El problema del árbol de empuje en sistemas de telecomunicaciones

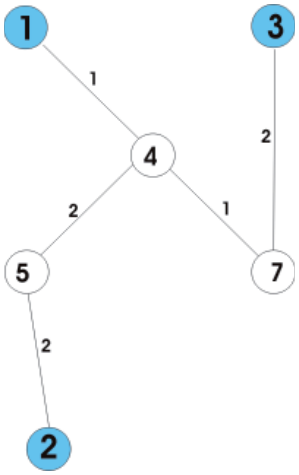
Karla V. Martínez Facundo, Jania A. Saucedo Martínez, J. Ángel Segura Ramiro, Miguel A. Urbano Vázquez, Roger Z. Ríos Mercado
Programa de Posgrado en Ingeniería de Sistemas, FIME, UANL
{karla, jania, angel, akuma, roger}@yalma.fime.uanl.mx

RESUMEN

En los sistemas de distribución de información en sistemas de telecomunicaciones se utiliza una combinación de "empujar" y "jalar" paquetes de información para obtener los datos precisos en el lugar adecuado. El problema principal de estos sistemas es el tráfico innecesario que se genera. Una forma de minimizar dicho tráfico es resolviendo un problema del "árbol de empuje" asociado. En este trabajo se presenta una descripción a detalle de este problema, desde la perspectiva de la optimización de flujo en redes, y una implementación computacional de un algoritmo heurístico para obtener soluciones aproximadas, basado en el método de Havet y Wennink.

PALABRAS CLAVE

Investigación de operaciones, flujo, redes, telecomunicaciones, heurística, árbol de Steiner.



ABSTRACT

In data distribution systems, a "push" and "pull" combination procedure is used to obtain the correct data in the correct place. The main problem in these systems is the unnecessary traffic that is generated. One way to minimize this traffic is to solve an associated Push Tree problem. A detailed description of this problem from the network flow programming perspective, and a heuristic scheme for approximate solutions, based on the method developed by Havet and Wennink, are discussed in this paper.

KEYWORDS

Operations research, network flow, telecommunications, heuristic, Steiner tree.

INTRODUCCIÓN

El presente trabajo cae en el marco de la optimización de problemas de flujos en redes el cual es un subcampo de la investigación de operaciones, que es la ciencia que brinda sustento científico a los problemas de toma de decisiones. Dentro de este subcampo, uno de los problemas importantes encontrados en el área de telecomunicaciones es el derroche de recursos del ancho de banda. Esto

se debe al tráfico innecesario de información que puede presentarse en la red causado por el envío de información no requerida. Este tipo de problema se puede resolver al plantearse como un problema de árbol de empuje óptimo, el cual se describe en detalle más adelante.

Recientemente, Havet y Wennink[1] presentan una heurística (algoritmo de solución aproximada) para encontrar el árbol de empuje (al cual referiremos como *PT* por sus siglas en inglés, *Push Tree*) para una red determinada.

El primer objetivo de este trabajo es el de presentar al lector una descripción formal del problema del árbol de empuje. Posteriormente, se describe e implementa la heurística de Havet y Wennink para la solución aproximada del mismo. Otra contribución de nuestro trabajo es el desarrollo de un programa de optimización basado en CPLEX[2] (paquete de bibliotecas de optimización para problemas de programación lineal entera). Finalmente se presenta una evaluación empírica de la heurística de Havet y Wennink y una comparación con la solución óptima al problema reportado por CPLEX.

El artículo está organizado de la siguiente manera. Primero se describe el problema del árbol de empuje, seguido después por su formulación. Posteriormente se plantea la formulación del problema del árbol de Steiner mínimo y una heurística para obtenerlo, ya que es necesario para encontrar el *PT* óptimo mediante el método presentado en.[1] Por último se muestran resultados de la experimentación computacional a partir de los cuales obtenemos conclusiones.

DESCRIPCIÓN GENERAL DEL PROBLEMA

En los sistemas de telecomunicaciones encargados de la distribución de información se utiliza una combinación de empujar y jalar paquetes de información para obtener los datos adecuados en el momento y lugar correctos. La función de "empujar" se emplea donde se encuentra la información y consiste en enviarla a todos los usuarios (estén interesados o no en ella), mientras que la función de "jalar" les corresponde a los usuarios que están interesados en obtenerla. En general el lugar de origen envía mensajes actualizados a un conjunto de nodos en la red, los cuales van a mantener réplicas actualizadas de la información y ésta se encontrará a la mano de los nodos que así la requieran. Las dos formas de distribución de información producen tráfico en la red.

El objetivo es encontrar la mejor forma de distribuir la información minimizando la cantidad total de tráfico respetando las restricciones tecnológicas del problema. Es decir, encontrar el conjunto de nodos (usuarios) y aristas (rutas) a través de los cuales serán trasmitidas las actualizaciones y requisiciones, de tal manera que se minimice el tráfico generado por las funciones de "empujar" y "jalar", tomando en cuenta los nodos requerimiento (los que necesitan información y la solicitan) y la tasa de actualización (cantidad de veces que la información cambia por unidad de tiempo). Este problema, introducido por Havet y Wennink,[1] se denomina el problema del árbol de empuje.

La solución óptima al problema del *PT* está caracterizada por un árbol enraizado en el nodo donde se genera la información. El problema del *PT* se enfoca a la minimización de costos variables asociados con la cantidad de tráfico generada por los diferentes mecanismos.

Si sólo se utilizara el mecanismo de "jalar" tratando de minimizar el costo variable generado por dicho tráfico, correspondería a solucionar el problema de ruta más corta entre todos los pares de nodos; en cambio si sólo utilizamos el mecanismo de "empujar" corresponde a encontrar la solución a un problema del mínimo árbol de Steiner (ver Apéndice). Una característica importante del problema del *PT* es la transición entre estos dos tipos de problemas cuando ocurren cambios en la tasa de actualización.

FORMULACIÓN DEL PROBLEMA DEL ÁRBOL DE EMPUJE

Para formular adecuadamente el problema del árbol de empuje se utilizan conceptos básicos de la teoría de grafos y de optimización de flujos en redes (véase por ejemplo el texto de Ahuja, Magnanti y Orlin).[3] En el Apéndice se incluyen algunos términos para darle mayor claridad a la exposición.

Sea $G = (V(G)\ E(G)$ un grafo no dirigido y sea $l(e) = l(u,v) > 0$ la longitud de la arista

$e=(u,v)\in E(G)$. Además sea $l(H)=\sum_{e\in E(H)} l(e)$ la longitud del subgrafo *H* de *G*. Entonces se puede modelar el problema del *PT* como sigue: Dado un grafo *G*, un nodo origen $s\in V(G)$, una tasa de actualización $\mu \geq 0$, un conjunto de nodos requerimiento $R\subseteq V(G)$, cada uno de estos con una tasa de requerimiento $r(v)\geq 0$, $\forall v\in R$, encontrar un subgrafo *PT* de G, con $s\in V(PT)$, y rutas P_v de v a cualquier nodo en *V*(*PT*), $\forall v\in R$, tal que:

$$\mu l(PT)+\sum_{v\in R} r(v)l(P_v)$$

sea mínimo.

El nodo origen *s* contiene una parte de la información la cual cambia μ veces por unidad de tiempo. Después de cada cambio, la información actualizada es "empujada" del nodo *s* a todos los nodos en *V(PT)* usando las aristas en *E(PT)*. Medimos la cantidad de tráfico generado por un mensaje al recorrer cierta distancia. La cantidad total de tráfico de actualización por unidad de tiempo es igual a $a\mu l(PT)$ donde a es el tamaño (número de bits) de un mensaje de actualización.

La información requerida del nodo *s* por los nodos requerimiento se almacena en *r(v)* que es el número de veces por unidad de tiempo que se necesita información en el nodo $v\in R$. Cada requerimiento es causado por un mensaje requisitor (de tamaño *b*) a ser enviado sobre la ruta P_v (sobre la cual se aplica el mecanismo de "jalar") del nodo v a un nodo en *PT* el cual contiene una copia actualizada de la información. Este nodo envía un mensaje de respuesta (de tamaño *c*) a v usando la misma ruta pero ahora en la dirección contraria. La cantidad total de tráfico de requerimiento por unidad de tiempo es igual a:

$$\sum_{v\in R} r(v)(b+c)l(P_v)$$

Por comodidad, se normaliza el tamaño del tráfico con $a=1$ y utilizando tasas de requerimiento ajustadas que son obtenidas de multiplicar la ecuación original por un factor (*b*+*c*). Esto resulta en un tráfico total que es igual a $\mu l(PT)+\sum_{v\in R} r(v)l(P_v)$. Es fácil ver que en la solución óptima *PT* es un árbol. Para probarlo, suponga que *PT*=(*V*(*PT*), *E*(*PT*)) contiene un ciclo *C*. Sea e una arista del ciclo. Entonces el subgrafo $PT'=(V(PT), E(PT)\backslash\{e\})$ todavía está conectado y todos los nodos en *V*(*PT*) aún están recibiendo mensajes de actualización. Sin embargo, el tráfico de actualización generado por *PT'* es $\mu l(e)$ y es más pequeño que el tráfico generado por *PT*.

Además, para un árbol dado *PT*, la ruta óptima P_v para un nodo requerimiento v es una ruta más corta de v a cualquier nodo en *PT*. Encontrar esta ruta es fácil, sin embargo la principal dificultad es encontrar el óptimo *PT*.

Suponga que μ es lo suficientemente pequeño (digamos $\mu < r(v), \forall v\in R$). Considere una solución al correspondiente problema del *PT* en el cual se tiene un nodo requerimiento que no está en *PT*. Por lo tanto la ruta por la cual se "jala" la información tiene una longitud positiva. Incluyendo todas las aristas en P_v, en el árbol de empuje se generará un tráfico de actualización adicional de $\mu l(P_v)$ que resultará en una reducción en el tráfico de requerimiento dado por $r(v)l(P_v)$. El efecto en la red es una reducción en el tráfico total. Todos los nodos en *R* deben ser incluidos en el árbol de empuje, y el óptimo *PT* es el mínimo árbol que conecta todos los nodos en $R\cup\{s\}$. Este árbol es conocido como el árbol de Steiner de $R\cup\{s\}$ en *G*. Encontrar el árbol de Steiner es un problema NP-difícil,[4] por lo tanto el problema del *PT* también es NP-difícil. Esto significa que cualquier algoritmo que pretenda encontrar la solución exacta al problema emplea un tiempo de cómputo que crece exponencialmente con el tamaño del problema en el peor de los casos.

Por otro lado si μ es suficientemente grande (digamos $\mu > \sum_{v\in R} r(v)$), entonces el óptimo *PT* consiste de un sólo nodo *s*. Sea *w* un nodo hoja de un árbol de empuje *PT*, y sea $e=(u,v)\in E(PT)$. Eliminar *w* y *e* de *PT* causa una reducción del tráfico de actualización dado por $\mu l(e)$ y un incremento en el tráfico de requerimiento de al menos $\sum_{v\in R} r(v)l(e)$. Otra vez el efecto en la red es la reducción del tráfico total. Este proceso puede ser repetido sólo cuando *s* está en *PT*. El problema del *PT* es equivalente a encontrar las rutas más cortas de cada nodo en *R* a *s*.

Se podría decir que el problema del *PT* cambia de ser un problema de árbol de Steiner a ser un problema de rutas más cortas cuando la tasa de actualización μ se incrementa.

FORMULACIÓN DEL PROBLEMA DEL ÁRBOL DE STEINER

Dado un grafo $G=(N,A)$ donde N es un conjunto de nodos y A es un conjunto de aristas, además de un conjunto de nodos requerimiento $R \subset N$ y un nodo $s \in R$ que representa el nodo fuente, una formulación matemática del problema del árbol de Steiner está dada como sigue.

Parámetros:

c_{ij} = *Costo por utilizar la arista* (i, j)

Variables de decisión:

x_{ij} = Flujo de la arista (i, j)
$z_{ij} = 1$ si la arista (i, j) se utiliza y 0 si no.
$w_i = 1$ si el nodo i es fuente y 0 si no.

Modelo:

$$\min \sum_{(i,j)\in A} c_{ij} z_{ij}$$

sujeto a:

$$(1) \quad x_{ij} \le z_{ij}(|R|-1), \ \forall (i,j) \in A$$
$$(2) \quad \sum_{j\in N} x_{ij} - \sum_{j\in N} x_{ji} = 0, \ \forall i \in N \setminus R$$
$$(3) \quad \sum_{j\in N} x_{ij} - \sum_{j\in N} x_{ji} = (|R|-1)w_i + (1-w_i)(-1), \ \forall i \in R$$
$$(4) \quad \sum_{i\in R} w_i = 1$$
$$(5) \quad z_{ij}, w_i \in \{0,1\}$$
$$(6) \quad x_{ij} \in Z^+$$

La función objetivo representa la búsqueda de la longitud mínima del árbol de Steiner que para nuestro modelo está estructurado como un problema de flujo en redes. La restricción (1) asegura que el flujo que pasa por la arista (i,j) no excede el requerimiento solicitado por los nodos. La restricción (2) muestra la ecuación de balance nodal para los nodos que no se encuentran en el conjunto R. La restricción (3) señala la ecuación de balance de los nodos requerimiento. La restricción (4) establece que únicamente existe un nodo fuente. Por último, la restricción (5) exige que las variables z_{ij} y w_i tomen valores binarios y la restricción (6) limita las variables a tomar valores enteros positivos.

HEURÍSTICA PARA ENCONTRAR EL ÁRBOL DE STEINER

La heurística que se utilizó para encontrar el árbol de Steiner consta de tres pasos:

1. Calcular las rutas más cortas entre todos los pares de nodos.
2. Calcular el mínimo árbol de expansión.
3. Construcción del árbol de Steiner.

A partir de un grafo dado se calculan las rutas más cortas entre cada par de nodos (utilizando el algoritmo de Floyd-Warshall)[3] y este valor reemplazará el peso de las aristas. Una vez obtenida esta información el siguiente paso es encontrar el mínimo árbol de expansión considerando únicamente los nodos requerimiento, después se agregará un elemento del conjunto $N\setminus R$. Para ilustrar la heurística se presenta un pequeño ejemplo del cual su grafo inicial se muestra en la figura 1.

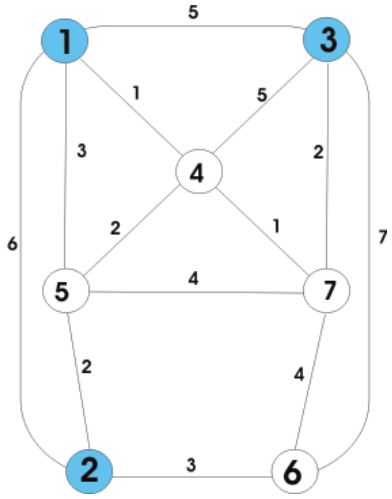


Fig. 1. Grafo inicial.

A partir del grafo inicial se genera la matriz de incidencia nodo-nodo con las distancias más cortas, en este caso desde el nodo 1 hasta el nodo 7, esto se muestra en la figura 2. Es fácil observar que se trata de una matriz simétrica.

	1	2	3	4	5	6	7
1	0	5	4	1	3	6	2
2	5	0	7	4	2	3	5
3	4	7	0	3	5	6	2
4	1	4	3	0	2	5	1
5	3	2	5	2	0	5	3
6	6	3	6	5	5	0	4
7	2	5	2	1	3	4	0

Fig. 2. Matriz de incidencia nodo-nodo de las distancias más cortas.

Con la matriz de la figura 2 se buscan todos los posibles árboles de expansión que puedan construirse con los subconjuntos formados con los nodos requerimiento (en este ejemplo nodos 1, 2 y 3) y cualquier otro nodo del grafo. Los pesos de los árboles encontrados se muestran en la tabla I.

Tabla I. Árboles mínimos de expansión encontrados.

Árbol	Conjunto de nodos a cubrir	Peso del árbol
1	(1,2,3)	9
2	(1,2,3,4)	8
3	(1,2,3,5)	9
4	(1,2,3,6)	12
5	(1,2,3,7)	9

De los árboles encontrados se selecciona el de menor peso, para nuestro ejemplo el árbol mínimo es el 2 con un peso de 8, este árbol es el mínimo árbol de Steiner el cual se muestra en la figura 3.

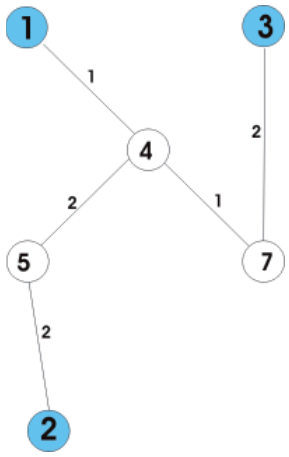


Fig. 3. Mínimo árbol de Steiner.

MÉTODO PARA ENCONTRAR EL ÁRBOL DE EMPUJE

Sea $T = (V, E)$ un árbol con origen en $s \in V$ y sea un conjunto de nodos requerimiento $R \subset V$. Removiendo del árbol T cualquier arista $e \in E$ se va a dividir al árbol en dos componentes. Denotemos V_e^1 al componente que contiene a s y sea V_e^2 el otro componente. Se define la tasa de requerimiento de una arista e como $\lambda(e) := \sum_{v \in V_e^2} r(v)$ .Si una arista e está en el ábol de empuje, este va a contribuir con $\mu l(e)$ al total de tráfico, si la arista e no está, contribuye con $\lambda(e) l(e)$. Si $\lambda(e) > \mu$ sería conveniente incluir a e en el árbol de empuje. Para cualquier $\mu > 0$, el conjunto de aristas e tal que $\lambda(e) > \mu$ forman un subárbol de T que contiene el nodo origen s, este subárbol entonces es el árbol de empuje óptimo para un μ dado.

Supongamos la red mostrada en la figura 3. Los números dentro de los nodos representan las correspondientes tasas de requerimiento y los números asociados con las aristas son los valores de $\lambda(e)$. Las longitudes de las aristas son irrelevantes para determinar el árbol de empuje óptimo, sin embargo para obtener el tráfico si se toman en cuenta. Se denota $w(e) = l(e) \cdot \min\{\lambda(e), \mu\}$ el tráfico que pasa a través de la arista e. El total del tráfico es entonces $w(t) = \sum_{e \in E} w(e)$.

En la figura 4 se muestran los niveles de tráfico que corresponden al árbol óptimo presentado en la figura 1 para diferentes valores de μ. La cantidad total de tráfico se observa en la gráfica de manera cóncava, no decreciente y parece que en algunas secciones se comporta de manera lineal con respecto a μ. La cantidad de tráfico generado por la función "jalar" es no decreciente. Cuando μ se incrementa el tamaño del árbol de empuje decrece.

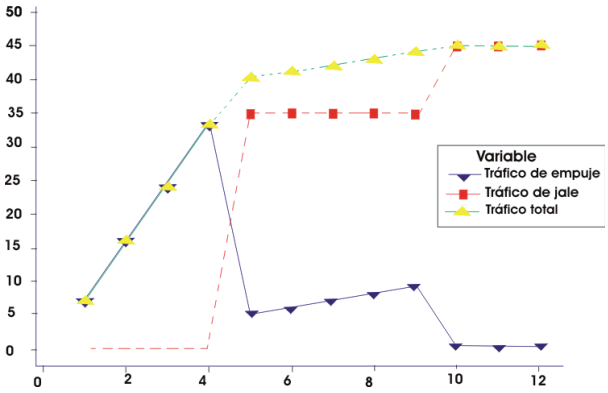


Fig. 4. Relación tráfico-tasa de actualización.

EXPERIMENTACIÓN COMPUTACIONAL

Para los casos-prueba que se han desarrollado para verificar la eficacia de la heurística implementada se utilizaron 2 herramientas computacionales: a) CPLEX 9.0 mediante su interfaz interactiva en un servidor V440 de 4 procesadores bajo el sistema operativo SolarisTM versión 9, ubicado en el Laboratorio de

Cómputo de Alto Rendimiento de la División de Posgrado en Ing. de Sistemas (PISIS) de la FIME, y b) lenguaje C con compilador GNU de Dev C++, en una laptop Dell Inspirion 9400 con 1 GB de memoria RAM y un procesador Intel Duo 1.66GHz, con sistema operativo Windows XP profesional. Para llevar a cabo nuestro análisis de comparación se crearon códigos en lenguaje C para obtener instancias con diferentes parámetros, dicho código genera: la cantidad de nodos, de aristas y su longitud, un conjunto de nodos requerimiento y su tasa de actualización; todo esto se incluye en un archivo de texto a partir del cual se construye un grafo de los siguientes datos: el número de nodos, número de aristas, lista de aristas con su respectivo peso, número de nodos requerimiento, nodo *s*, lista de nodos requerimiento con su respectiva tasa de actualización.

Se generaron grafos de 50, 100 y 150 nodos con 5%, 10% y 15% nodos requerimiento del total de nodos, todos los grafos con densidad del 10%. Para cada combinación se obtuvieron 3 distintas instancias. Los resultados se muestran en la tabla II. Las primeras dos columnas muestran la información de cada instancia (el número de nodos y la cantidad de nodos requerimiento, representado por un porcentaje del total de nodos). En la tercera columna se presenta el tiempo en segundos que tardó CPLEX para llegar a la solución mostrada en la

Tabla II. Resultados de la experimentación.

Nodos	% de nodos requerimiento	Cplex				Solución Heurística
		Tiempo(seg)	Solución	Cota	IOR(%)	
50	5	1.54	61	61	0.00%	61
50	5	1.36	35	35	0.00%	35
50	5	0.8	36	36	0.00%	36
50	10	3.3	42	42	0.00%	43
50	10	147.68	83	83	0.00%	87
50	10	1.78	45	45	0.00%	50
50	15	6.67	60	60	0.00%	61
50	15	6.58	61	61	0.00%	61
50	15	713.81	95	95	0.00%	98
100	5	2003.55	49	49	0.00%	50
100	5	695.26	34	34	0.00%	36
100	5	1996.41	40	40	0.00%	44
100	10	3600.02	47	42.2	10.17%	47
100	10	3600.01	52	34.2	34.21%	55
100	10	3600.02	52	34.5	33.58%	59
100	15	3600.01	66	44.5	32.45%	70
100	15	3600.02	74	50.1	32.27%	79
100	15	3600.02	83	60.8	26.70%	89
150	5	3600.03	36	19.4	46.03%	39
150	5	3600.02	42	21.1	49.70%	47
150	5	3600.02	47	26.6	43.24%	49
150	10	3600.02	64	43.4	32.05%	70
150	10	3600.01	76	35.7	53.02%	71
150	10	3600.01	83	42.0	49.33%	82
150	15	3600.06	99	53.5	45.92%	109
150	15	3600.01	83	40.2	51.56%	90
150	15	3600.02	79	47.4	39.90%	92

cuarta columna, esta representa el costo en unidades para satisfacer la tasa de actualización de los nodos requerimiento. En la quinta columna podemos ver la cota inferior encontrada por CPLEX de cada uno de los problemas. En la sexta columna se observa el intervalo de optimalidad relativa (IOR) alcanzado para cada uno de los problemas, es decir, la diferencia relativa entre la mejor solución factible y la mejor cota inferior encontradas por CPLEX. En dicha columna, un valor de 0.00% indica que la solución de CPLEX es óptima. En la última columna se observa el valor de la solución encontrado por la heurística que se implementó. El tiempo de ejecución de la heurística no se muestra ya que es relativamente pequeño (menos de 5 segundos en cualquiera de las instancias), comparado con el de CPLEX.

Podemos observar que los resultados obtenidos por la heurística para problemas de 50 nodos y 5% nodos requerimiento la solución encontrada es óptima mientras que para los de 10% nodos requerimiento la solución de la heurística difiere de la óptima en 3.3 unidades en promedio, y 1.3 unidades promedio para las de dimensión de 15% nodos requerimiento; debemos aclarar que el tiempo de ejecución para encontrar la solución óptima por CPLEX depende en gran medida de la estructura del grafo.

Además para casos pruebas con 100 y 150 nodos ya sea con 5, 10 y 15% nodos requerimiento el tiempo que usa CPLEX para obtener una solución es grande, a comparación de la heurística que da una solución en tiempo razonable (menos de 5 segundos para las instancias más grandes), la cual puede ser usada como una buena cota superior para estos problemas.

CONCLUSIONES

Se pudo constatar que empleando la heurística desarrollada se obtuvieron buenos resultados en un tiempo razonablemente pequeño. La aplicación de este método es de gran ayuda en el área de telecomunicaciones para minimizar el tráfico en las redes de información.

AGRADECIMIENTOS

Los primeros cuatro autores agradecen al CONACYT por su apoyo económico como becarios del Programa de Maestría en Ciencias en Ing. de Sistemas de la FIME.

REFERENCIAS

1. F. Havet y M. Wennink. The push tree problem. Networks, 44(4):281-291, 2004.
2. ILOG, S.A. CPLEX 9.0 Online Documentation, 2003. http://yalma.fime.uanl.mx/cplex-manual/
3. R.K. Ahuja, T.L. Magnanti y J.B. Orlin. Network Flows. Prentice-Hall, Englewood Cliffs, EUA, 1993.
4. M.R. Garey y D.S. Johnson. Computers and Intractability: A Guide to the Theory of NP-Completeness. W.H. Freeman and Company. New York, EUA. 1979.

APÉNDICE

En este apartado se incluye la definición de algunos términos de teoría de grafos y redes usados durante la exposición del trabajo.

Ruta: Una secuencia de nodos adyacentes $i_1 - i_2 - \ldots - i_r$ en un grafo en la cual no hay repetición de nodos.

Ciclo: Ruta $i_1 - i_2 - \ldots - i_r$ agregándole la arista (i_r, i_1).

Nodo hoja: Vértice de un grafo que tiene solo una arista incidente.

Matriz de incidencia nodo-nodo: Representación del grafo en una matriz de $n \times n$ $H = \{h_{ij}\}$ donde h_{ij} es 1 si la arista $(i, j) \in A$ y 0 en otro caso.

Densidad de un grafo: Cociente entre el número m de aristas y el número de aristas del grafo completo de n vértices. Se denota como: $d = \frac{2m}{n(n-1)}$

Árbol de expansión: Grafo conexo sin ciclos que abarca todos los nodos de una red no dirigida. Su costo es la suma de los costos de cada uno de los arcos que lo conforman.

Mínimo árbol de expansión: Árbol de expansión de costo mínimo.

Problema del Árbol de Steiner: Dado un subconjunto $S \subseteq N$ de nodos, llamados nodos requerimiento. Se desea determinar el árbol de mínimo costo (no necesariamente árbol de expansión) que debe contener todos los nodos en S y opcionalmente, algunos nodos en $N \backslash S$.