Informe parámetros BRITE

1.- Introducción

Para la generación de topologías de red aleatorias hemos utilizado la herramienta BRITE (Boston University Representative Internet Topology Generator) desarrollada por Alberto Medina, Anukool Lakhina, Ibrahim Matta, John Byers, Department of Computer Science, Boston University.

Este, es un generador de topologías flexible que no está limitado a una única forma de generar topologías, es más, soporta múltiples modelos de generación. Por eso vemos necesario la explicación de los parámetros de generación de topologías.

Funcionamiento de BRITE: BRITE divide la generación de topologías en 4 pasos:

1.-Posicionamiento de nodos en el plano

2.-Interconexión de los nodos

3.-Asignación de atributos a los componentes de la topología (delay y ancho de banda a los enlaces…)

4.-Salida de la topología al formato especificado

2.-Ejecución de BRITE

Para ejecutar BRITE disponemos de 2 opciones, ejecutarlo por interfaz gráfica (GUI) implementado con Java, o por línea de comandos, implementado en C++.

Para el proyecto DEN2NE, nosotros hemos utilizado BRITE por línea de comandos, así pudimos automatizar la generación de las topologías. De esta forma la entrada de BRITE por línea de comandos es la siguiente:

```
$ bin/brite <archivo de configuración de entrada> <archivo brite de salida>
<seed_file>
```

Como vemos, para la ejecución de BRITE necesitamos un archivo de configuración de entrada en el que se definen los parámetros que luego tendrá el archivo brite de salida. Estos archivos de configuración tendrán una estructura como el siguiente. Para la construcción de estos archivos de configuración desarrollamos un programa en Python que genera estos archivos automáticamente, “generador_brite.py” que se encuentra en el repositorio del proyecto:

https://github.com/NETSERV-UAH/BRITE.

```
BriteConfig

BeginModel
        Name = 1
        N = 10
        HS = 1000
        LS = 100
        NodePlacement = 1
        GrowthType = 1
        m = 2
        alpha = 0.15
        beta = 0.2
        BWDist = 1
        BWMin = 10.0
        BWMax = 1024.0
EndModel



BeginOutput
        BRITE = 1
        OTTER = 0
        DML = 0
        NS = 0
        Javasim = 0
EndOutput
```

```
BriteConfig

BeginModel
        Name = 2
        N = 10
        HS = 1000
        LS = 100
        NodePlacement = 1
        m = 1
        BWDist = 1
        BWMin = 10.0
        BWMax = 1024.0
EndModel



BeginOutput
        BRITE = 1
        OTTER = 0
        DML = 0
        NS = 0
        Javasim = 0
EndOutput
```

Figura 1: Ejemplo de archivo de configuración de entrada BRITE. Izquierda RTWaxman, derecha RTBarabasi-Albert [3]

3.-Explicación de los parámetros de entrada

+ Modelo de la topología (Name):

Indican el modelo de generación de la topología utilizada. Puede ser:

-Router Waxman = 1

-AS Waxman = 3

-Router Barbasi = 2

-AS Barbasi = 4

-Top Down = 5

-Bottom Up = 6

-Router FIle = 7

-AS File = 8

+Número de nodos de la topología (N): Su valor debe ser un entero entre: 1<= N <= HS * HS

+HS y LS (Dimensiones del plano):

HS: Longitud del lado del plano cuadrado donde se encuentra la topología

LS: Tamaño de los cuadros interiores

Estos valores se han dejado los de por defecto: HS=1000 y LS=100

+Node Placement: Cómo se posicionan los nodos en el plano. Dos opciones:

-Random (1): Se colocan en el plano de forma totalmente aleatoria. Es el seleccionado para las topologías generadas

-Heavy Tailed (2): Se colocan por zonas con mayor concentración de nodos

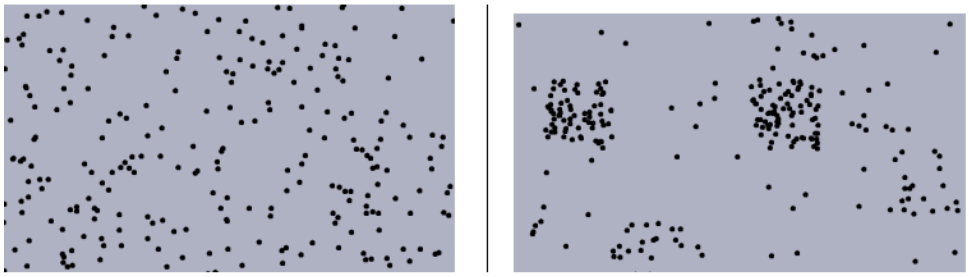

Figura 2: Node Placement: Random (izquierda), Heavy-Tailed (derecha) [1]

+Growth Type: Cómo los nodos se introducen en la topología.

-Incremental (1): Posiciona los nodos uno a uno a medida que se unen a la topología. En este punto solo se seleccionan como nodos de elección aquellos que ya se han incluido en la topología. Es el seleccionado para las topologías Waxman generadas. Las topologías Barabasi no necesitan esta variable

-Random (2): posiciona todos los nodos a la vez antes de añadir los enlaces. En este punto, un nodo es aleatoriamente seleccionado y m enlaces son usados para conectarlo con m vecinos candidatos del total de nodos. [2]

+ Número de enlaces por nuevo nodo (m): Número de nodos vecinos a los que un nuevo nodo se conecta cuando se une a la red. En otras palabras, el número de nuevos enlaces que se añaden a la topología. Cuanto mayor es el valor de m, más densa es la topología generada.

Hay que descatar que BRITE puede mostrar los enlaces unidireccionales o bidireccionales. Si los enlaces son unidireccionales m coincide con el grado de la topología, mientras que si los enlaces so bidireccionales entones el grado de la topología será 2m. Para saber si BRITE ha definido los enlaces como unidireccionales o bidireccionales nos fijamos en los archivos de salida de BRITE. En el ejemplo de la Figura 3 vemos que son bidireccionales pues las columnas 5 y 6 de los nodos, son el número de enlaces entrantes y salientes respectivamente. Si sumamos por un lado los enlaces de salida de todos los nodos y por otro los de entrada, vemos que obtenemos 40 enlaces de entrada y de salida, que son el doble de los enlaces unidireccionales que define en la sección de abajo BRITE. Por tanto, concluimos que los enlaces son considerados bidireccionales, y para este ejemplo el grado es 4 = 2*m = 40/10. En todas topologías los enlaces son bidireccionales, por tanto: **Grado = 2m**

+Alpha y Beta: Parámetros específicos del modelo Waxman. Se han fijado a: Alpha = 0.15, Beta = 0.2

+BWDist, BWMin, BWMax: Parámetros relacionados con el ancho de banda de los enlaces. Para la generación de las topologías no se han tenido en cuenta.

+Archivo de salida (BeginOutput): Seleccionamos siempre el formato ".brite" pues para el que se ha diseñado una interfaz para la generación de los grafos en el algoritmo de estudio, a partir de los archivos ".brite" generados.

```
1   Topology: ( 10 Nodes, 20 Edges )
2   Model ( 1 ): 10 1000 100 1 1 2 0.15 0.2 1 10 1024
3
4   Nodes: (10)
5   0 724.00 173.00 5 5 -1 RT_NODE
6   1 684.00 416.00 5 5 -1 RT_NODE
7   2 300.00 692.00 5 5 -1 RT_NODE
8   3 480.00 702.00 4 4 -1 RT_NODE
9   4 909.00 886.00 3 3 -1 RT_NODE
10  5 251.00 603.00 3 3 -1 RT_NODE
11  6 51.00 76.00 4 4 -1 RT_NODE
12  7 853.00 943.00 4 4 -1 RT_NODE
13  8 582.00 71.00 3 3 -1 RT_NODE
14  9 438.00 405.00 4 4 -1 RT_NODE
15
16  Edges: (20):
17  0 2 1 472.90 1.58 10.00 -1 -1 E_RT U
18  1 2 0 670.18 2.24 10.00 -1 -1 E_RT U
19  2 3 1 351.30 1.17 10.00 -1 -1 E_RT U
20  3 3 0 582.56 1.94 10.00 -1 -1 E_RT U
21  4 4 2 639.15 2.13 10.00 -1 -1 E_RT U
22  5 4 3 466.79 1.56 10.00 -1 -1 E_RT U
23  6 5 3 249.48 0.83 10.00 -1 -1 E_RT U
24  7 5 2 101.60 0.34 10.00 -1 -1 E_RT U
25  8 6 2 664.42 2.22 10.00 -1 -1 E_RT U
26  9 6 1 718.53 2.40 10.00 -1 -1 E_RT U
27  10 7 4 79.91 0.27 10.00 -1 -1 E_RT U
28  11 7 6 1181.06 3.94 10.00 -1 -1 E_RT U
29  12 8 6 531.02 1.77 10.00 -1 -1 E_RT U
30  13 8 0 174.84 0.58 10.00 -1 -1 E_RT U
31  14 9 8 363.72 1.21 10.00 -1 -1 E_RT U
32  15 9 5 272.35 0.91 10.00 -1 -1 E_RT U
33  16 0 7 780.73 2.60 10.00 -1 -1 E_RT U
34  17 0 9 368.27 1.23 10.00 -1 -1 E_RT U
35  18 1 7 553.43 1.85 10.00 -1 -1 E_RT U
36  19 1 9 246.25 0.82 10.00 -1 -1 E_RT U
```

Figura 3: Ejemplo de archivo .brite de salida [3]

Field	Meaning
NodeId	Unique id for each node
xpos	x-axis coordinate in the plane
ypos	y-axis coordinate in the plane
indegree	Indegree of the node
outdegree	Outdegree of the node
ASid	id of the AS this node belongs to (if hierarchical)
type	Type assigned to the node (e.g. router, AS)

Field	Meaning
EdgeId	Unique id for each edge
from	node id of source
to	node id of destination
length	Euclidean length
delay	propagation delay
bandwidth	bandwidth (assigned by *AssignBW* method)
ASfrom	if hierarchical topology, AS id of source node
ASto	if hierarchical topology, AS id of destination node
type	Type assigned to the edge by classification routine

Figura 4: Salida archivo .brite. Izquierda Nodos, derecha Enlaces [1]

4.- Explicación de los modelos de generación de la topología

BRITE, como generador de topologías flexible permite la generación de varios modelos topológicos, y por eso centra su arquitectura sobre el concepto de modelo

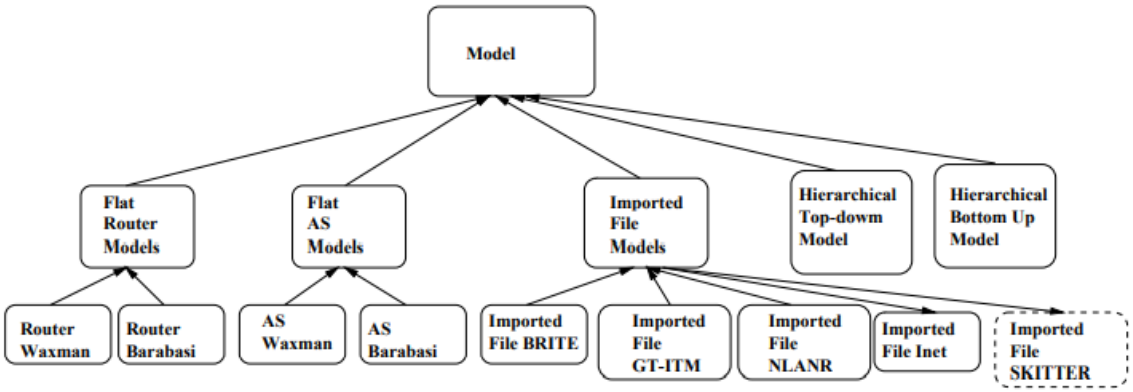

Figura 4: Modelos utilizados en BRITE [1]

Como vemos, BRITE permite la generación de topologías basadas en modelos de Router, de AS, topologías importadas, y topologías jerárquicas Top-down o BottomUp.

4.1 Flat Router-level Models

BRITE contiene la clase RouterModel derivada de la clase Model. La idea de esta clase es generar topologías a nivel de router, separadas de modelos para otros entornos como AS, LANS, etc.

Los modelos de topologías a nivel de router que proporciona BRITE son RouterWaxman y RouterBarabasiAlbert

4.1.1 Router Waxman:

Básicamente hace referencia a un modelo de generación de topologías aleatorias usando el modelo de probabilidad Waxman para la interconexión de los nodos de la topología, la cual es:

$$P(u, v) = \alpha e^{-d/(\beta L)}$$

Donde $0 < \alpha, \beta \leq 1$, d es la distancia Euclídea desde el nodo u al nodo v y L es la máxima distancia entre dos nodos cualesquiera. Un aumento de α produce un aumento en la densidad de enlaces, mientras que un aumento en β aumenta el ratio de enlaces más largos sobre los enlaces más pequeños. Ante la inviabilidad de modelar las topologías con 4 parámetros (nº nodos, grado, α y β) por la sobrecarga de pruebas, decidimos fijar los valores α y β, siguiendo el modelado de Zegura en [4] donde propone para modelar redes de Internet utilizar α=0.2 y β=0.15.

La diferencia entre Router Waxman y ASWaxman solamente difiere en que la topología Router Waxman representa routers.

4.1.2 Router Barabasi-Albert (BA)

El modelo Barabasi-Albert (BA) es un algoritmo para generar redes aleatorias complejas libres de escala. Libres de escala quiere decir que el algoritmo sigue leyes exponenciales. Este modelo se caracteriza por dos conceptos:

- Growth (Crecimiento). El número de nodos en la red se incrementa con el tiempo

- Preferencial Attachment (Conexión preferencial). Cuanta mayor conectividad tenga un nodo, mayor probabilidad tiene de recibir nuevos enlaces.

La red comienza con un conjunto de N_0 nodos conectados. Los nuevos nodos son añadidos a la red de uno en uno. Cada nuevo nodo añadido a la red se conecta a N <= N_0 nodos existentes con una probabilidad que es proporcional al número de enlaces existentes. La probabilidad de conectar un nuevo nodo con el nodo i es:

$$p_i = \frac{k_i}{\sum_j k_j}$$

donde k_i es el grado del nodo objetivo i y $\sum_j k_j$ es el sumatorio de los grados de salida de todos los nodos que se han unido previamente a la topología.

4.2 Modelos a nivel AS

Los modelos de nivel AS son muy similares a las topologías a nivel de router, con la diferencia de que en los modelos AS, los nodos AS posicionados en el plano tienen la capacidad de contener topologías asociadas.

4.3 Topologías jerárquicas

BRITE soporta la generación de dos topologías jerárquicas:

4.3.1 Topología jerárquica Top-down:

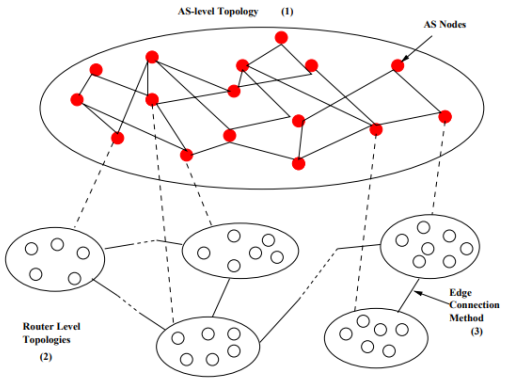

Figura 5: Topología Top-down [1]

Para esta topología BRITE genera primero una topología de nivel AS (1) entre las posibles (Waxma, Barabasi, importada…). Posteriormente BRITE generará topologías a nivel de router (2) usando los distintos modelos de generación posibles. BRITE interconectará estas topologías a nivel de router según la conectividad de la topología AS. BRITE proporciona 4 mecanismos de interconexión basándose en el popular generador de topologias GT-ITM. Estos 4 mecanismos son, considerando (i, j) un enlace a nivel AS, entonces escoge un nodo u de la topología a nivel de router, asociada al nodo AS i, RT(i), y un nodo v de la topología a nivel de router, asociada al nodo AS j, RT(j):

- **Aleatorio:** u es seleccionado de forma aleatoria de RT(i), y u es también aleatorio de RT(j)
- **Por menor grado:** u y v son nodos con el menor grado en RT(i) y RT(j)
- **Por menor grado sin ser nodos hoja:** u y v son nodos con el menor grado en RT(i) y RT(j) sin ser nodos hoja

- **Con grado mayor que K:** u y v son nodos con grado mayor o igual a K en RT(i) y RT(j)

La topología final es una topología a nivel de router formada por las topologías individuales de cada nodo AS.

4.3.2 Topología jerárquica Bottom-up

Para esta topología BRITE primero genera una topología a nivel de router. Posteriormente BRITE asigna a cada nodo AS un número de routers según el tipo de asignación seleccionada por el usuario. Estos tipos de asignación son:

- **Constante:** Asigna a cada AS un número igual de nodos. Ej.: NumNodes/NumAS
- **Uniforme:** Asigna un numero uniformemente distribuido en el rango [1, NumNodes]
- **Exponencial:** Asigna un valor exponencialmente distribuido que significa NumNodes/NumAS
- **Heavy-tailed:** Asigna un valor de una distribución heavy-tailed truncada entre 1 y NumNood

Una vez definido el número de routers por nodo AS, BRITE selecciona los nodos que pertenecen a cada nodo AS mediante dos mecanismos:

- **Selección aleatoria:** Selecciona un nodo aleatorio para asignarlo al nodo AS i, hasta que dicho nodo alcanza el número de routers que tiene asignados. Se repite para cada nodo AS
- **Paseo aleatorio:** Realiza un paseo aleatorio a través del grafo, en cada paso se escoge un nodo vecino aleatorio. Cada nodo visitado es asignado al AS i hasta que este se llena. Se repite para cada nodo AS.

4.4 Topologías importadas

BRITE también permite la generación de topologías a partir de ficheros importados de otros generadores de topologías. Por ejemplo, podríamos generar una topología top-down a partir de ficheros de otros generadores como GT-ITM. BRITE permite importar las siguientes topologías: BRITE, GT-ITM, NLANR y Inet.

5.-Funcionamiento de las semillas en BRITE

Con el objetivo de que estas pruebas sean reproducibles por cualquier persona, en la generación de las topologías hemos utilizado nuestras propias semillas, de forma que cualquier persona que ejecute el script “autogenerador.sh” obtendrá las mismas topologías que obtenemos nosotros.

Pero, aunque esté todo automatizado, vamos a explicar el funcionamiento de las semillas en BRITE. Como hemos visto en el apartado 2 de este informe, al ejecutar BRITE por línea de comandos:

```
$ bin/brite <archivo de configuración de entrada> <archivo brite de salida>
<seed_file>
```

Vemos que necesitamos un “seed_file”, una semilla. BRITE tras esta ejecución modifica el seed_file. Esto lo hace para evitar que una repetición secuencial de este comando, sin cambiar de seed_file genere topologías idénticas. Ante esta operativa de BRITE, nos hemos visto obligados a crearnos 2 carpetas distintas con las mismas semillas. Estas son “Carpeta_seeds/” y “Carpeta_seeds_backup/”. Como necesitamos 10 topologías brite distintas pero que se

generen a partir de los mismos parámetros (mismo archivo de configuración), entonces tenemos 10 semillas, nombradas como “seedx” donde la x es un número del 1 al 10.

En “Carpeta_seeds_backup/” se encuentran una copia de las semillas que vamos a utilizar para cada ejecución. Así, estas semillas no se van a introducir nunca en el comando de BRITE, y por tanto estas semillas nunca cambiarán. Las semillas que meteremos en el comando BRITE serán las que se encuentran en el directorio “Carpeta_seeds/”. Las semillas de esta carpeta son las que introduciremos en el comando BRITE y, por tanto, estas semillas cambiarán tras ejecutarse el comando BRITE. Por eso, antes de cada ejecución del comando BRITE copiamos las semillas que se encuentran en “Carpeta_seeds_backup/” a la “Carpeta_seeds/”, para utilizar siempre las mismas semillas y no las que me cambia BRITE. De esta forma sabemos con qué semillas estamos ejecutando BRITE todo el tiempo.

6.- Topologías generadas para las pruebas

Para la realización de pruebas sobre el algoritmo DEDENNE se han generado mediante BRITE topologías aleatorias basadas en los modelos RTWaxman y RTBarabasi-Albert. Estas topologías están parametrizadas de 10 hasta 200 nodos con un incremento de 10 nodos, grado de conectividad de 2 (poco conectivo), 4 (conectividad normal), 6 (alta conectividad), por lo que, al ser enlaces bidireccionales, el parámetro m es 1, 2 y 3 respectivamente. Como se ha comentado antes, en los modelos RTWaxman los valores de α y β se han fijado a 0.2 y 0.15 respectivamente. Además, se han generado, para cada conjunto de parámetros idénticos (archivo de configuración), 10 topologías distintas, es decir, difieren en la forma en la que se interconectan los nodos en el plano. De esta forma se obtienen un total de 1200 topologías distintas (2x20x3x10).

Estas topologías se encuentran organizadas en un sistema de carpetas que sigue el siguiente orden (partiendo desde la raíz del repositorio):

Archivos_brite/<Modelo topológico>/<númeronodos>/<grado>/<semilla>/archivobrite.brite

Donde Modelo topológico es o Waxman o Barabasi; númeronodos va de 10 a 200 con un incremento de 10; grado es el grado de la topología que puede ser 2,4 o 6; y semilla es la semilla utilizada que es un entero del 1 al 10 ambos incluidos; y archivobrite.brite es el nombre que le hemos puesto a todos los archivos generados por BRITE

Como adición, para no tener que tratar con el formato de los ficheros “.brite”, creamos el programa “parser.py” el cuál lee los ficheros .brite y guarda los datos importantes en dos ficheros, “Nodos.txt” y “Enlaces.txt”. En Nodos.txt guardamos para cada topología sus nodos y sus posiciones x e y en el plano. En Enlaces.txt guardamos, por cada enlace en la topología, los dos nodos que interconecta y la distancia entre ellos. Todo esto separado entre ‘;’

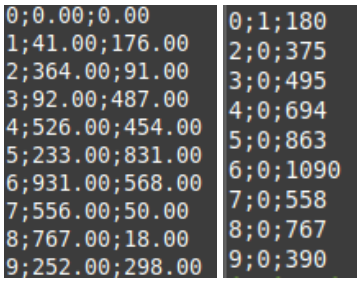

Figura 6: Izquierda: Nodos.txt | Derecha: Enlaces.txt. Se corresponde con un Barabasi/10/2/1/.

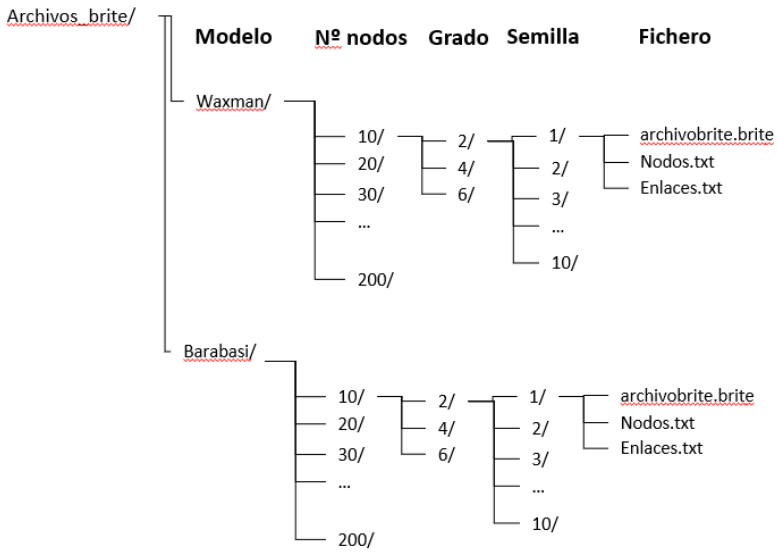

Figura 6: Estructura final del directorio Archivos_brite

7. Automatización de la generación de topologías: autogenerador.sh

La generación de todos los archivos que se han realizado para la prueba se ha automatizado a través de un script "autogenerador.sh" que se encuentra accesible en el repositorio Github. Este script automatiza todo el proceso de la generación de topologías. Genera los archivos de configuración, los cuales guarda en las carpetas "Archivos_conf_Waxman" y "Archivos_conf_Barabasi" en función de si son modelos Waxman o Barabasi. Su nombre es "archivoconf<n_nodos>x<m>.conf", donde n_nodos es el número de nodos de la topología y m es el parámetro m descrito en el apartado 3 de este informe, este valor m como hemos comentado tomará los valores 1, 2 y 3. Posteriormente el script ejecuta BRITE 10 veces a partir del mismo archivo de configuración, pero con distinta semilla, así conseguimos las 10 topologías que, teniendo los mismos parámetros, son distintas gracias a las semillas. Y por último se ejecuta parser.py que transforma la salida de BRITE, en dos archivos Nodos.txt y Enlaces.txt.

8.-Bibliografía

[1] Medina, Alberto & Lakhina, Anukool & Matta, Ibrahim & Byers, John. (2001). BRITE: an approach to universal topology generation. 346-353. 10.1109/MASCOT.2001.948886.

[2] Medina, Alberto & Matta, Ibrahim & Byers, John. (2000). On the Origin of Power Laws in Internet Topologies. Computer Communication Review. 30. 10.1145/505680.505683.

[3] https://github.com/NETSERV-UAH/BRITE

[4] E. W. Zegura, K. L. Calvert and S. Bhattacharjee, "How to model an internetwork," Proceedings of IEEE INFOCOM '96. Conference on Computer Communications, 1996, pp. 594-602 vol.2, doi: 10.1109/INFCOM.1996.493353.