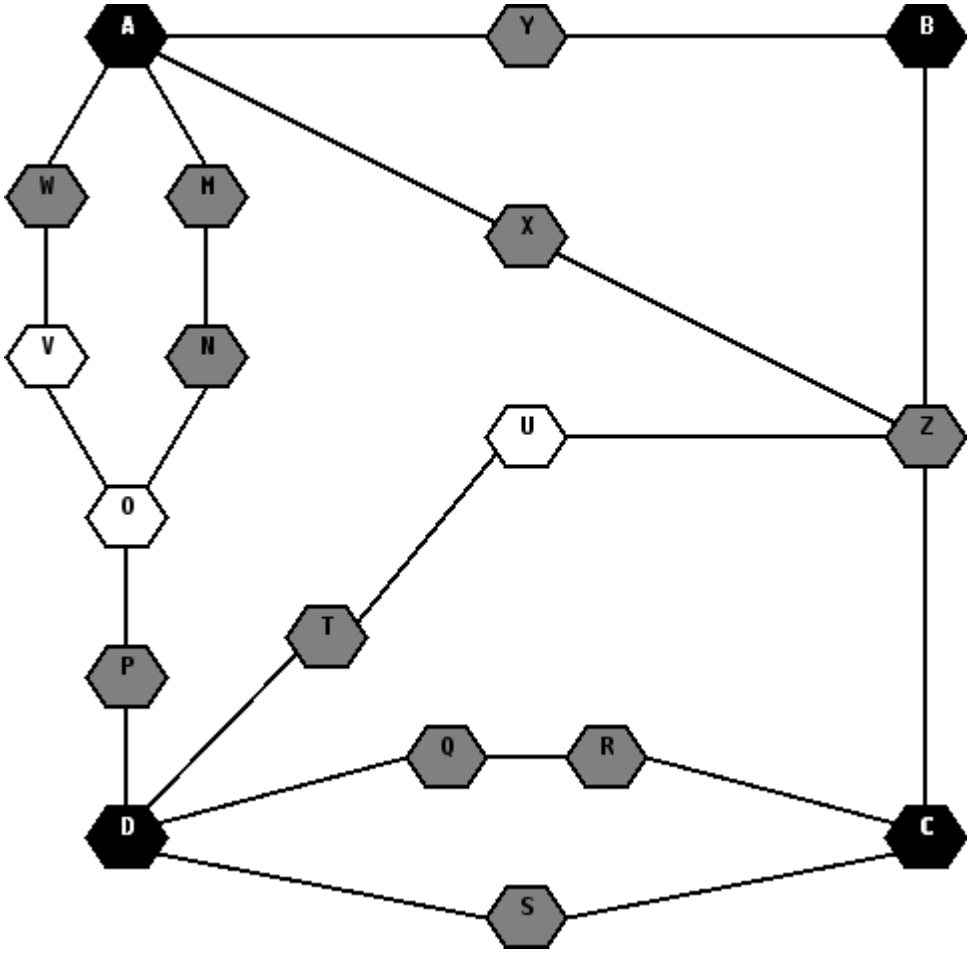
Los deltas adyacentes a alfas son obviamente alfa-simétricos, por lo cual se los convierte en gamas.



¿ Qué sucede con N ?
Al trazar la ruta D->A, también se aprendió fortuitamente que la ruta N->A es NM,MA.
Si se trazara la ruta A->N y resultara ser AM,MN entonces...
...N sería alfa simétrico.

Por ejemplo, si eso efectivamente sucede:

Entonces N es gama.



¿ Y qué sucede con O ?

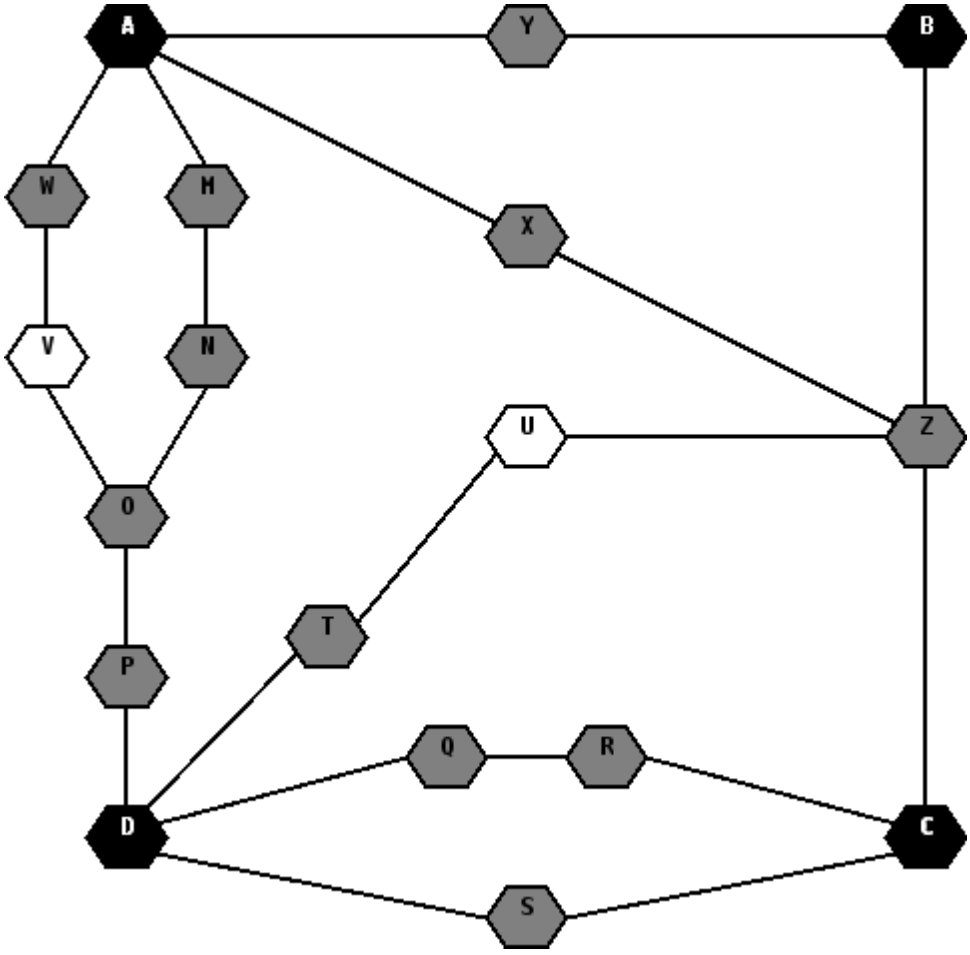
Al trazar la ruta D->A, también se aprendió fortuitamente que la ruta O->A es ON,NM,MA.
Y al trazar la ruta A->D, también se aprendió fortuitamente que la ruta O->D es OP,PD.

Por lo tanto, si se trazara la ruta de A a O, y resultara ser AM,MN,NO
y/o
si se trazara la ruta de D a O, y resultara ser DP,PO

Entonces O sería alfa-simétrico.

Por ejemplo, si se eso efectivamente sucede:

Entonces O es gama.



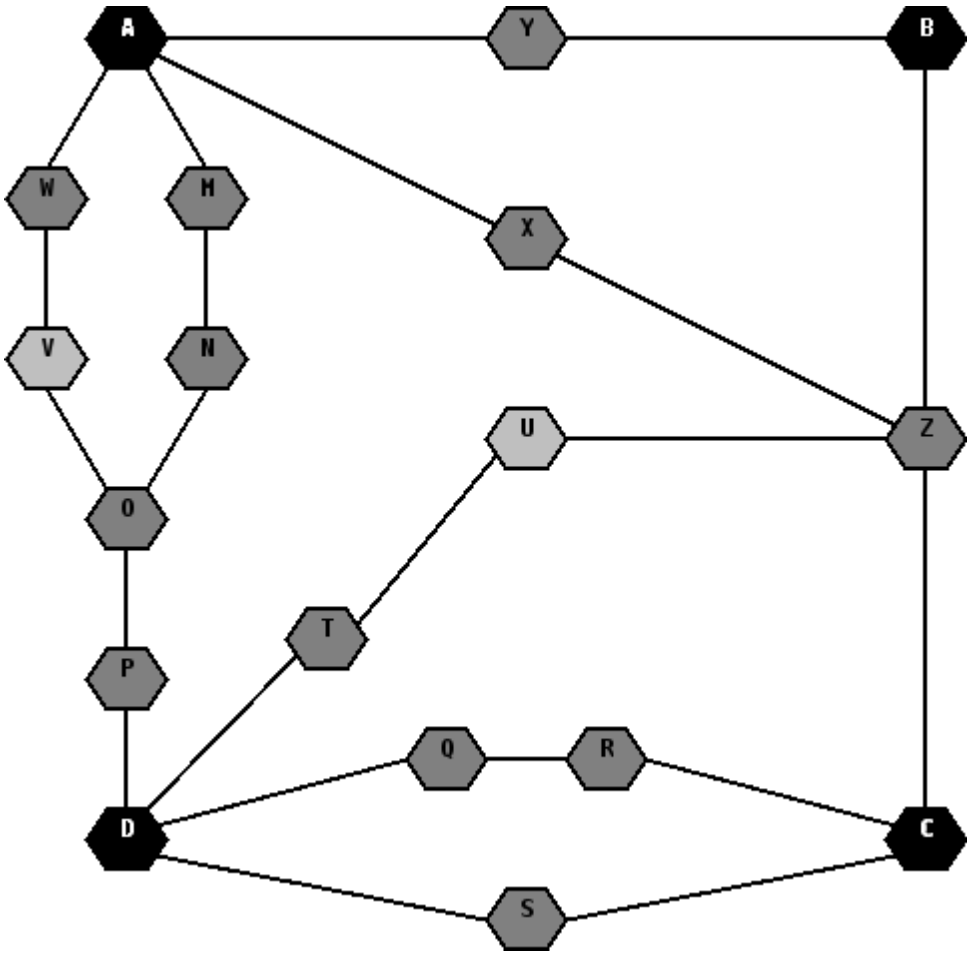
Se puede hacer lo mismo con V (desde D)
y con U (desde B y desde D).

Para variar, si se asume por ejemplo que ninguna de las 3 rutas resulta simétrica...

...entonces U y V no son alfa-simétricos.

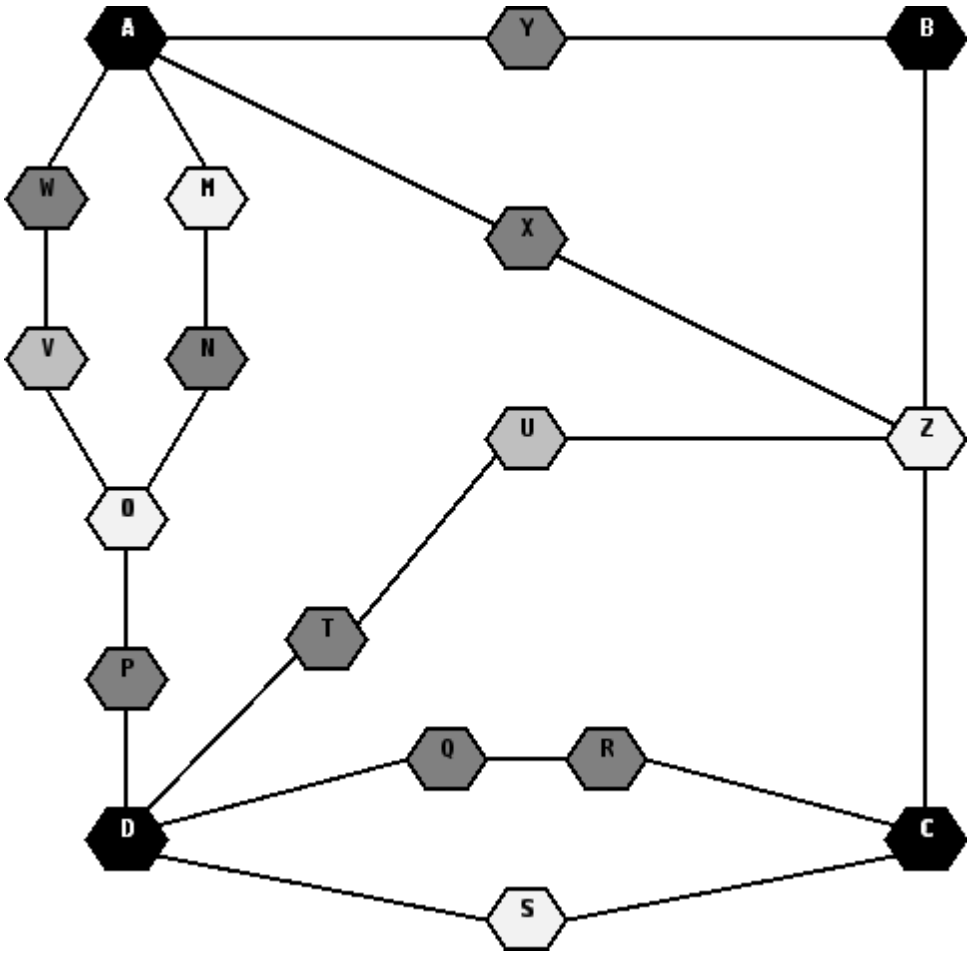
Con lo cual nunca podrían ser betas.

Sino que en realidad son épsilons:

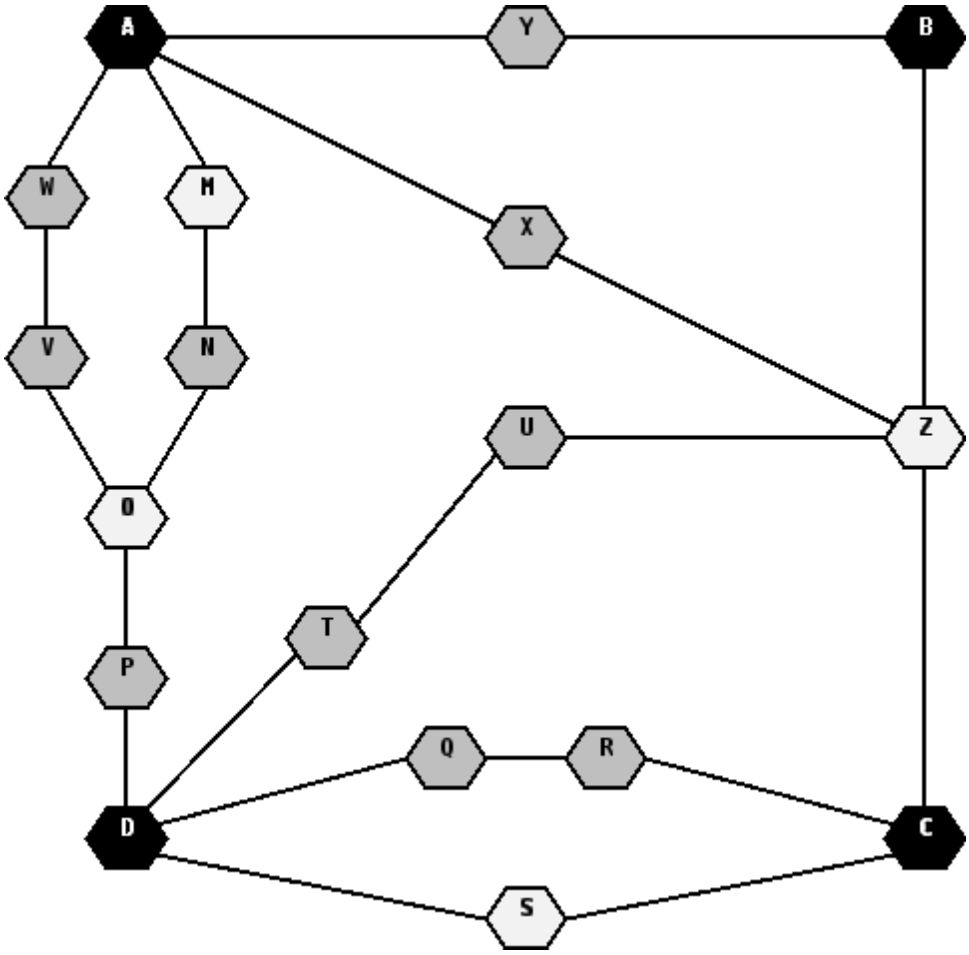


Ahora es necesario intentar convertir gamas en betas.
Se envían paquetes *timestamp* a todos los gamas.

Si por ejemplo M, O, S y Z resultan soportar *timestamps*, entonces son betas:



Y el resto de los gamas, es decir, los que no soportan *timestamps*, son deltas:



3.1.2. Creación de la matriz de ruteo

El algoritmo 3.1.2 describe la creación de la matriz de ruteo. Las rutas medibles son aquellas que van entre dos alfas o entre un alfa y un beta. Todas las rutas entre alfas se conocen, pero la mayoría de las rutas entre un alfa y un beta no fueron descubiertas en el proceso de clasificación de nodos. Por eso la primera etapa de este algoritmo es aprender las rutas entre todo par posible alfa-beta.

Si la ruta del alfa al beta es desconocida, simplemente se hace un *traceroute* (líneas 3 y 4).

Pero si la ruta del beta al alfa es desconocida, no se puede hacer un *traceroute* para solucionarlo, porque los betas no son controlables. Entonces se crea un enlace virtual del beta al alfa y se considera que la ruta de regreso del beta al alfa es la que atraviesa única y simplemente ese enlace virtual (líneas 5 y 6).

Luego de determinadas todas las rutas de ida y de vuelta entre todo par posible alfa-beta, ya son conocidas todas las rutas medibles (es decir, las rutas alfa-alfa y alfa-beta) y se puede finalmente crear la matriz de ruteo.

El procedimiento consiste en recorrer la lista de rutas medibles e ir agregando las correspondientes filas en la matriz de ruteo (líneas 7 a 11). Antes de poder agregar cada una de esas filas, es necesario que existan todas las columnas correspondientes a los enlaces atravesados por la ruta a la que se refiere la fila que se desea agregar (líneas 8 a 10). Esto quiere decir que, por ejemplo, no se puede dar de alta la fila correspondiente a la ruta $A->D = AB + BC + CD$ si antes no existen las columnas correspondientes a los enlaces *AB*, *BC* y *CD*.

Cuando se agrega a la matriz de ruteo la fila correspondiente a una ruta (línea 11), la fila agregada contendrá *1* en las posiciones correspondientes a las columnas que representan a los enlaces atravesados por la ruta, y *0* en el resto de las posiciones.

La complejidad del algoritmo 3.1.2 es $O(n_\alpha^2 l)$, donde n_α es la cantidad de alfas y l es la máxima cantidad de enlaces en una ruta. El análisis que permite llegar a ese resultado se presenta a continuación.

La complejidad de los ciclos anidados de las líneas 1 y 2 es exactamente $n_\alpha n_\beta$, donde n_β es la cantidad de betas. La complejidad de los ciclos anidados de las líneas 7 y 8 es $O(rl)$, donde r es la cantidad de rutas medibles, es decir, la cantidad de filas de la matriz de ruteo, y l es la máxima cantidad de enlaces en una ruta.

La complejidad total es entonces:

$$O(n_\alpha n_\beta + rl)$$

Pero esta expresión se puede simplificar. Como la cantidad de filas en la matriz de ruteo es $O(n_\alpha^2)$, la complejidad total se puede escribir como:

$$O\left(n_\alpha (n_\beta + n_\alpha l)\right)$$

Como pocos nodos en la red soportan *ICMP timestamp*, la cantidad de betas es inferior a la

cantidad de alfas, y entonces es fácil de ver que la complejidad total queda:

$$O\left(n_\alpha{}^2 I\right)$$

```
    Entrada:
         Lista de nodos
         Lista de rutas
    Salida:
         Matriz de ruteo
         Lista de nodos actualizada
         Lista de rutas actualizada
 1  Para cada alfa a
 2       Para cada beta b
 3            Si no se conoce la ruta ab
 4                 Trazarla
 5            Si no se conoce la ruta ba
 6                 Asignarle un enlace virtual ba
 7  Para cada ruta medible r
 8       Para cada enlace e
 9            Si la matriz no tiene la columna e
10                 Agregar a la matriz la columna e
11       Agregar a la matriz la fila correspondiente a r
```

Algoritmo 3.1.2. *Creación de la matriz de ruteo.*

3.2. Minimización del sesgo

Como se detalló en la sección 1 del capítulo 2, la matriz de ruteo típicamente tiene rango deficiente (es decir, que la cantidad de ecuaciones linealmente independientes es menor que la cantidad de incógnitas).

Consecuentemente, el sistema no tiene una solución única. Esto implica que al hacer el cálculo, se elige una solución en especial, cual significa adoptar algún criterio.

Por ejemplo, [Chua05] elige como criterio asignarle cero a los valores de la métrica para los enlaces que no son atravesados por ninguna de las rutas monitoreadas.

Adoptar un criterio introduce inevitablemente un sesgo. Pero la diferencia entre este sesgo y los demás que también aparecen (por ejemplo, el sesgo introducido por la hipótesis de independencia

entre las demoras en los enlaces) es que este sesgo puede ser eliminado.

Si se puede transformar el sistema de forma tal que la matriz de ruteo no presente rango deficiente(que la cantidad de ecuaciones linealmente independientes sea igual a la cantidad de incógnitas) entonces el sistema deja de ser indeterminado y consecuentemente la solución es única.

Luego, ya no es necesario imponer un criterio para elegir la solución, y por lo tanto el sesgo en cuestión desaparece.

Esto es lo que se propone solucionar en [Zhao05] y [Zhao06]. En esta sección se presenta una manera enormemente más económica de llevarlo a cabo que en dichos trabajos, representantes del estado del arte actual.

Primeramente se definen dos conceptos fundamentales:

Identificabilidad. Un enlace es *identificable* cuando su variable asociada se puede escribir en función de los datos.

Red abierta. Una red es *abierta* cuando puede crearse o destruirse flujo en cada uno de sus nodos.

Dada la red abierta de la figura 3.2.1, donde el nodo intermedio Z no es alfabeto (es decir, no es ni controlable ni sincronizable, con lo cual no puede ser extremo de ninguna medición), las mediciones que se pueden tomar son: la ruta desde A hasta B, la ruta desde B hasta A, el roundtrip entre A y Z y el roundtrip entre B y Z.

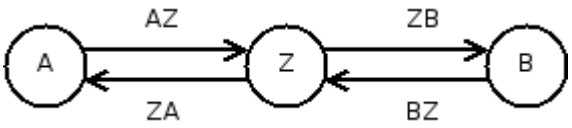


Figura 3.2.1. *Nodo intermedio que no se puede controlar ni sincronizar.*

Lema 1. Los enlaces AZ, ZB, BZ y ZA no son identificables.

Demostración.
Como los enlaces son AZ, ZB, BZ y ZA, entonces las ecuaciones que se pueden plantear son:

$$\begin{cases} AZ + ZB = AB_{obs} \\ BZ + ZA = BA_{obs} \\ AZ + ZA = AZA_{obs} \\ BZ + ZB = BZB_{obs} \end{cases}$$

Escribiéndolo con la notación matricial $R \cdot L = M$ queda:

$$\begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} \cdot \begin{pmatrix} AZ \\ ZB \\ BZ \\ ZA \end{pmatrix} = \begin{pmatrix} AB_{obs} \\ BA_{obs} \\ AZA_{obs} \\ BZB_{obs} \end{pmatrix}$$

La matriz de ruteo tiene rango 3, luego los 4 enlaces no pueden ser identificables. □

Se puede conjeturar que los enlaces conectados a no alfabetos no son identificables.

Según el lema, los pares de enlaces AZ, ZB y BZ, ZA no son identificables, pero sí lo son sus concatenaciones, es decir, los enlaces virtuales AZB y BZA.

Teorema 1. Los enlaces identificables son las concatenaciones delos enlaces existentes entre los alfabetos de las rutas medibles.

Demostración.
Dada una ruta cualquiera, la misma tiene necesariamente alfabetos en sus extremos, y los nodos intermedios serán alfabetos y no alfabetos. Por ejemplo:

$$a_1 \to z_1 \to z_2 \to a_2 \to z_3 \to z_4 \to z_5 \to a_3 \to a_4 \to z_6 \to z_7 \to z_8 \to a_5$$

donde la letra *a* se usó para denotar a los alfabetos y la *z* para los no alfabetos.
Agrupando los no alfabetos consecutivos, se puede escribir así:

$$a_1 \to (z_1 \to z_2) \to a_2 \to (z_3 \to z_4 \to z_5) \to a_3 \to a_4 \to (z_6 \to z_7 \to z_8) \to a_5$$

Y en general, se puede escribir así:

$$a_1 \to (z_1 \to \ldots \to z_i) \to a_2 \to (z_{i+1} \to \ldots \to z_j) \to a_3 \to \ldots \to a_n$$

Se ve claramente que se puede particionar en segmentos determinados por los alfabetos. Un segmento cualquiera se puede escribir así:

$$a_i \to (z_k \to z_{k+1} \to z_{k+2} \to \ldots) \to a_j$$

Si se aisla el nodo z_k queda:

$$a_i \rightarrow z_k \rightarrow \left(z_{k+1} \rightarrow z_{k+2} \rightarrow \ldots \right) \rightarrow a_j$$

Por el lema 1, los enlaces anterior y posterior al nodo z_k no son identificables. Lo mismo sucede con cualquier otro z_m intermedio:

$$a_i \rightarrow \left(z_k \rightarrow z_{k+1} \rightarrow \ldots \rightarrow z_{m-1} \right) \rightarrow z_m \rightarrow \left(z_{m+1} \rightarrow z_{m+2} \rightarrow \ldots \right) \rightarrow a_j$$

Por lo tanto, queda demostrado el teorema. □

El teorema permite derivar el algoritmo 3.2.1 para la búsqueda de los enlaces identificables.

```
    Entrada:
         Lista de nodos
         Matriz de ruteo R
    Salida:
         Lista de enlaces virtuales identificables
 1  Para cada ruta p  (o sea, para cada fila de R)
 2       Crear un enlace virtual e vacío
 3       Para cada enlace xy
 4            Agregar xy a e
 5            Si y es un alfabeto
 6                 Agregar e a la lista de enlaces virtuales identificables
 7                 Crear un enlace virtual e vacío
```

Algoritmo 3.2.1. *Búsqueda de los enlaces identificables.*

Dadas *r* la cantidad de rutas y *l* la máxima cantidad de enlaces en una ruta, el ciclo externo del algoritmo (línea 1) consta de *r* iteraciones y el ciclo interno (línea 3) de a lo sumo *l*. Luego la complejidad del algoritmo 3.2.1 es evidentemente $O(rl)$.

Esto es mucho menor que la complejidad $O(rkl^2)$ del algoritmo 2.1.4, propuesto por Zhao et al. Se puede apreciar que la diferencia es tan abismal como la diferencia entre n y n^2.

En el capítulo 4 se comparan los dos métodos (el algoritmo 2.1.4, propuesto por Zhao et al. y el algoritmo 3.2.1, presentado en este trabajo), obteniéndose la misma matriz de ruteo.

Además el nuevo algoritmo es claramente acoplable a los algoritmos de determinación de las rutas también aportados por el presente trabajo (sección 3.1). Puntualmente, se puede integrar con el algoritmo 3.1.2. Esto quiere decir que no es necesario ejecutar primero el algoritmo 3.1.2 y luego el

algoritmo 3.2.1, sino que se pueden combinar en un único algoritmo, lo cual es positivo porque significa que la complejidad total es menor que la suma de las respectivas complejidades.

La integración es posible gracias a la equivalencia de los ciclos anidados de las líneas 8 y 9 del algoritmo 3.1.2 con los del algoritmo 3.2.1 y el hecho de que no existe una relación de precedencia en los dos pares de ciclos. El algoritmo 3.2.2 consiste en la integración de los algoritmos 3.1.2 y 3.2.1.

```
   Entrada:
        Lista de nodos
        Lista de rutas
   Salida:
        Matriz de ruteo
        Lista de nodos actualizada
        Lista de rutas actualizada
 1 Para cada alfa a
 2      Para cada beta b
 3           Si no se conoce la ruta ab
 4                Trazarla
 5           Si no se conoce la ruta ba
 6                Asignarle un enlace virtual ba
 7 Para cada ruta medible p
 8      Crear un enlace virtual e vacío
 9      Para cada enlace xy de la ruta p
10           Agregar xy a e
11           Si y es un alfabeto
12                El enlace virtual e es identificable
13                Si la matriz no tiene la columna e
14                     Agregar a la matriz la columna e
                  Crear un enlace virtual e vacío
15      Agregar a la matriz la fila correspondiente a p
```

Algoritmo 3.2.2. *Creación de la matriz de ruteo en términos de enlaces identificables.*

La complejidad del algoritmo 3.2.2 es la misma que la del algoritmo 3.1.2. En la sección anterior se determinó que dicha complejidad es:

$$O\left(n_{\alpha}^{2} l\right)$$

Como se detalló anteriormente, la complejidad del algoritmo de Zhao et al. para llevar a cabo la misma tarea que el algoritmo 3.2.2 es:

$$O\left(r k l^{2}\right)$$

Teniendo en cuenta que $r = n_\alpha(n_\alpha - 1)$ la complejidad del nuevo algoritmo es mucho menor. Pero además, cabe destacar que el nuevo algoritmo permite obtener mucha más información, porque no solamente se ocupa de los nodos alfa sino también de los nodos beta.

4. RESULTADOS EXPERIMENTALES

Con el objeto de evaluar el desempeño de los métodos propuestos, se llevaron a cabo simulaciones con datos reales de los proyectos CAIDA y DIMES, así como también con redes aleatorias[§].

Para posibilitar la verificación y comparación, fueron implementados todos los algoritmos de los capítulos 2 (algoritmos actuales) y 3 (algoritmos nuevos). En ambos algoritmos, se utilizó el mismo módulo para trazar rutas. La ruta entre dos nodos corresponde a alguno de los caminos con longitud mínima entre ellos.

En la sección 4.1 se presentan los resultados de aplicar los algoritmos en distintas topologías.

Los algoritmos también fueron utilizados con mediciones reales de rutas en Internet tomadas por el proyecto DIMES. Los resultados se presentan en la sección 4.2.

En la sección 4.3 se analizan los resultados y se extraen conclusiones.

Para llevar a cabo las simulaciones se desarrolló una herramienta, que se describe en el apéndice B.

Objetivos

La experimentación se llevó a cabo con los siguientes objetivos:

- Verificar que los MILSes (enlaces virtuales identificables mínimos) encontrados por los nuevos algoritmos coinciden con los del algoritmo 2.1.4 presentado por Zhao, Chen et al. [Zhao05, Zhao06].
- Estudiar el beneficio de utilizar betas (nodos no controlables pero sincronizables). Es decir, cuantificar el incremento en la cantidad de enlaces virtuales identificables mínimos, representantes de la granularidad, contra el incremento en la cantidad de nodos en la red que soportan de *ICMP timestamps*.
- Aplicar la metodología a mediciones en Internet pertenecientes al proyecto DIMES. La idea es por un lado utilizar datos reales y por el otro estudiar el proceso de determinar

§ Una red generada al azar, donde la probabilidad de que un par cualquiera de nodos estén conectados es idéntica para todos los pares posibles de nodos.

los nodos a partir de las mediciones, inverso al aplicado en los experimentos detallados anteriormente.

4.1. Simulaciones en diferentes topologías

Los algoritmos de los capítulos anteriores fueron aplicados en simulaciones sobre diversas topologías. En esta sección se describen primeramente las redes utilizadas y el procedimiento utilizado para llevar a cabo las simulaciones. Luego se presentan los resultados obtenidos y se lleva a cabo el análisis de los mismos.

Las redes utilizadas

Primeramente se utilizó una red cuya topología es representativa de la topología de Internet a nivel ruteadores. Dicha red es la red de ruteadores correspondiente al proyecto CAIDA

Para las simulaciones se tomó la componente conexa más grande de la red. En la sección 3.1.1 se explica que la metodología también es aplicable a redes disconexas. Luego de seleccionar la mayor componente conexa, la red tiene las siguientes características:

- Aproximadamente 90.000 nodos.
- Aproximadamente 250.000 enlaces físicos.
- La distribución de grados[¶] de sus nodos corresponde a una ley de potencias.

Se tomaron 100 alfas, lo que arroja una densidad de nodos controlables del 0,11%.

La segunda red utilizada fue una red aleatoria con los siguientes parámetros:

- 10.000 nodos.
- probabilidad de conexión $p = 0,01$.

El procedimiento mediante el cual fue construida la red aleatoria es el siguiente:

1. Se crearon los 10.000 nodos.
2. Se iteró sobre todos los pares posibles de nodos (en este caso 49.995.000) y se creó el correspondiente enlace con probabilidad p.

¶ En un grafo, el grado de un nodo es la cantidad de aristas que lo tienen como extremo. Análogamente, en una red, el grado de un nodo es la cantidad de enlaces físicos a los que está conectado.

La red quedó de esta manera conformada con las siguientes características:

Característica	*Valor*	*Comentario*
Cantidad de nodos	10.000	Fijado por parámetro
Cantidad de enlaces físicos	499.955	Aproximadamente el 1% de la cantidad de enlaces posibles, de acuerdo a lo establecido en el parámetro *p*.
Distribución de grados	normal	Se puede explicar mediante el teorema del límite central, porque para cada nodo, su grado es la suma de 9.999 variables aleatorias independientes e idénticamente distribuidas (Bernoulli con idéntico parámetro *p*).
Grado medio	99.99	

La elección de una red aleatoria de grado medio aproximadamente 100 se basa en el hecho de que si Internet fuera una red aleatoria, entonces sus ruteadores deberían tener grado medio 100, como se demuestra en [Dallasta06].

Para obtener la misma densidad de alfas que se utilizó en las simulaciones con la red CAIDA (0.11%), en la red aleatoria se tomaron 11 alfas.

En la figura 4.1.1 se representan 4 curvas. Por un lado, se puede apreciar la distribución empírica de los grados de los nodos de la red aleatoria (círculos) comparada con una ley *gaussiana* de media 100 y desvío estándar 10 (línea completa), y la distribución empírica de los grados de los nodos de la red CAIDA (triángulos) comparada con una ley de potencias (línea punteada).

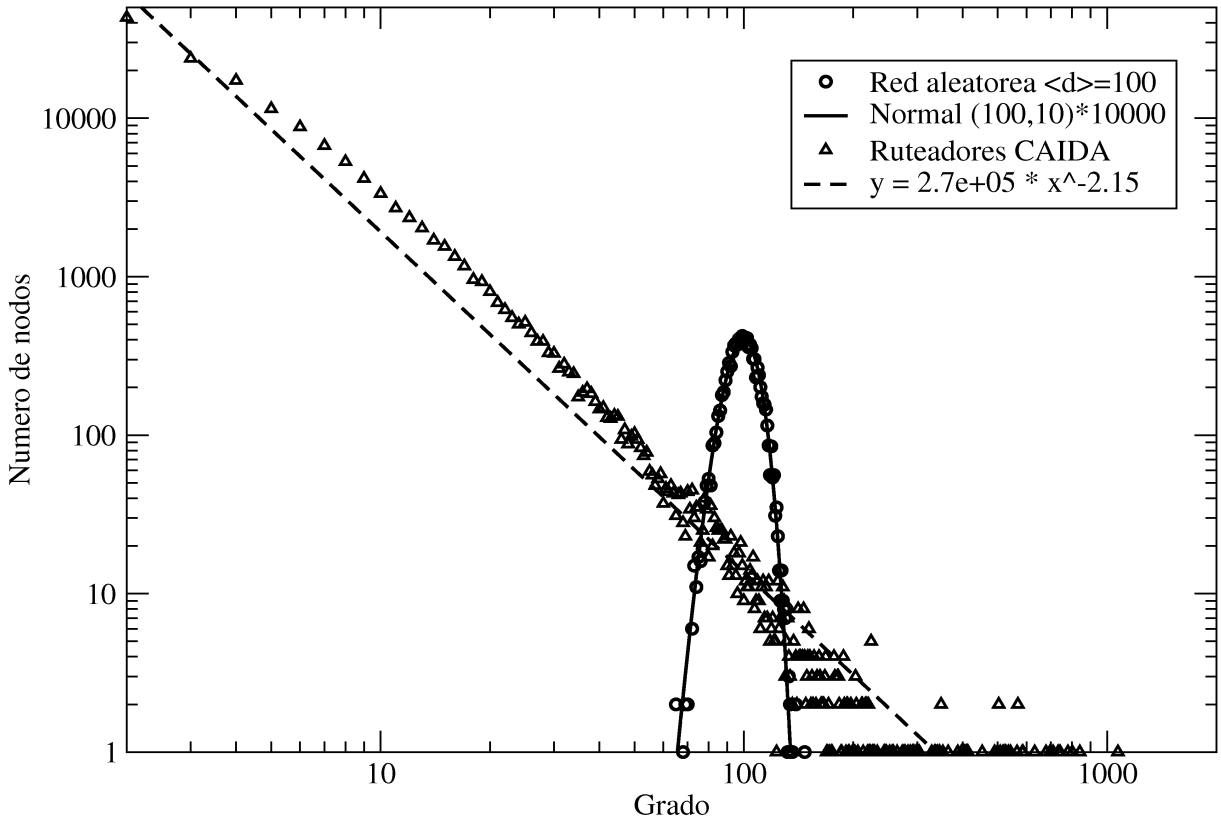


Figura 4.1.1. *Distribuciones de los grados de los nodos de las redes utilizadas. La distribución de la red aleatoria es similar a una ley normal con media 100 y desvío estándar 10, y la distribución de la red CAIDA es similar a una ley de potencias.*

¿Por qué presentan las redes distribuciones de grados tan disímiles?

En la red CAIDA, representativa de Internet a nivel ruteadores, muchos nodos tienen grados bajos, porque se encuentran por así decirlo, en la periferia. Los nodos que conectan entre sí a los racimos de nodos periféricos son relativamente pocos, pero tienen grados altos. Esto se puede apreciar en la figura 4.1.2 (a), que representa a una red con las mencionadas características. Se puede observar la organización en racimos.

Por su parte, en la red aleatoria, no existe la noción de nodos "periféricos" o "centrales" debido a que todo par de nodos tiene idéntica probabilidad de conexión *p*. De ese modo, todos los nodos tienden a tener grados parecidos, motivo por el cual a veces se la denomina "homogénea". Esto se puede apreciar en la figura 4.1.2 (b), que representa a una red de esas características. El grado de cada nodo es una variable aleatoria binomial, que cuando la cantidad de nodos es grande se aproxima a una distribución normal. Esta distribución tiene mucha menor varianza que una distribución como la de potencias, que tiene una cola más pesada.

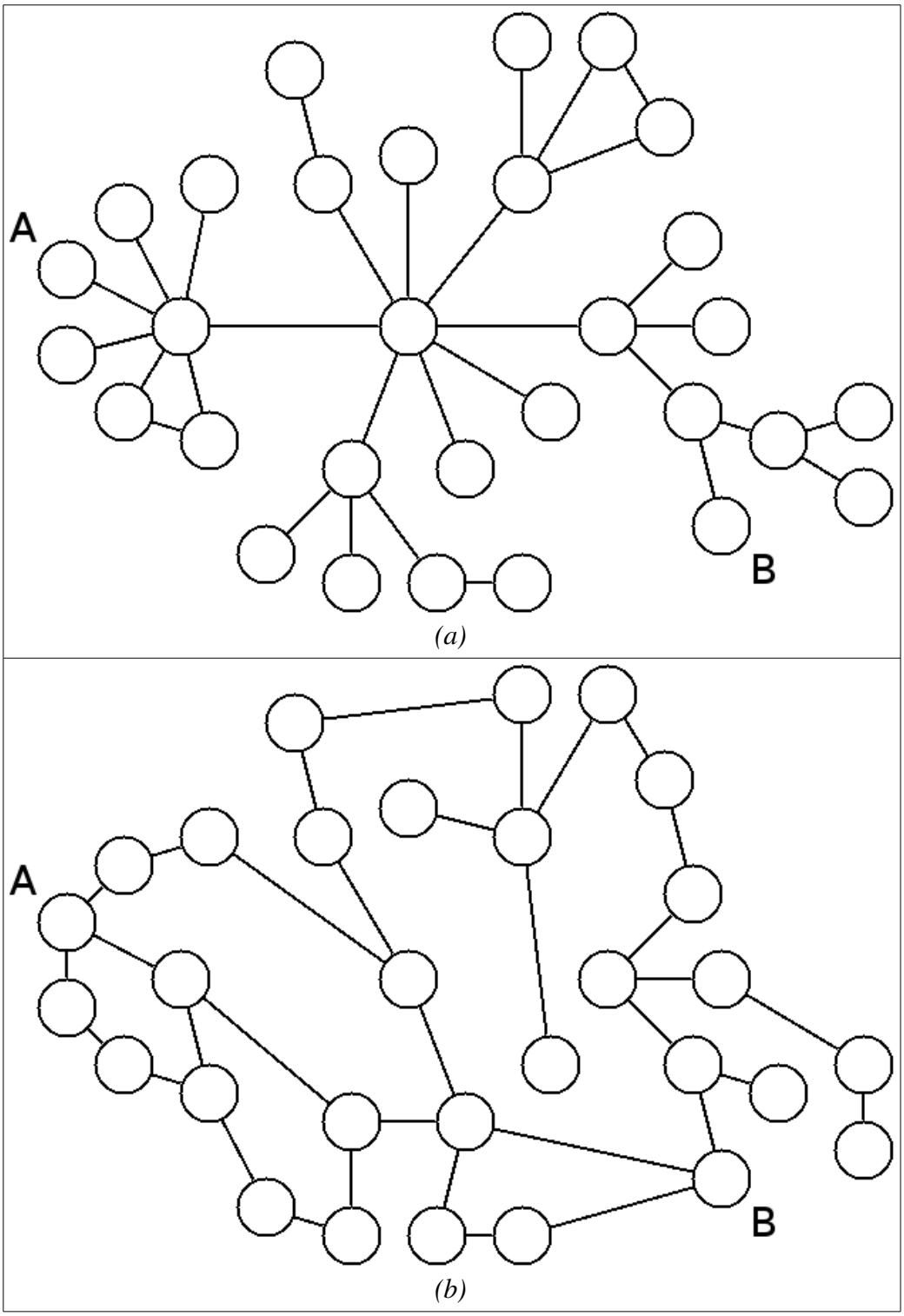


Figura 4.1.2. *Topologías estudiadas. La red (a) tiene una distribución de grados que sigue una ley de potencias, como la de red CAIDA. La red (b) tiene una distribución de grados que sigue una ley normal, como la de una red aleatoria.*

El procedimiento

A continuación se describe el método aplicado a cada una de las redes.

Para cada red, se estudió su comportamiento para distintos niveles de soporte de *ICMP timestamps*. En particular, se utilizaron los siguientes niveles:

- 0 a 100%, en intervalos de 10%. El objetivo es estudiar el comportamiento para todo el rango de valores posibles del nivel de soporte.
- 0 a 10%, en intervalos de 1%. El objetivo es estudiar el comportamiento para niveles bajos de soporte. Como se detalla más adelante, este rango resulta particularmente interesante, debido a que es el que más frecuentemente aparece en la realidad.

Por cada nivel de soporte, se llevaron a cabo 100 iteraciones. En cada iteración se eligieron al azar los alfas (nodos controlables) y luego se ejecutaron los algoritmos de los capítulos 2 (Chen et. al) y 3 (los presentados en este trabajo). Luego de cada ejecución de los algoritmos se compararon los resultados para verificar que los resultados son los mismos. También se registró el valor de las siguientes variables:

- Cantidad de betas
- Cantidad de nodos no alfabetos
- Cantidad de enlaces físicos
- Cantidad de enlaces virtuales identificables mínimos (MILSes)

Luego de efectuadas las 100 iteraciones para cada nivel de soporte, se calculó el promedio y desvío estándar de las variables observadas.

El algoritmo 4.1.1 resume el proceso de simulación.

```
   Entrada:
        Las redes a utilizar (CAIDA y aleatoria).
   Salida:
        Promedio y desvío estándar de cada variable observada para
        cada punto de cada red.
1  Para cada red
2       Para cada nivel de soporte de ICMP timestamps
        (0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 20, 30, 40, 50, 60, 70, 80,
        90 y 100%)
3            Iterar 100 veces
4                 Elegir alfas al azar
5                 Ejecutar algoritmos de los capítulos 2 y 3
6                 Verificar coincidencia de los resultados
7                 Tomar nota de las variables observadas
8                       Cantidad de betas
                        Cantidad de nodos no alfabetos
                        Cantidad de enlaces físicos
                        Cantidad de enlaces virtuales identificables
                        mínimos (MILSes)
9            Calcular el promedio y el desvío estándar de la muestra
             de 100 iteraciones de cada una de las variables
             observadas
```

Algoritmo 4.1.1. *Procedimiento utilizado para llevar a cabo las simulaciones.*

Resultados

A continuación se presentan los resultados obtenidos y el correspondiente análisis. Para facilitar la comprensión de los resultados, en esta sección los mismos se presentan únicamente en formato gráfico. Los valores en formato tabla aparecen en el apéndice A.

En todos los casos, los enlaces virtuales determinados por el algoritmo de Zhao et al. 2.1.4 y el nuevo algoritmo 3.2.1 propuesto por el presente trabajo fueron idénticos.

En la figura 4.1.3 se pueden visualizar los valores de las variables observadas para ambas redes, para los distintos porcentajes de soporte de *ICMP timestamps*. En las figuras (a) y (b) se muestran las cantidades de nodos beta y no-alfabetos respectivamente. En la figura (c) se muestran las cantidades de enlaces físicos. Como la red aleatoria tiene la décima parte de la cantidad de nodos de la red CAIDA, las curvas en las figuras (a), (b) y (c) se encuentran corridas aproximadamente una década. En la figura (d) se muestran las cantidades de enlaces virtuales, divididas por el total de enlaces físicos totales de cada red; eso permite tener una idea de qué porcentaje de la red se logra descubrir.

En las páginas subsiguientes de esta misma sección se desarrolla el análisis de las variables observadas.

Los gráficos se trazaron en base a los resultados de las simulaciones, contenidos en las tablas A.1 (red CAIDA) y A.2 (red aleatoria).

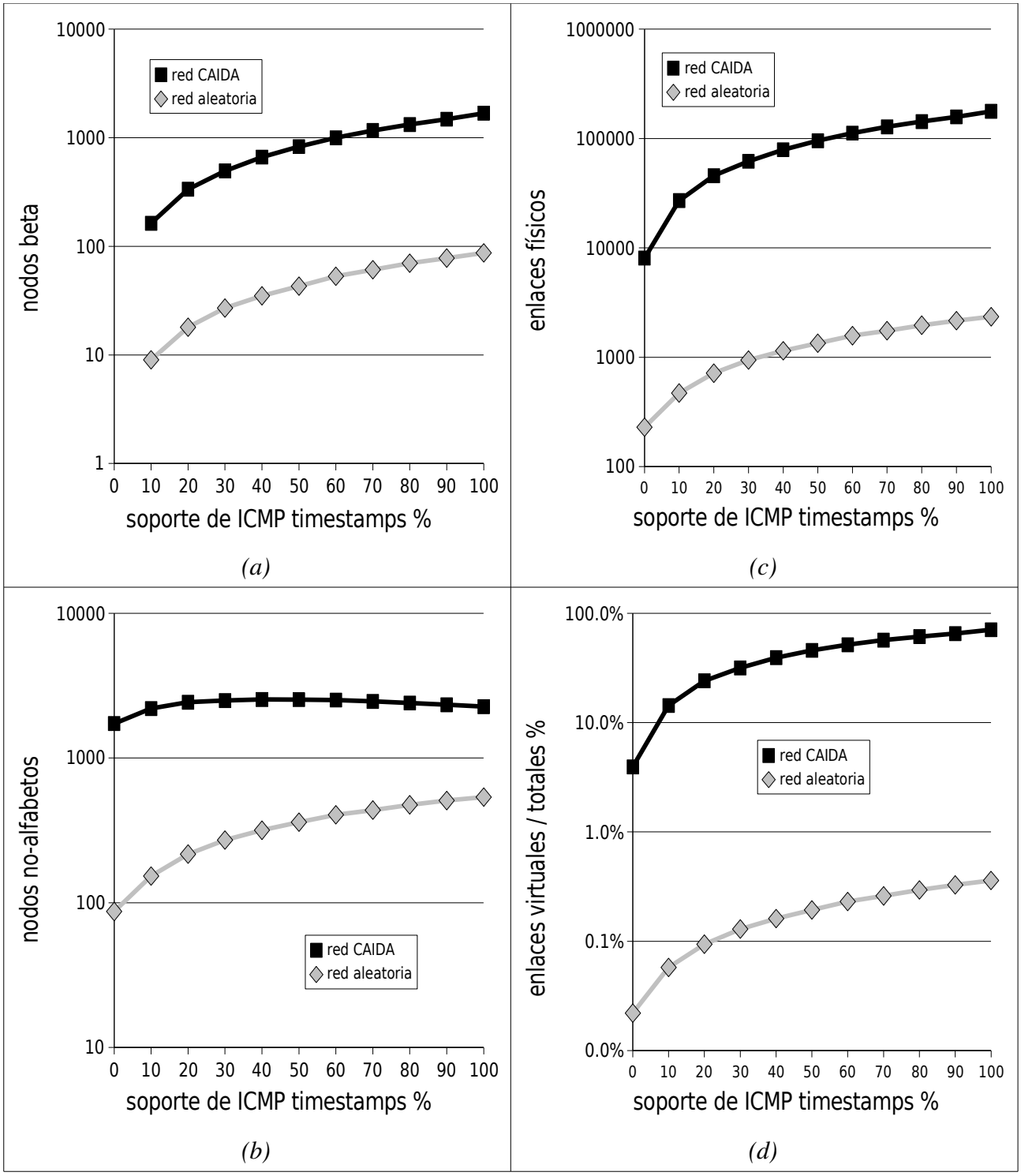


Figura 4.1.3. *Variables observadas, para ambas redes, para los distintos porcentajes de soporte de ICMP timestamps.*

Cantidad de enlaces virtuales

La primera variable a analizar es la cantidad de enlaces calculables individualmente, es decir, los enlaces virtuales. Dicha variable es la representada en la figura 4.1.3(d).

El primer punto en cada línea trazada, tomado de la primera fila de cada tabla (A.1, A.2), corresponde al caso en que absolutamente ningún nodo de la red soporta *ICMP timestamps*. En ese escenario no existen los nodos beta, y la cantidad de enlaces calculables (es decir, los enlaces virtuales) es $O(n^2)$[¶], donde n es la cantidad de nodos controlables[§]. Dicha situación es la estudiada por Zhao et al. [Zhao05, Zhao06], quienes, como se detalló en los capítulos 2 y 3, directamente desestiman la posibilidad de que los nodos soporten *ICMP timestamps*.

Aunque el porcentaje de nodos que soportan *ICMP timestamps* sea bajo, como indican Zhao et al., una pequeña cantidad alcanza para obtener un volumen mucho mayor de información acerca de la red que si no se los tiene en cuenta: por ejemplo, para la red CAIDA, en la segunda fila de la tabla A.1 se puede apreciar que con tan sólo un 10% de soporte de *ICMP timestamps* entre los nodos de la red, se pasa de 9848 a 35810 enlaces virtuales, es decir, se obtienen casi 4 veces más enlaces calculables individualmente que si no se aprovecha el soporte de *ICMP timestamps*.

En las tablas A.3 y A.4 (para las redes CAIDA y aleatoria respectivamente) se muestra la cantidad de enlaces calculables dado el porcentaje de nodos que soportan *ICMP timestamps*, relativa a la cantidad de enlaces virtuales en el caso en el que no se aprovecha el soporte. Es decir, la cantidad de enlaces virtuales, relativa al 0% de soporte *ICMP timestamps* (tercera columna de las tablas A.3 y A.4) se calcula como:

$$a_i = \frac{v_i}{v_0} \tag{4.1.1}$$

donde v_0 es la cantidad de enlaces virtuales sin soporte de *ICMP timestamps* (es decir, la primera fila de las tablas A.3 y A.4) y v_i es la cantidad de enlaces virtuales para i% de soporte de *ICMP timestamps*. Por ejemplo, para 10% de soporte de *timestamps* (es decir, la segunda fila de la tabla A.3, correspondiente a la red CAIDA) el cálculo fue:

$$\frac{35810}{9848} = 3.64 \tag{4.1.2}$$

Cantidad de información que se puede obtener

Otra característica interesante a observar es la diferencia entre el comportamiento de las redes.

¶ El orden es n^2, pero el valor no es necesariamente exactamente n^2, sino que puede ser un poco menor, debido a los enlaces compartidos entre las distintas rutas.

§ Es decir, la cantidad de nodos alfa, que para la red CAIDA se fijó en 100 y para la red aleatoria se fijó en 11.

En la figura 4.1.3(d) las cantidades de enlaces virtuales fueron divididas por el total de enlaces físicos de cada red, con el propósito de permitir apreciar cuánto se puede aprender sobre cada red.

Se observa fácilmente que, dado un determinado nivel de soporte de *timestamps*, en la red CAIDA se puede descubrir un porcentaje de los nodos totales mucho mayor que en la red aleatoria.

Esto se debe a que en la red aleatoria hay mayor cantidad de rutas posibles entre dos nodos (como ejemplo, ver los nodos *A* y *B* de la figura 4.1.2). En una red como la de CAIDA hay pocas alternativas para las rutas, y por lo tanto los enlaces se comparten más. Esto implica que un mismo enlace aparece en muchas rutas, y por lo tanto tiene mucha mayor probabilidad de aparecer en una ruta cualquiera que en un enlace en una red aleatoria equivalente. Esto permite aprender más sobre los enlaces, y consecuentemente, más sobre la red.

Este resultado coincide con las conclusiones de [Dallasta06], donde se observa que en las redes con distribuciones de grados heterogéneas, como la red CAIDA, es más fácil relevar información sobre la red que en las redes con distribuciones de grados homogéneas, como es el caso en las redes aleatorias.

Variabilidad

En la figura 4.1.4 se trazan las curvas de enlaces virtuales por separado y en escala lineal en vez de logarítmica, para poder apreciar sus características individualmente. Las subfiguras (a) y (b) corresponden a las redes CAIDA y aleatoria respectivamente.

Para cada porcentaje, la altura de la curva representa la media de la muestra de 100 iteraciones, y la línea vertical representa el desvío estándar muestral.

En estadística, el coeficiente de variabilidad, que es una medida de dispersión, se define como el cociente entre el desvío estándar y la media:

$$CV = \frac{\sigma}{\mu} \tag{4.1.3}$$

En el gráfico, para todos los porcentajes, se observa que en el caso de la red aleatoria, el desvío es mayor y la media es menor, comparados con los de la red CAIDA. Luego, el coeficiente es mayor, en todos los puntos, para la red aleatoria.

Consecuentemente, se observa mayor variabilidad en la red homogénea, y eso se debe a que se puede descubrir mayor cantidad de información en una red inhomogénea (como la red de ruteadores obtenida por el proyecto CAIDA) ya que la mayoría de los enlaces coinciden con las rutas de los caminos más cortos. Esto se debe a la organización en racimos, tal como se ve en la figura 4.2.1(a). En la red homogénea siempre quedan enlaces sin descubrir que dependen más fuertemente de la ubicación de los nodos alfa que en el caso de la red heterogénea. Esto quiere decir que la red aleatoria es mucho más sensible a cuáles sean los nodos controlables en cuanto a lo que se puede aprender acerca de su estructura. Como en cada iteración los nodos alfa que se eligen son distintos, en la red con distribución de grados gaussiana se pone de manifiesto esa mayor variabilidad.

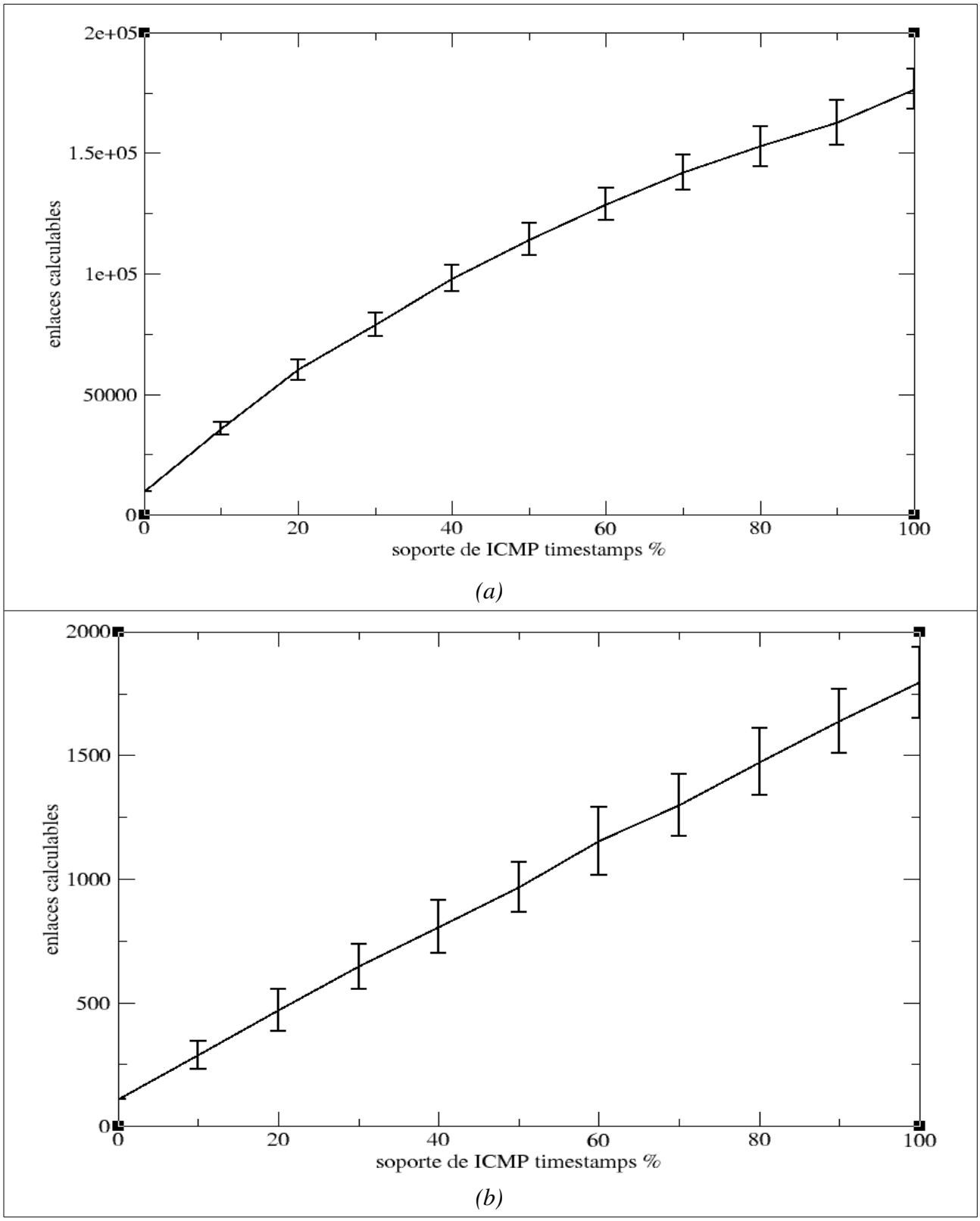


Figura 4.1.4. *Cantidad de enlaces virtuales, con líneas que indican el desvío estándar muestral en cada punto, para las redes CAIDA (a) y aleatoria (b). En la red aleatoria (b) se observa que el desvío es mayor.*

Forma de la curva

También resulta interesante analizar la forma de las curvas de enlaces virtuales. En la figura 4.1.4(b) se observa a simple vista una aparente linealidad en el caso de la red aleatoria. En el caso de la red CAIDA, representado en la figura 4.1.4(a), si bien no se observa una línea recta, la curvatura no aparenta ser muy pronunciada.

Esto motiva el siguiente análisis de regresión.

Para el caso de la red CAIDA, la función a ajustar es la descripta por la tabla A.3, con el nivel porcentual de soporte de *ICMP timestamps* (primera columna de la tabla A.3) como variable independiente y la cantidad de enlaces obtenidos con ese nivel de soporte de *timestamps*, relativa a la cantidad con nivel de soporte del 0% (tercera columna de la tabla A.3) como variable dependiente.

Al llevar a cabo un ajuste lineal, se obtiene la función:

$$y = 2{,}515 + 0{,}16399\,x \tag{4.1.4}$$

siendo el error cuadrático medio 0,6207..

Sin embargo, al llevar a cabo un ajuste cuadrático, se obtiene la función:

$$y = 1{,}2092 + 0{,}25104\,x - 0{,}00087051\,x^2 \tag{4.1.5}$$

siendo el error cuadrático medio 0,0296. Este valor es mucho menor que para el caso lineal, y evidencia un ajuste casi perfecto.

En la figura 4.1.5(a) se grafican la curva y su ajuste cuadrático.

Si bien el ajuste cuadrático arrojó mucha mayor precisión que el ajuste lineal, resulta interesante observar lo siguiente:

Los porcentajes altos de soporte de *timestamps* son irreales. Esto significa que el rango de porcentajes que se pueden hallar en la realidad es una pequeña parte del rango simulado. Como la curvatura es relativamente suave, el ajuste lineal no es del todo errado, lo cual, sumado a lo anterior, podría llevar a conjeturar que la aproximación mediante una recta dentro del rango real es muy buena. Esto se estudia en detalle más adelante.

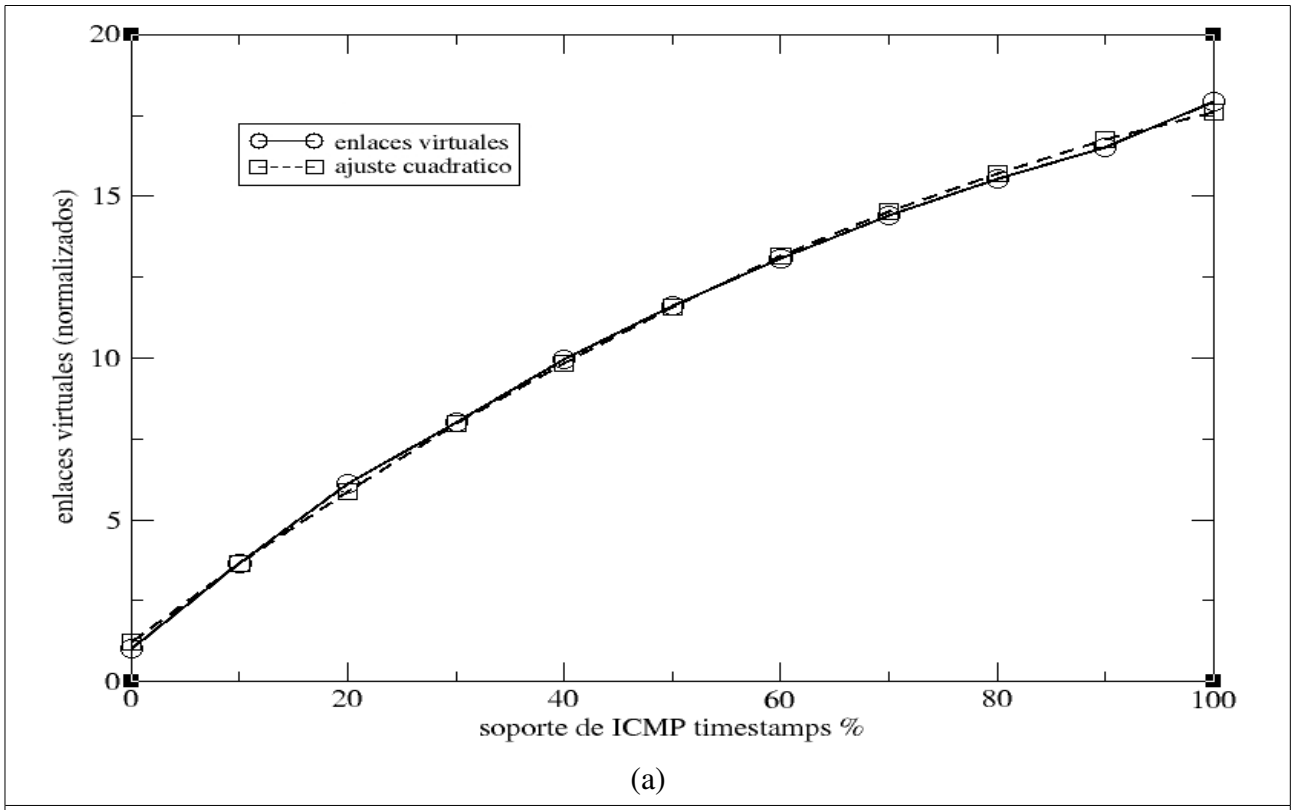
Para el caso de la red aleatoria, la función a ajustar es la descripta por la tabla A.4 (cantidad de enlaces virtuales en la red aleatoria para los distintos niveles de 0% a 100% de timestamps, relativa a la cantidad para 0% de timestamps).

Al llevar a cabo un ajuste lineal, se obtiene la función:

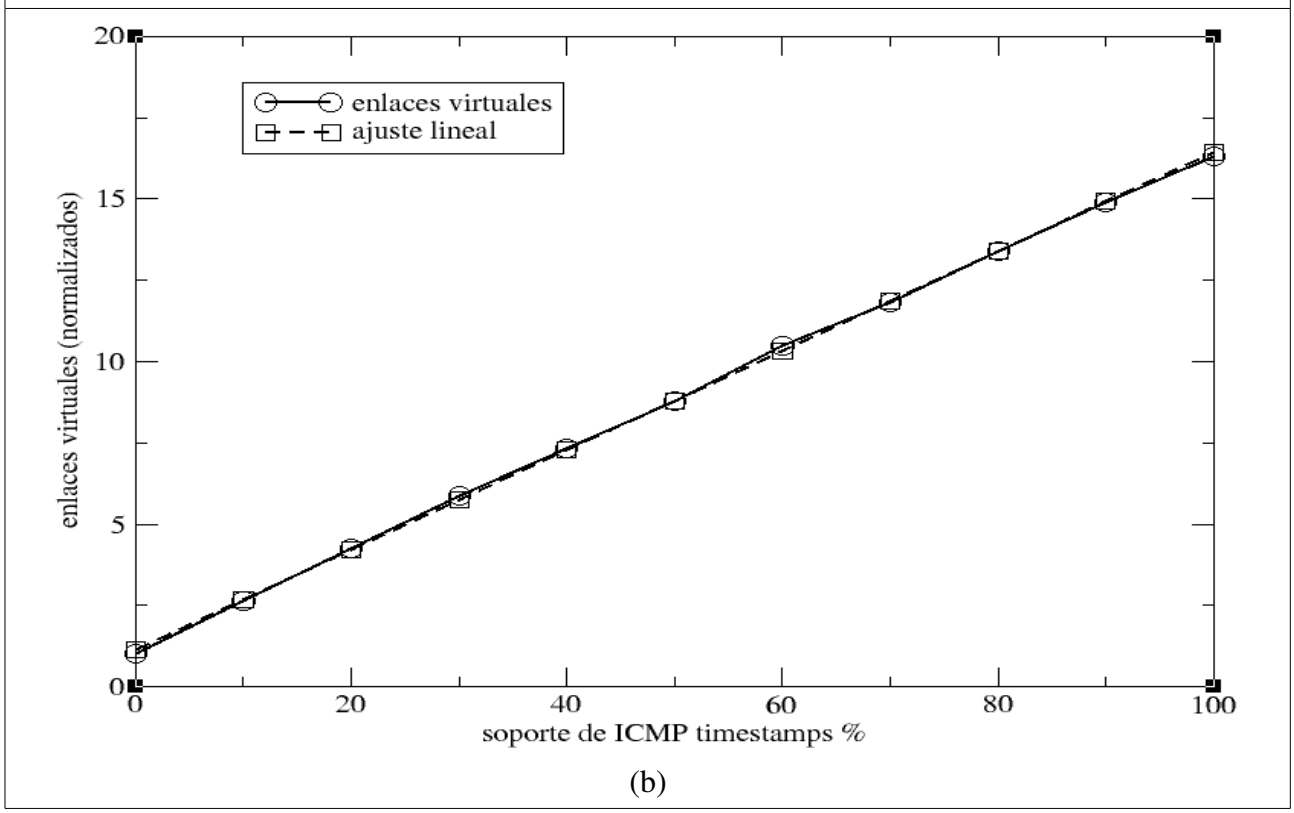
$$y = 1{,}1559 + 0{,}15294\,x \tag{4.1.6}$$

siendo el error cuadrático medio 0,0088 y el coeficiente de correlación lineal 0,9998. Esto significa que el ajuste lineal es excelente.

En la figura 4.1.5(b) se grafican la curva y su ajuste lineal.



(a)



(b)

Figura 4.1.5. *Ajustes de la curva de crecimiento de los enlaces calculables.*
(a) ajuste cuadrático de la curva de red CAIDA
(b) ajuste lineal de la curva de la red aleatoria

Cantidades de cada tipo de nodo

A continuación se presenta el análisis de las cantidades que se logran descubrir de cada tipo de nodo. Las variables cuantificadas son la cantidad de nodos alfa, la cantidad de nodos beta y la cantidad de nodos no-alfabetos, que en este caso, por analizarse la ejecución completa, consiste únicamente de nodos delta y épsilon, ya que los gama se procesan por completo y se desdoblan en betas y épsilons (ver algoritmo 3.1.1).

En la figura 4.1.6 se representan las cantidades de cada tipo de nodo, para las redes CAIDA (a) y aleatoria (b).

La cantidad de nodos alfa se dejó fija para cada red. En la red CAIDA se fijó en 100, y en la red aleatoria se fijó en 11. En ambas redes la densidad de alfas (es decir, la cantidad de nodos alfa que hay en el total de nodos de la red) quedó así fijada en 0,11%.

En ambas redes, la cantidad de nodos beta crece linealmente con el porcentaje de soporte de *ICMP timestamps*. Esto es natural, porque el mencionado soporte es la característica más importante en la clasificación de un nodo como beta o no-alfabeto. Llevando a cabo una regresión lineal para la variable en la red CAIDA, se obtiene que el coeficiente de correlación es 0,9999. Para la red aleatoria, se obtiene un coeficiente de correlación muy similar: 0,99989.

La cantidad de nodos no-alfabetos tiene comportamientos diferentes en las dos redes. En la red CAIDA, crece rápidamente a medida que se agregan los primeros nodos que soportan *timestamps*, pero luego decrece, a medida que los nodos beta van llenando los espacios entre los nodos alfa. En la red aleatoria, crece rápidamente y luego se desacelera, pero no llega a retroceder. El comportamiento es completamente distinto porque, para la misma concentración de alfas, la red aleatoria mantiene siempre una cantidad de nodos no-analfabetos significativa, ya que no se comparten enlaces como en el caso de la red inhomogénea.

Por último, es interesante observar que, cuando todos los nodos en una red soportan *ICMP timestamps*, la única razón por la cual pueden ser clasificados como épsilons es la carencia de alfa-simetría. En la red CAIDA, se observa que en el escenario en el que el 100% de los nodos soportan timestamps, más del 50% de los nodos clasificados son no-alfabetos. En similares condiciones, en la red aleatoria los nodos-no alfabetos superan el 80%. La mayor cantidad de nodos no-alfabetos encontrada en la red aleatoria se debe a que es mayor la cantidad de épsilons, y esto último se debe a que en una red aleatoria resulta más difícil encontrar una ruta simétrica, y por consiguiente la alfa-simetría.

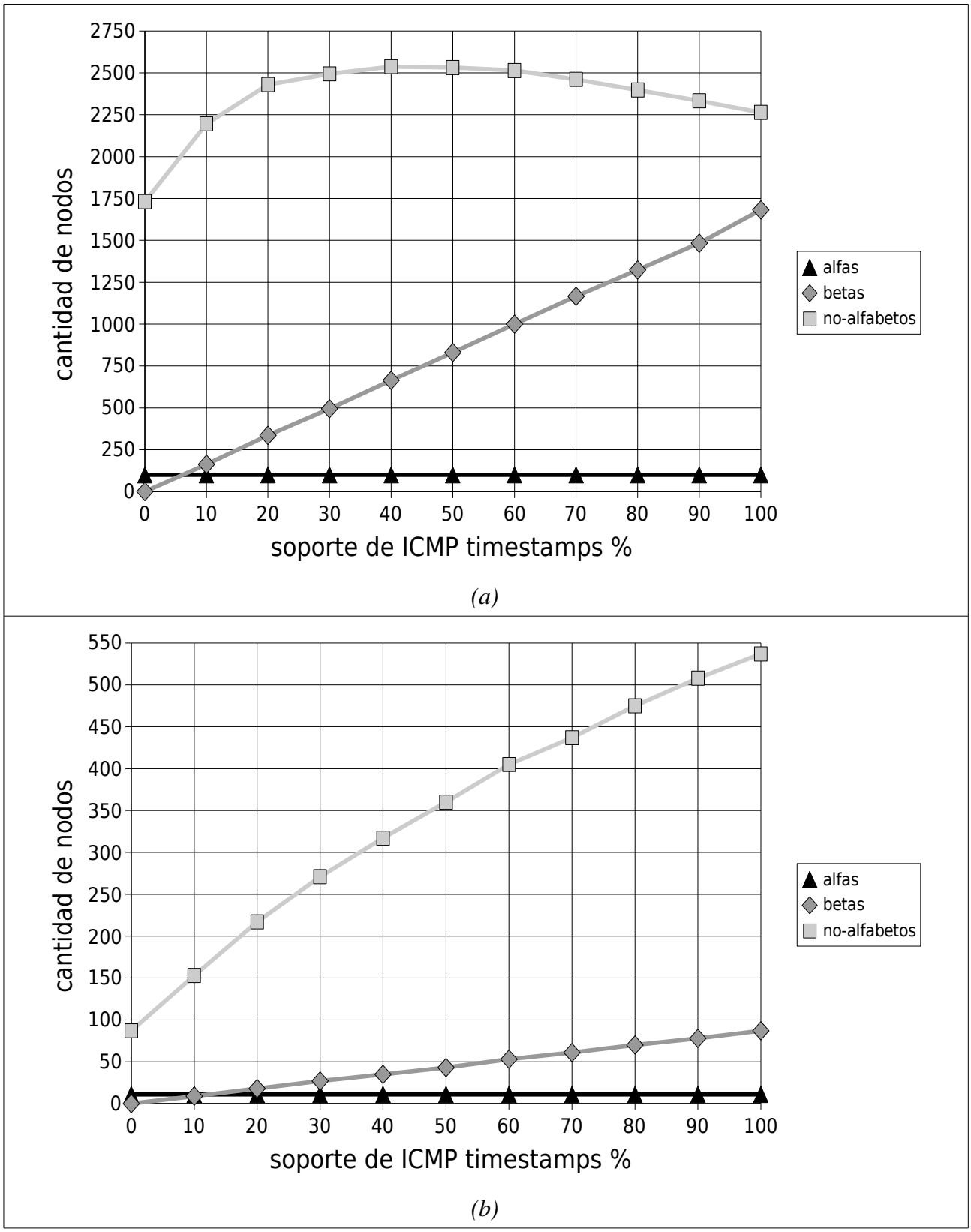


Figura 4.1.6. *Cantidades de los distintos tipos de nodos que se logran descubrir, para las redes CAIDA (a) y aleatoria (b).*

Cantidades de enlaces físicos y virtuales

Resulta interesante comparar el comportamiento de ambas redes en lo que respecta a las cantidades que presentan de enlaces físicos y virtuales que se llegan a descubrir para los diferentes niveles de soporte de *ICMP timestamps*.

En las tablas A.5 y A.6 se encuentran calculados los cocientes entre las cantidades de enlaces físicos y virtuales descubiertos para distintos niveles de soporte *ICMP* en las redes CAIDA y aleatoria respectivamente.

La representación gráfica aparece en la figura 4.1.7.

El mencionado cociente puede interpretarse como el número de enlaces físicos que contiene cada enlace virtual, en promedio. Por ejemplo, en la figura 4.1.7, para la red aleatoria,para 0% de soporte de *ICMP timestamps*, el cociente es aproximadamente 2, y esto significa que, que promedio, cada enlace virtual consta de 2 enlaces físicos. Cuanto mayor sea la relación, menor es la granularidad alcanzada.

En la red CAIDA, el cociente es menor que 1, y eso también evidencia, aunque en menor medida, pérdida de resolución. El resultado se debe a que muchos enlaces físicos aparecen en varios enlaces virtuales porque en esta situación tienden a quedar determinados enlaces virtuales correspondientes a las rutas completas entre todo par de alfas (es decir, un grafo completo entre los alfas), como concluye [Zhao05].

Idealmente, en el caso del conocimiento completo de la red, el cociente sería 1, porque todos los enlaces físicos se podrían calcular individualmente (es decir, habría exactamente 1 enlace físico por cada enlace virtual).

Por último, observando la curva correspondiente a la red CAIDA, se advierte que inicialmente se aleja del 1. Esto no representa menor cantidad de información, porque como se puede observar en la figura 4.1.3 (c), la cantidad de enlaces físicos descubiertos crece ampliamente: según la tabla A.1, al pasar de 0 a 10% de soporte de *ICMP timestamps,* la cantidad de enlaces físicos pasa de 8.068 a 27.043 (es decir, 3,35 veces), mientras que la cantidad de enlaces virtuales pasa de 9.848 a 35.810 (es decir, 3,64 veces). Esto muestra por qué se reduce levemente el cociente, pero también indica la gran cantidad de nuevos enlaces físicos descubiertos.

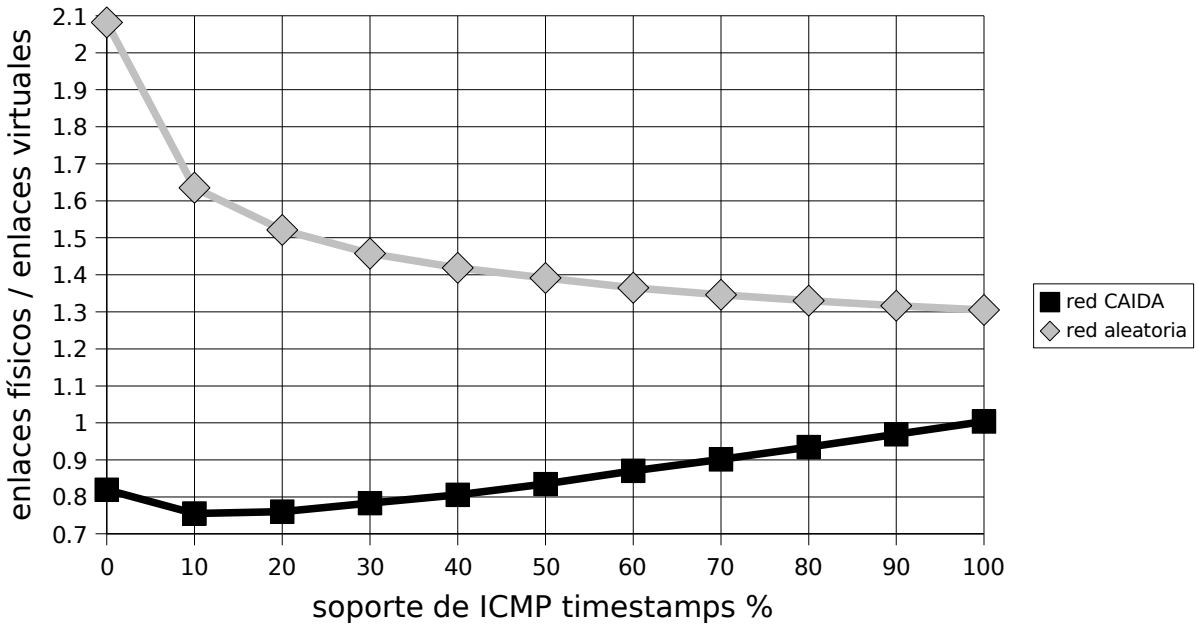


Figura 4.1.7. *Cociente entre las cantidades descubiertas de enlaces físicos y virtuales, para las redes CAIDA y aleatoria. Se observa que en la red CAIDA el cociente es menor a 1, y en la red aleatoria el cociente es mayor a 1.*

Valores pequeños del porcentaje de soporte de *ICMP timestamps*

Como se analizó más arriba, con tan sólo 10% de soporte de *ICMP timestamps* en la red la ganancia en cantidad de información es enorme. Además, en la realidad, el soporte de *ICMP timestamps* es relativamente bajo. Resulta entonces de interés estudiar el comportamiento de las variables más minuciosamente entre el 0% y el 10%.

Para estudiar esa región, se llevaron a cabo simulaciones para valores del parámetro entre el 0% y el 10% en incrementos de 1%. Los valores de las variables observadas se encuentran en las tablas A.7 (red CAIDA) y A.8 (red aleatoria).

La figura 4.1.8 representa las cantidades de enlaces virtuales dividido el total de enlaces físicos de cada red (esto último, hecho con el propósito facilitar la visualización de cuánto se puede aprender sobre cada red), para porcentajes pequeños de soporte de ICMP timestamps, y es análoga a la figura 4.1.3(d), que se ocupa del rango completo de porcentajes.

En la figura 4.1.8 se puede apreciar cómo crece el conocimiento sobre cada red, incluso en el escenario donde el nivel de soporte de *ICMP timestamps* es muy bajo. Se observa que la cantidad de información que la metodología descripta en este trabajo permite adquirir es significativamente mayor que con la técnica correspondiente al estado del arte actual.

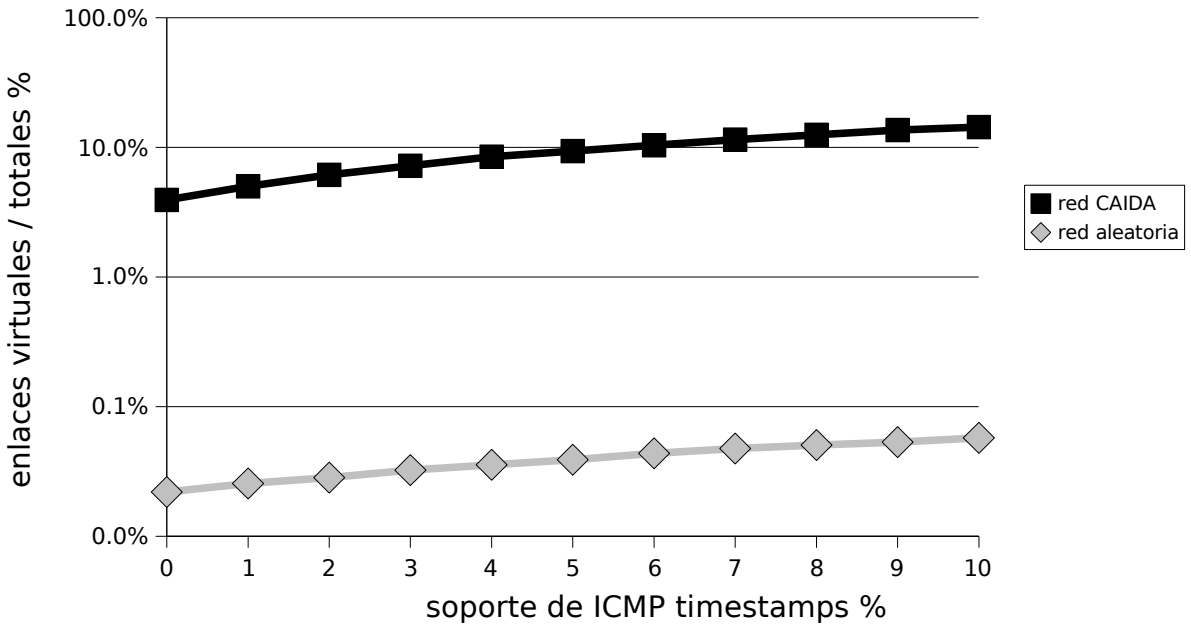


Figura 4.1.8. *Cantidad de enlaces virtuales para porcentajes pequeños de soporte de ICMP timestamps en las redes CAIDA y aleatoria.*

Cantidad de enlaces virtuales en el rango reducido

Se observa que aunque el nivel de soporte de *ICMP timestamps* sea tan bajo como el 1%, de todas maneras el incremento obtenido en la cantidad de enlaces virtuales (es decir, calculables individualmente) es considerable.

Por ejemplo, para la red CAIDA, en la segunda fila de la tabla A.7 se puede apreciar que con tan sólo un 1% de soporte de *ICMP timestamps* entre los nodos de la red, se pasa de 9.848 a 12.541 enlaces virtuales, es decir, se obtiene un 27% más de enlaces calculables individualmente que si no se aprovecha el soporte de *ICMP timestamps*.

Las tablas A.9 y A.10, que contienen los valores numéricos para la cantidad de enlaces calculables dado el porcentaje de nodos que soportan *ICMP timestamps*, en forma relativa a la cantidad de enlaces virtuales en el caso en el que no se aprovecha el soporte, y para el rango reducido, se calculan en base a las tablas A.7 y A.8 en forma análoga a como, para el rango completo, se calculan las tablas A.3 y A.4 en base a las tablas A.1 y A.2, como se explicó más arriba en esta misma sección.

En la figura 4.1.9 se trazan las curvas de enlaces virtuales por separado y en escala lineal en vez de logarítmica, para poder apreciar sus características individualmente. Las líneas verticales representan el desvío estándar en cada punto de la muestra de 100 iteraciones que se llevaron a cabo para ese punto. Las subfiguras (a) y (b) corresponden a las redes CAIDA y aleatoria respectivamente. Se sigue observando menor variabilidad en la red CAIDA, producto de la menor sensibilidad de esa red a la elección aleatoria de los nodos alfa en cada iteración, explicada más arriba.

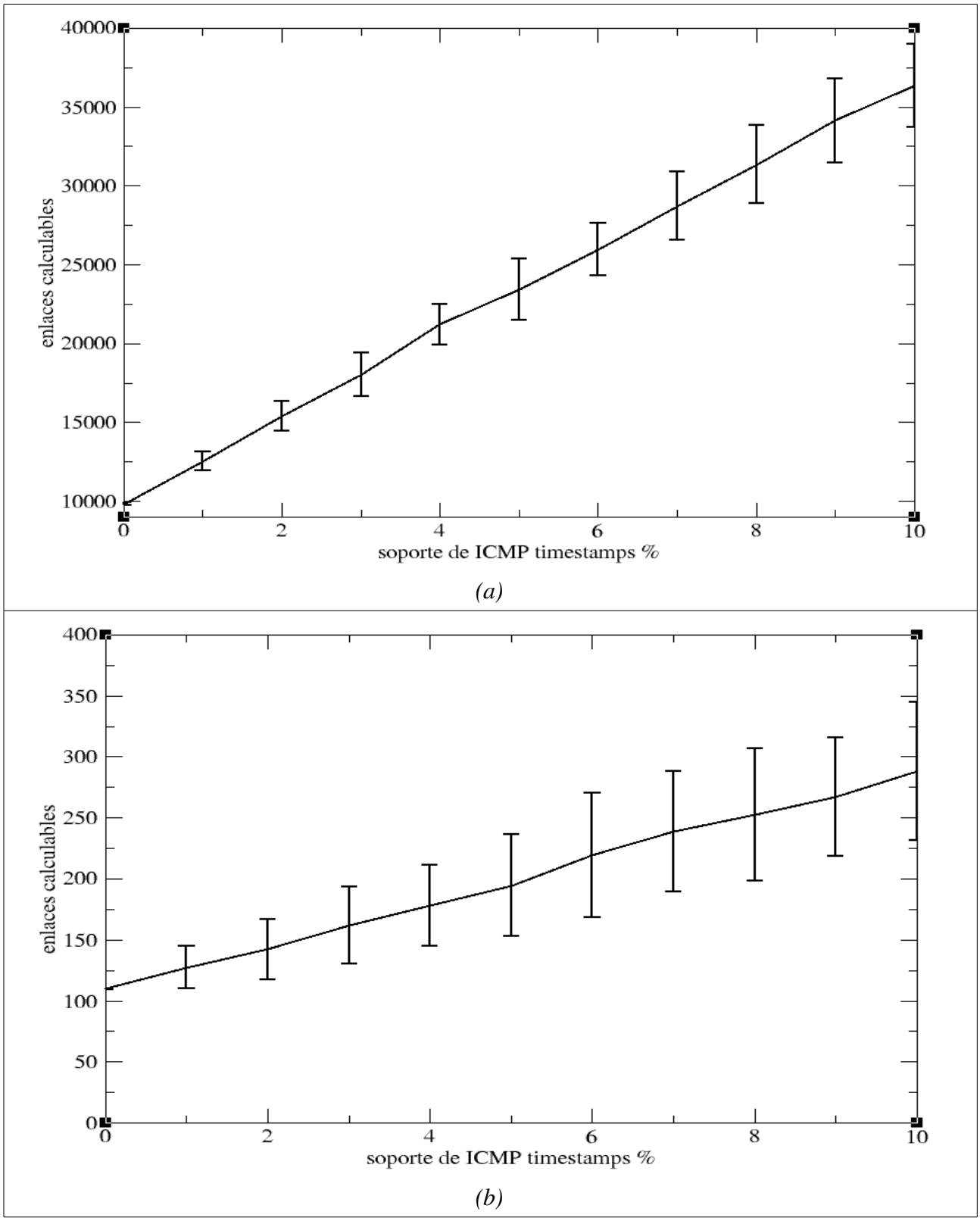


Figura 4.1.9. *Cantidad de enlaces virtuales en el rango reducido, con líneas que indican el desvío estándar muestral en cada punto, para las redes CAIDA (a) y aleatoria (b).*
En la red aleatoria (b) se observa que el desvío es mayor.

Forma de la curva en el rango reducido

En el rango completo, la curva de enlaces virtuales para la red CAIDA resulta cuadrática, aunque con una curvatura no demasiado pronunciada (coeficiente de correlación lineal 0,9887). Para el rango reducido, aparenta ser casi exactamente lineal.

Llevando a cabo un ajuste lineal, se obtiene la siguiente recta:

$$y = 1{,}0264 + 0{,}26836\,x \tag{4.1.7}$$

siendo el error cuadrático medio 0,0107 y el coeficiente de correlación lineal 0,9993. Esto significa que el ajuste lineal es casi perfecto. En la figura 4.1.10(a) se puede apreciar la superposición de la función y su ajuste lineal.

La observación interesante es que la curva presenta linealidad para el rango reducido, que justamente es el más importante (debido a que está constituido por niveles de soporte de *timestamps* verosímiles).

Por su parte, la red aleatoria conserva en el rango reducido la linealidad que ya se observó en el rango completo.

Al llevar a cabo un ajuste lineal, se obtiene la función:

$$y = 0{,}98455 + 0{,}16327\,x \tag{4.1.8}$$

siendo el error cuadrático medio 0,0004 y el coeficiente de correlación lineal 0,99923. Esto significa que el ajuste lineal es excelente.

En la figura 4.1.10(b) se grafican la curva y su ajuste lineal.

Comparando las dos rectas, se puede observar que la pendiente de la recta correspondiente a la red CAIDA es mucho mayor que la correspondiente a la red aleatoria. Esto significa que, a medida que se incrementa el porcentaje de soporte de *ICMP timestamps*, en la red CAIDA se incrementa más rápidamente la cantidad de información que se puede obtener. Esto se debe a que aumentar el nivel de soporte de *timestamps* equivale a agregar betas, y como se detalló más arriba, en la red CAIDA los nodos se comparten más entre las rutas, lo cual permite que un nuevo beta arroje luz sobre una mayor proporción de las rutas existentes que en la red aleatoria.

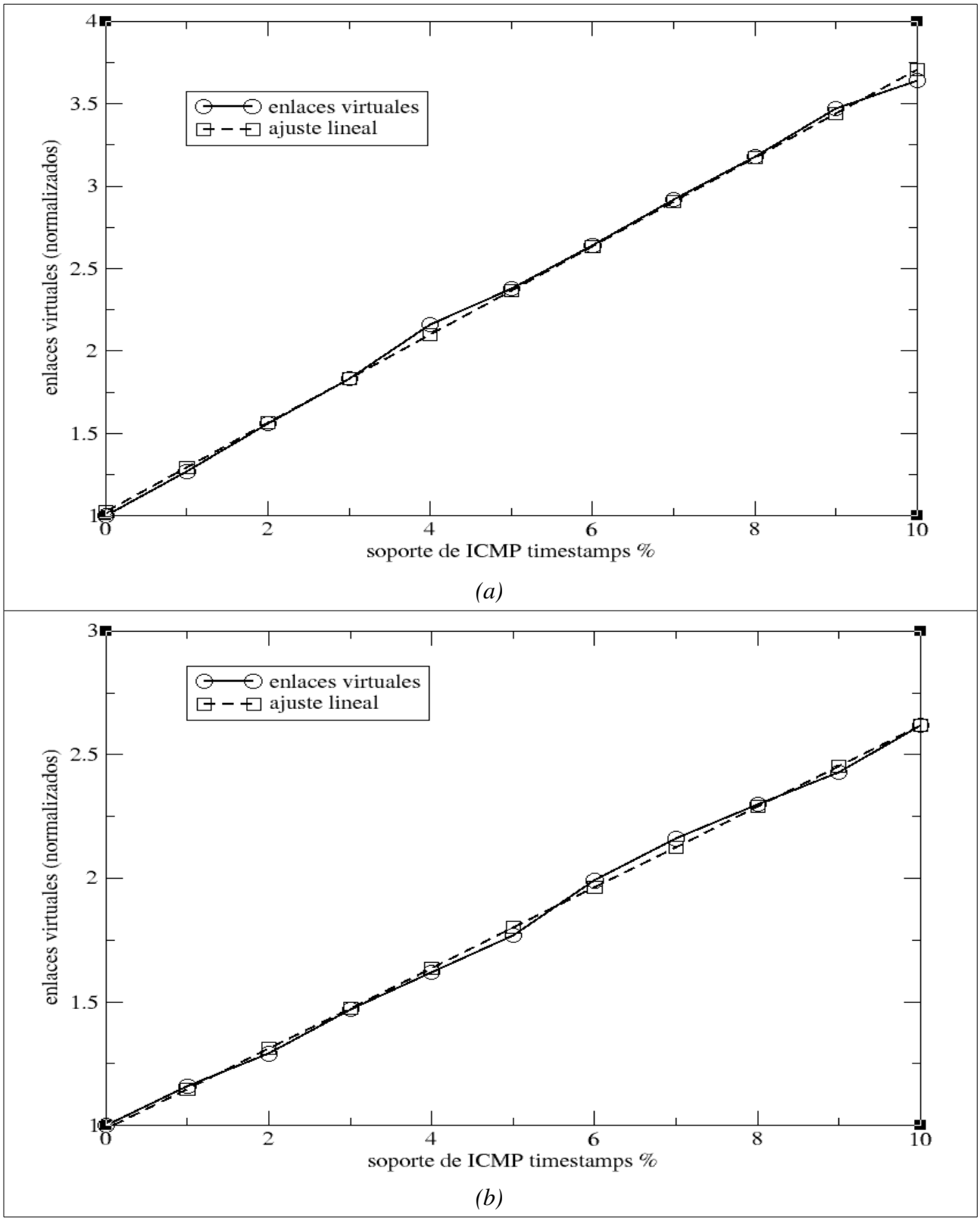


Figura 4.1.10. *Ajustes de la curva de crecimiento de los enlaces calculables en el rango reducido.*
(a) ajuste cuadrático de la curva de red CAIDA
(b) ajuste lineal de la curva de la red aleatoria

Cantidades de enlaces físicos y virtuales en el rango reducido

No se observan nuevos detalles en esta área, ya que las curvas presentan las mismas propiedades que en el rango completo.

Los datos corresponden a las tablas A.11 y A.12 (para las redes CAIDA y aleatoria respectivamente) y se grafican en la figura 4.1.11.

El cociente para la red aleatoria se acerca al 1 (caso ideal) y en la red CAIDA se aleja levemente, lo cual no es malo, considerando el enorme crecimiento en la cantidad de enlaces físicos descubiertos, que se puede visualizar en la figura 4.1.3(c) y leer en la tablas A.7 (red CAIDA) y A.8 (red aleatoria).

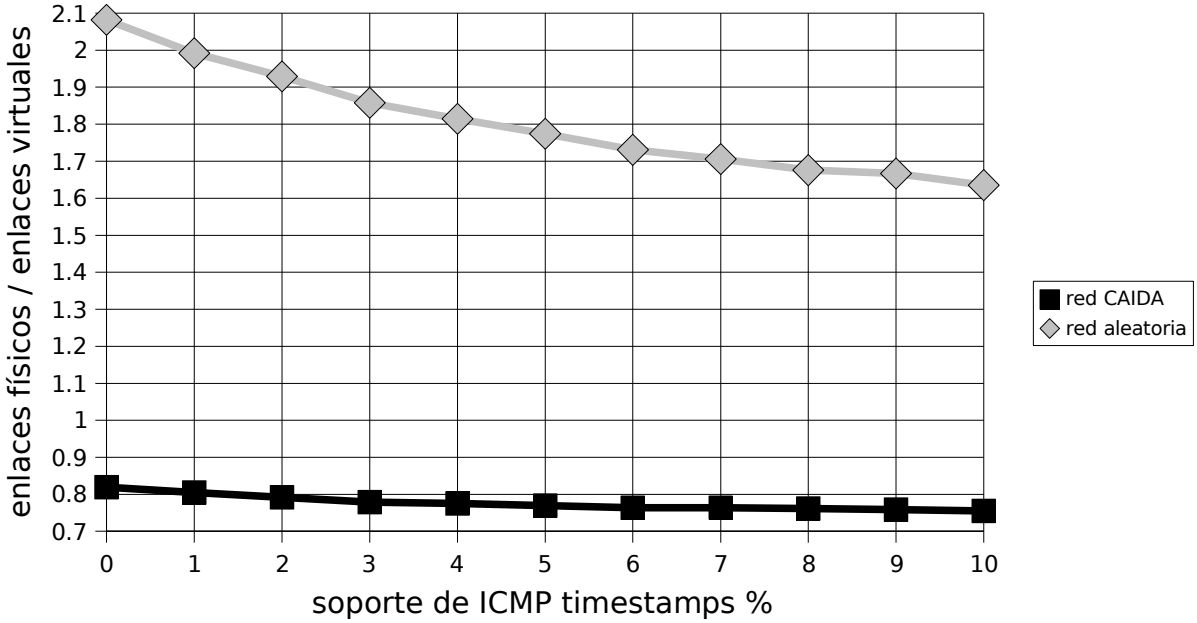


Figura 4.1.11. *Cociente entre las cantidades descubiertas de enlaces físicos y virtuales, para el rango reducido, en las redes CAIDA y aleatoria.*

4.2. Procesamiento de datos reales

Experimentar con datos del proyecto DIMES resulta interesante por varios motivos. El primer motivo es que se trata de mediciones reales en Internet, lo cual permite observar si la teoría funciona también en este caso.

El segundo motivo es que se partir de los valores de las mediciones es básicamente el proceso inverso al efectuado hasta el momento: en esta sección se determinan los nodos a partir de las mediciones.

Un tercer motivo consiste en estudiar el efecto de considerar alfabetos[¶] adicionales en la red, es decir, considerar como alfabetos a nodos que no necesariamente lo son.

Los datos utilizados

Se utilizaron datos del proyecto DIMES. Se partió de un conjunto de mediciones en Internet, con las siguientes características:

- Se midieron 200.000 rutas.
- Cada ruta fue medida 4 veces.
- Por cada ruta se cuenta con una lista de los nodos que la componen, y por cada nodo se cuenta con la siguiente información:
  - Número de orden (por ejemplo, el primer nodo de la ruta recibe el número 1).
  - Dirección IP
  - Tiempo promedio (entre las 4 mediciones) desde el inicio hasta el nodo.

En la tabla 4.2.1 se puede ver un ejemplo de una ruta medida.

La demora observada para una ruta fue calculada como el tiempo hasta el último nodo, menos el tiempo hasta el primer nodo. Si se considera que la ruta en su totalidad es una medición de punta a punta, entonces la medición de la ruta de la figura 4.2.1 es de 119 *ms*. Sin embargo, se experimentó con la posibilidad de desglosar una ruta medida en rutas más pequeñas. Esto se explica más adelante.

Número de ruta	Número de orden	Dirección IP	Tiempo promedio (*ms*)
21685	1	66.185.132.33	0
21685	2	66.185.138.64	27
21685	3	66.185.152.182	38
21685	4	66.185.147.193	73
21685	5	209.58.47.1	81
21685	6	216.6.84.13	99
21685	7	58.71.0.146	78
21685	8	203.111.239.54	108
21685	9	203.111.239.201	119

Tabla 4.2.1. *Ejemplo de ruta medida por el proyecto DIMES.*

¶ Los alfabetos, según se explica en la sección 3.1, son los nodos capaces de ser extremos de una medición.

El experimento

El análisis a partir de las mediciones de rutas es en cierta forma el proceso inverso a la experimentación descripta en las secciones anteriores de este capítulo. En esta sección, no se parte de los nodos para determinar las rutas y luego medirlas, sino que se parte de rutas medidas y se llega a los nodos.

Como se detalló en el capítulo 3, los alfabetos son extremos de las mediciones, y a la hora de determinar cuáles rutas son medibles es indispensable desglosarlos en alfas y betas. Sin embargo, como en el escenario planteado en esta sección no es necesario determinar cuáles rutas son medibles, porque justamente las rutas ya fueron medidas, los alfas y betas son considerados colectivamente alfabetos.

El primer paso del procedimiento consistió en recorrer el conjunto de mediciones y marcar como alfabetos a todos los nodos que aparecían como extremo de alguna medición.

Obtenida la lista de alfabetos, se volvió a recorrer el conjunto de mediciones, y cuando una ruta medida presentaba un nodo intermedio alfabeto, se tomó como dos mediciones. Es decir, si en la ruta:

$$n_1 \rightarrow n_2 \rightarrow \ldots \rightarrow n_{i-1} \rightarrow n_i \rightarrow n_{i+1} \rightarrow \ldots \rightarrow n_{j-1} \rightarrow n_j$$

el nodo n_i había sido marcado como alfabeto en la primera pasada, entonces la medición se tomó como las siguientes dos mediciones por separado:

- $n_1 \rightarrow n_2 \rightarrow \ldots \rightarrow n_{i-1} \rightarrow n_i$
- $n_i \rightarrow n_{i+1} \rightarrow \ldots \rightarrow n_{j-1} \rightarrow n_j$

Este método se aplicó en cada ruta medida hasta que ninguna medición quedó con alfabetos intermedios.

Cuando una ruta medida es desglosada en rutas más pequeñas, la demora observada se calcula como en el siguiente ejemplo: si el nodo 209.58.47.1 fuera considerado alfabeto (extremo de medición), entonces la medición de la tabla 4.2.1 podría traducirse en dos mediciones por separado:

- La ruta 66.185.132.33 – 209.58.47.1, con demora observada 81 *ms*.
- La ruta 209.58.47.1 – 203.111.239.201, con demora observada 38 *ms*.

Se descartaron las mediciones que presentaban inconsistencias, de acuerdo a los siguientes criterios:

- Se consideraron únicamente rutas acíclicas. Consecuentemente, se descartaron las mediciones en las que una misma dirección IP aparecía más de una vez.
- Se descartaron las mediciones que arrojaban valores negativos. Por ejemplo, en la ruta

de la tabla 4.2.1, si los nodos 209.58.47.1 y 58.71.0.146 fueran considerados alfabetos, entonces la demora observada sería -3 ms. Luego, la ruta entre esos dos nodos es descartada. El proceso de subdivisión de las rutas se explica en detalle más adelante.

- Se descartaron las mediciones que arrojaban valores excesivamente grandes y apartados de la realidad, que evidenciaban errores. Por ejemplo, si una ruta arroja una demora observada de 293284 ms, es descartada.

A continuación, se aplicó el algoritmo 3.2.1 para determinar los enlaces virtuales, y se verificó que los mismos eran correctos.

Seguidamente se estudió el efecto de convertir en alfabetos a nodos que no aparecían como extremo de ninguna medición. Se hicieron 100 iteraciones convirtiendo en alfabetos al 10% de los nodos a los que les correspondía ser no-alfabetos. Luego se repitió para 20%, 30%, y así hasta llegar al 100% en intervalos de 10%.

Los resultados se presentan en la tabla 13 del anexo A.

En la figura 4.2.1 se grafican los resultados de la tabla A.13. Se puede apreciar que, a medida que se agregan alfabetos adicionales, se incrementa la cantidad de enlaces calculables.

Sin embargo, la cantidad de enlaces físicos decrece. Esto se debe a que, de acuerdo a la política establecida, cuando hay un alfabeto en el medio de una medición, la misma se toma como 2 mediciones por separado. Al analizar las mediciones reales en trozos más pequeños, se detectan más inconsistencias, y eso lleva a descartar mediciones. Consecuentemente, la cantidad de enlaces físicos abarcados en el conjunto de mediciones decrece.

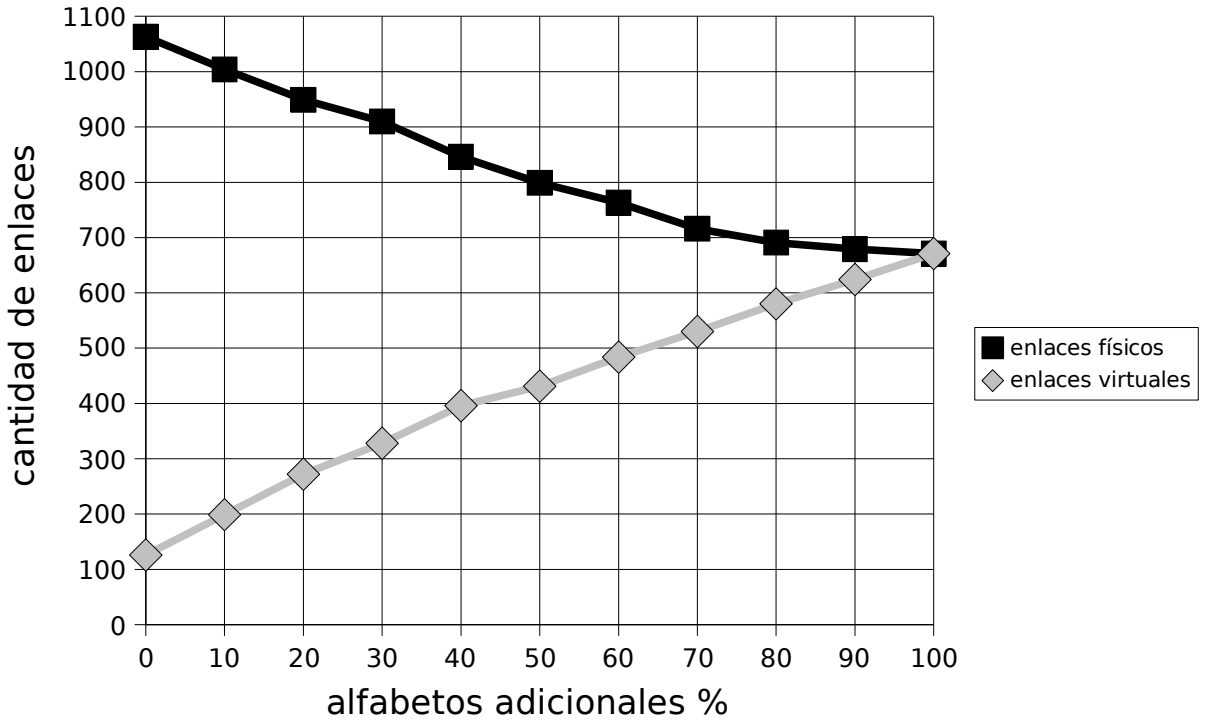


Figura 4.2.1. *Enlaces físicos y virtuales al agregar alfabetos adicionales al conjunto de mediciones de DIMES.*

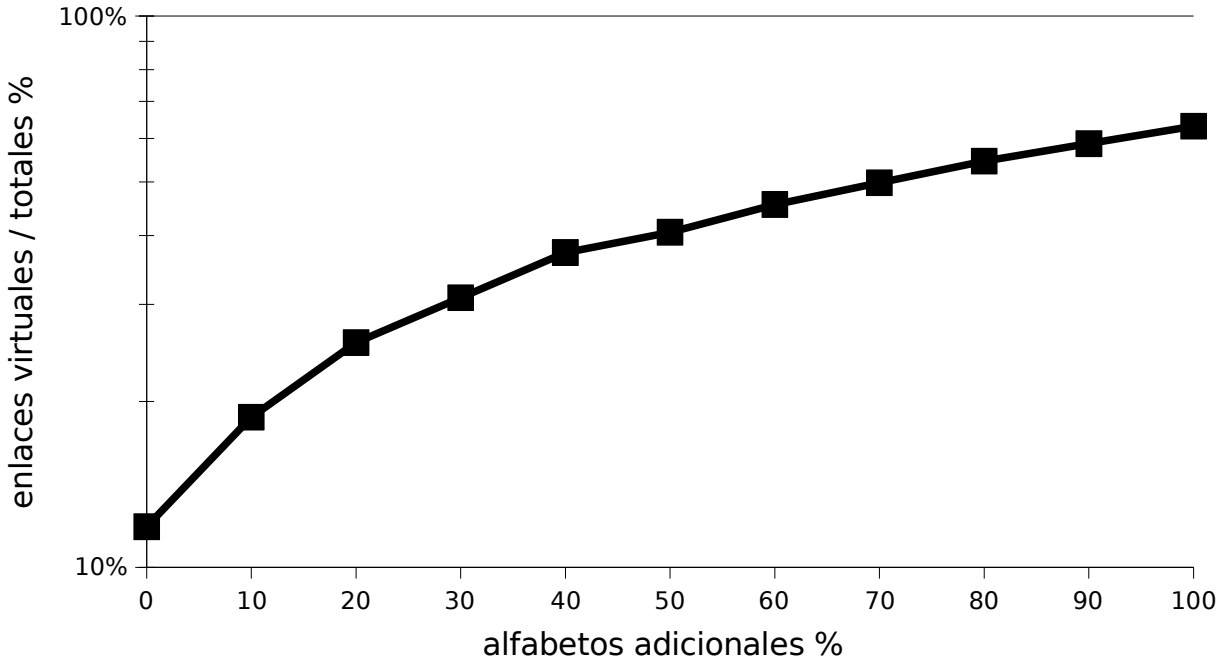


Figura 4.2.2. *Cantidad de enlaces virtuales en los datos de DIMES, relativa a la cantidad total de enlaces físicos involucrados. Esta figura es análoga a la figura 4.1.3(d).*

A la derecha de la figura 4.2.1 se puede observar que cuando todos los nodos son alfabetos, los enlaces físicos cuyas mediciones no resultaron inconsistentes se pueden calcular individualmente.

Para comparar el crecimiento de la cantidad de enlaces virtuales en esta red con los de las redes de la sección 4.1, se trazó la figura 4.2.2, análoga a la 4.1.3(d).

Comparando las figuras 4.2.2 y 4.1.3(d) se observa que la curva obtenida con los datos de DIMES se asemeja a la curva obtenida para la red CAIDA, pero crece más lentamente porque, debido a las inconsistencias, muchos enlaces físicos son eliminados.

A continuación se analiza la forma de la curva.

Al llevar a cabo un ajuste cuadrático, se obtiene la función:

$$y = 1{,}0567 + 0{,}054404\,x - 0{,}00012249\,x^2 \qquad (4.2.1)$$

siendo el error cuadrático medio 0,0027. Este valor indica que el ajuste cuadrático es excelente.

En la figura 4.2.3 se grafican la curva y su ajuste cuadrático. Es análoga a la figura 4.1.5(a) de la red CAIDA.

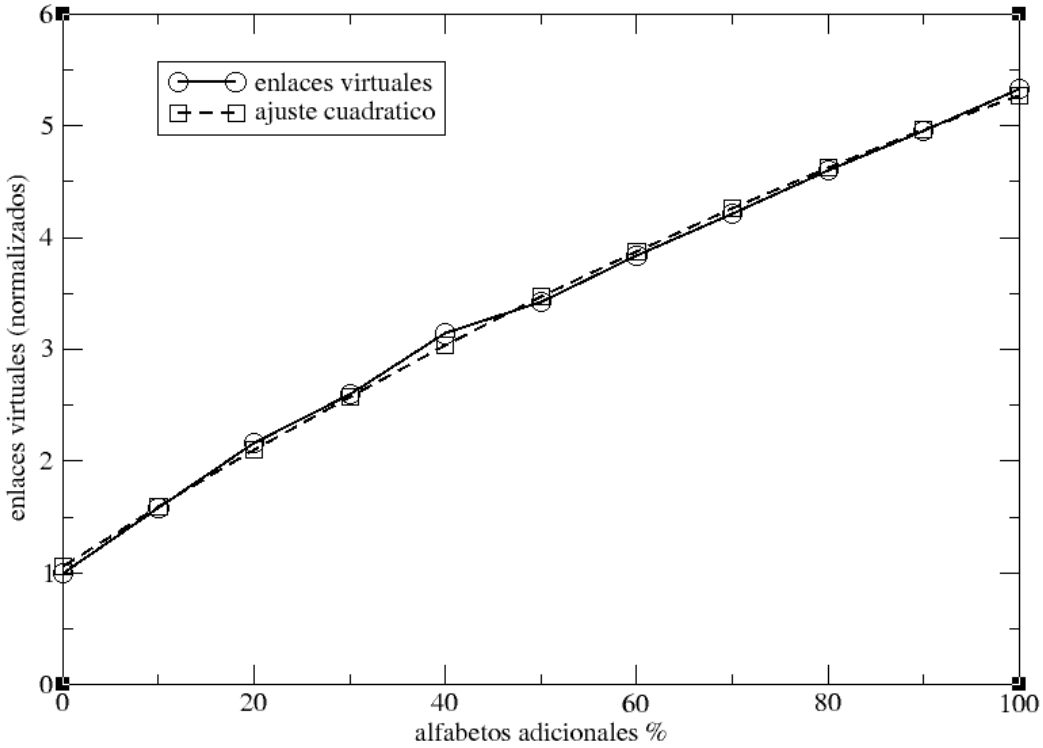


Figura 4.2.3. *Ajuste cuadrático de la curva de crecimiento de los enlaces virtuales.*

Cabe destacar que en ambas redes la curva tiene forma cuadrática.

En las dos ecuaciones (4.1.5 y 4.2.1) los coeficientes son del mismo orden, pero los de la red CAIDA son un poco mayores.

El coeficiente lineal de CAIDA es 4.6 veces el de la red aleatoria. Esto es porque en la red CAIDA se aprende una cantidad mayor de información con respecto al total.

El coeficiente cuadrático de CAIDA es 7.1 veces el de la red aleatoria. La curva trazada a partir de los datos de DIMES es más chata que la curva de la red CAIDA.

En la figura 4.2.4 se puede apreciar la evolución del cociente entre la cantidad de enlaces virtuales y la cantidad de enlaces físicos. La proporción aumenta a medida que se agregan alfabetos adicionales, hasta que llega al 100% cuando el 100% de los nodos se consideran alfabetos. Esa es la situación en la cual todos los enlaces físicos se pueden calcular individualmente y entonces la red se conoce en su totalidad.

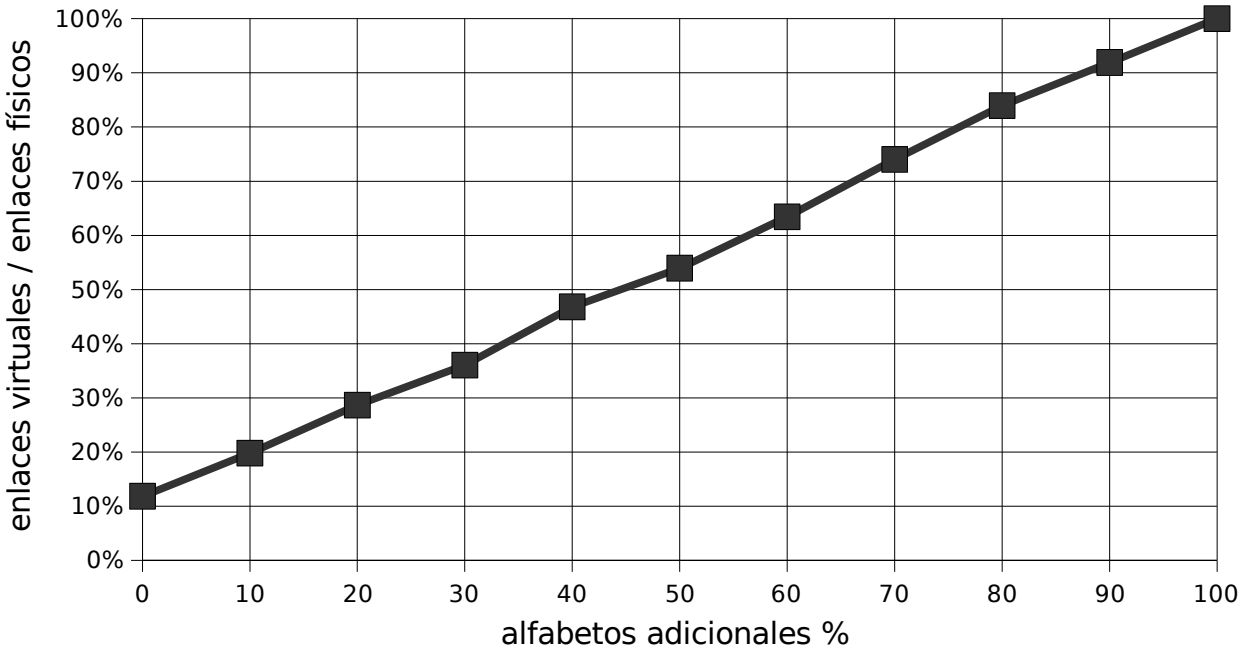


Figura 4.2.4. Relación entre la cantidad de enlaces virtuales y físicos *al agregar alfabetos adicionales al conjunto de mediciones de DIMES.*

4.3. Conclusiones

Los primeros capítulos de este trabajo señalan, desde la teoría, que se puede utilizar el soporte de *ICMP timestamps* de los nodos de una red para incrementar el conocimiento que se puede obtener de ella. Asimismo, explican el procedimiento para extraer la mayor cantidad posible de información.

Las simulaciones presentadas en este capítulo permitieron comprobar el beneficio de aprovechar los nodos que soportan *ICMP timestamps* (aunque éstos sean pocos) obteniéndose mucha más información a nivel de los enlaces que si no se los tiene en cuenta.

La experimentación efectuada involucró a topologías tanto homogéneas (como la red aleatoria) como no homogéneas (como la red CAIDA a nivel de ruteadores) y a datos reales (como las mediciones del proyecto DIMES y la topología de la red CAIDA) como sintéticos (como la red aleatoria).

Las simulaciones en las redes CAIDA y aleatoria arrojaron similitudes y diferencias, que se resumen y comentan en la tabla 4.3.1. Las diferencias encontradas coinciden con lo enunciado en [Dallasta06].

El experimento con los datos del proyecto DIMES permitió comprobar, por un lado, que la metodología es aplicable a mediciones reales, y por el otro, que cuando la totalidad de los nodos de la red son alfabetos, todos los enlaces de la red pueden descubrirse y calcularse individualmente.

Los experimentos con datos reales (CAIDA y DIMES) coincidieron en la forma (cuadrática) y el orden de la cantidad de enlaces virtuales que se logran descubrir relativa al total de enlaces físicos de cada red. En el caso de DIMES se puede ver una pequeña desaceleración, que se debe a las inconsistencias inherentes al proceso de real de toma de datos.

También se verificó en este capítulo que los enlaces virtuales calculables y mínimos obtenidos con los algoritmos actuales coinciden con los determinados por los algoritmos nuevos, que tienen una complejidad de cálculo mucho menor.

Red	**CAIDA**	**aleatoria**	
Topología	jerárquica, en racimos, heterogénea	al azar, homogénea	
Distribución de los grados de los nodos	de potencias, cola pesada	normal, cola liviana	
Densidad de alfas utilizada	0.11%	0.11%	en las dos redes se utilizó la misma proporción de alfas en relación al total de nodos de cada red
Cantidad de información obtenida	alta	baja	aumenta cuando los nodos y enlaces se comparten entre muchas rutas
Variabilidad en la cantidad de información	baja	alta	aumenta cuando la cantidad de información que se puede obtener depende fuertemente de la ubicación de los alfas
Curva de crecimiento de los enlaces virtuales (0 a 100% de ICMP)	cuadrática (ec. 4.1.5)	lineal (ec. 4.1.6)	la curva de la red CAIDA no es muy pronunciada
Curva de crecimiento de los enlaces virtuales (0 a 10% de ICMP)	lineal, con pendiente grande (ec. 4.1.7)	lineal, con pendiente chica (ec. 4.1.8)	la pendiente grande implica que la cantidad de información obtenida crece más rápidamente, y eso es característico de las redes con cola pesada
Cantidad de nodos beta	lineal	lineal	porque el soporte de *timestamps* es la característica determinante
Cantidad de enlaces físicos descubiertos	lineal	lineal	porque la cantidad de nodos beta crece linealmente

Tabla 4.3.1. *Similitudes y diferencias entre las redes CAIDA y aleatoria.*

5. CONCLUSIONES

En este breve capítulo se resume el trabajo realizado, se destacan las principales contribuciones y se describen líneas posteriores de investigación.

5.1 Contribuciones de esta tesis

Primeramente se llevó a cabo un estudio del estado del arte. Se analizaron artículos de las principales líneas de investigación de la tomografía de una red, así como también otros relacionados. Además, se examinaron proyectos dedicados al estudio de la topología de Internet, como CAIDA y DIMES.

Del relevamiento surgió que las técnicas más avanzadas para llevar a cabo la tomografía de una red no aprovechan el soporte de *ICMP timestamps*. Esto no es casual, sino que se debe a una decisión fundada en el argumento de que la mayoría de los ruteadores en Internet no proveen soporte de *ICMP timestamps*, y que no es una buena idea depender de algo que no puede garantizarse.

El presente trabajo sostiene que el hecho de que una determinada característica de infraestructura no esté presente en todos los nodos de una red no es motivo suficiente como para descartar la posibilidad de aprovecharla. Surgieron entonces preguntas como: "¿Es posible utilizar el soporte de *ICMP timestamps* de los pocos nodos que lo soportan, y obtener de ese modo una cantidad de información mucho mayor que si directamente se desestimara la posibilidad de tenerlos en cuenta? Y, si es posible, ¿cómo aprovecharlo?"

Uno de los aportes de esta tesis fue responder esas preguntas. Se ha creado un nuevo procedimiento que, efectivamente, posibilita aprovechar el soporte de *ICMP timestamps* de los nodos que lo tengan, y aunque estos sean pocos, permite obtener un mayor volumen de información sobre la red.

La nueva técnica comienza con los nodos controlables de la red (es decir, aquellos nodos a los que se tiene acceso), y se basa en el conocido hecho de que, cuando un nodo soporta *ICMP timestamps*, es posible sincronizarlo a los nodos controlados. Si es posible sincronizar dos nodos, entonces es posible medir la demora de la ruta entre ellos en ambos sentidos. Y es importante poder medir la demora en ambos sentidos porque, por ejemplo, en momentos de congestión, un paquete

puede no tardar el mismo tiempo yendo desde el nodo *A* hasta el nodo *B* que en el sentido contrario.

Hasta ahora, la medición en ambos sentidos sólo se había efectuado entre nodos controlables. Utilizar a los nodos no controlables (pero sincronizables) como extremos de medición complica el análisis, porque además de las rutas entre nodos controlables, aparecen rutas entre pares de nodos en los cuales uno es controlable y el otro no.

Para atacar el problema, se desarrolló una clasificación de nodos, según la cual los nodos controlables son alfas, y los no controlables se subdividen en betas (sincronizables), épsilons (no sincronizables) y las categorías intermedias (gama y delta) para los nodos cuya sincronizabilidad todavía no fue determinada por el sistema. Las rutas medibles son las que van entre dos alfas o entre un alfa y un beta.

El siguiente paso consiste en determinar los *MILS*es del sistema para minimizar el sesgo de la estimación. El *MILS* es concepto del estado del arte actual que se refiere a un enlace virtual (es decir, un enlace compuesto por la concatenación de 1 ó más enlaces físicos) con dos características adicionales: es identificable (la demora en el enlace se puede calcular individualmente) y es mínimo (si se le quitara cualquiera de los enlaces físicos que lo componen, dejaría de ser identificable).

El mejor algoritmo hasta el momento para determinar los *MILS*es consistía en tomar cada ruta y por cada subconjunto posible de enlaces físicos consecutivos verificar mediante la proyección sobre el subespacio generado por las rutas si el enlace era identificable. La complejidad de ese algoritmo es $O(rkl^2)$, donde r es la cantidad total de rutas, k es la cantidad de rutas a medir, y l es la longitud media de una ruta.

El nuevo algoritmo propuesto consiste simplemente en tomar cada ruta y partirla de acuerdo a los nodos alfabetos. La complejidad resulta mucho menor: $O(rl)$.

Se implementaron los algoritmos de varias líneas de investigación actuales, así como también los propuestos en este trabajo. La lista completa de algoritmos implementados se encuentra en el apéndice B. Los principales son:

- Algoritmos de selección de rutas a monitorear y cálculo de MILSes, de Chen, Zhao et al. [Chen04, Zhao06].
- Algoritmos de selección de rutas a monitorear y predicción estadística, de Chua et al. [Chua05].
- Algoritmos de clasificación de nodos, creación de la matriz de ruteo y determinación de los enlaces identificables, presentados en esta tesis.

Se llevaron a cabo simulaciones con topologías homogéneas (como la red aleatoria), topologías no homogéneas (como la red CAIDA) y con datos reales (como las mediciones del

proyecto DIMES y la topología de la red CAIDA) y sintéticos (como la red aleatoria). Al trabajar con los datos de DIMES, también se estudió el proceso inverso, de clasificar los nodos a partir de las rutas medidas, en vez de determinar las rutas medidas en base a los nodos clasificados.

Los experimentos comprobaron que los resultados arrojados por los métodos actuales y nuevos son iguales. Cabe destacar que la complejidad de los nuevos algoritmos es mucho menor.

Los experimentos también permitieron comprobar que efectivamente la información que se puede obtener aprovechando el soporte de *ICMP timestamps* es significativamente mayor, aún cuando muy pocos nodos en la red tengan el soporte.

Los resultados experimentales sobre la cantidad de información que se puede extraer de cada red señalan que las cantidades de nodos y enlaces relevados en una red con distribución de grados de cola pesada (como CAIDA) es mayor que en la topología aleatoria, lo cual coincide con lo observado en [Dallasta06].

Se observó que algunas variables crecen linealmente con el nivel de soporte de *ICMP timestamps* en la red, como la cantidad de nodos beta, la cantidad de enlaces físicos descubiertos y la cantidad de enlaces virtuales (aunque esta última sólo es completamente lineal para porcentajes pequeños de soporte de *timestamps*).

La pendiente de la recta es mayor en el caso de la red CAIDA, lo cual significa que la cantidad de información que se puede adquirir crece más rápidamente, y se debe a que la red CAIDA es una red de cola pesada. Por otra parte, conocer las pendientes permite tener una idea de qué esperar.

El experimento con los datos del proyecto DIMES permitió comprobar que:

- la metodología es aplicable a mediciones reales
- cuando la totalidad de los nodos de la red son alfabetos, todos los enlaces de la red pueden descubrirse y calcularse individualmente.

Cabe destacar que los experimentos con datos reales (CAIDA y DIMES) coincidieron en la forma (cuadrática) y el orden de la cantidad de enlaces virtuales que se logran descubrir relativa al total de enlaces físicos de cada red. En el caso de DIMES se observa una pequeña desaceleración, que se debe a las inconsistencias que presentan los datos en un proceso de medición real.

5.2 Futuras líneas de investigación

Dadas las innovaciones presentadas en este trabajo, se abre una gran cantidad de cuestiones a investigar más detalladamente. A continuación se describen algunas de las más importantes.

El aspecto más importante a estudiar, y que constituye el siguiente paso en la investigación, es el estadístico. Esto es, encarar la metodología desde el punto de vista de la estimación de las demoras en los enlaces como variables aleatorias, y determinar una cota del error.

También es necesario estudiar la cantidad de mediciones que se deben realizar sobre cada ruta.

Por último, es igualmente interesante estudiar la influencia de la cantidad y ubicación de los alfas en la estimación, para determinar cuántos alfas se necesitan para mejorar una estimación y en qué partes de la red.

La segunda cuestión a estudiar en detalle es la política de exploración. Luego de trazadas las rutas y clasificados los nodos, queda un conjunto de nodos delta, que pueden ser investigados recursivamente para abarcar una porción mayor de la red. Dado el tamaño de Internet, hasta ahora la política utilizada consistió en no aplicar la metodología recursivamente en los deltas surgidos durante el proceso de clasificación de nodos. Queda para el futuro estudiar otras políticas que permitan aumentar la porción abarcada manteniendo un crecimiento controlado.

Una tercera línea sumamente interesante es integrar las nuevas metodologías con el proyecto DIMES, de modo tal que se comiencen a utilizar como estrategia para determinar las rutas más convenientes para medir. También es importante desarrollar mejores métodos para la resolución de inconsistencias en las mediciones.

ANEXO A. DATOS EXPERIMENTALES

% de soporte de *ICMP timestamps*	nodos beta	nodos no alfabetos	enlaces físicos	enlaces virtuales
0	0	1731	8068	9848
10	163	2196	27043	35810
20	336	2430	45788	60229
30	495	2494	61897	79073
40	664	2537	78963	98038
50	830	2532	95461	114365
60	1000	2514	112142	128858
70	1166	2461	128042	142030
80	1323	2398	142901	152962
90	1483	2333	157801	162833
100	1682	2264	177343	176641

Tabla A.1. *Variables observadas. Red CAIDA. Rango completo.*
Referenciada en la sección 4.1.

% de soporte de *ICMP timestamps*	nodos beta	nodos no alfabetos	enlaces físicos	enlaces virtuales
0	0	87	229	110
10	9	153	471	288
20	18	217	715	470
30	27	271	943	647
40	35	317	1145	807
50	43	360	1344	966
60	53	405	1574	1153
70	61	437	1749	1299
80	70	475	1962	1475
90	78	508	2159	1640
100	87	537	2344	1796

Tabla A.2. *Variables observadas. Red aleatoria. Rango completo.*
Referenciada en la sección 4.1.

% de soporte de *ICMP timestamps*	enlaces virtuales	enlaces virtuales, relativo al 0% de soporte *ICMP timestamps*
0	9848	1
10	35810	3.64
20	60229	6.12
30	79073	8.03
40	98038	9.96
50	114365	11.61
60	128858	13.08
70	142030	14.42
80	152962	15.53
90	162833	16.53
100	176641	17.94

Tabla A.3. *Relación entre la cantidad de enlaces calculables para distintos niveles de soporte de ICMP timestamps y la cantidad cuando no se contempla su uso. Red CAIDA. Rango completo. Referenciada en la sección 4.1.*

% de soporte de *ICMP timestamps*	enlaces virtuales	enlaces virtuales, relativo al 0% de soporte *ICMP timestamps*
0	110	1
10	288	2.62
20	470	4.27
30	647	5.88
40	807	7.34
50	966	8.78
60	1153	10.48
70	1299	11.81
80	1475	13.41
90	1640	14.91
100	1796	16.33

Tabla A.4. *Relación entre la cantidad de enlaces calculables para distintos niveles de soporte de ICMP timestamps y la cantidad cuando no se contempla su uso. Red aleatoria. Rango completo. Referenciada en la sección 4.1.*

% de soporte de *ICMP timestamps*	enlaces físicos	enlaces virtuales	relación
0	8068	9848	0.82
10	27043	35810	0.76
20	45788	60229	0.76
30	61897	79073	0.78
40	78963	98038	0.81
50	95461	114365	0.83
60	112142	128858	0.87
70	128042	142030	0.90
80	142901	152962	0.93
90	157801	162833	0.97
100	177343	176641	1.00

Tabla A.5. *Cociente entre la cantidad de enlaces físicos y virtuales que se descubren en la red. Red CAIDA. Rango completo.*
Referenciada en la sección 4.1.

% de soporte de *ICMP timestamps*	enlaces físicos	enlaces virtuales	relación
0	229	110	2.08
10	471	288	1.64
20	715	470	1.52
30	943	647	1.46
40	1145	807	1.42
50	1344	966	1.39
60	1574	1153	1.37
70	1749	1299	1.35
80	1962	1475	1.33
90	2159	1640	1.32
100	2344	1796	1.31

Tabla A.6. *Cociente entre la cantidad de enlaces físicos y virtuales que se descubren en la red. Red aleatoria. Rango completo.*
Referenciada en la sección 4.1.

% de soporte de *ICMP timestamps*	nodos beta	nodos no alfabetos	enlaces físicos	enlaces virtuales
0	0	1731	8068	9848
1	16	1808	10091	12541
2	33	1879	12184	15391
3	49	1916	14057	18045
4	69	1992	16454	21230
5	82	2026	18031	23435
6	99	2064	19831	25969
7	117	2106	21950	28732
8	135	2148	23878	31344
9	152	2185	25901	34144
10	163	2196	27043	35810

Tabla A.7. *Variables observadas. Red CAIDA. Rango reducido.*
Referenciada en la sección 4.1.

% de soporte de *ICMP timestamps*	nodos beta	nodos no alfabetos	enlaces físicos	enlaces virtuales
0	0	87	229	110
1	1	95	255	128
2	2	100	274	142
3	3	106	301	162
4	3	113	323	178
5	4	119	346	195
6	5	128	379	219
7	6	136	406	238
8	7	140	424	253
9	8	146	445	267
10	9	153	471	288

Tabla A.8. *Variables observadas. Red aleatoria. Rango reducido.*
Referenciada en la sección 4.1.

% de soporte de *ICMP timestamps*	enlaces virtuales	enlaces virtuales, relativo al 0% de soporte *ICMP timestamps*
0	9848	1
1	12541	1.27
2	15391	1.56
3	18045	1.83
4	21230	2.16
5	23435	2.38
6	25969	2.64
7	28732	2.92
8	31344	3.18
9	34144	3.47
10	35810	3.64

Tabla A.9. *Relación entre la cantidad de enlaces calculables para distintos niveles de soporte de ICMP timestamps y la cantidad cuando no se contempla su uso. Red CAIDA. Rango reducido. Referenciada en la sección 4.1.*

% de soporte de *ICMP timestamps*	enlaces virtuales	enlaces virtuales, relativo al 0% de soporte *ICMP timestamps*
0	110	1
1	128	1.16
2	142	1.29
3	162	1.47
4	178	1.62
5	195	1.77
6	219	1.99
7	238	2.16
8	253	2.30
9	267	2.43
10	288	2.62

Tabla A.10. *Relación entre la cantidad de enlaces calculables para distintos niveles de soporte de ICMP timestamps y la cantidad cuando no se contempla su uso. Red aleatoria. Rango reducido. Referenciada en la sección 4.1.*

% de soporte de *ICMP timestamps*	enlaces físicos	enlaces virtuales	relación
0	8068	9848	0.82
1	10091	12541	0.80
2	12184	15391	0.79
3	14057	18045	0.78
4	16454	21230	0.78
5	18031	23435	0.77
6	19831	25969	0.76
7	21950	28732	0.76
8	23878	31344	0.76
9	25901	34144	0.76
10	27043	35810	0.76

Tabla A.11. *Cociente entre la cantidad de enlaces físicos y virtuales que se descubren en la red. Red CAIDA. Rango reducido. Referenciada en la sección 4.1.*

% de soporte de *ICMP timestamps*	enlaces físicos	enlaces virtuales	relación
0	229	110	2.08
1	255	128	1.99
2	274	142	1.93
3	301	162	1.86
4	323	178	1.81
5	346	195	1.77
6	379	219	1.73
7	406	238	1.71
8	424	253	1.68
9	445	267	1.67
10	471	288	1.64

Tabla A.12. *Cociente entre la cantidad de enlaces físicos y virtuales que se descubren en la red. Red CAIDA. Rango reducido. Referenciada en la sección 4.1.*

alfabetos adicionales	enlaces físicos	enlaces virtuales	enlaces físicos / enlaces virtuales
0%	1063	126	12%
10%	1004	199	20%
20%	949	272	29%
30%	910	328	36%
40%	846	396	47%
50%	799	431	54%
60%	763	484	63%
70%	716	530	74%
80%	691	580	84%
90%	679	624	92%
100%	671	671	100%

Tabla A.13. *Enlaces físicos y virtuales al agregar alfabetos adicionales al conjunto de mediciones de DIMES.*
Referenciada en la sección 4.2.

ANEXO B. HERRAMIENTA DE SIMULACIÓN

Para llevar a cabo las simulaciones del capítulo 4, se desarrolló una herramienta. Los principales algoritmos implementados se enumeran en la tabla B.1.

La herramienta se encuentra disponible en http://cnet.fi.uba.ar/azylber/

Número	Nombre	Autores	Referencia
2.1.1	Selección de rutas a monitorear	Y. Chen et al.	[Chen04]
2.1.2	Selección de rutas a monitorear	D. Chua et al.	[Chua05]
2.1.3	Estimación de las demoras en los enlaces y predicción para las rutas no monitoreadas	D. Chua et al.	[Chua05]
2.1.4	Búsqueda de todos los MILSes	Zhao et al.	[Zhao06]
3.1.1	Clasificación de nodos	presentado en esta tesis	
3.1.2	Creación de la matriz de ruteo	presentado en esta tesis	
3.2.1	Búsqueda de los enlaces identificables	presentado en esta tesis	

Tabla B.1. *Principales algoritmos implementados.*

Se implementaron también otros algoritmos específicos, como el procedimiento de la simulación (4.1.1) y los métodos para filtrar los datos de DIMES, explicados en la sección 4.2.

Las operaciones básicas con matrices ralas fueron implementadas.

Se tomaron 3 funciones de la librería GSL[§]:

- Descomposición en valores singulares
- Descomposición QR con pivoteo
- Eliminación de Gauss

§ GNU Scientific Library - http://www.gnu.org/software/gsl/

Anexo B – Herramienta de simulación

BIBLIOGRAFÍA

[Aaronson] Scott Aaronson,
"The Complexity Zoo",
http://www.cse.unl.edu/~cbourke/latex/ComplexityZoo.pdf

[Abilene] Abilene – Advanced Networking for Leading-edge Research and Education.
http://abilene.internet2.edu/

[Anagnostakis03] Kostas G. Anagnostakis, Michael Greenwald, Raphael S. Ryger,
"cing: Measuring Network-Internal Delays using only Existing Infrastructure",
IEEE Infocom, April 2003.

[Caida] CAIDA – Cooperative Association for Internet Data Analysis.
http://www.caida.org/

[Carmi06] Shai Carmi, Shlomo Havlin, Scott Kirkpatrick, Yuval Shavitt, Eran Shir,
"MEDUSA - New Model of Internet Topology Using k-shell Decomposition",
http://arxiv.org/abs/cond-mat/0601240 2006.

[Chen03] Yan Chen, David Bindel, Randy H. Katz,
"Tomography-based overlay network monitoring" ,
ACM Internet Measurement Conference, p 216-231, 2003.

[Chen04] Yan Chen, David Bindel, Hanhee Song, Randy H. Katz,
"An algebraic approach to practical and scalable overlay network monitoring" ,
ACM SIGCOMM, p 55-66, 2004.

[Chua05] David B. Chua, Eric D. Kolaczyk, Mark Crovella,
"Efficient Monitoring of End-to-End Network Properties",
Proceedings of Infocom 2005, March 2005.

[Dallasta06] L. Dall'Asta, I. Alvarez-Hamelin, A. Barrat, A. Vazquez and A.Vespignani
"Exploring networks with traceroute-like probes: theory and simulations" ,
Theoretical Computer Science 355, 6-24, 2006.

[Dimes] NetDimes – Journey to map the Internet.
http://www.netdimes.org/

[Govindan00] R. Govindan and H. Tangmunarunkit,
"Heuristics for Internet Map Discovery",
Proceedings of Infocom 2000, March 2000.

[Mahajan03] Ratul Mahajan, Neil Spring, David Wetherall, Thomas Anderson,
"User-level Internet Path Diagnosis",
ACM Sosp, October 2003.

[Rabbat04] M. Rabbat, R. Nowak, M. Coates,
"Multiple Source, Multiple destination Network Tomography",
IEEE Infocom, March 2004.

[RFC792] RFC 792 – Internet Control Message Protocol (ICMP)
http://www.faqs.org/rfcs/rfc792.html

[Shavitt04] Yuval Shavitt, Xiaodong Sun, Avishai Wool, Bulent Yener,
"Computing the Unmeasured: An Algebraic Approach to Internet Mapping",
IEEE J. on Selected Areas in Communications, v. 22, n. 1, p. 67-78, 2004.

[Vardi96] Y. Vardi,
"Network Tomography: estimating source-destination traffic intensities from link data",
J. Am. Statist. Assoc., vol. 91, pp. 365–377, 1996.

[Zhang00] Y. Zhang, V. Paxson, S. Shenker,
"The stationarity of Internet path properties: routing, Loss and Throughput",
ACIRI Technical Report, May 2000.

[Zhang01] Y. Zhang et al.,
"On the constancy of Internet path properties",
Proceedings of SIGCOMM IMW, 2001.

[Zhao05] Y. Zhao, Y. Chen, and D.S. Bindel,
"Scalable and deterministic overlay network diagnosis" ,
Poster, ACM SIGCOMM 2005

[Zhao06] Yao Zhao, Yan Chen, David Bindel,
"Towards unbiased end-to-end network diagnosis",
ACM SIGCOMM, 2006