UNIVERSIDAD AUTÓNOMA DE QUERÉTARO

FACULTAD DE INGENIERÍA

EL MODELO DE FLUJO MÁXIMO Y SU DUAL EN EL ANÁLISIS DE REDES DE TRANSPORTE

TESIS

QUE COMO PARTE DE LOS REQUISITOS PARA OBTENER EL GRADO DE

LICENCIADA EN MATEMÁTICAS APLICADAS

PRESENTA

MARÍA FERNANDA FLORES JUÁREZ

DIRIGIDA POR

DR. ERIC MORENO QUINTERO

SANTIAGO DE QUERÉTARO, QUERÉTARO, 2019

UNIVERSIDAD AUTÓNOMA DE QUERÉTARO

FACULTAD DE INGENIERÍA

LICENCIATURA EN MATEMÁTICAS APLICADAS

EL MODELO DE FLUJO MÁXIMO Y SU DUAL EN EL ANÁLISIS DE REDES DE TRANSPORTE

TESIS

Que como parte de los requisitos para obtener el grado de Licenciada en Matemáticas Aplicadas

Presenta
María Fernanda Flores Juárez

Dirigida por
Dr. Eric Moreno Quintero

Dr. Eric Moreno Quintero Presidente	Firma
MC. Luisa Ramírez Granados Secretaria	Firma
MC. Verónica Josefina Soria Anguiano Vocal	Firma
LMA. Berenice Medina Martínez Suplente	Firma

Centro Universitario
Querétaro, Qro.
Octubre 2019
México

RESUMEN

La vida diaria puede ser en muchos sentidos ajustada como sistema de gráficas y redes. El cuerpo humano consiste en sistemas interconectados que pueden modelarse, la sociedad requiere de redes de comunicación y transporte, Internet es una red que conecta a las computadoras alrededor del mundo, las manufactureras requieren redes de fabricación y suministro, las plataformas sociales tienen su base en una red de contactos entre usuarios, recursos como agua y petróleo requieren de una red de tuberías, por mencionar algunos.

Uno de los problemas más importantes en el campo de la optimización que utiliza teoría de redes, es el problema de flujo máximo, el cual consiste en determinar la máxima cantidad de flujo en una red dirigida desde el nodo fuente hasta el nodo destino respetando las capacidades de los arcos, donde todo flujo a través de la red se deriva del nodo origen y termina en el nodo destino.

Debido a la importancia del modelo de red y lo que puede fungir en ella como un flujo, en el presente se aborda el problema de Flujo Máximo y su dual; el problema de Corte Mínimo. En primer capítulo se introducen los conceptos básicos de teoría de gráficas y teoría de redes, continuando en el segundo con los antecedentes del problema y la representación de una red como programa lineal, dando énfasis en las propiedades de la matriz de restricciones correspondiente en la red. Se sigue con la importancia del dual en el problema de Flujo Máximo, para concluir la sección presentando el Teorema de Flujo Máximo-Corte Mínimo.

Buscando que el lector pueda aplicar lo antes explicado, el tercer capítulo aborda los métodos de solución por programación lineal y el algoritmo de Ford y Fulkerson, así como instrucciones para su resolución en software libre. El cuarto capítulo consta de diversas modificaciones al problema de flujo máximo, siendo éstos múltiples fuentes y destinos, capacidades en los nodos, cotas inferiores en los arcos y Flujo Máximo a Costo Mínimo. Se expone entonces un ejemplo real aplicado a la red ferroviaria mexicana, donde se encontró un flujo máximo representativo de la carga ferroviaria que se mueve entre Estados Unidos y el centro de México, así como el corte mínimo que los desconecta. La sección cierra con aplicaciones encontradas en la literatura. En cada tópico se exponen ejemplos tanto de creación propia como encontrada diversas fuentes.

Se concluye retomando brevemente la utilidad del modelar problemáticas como redes, la relevancia del problema de flujo máximo y su dual, así como breves consideraciones de problemas y aplicaciones relacionados con los tópicos de esta tesis que no fue posible cubrir pero que pueden resultar de interés para trabajos futuros.

SUMMARY

Daily life can be adjusted in many ways as a system of graphics and networks. The human body consists of interconnected systems that can be modeled, society requires communication and transport networks, the Internet is a network that connects computers around the world, manufacturers require manufacturing and supply networks, Social platforms are based on a network of contacts between users, resources such as water and oil require a network of pipes, to name a few.

One of the most important problems in the field of optimization using network theory is the maximum flow problem, which consists in determining the maximum amount of flow in a network directed from the source node to the destination node respecting the capabilities of the arcs, where all flow through the network is derived from the origin node and ends at the destination node.

Due to the importance of the network model and what can function in it as a flow, the problem of Maximum Flow and its dual; the Minimum Cut problem are addressed in this thesis. In the first chapter the basic concepts of graph theory and network theory are introduced, continuing in the second with the background of the problem and the representation of a network as a linear program, emphasizing the properties of the corresponding constraint matrix in the network . The importance of the dual in the problem of Maximum Flow continues, to conclude the section presenting the Maximum Flow-Minimum Cut Theorem.

Seeking that the reader can apply the above, the third chapter discusses methods of solution by linear programming and Ford and Fulkerson´s algorithm, as well as instructions for resolution in free software. The fourth chapter consists of several modifications to maximum flow´s problem, considering multiple sources and destinations, capacities in nodes, lower levels in the arcs and Maximum Flow at Minimum Cost. Then, a real example then is applied to the Mexican rail network, where a maximum possible flow representative of the freight moved was found between the United States and central Mexico, as well as the minimum cut that disconnects them. The section closes with applications found in the literature. In each topic examples of both own creation and found various sources are presented.

It is concluded by briefly recalling the usefulness of modeling problems such as networks, the relevance of maximum and dual flow problem, as well as brief considerations of problems and applications related to the topics of this thesis that could not be covered but that may be of interest to future work.

DEDICATORIAS

A mis padres, hermanos y amigos.
A mi pareja, Oscar Jordan Parra,
quién pasó de compañero de clase
a compañero de vida.

"Escuchad mis palabras, sed testigos de mi juramento:

La noche se avecina, ahora empieza mi guardia. No terminará
hasta el día de mi muerte. No tomaré esposa, no poseeré tierras,
no engendraré hijos. No llevaré corona, no alcanzaré la gloria.
Viviré y moriré en mi puesto. Soy la espada en la oscuridad.
Soy el vigilante del Muro. Soy el fuego que arde contra el frío,
la luz que trae el amanecer, el cuerno que despierta a los durmientes,
el escudo que defiende los reinos de los hombres.
Entrego mi vida y mi honor a la Guardia de la Noche,
durante esta noche y todas las que estén por venir."

Juramento de la Guardia de la Noche. GEORGE R.R. MARTIN

AGRADECIMIENTOS

En primera instancia, quiero agradecer a mis padres, por su apoyo incondicional durante mis estudios y por ser mi ejemplo de vida. Gracias a mis hermanos por acompañarme, a mis hermanas por su alegría, a mis suegros por sus atenciones, a mis amigos por su empatía, a mis maestros por todo lo aprendido, a mis alumnos y asesorados por lo que me han enseñado de mí misma.

Agradezco al Instituto Mexicano del Transporte, por aceptarme como practicante y posteriormente como tesista, así como a los Doctores y compañeros que me prestaron su apoyo y ayuda. Gracias también a mi Alma Máter y su planta docente, que me llevó a una comprensión y gusto por las matemáticas más allá de la que esperé tener.

Encarecidas gracias a mi asesor de tesis, el Dr. Eric Moreno, por su paciencia y profesionalismo que permitió llevar este proyecto a término.

Finalmente, mis más grandes agradecimientos y respetos a mi pareja Oscar Jordan Parra, quién ha convertido esta etapa en digna de ser vivida.

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

1. INTRODUCCIÓN

1.1 Conceptos básicos de Teoría de Gráficas

A fin de abordar los conceptos de Teoría de Redes, en la que se fundamentará el presente trabajo, es necesario iniciar con las nociones de Teoría de Gráficas, la cual tiene sus cimientos en la Matemática Discreta.

Definición 1.1.1.- Una gráfica $G = (V, A)$ es una estructura combinatoria constituida por un conjunto $V = V(G)$ de elementos llamados vértices (o nodos) y un conjunto $A = A(G)$ de pares no ordenados de vértices distintos llamados aristas (o arcos).

Históricamente, el concepto de gráfica surge con el problema de los siete puentes de Königsberg, clásico problema matemático resuelto por Leonard Euler en 1736 y cuya resolución se considera dio origen a la teoría de gráficas. Su nombre se deriva de la ciudad de Königsberg, actualmente Kaliningrado, en Rusia. La ciudad estaba dividida en cuatro secciones por el río Pregel, y estaban unidas por siete puentes como se muestra en la figura 1.1.1 a continuación.

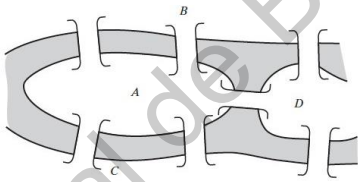


Figura 1.1.1 Problema de los puentes de Königsberg [1]

La cuestión a resolver era la posibilidad de cruzar los siete puentes una única vez (pudiendo pasar por las cuatro zonas de la ciudad más de una vez si era necesario) y regresar al punto de inicio.

La importancia de la formulación de Euler fue enfocarse meramente en las zonas y en los puentes, planteando el problema como gráfica, como se ve en la figura 1.1.2.

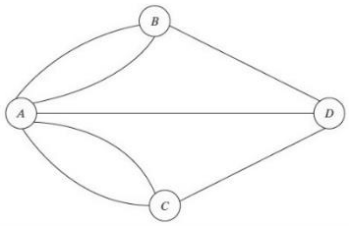


Figura 1.1.2 Representación gráfica al problema de los puentes de Königsberg[2]

[1] Taha, H. (2012), Puentes de Königsberg. Ilustración. Investigación de Operaciones, 9na edición.
[2] Taha, H.,2012, Representación en forma de red al problema de puentes de Königsberg. Figura. Investigación de Operaciones, 9na edición.

La respuesta encontrada fue negativa, ya que Euler se fundamentó en el hecho de que es necesario llegar a cada una de las cuatro zonas, y al llegar a cualquiera de ellas por un camino, debe forzosamente salir por otro, además, debe volverse al mismo punto de инicio, por lo que cada vértice debe contar con un número par de aristas, lo que no se cumple en este problema.

En la ciudad de Kaliningrado quedan algunos de los puentes originales. Dos de ellos fueron destruidos durante la Segunda Guerra Mundial, sólo cinco conectan a la ciudad actualmente; además, de éstos, sólo dos corresponden a los puentes originales. Las figuras 1.1.3 y 1.1.4 ilustran la zona de puentes que motivó el célebre problema en un mapa reciente de Kaliningrado.

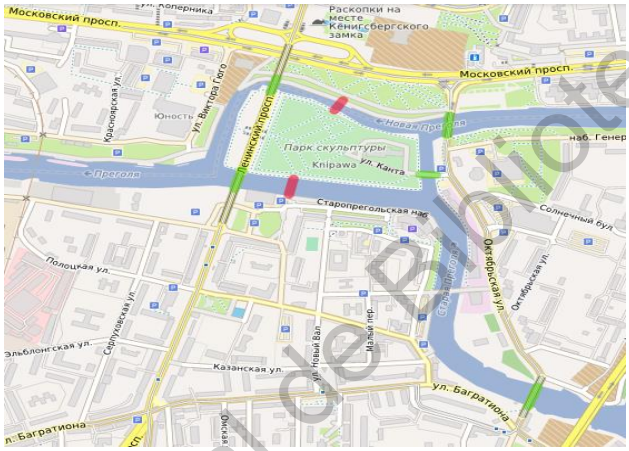


Figura 1.1.3 Actual ciudad de Kaliningrado; los puentes que se conservan del planteamiento de Euler marcados con verde y los faltantes con rojo [3]

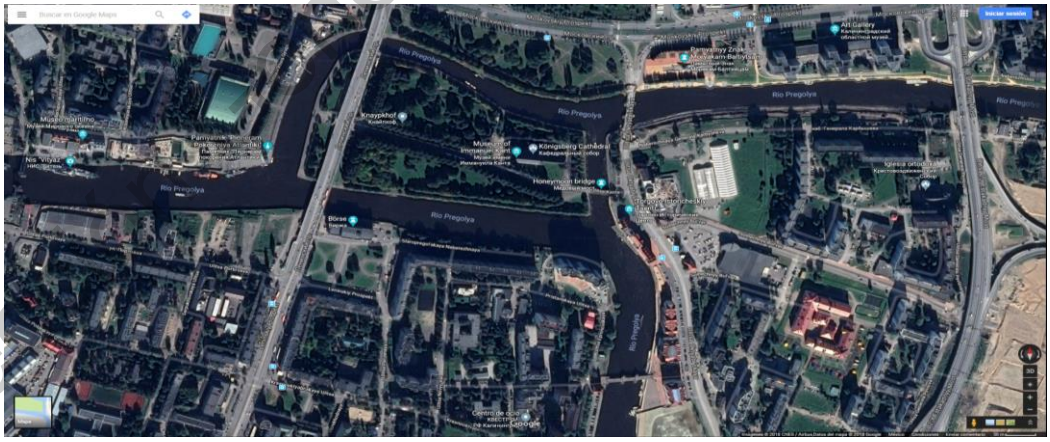


Figura 1.1.4 Foto satelital de la actual Kaliningrado[4]

[3] Fundación OpenStreetMap (OSMF)(s.f.). [Mapa de Kaliningrado, Rusia en OpenStreetMap]. Recuperado el 13 de Abril, 2018, de: https://www.openstreetmap.org/node/27048976#map=16/54.7066/20.5105.

[4] Google. (s.f.). [Foto satelital de Kaliningrado, Rusia en Google maps]. Recuperado el 13 de Abril, 2018, de: https://www.google.com.mx/maps/@54.7058266,20.5133849,2074m/data=!3m1!1e3

Una vez establecido el concepto de gráfica, es conveniente caracterizar esta estructura definiendo posibles formas en que puede representarse.

Definición 1.1.2.- Si la arista $a = (u, v) = uv$ relaciona los vértices u y v, se dice que u y v son vértices adyacentes y también que el vértice u (o bien v) y la arista a son incidentes. De otro modo, los vértices se llaman independientes. Las aristas $a = uv$ y $b = wz$ son aristas independientes si no tienen nodos en común, es decir, $(u, v) \cap (w, z) = \emptyset$.

Definición 1.1.3.- El orden de la gráfica G es número de nodos que contiene y se denota por $|V(G)|$.

Definición 1.1.4.- El número de aristas de G se denota $|A(G)|$ y este número es su tamaño. Un ejemplo se muestra en la figura 1.1.5.

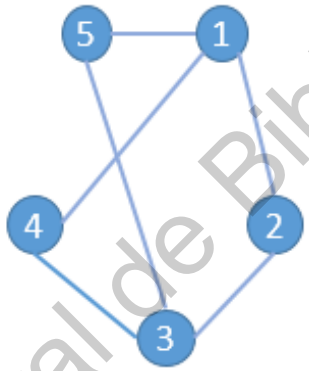


Figura 1.1.5 Ejemplo de gráfica de orden 5 y tamaño 6

Definición 1.1.5.- Una arista de la forma (i, i) donde $i \in A$, se denomina bucle o rizo.

Definición 1.1.6.- Los arcos son aristas que poseen dirección (representada por una flecha) y se dice que la gráfica G es dirigida u orientada.

Definición 1.1.7.- Sea una gráfica $G = (V, A)$ con $v \in V$, el grado de entrada del vértice v se define como $d^-(v) = |\Gamma^-(v)| = |\{w \in V : (w, v) \in A\}|$

Definición 1.1.8.- Sea una gráfica $G = (V, A)$ con $v \in V$, el grado de salida del vértice v se define como $d^+(v) = |\Gamma^+(v)| = |\{w \in V : (v, w) \in A\}|$

Ejemplos de los últimos dos conceptos se muestran en las figuras 1.1.6 y 1.1.7.

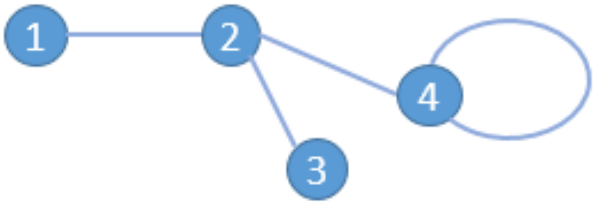


Figura 1.1.6 Gráfica con bucle en el nodo 4

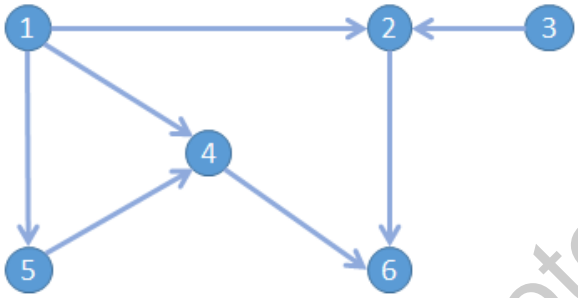


Figura 1.1.7 Ejemplo de gráfica dirigida, donde $d^{+}(4) = 1$, $d^{-}(4) = 2$

Definición 1.1.9.- Una gráfica $G' = (V', A')$ es un subgráfica de $G = (V, A)$ si $V' \subseteq V$ y $A' \subseteq A$. Cuando $V = V'$, la subgráfica G' se llama subgráfica generadora de G. Análogamente, dada una subgráfica $G' = (V', A')$ que contiene todas las aristas que unen en G a dos nodos de V', se dice que G' es un gráfica inducida por V' y se denota por $G[V']$. Ejemplos de esto se muestran las figuras 1.1.8 y 1.1.9 enseguida.

Figura 1.1.8 La gráfica *B* es subgráfica de *A*

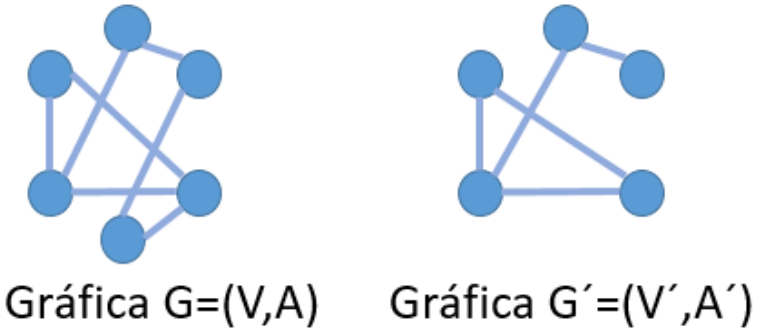


Figura 1.1.9 La subgráfica *G´* es una gráfica inducida por *V´*

Definición 1.1.10.- Dada una gráfica $G = (V, A)$, una secuencia de nodos $a_0, a_1, \ldots, a_n$ donde $(a_{i-1}, a_i) \in A$, $1 \leq i \leq n$, y $(a_{i-1}, a_i) \neq (a_{j-1}, a_j)$ si $i \neq j$ se llama recorrido *R* de longitud n entre a_0 y a_n. Se hace notar que en un recorrido, todas las aristas de éste son distintas.

Definición 1.1.11.- Un circuito es un recorrido cerrado, es decir, un recorrido en el cual $a_0 = a_n$. Cuando todos los nodos de R son distintos se tiene un camino, y un ciclo es un camino cerrado. La gráfica de la figura 1.1.10 muestra un circuito y un ciclo.

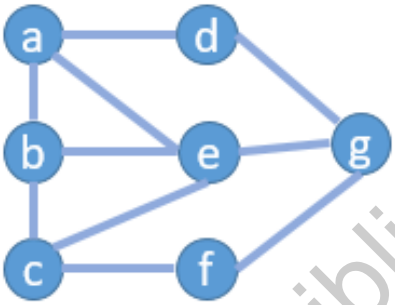


Figura 1.1.10 En la gráfica, la secuencia *a,d,g,e,b,c,e,a* es un circuito, mientras *a,d,g,e,a* es un *ciclo*

Definición 1.1.12.-Un *ciclo Hamiltoniano* es un ciclo que usa todos los vértices de la gráfica solamente una vez, un ejemplo se muestra en la figura 1.1.11.

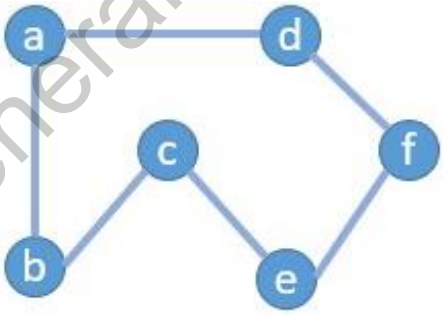


Figura 1.1.11 La secuencia *a,b,c,e,f,d,a* es un ejemplo de ciclo Hamiltoniano

Definición 1.1.13.- Una *gráfica conexa* es aquélla en la cual entre todo par de nodos existe un camino. Un ejemplo se ve en la figura 1.1.12.

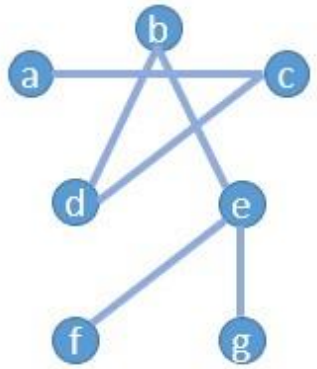


Figura 1.1.12 Ejemplo de gráfica conexa

Definición 1.1.14.- Una gráfica G *no conexa* consta de dos o más componentes G_i donde cada $G_i = (V_i, A_i)$ es una subgráfica inducida que es conexa y maximal respecto de esta propiedad, en el sentido que, si $w \in V \setminus V_i$, entonces la subgráfica inducida $G[V_i \cup \{w\}]$ es no conexa. En la figura 1.1.13, se observa la gráfica obtenida al suprimir el arco dc, obteniendo una gráfica G´ no conexa con dos componentes.

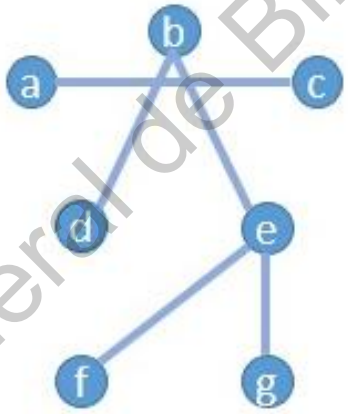


Figura 1.1.13 Gráfica no conexa

Definición 1.1.15.- Se dice que el vértice $w \in V(G)$ es un *vértice de corte* si el número de componentes de $G - \{w\}$ es mayor al número de componentes de G. En particular, si G es conexa, la supresión de un nodo de corte desconecta G en dos o más componentes. En la gráfica de la figura1.1.12, c es un vértice de corte.

Definición 1.1.16.- El *diámetro* es la longitud del recorrido más corto para unir los dos vértices más alejados de la gráfica. Por ejemplo, en la figura 1.1.12, hay dos parejas de vértices que cumplen ser las más alejadas entre sí respectivamente; $\{a, f\}$ y $\{a, g\}$, siendo el recorrido más corto de longitud 5 en ambos casos, luego, el diámetro de la gráfica de la figura 1.1.12 es 5.

1.2 Conceptos básicos de la Teoría de Redes

El concepto de red surge cuando las aristas de una gráfica tienen asociados pesos que representan magnitudes tales como distancias, costos, capacidades, flujos de datos o de energía, demandas, etc. Así, la Teoría de Redes tiene sus antecedentes en la Teoría de Gráficas. La idea de red se ha utilizado en diversidad de áreas tales como la Economía, la Biología, la Electrónica, las Comunicaciones, la Administración y las Ciencias Sociales, por mencionar algunas, donde el análisis de los sistemas de interés se realizan mediante el uso de redes, lo que se debe a la capacidad de representación y organización de información que estos modelos ofrecen en su estructura.

Para muestra, se tienen los siguientes ejemplos:

Ejemplo 1.2.1.- La figura 1.2.1 da la distancia en millas de los vínculos factibles que conectan nueve cabezales de pozos de gas natural localizados a una cierta distancia de la costa con un punto de distribución. En este caso, las aristas representan las conexiones entre pozos, y los pesos de la red corresponden a las distancias.

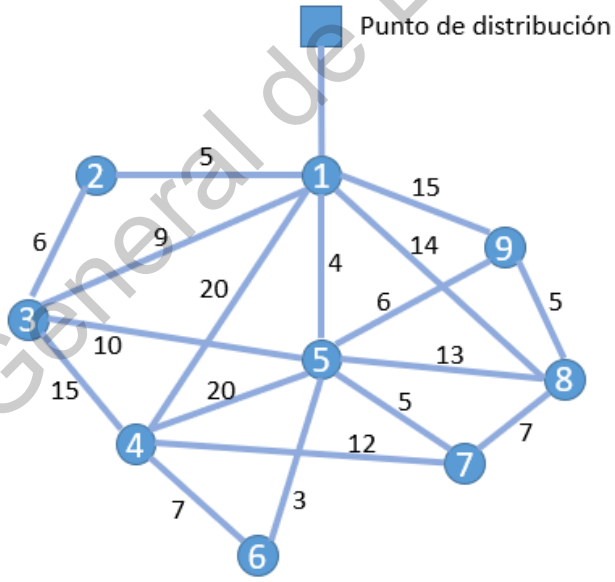


Figura 1.2.1 Red correspondiente al ejemplo 1.2.1 [5]

Ejemplo 1.2.2.-

Considérese la siguiente red donde el número asociado a cada arco es la altura máxima (cientos de metros) sobre el nivel del mar que se registra en la carretera que une a las ciudades (vértices). Por ejemplo, entre las ciudades 2 y 5 la máxima altura

[5] Taha, H.,2012. FIGURA 6.9, Red para el problema 4, conjunto 6.2a. Figura. Investigación de Operaciones, 9na edición.

que se registra es 12 (i.e. 1200 metros sobre el nivel del mar). Así, en este caso los pesos asociados corresponden a alturas.

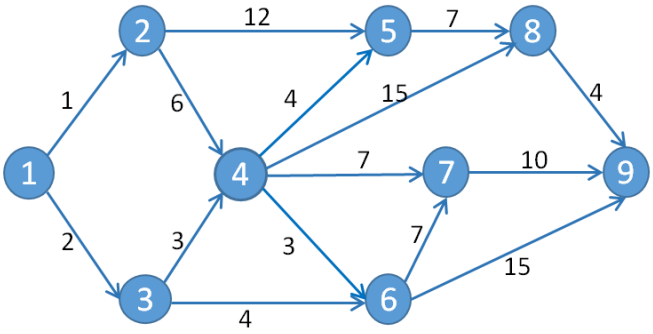


Figura 1.2.2 Red correspondiente al ejemplo 1.2.2[6].

Definición 1.2.1.- Una red $R = [G, s, t, q]$ es una gráfica dirigida $G = [V(G), A(G)]$ con s el vértice fuente (source), t (target) el vértice destino y q una función llamada capacidad tal que $q: A(G) \rightarrow \{0\} \cup \mathbb{R}^+$. Lo siguiente es considerar una red de flujo, donde un flujo factible en una red $R = [G, s, t, q]$ es una función $x: A(G) \rightarrow \{0\} \cup \mathbb{R}^+$ tal que:

i) $$\sum_{j \in \Gamma^+(i)} x(i,j) - \sum_{k \in \Gamma^-(i)} x(k,i) = \begin{cases} v, si\ i = s \\ 0, si\ i \neq s, t \\ -v, si\ i = t \end{cases}$$

ii) $$0 \leq x(i,j) \leq q(i,j), para\ todo\ (i,j) \in A(G)$$

A v se le llama valor del flujo x y a las ecuaciones i), ii) se les conoce como ecuaciones de conservación del flujo. Se hace notar que en la ecuación i), ya que no hay demandas en los nodos de paso, el flujo de entrada es igual al de salida, por lo que la suma del flujo entrante menos la suma del flujo que sale es igual 0 si $i \neq s, t$.

1.3 Representación matricial de una Red

Al trabajar con redes, se toman en cuenta dos características importantes: la topología de la red (que se refiere a la estructura de la red: aristas y nodos) y los pesos tales como costos, capacidades, ofertas y demandas. Las matrices por su estructura constituyen una importante herramienta computacional al momento de hacer representaciones de redes y aplicar algoritmos. Se considerarán dos matrices asociadas a una gráfica, la matriz de incidencia nodos-arcos y la matriz de adyacencia nodo a nodo.

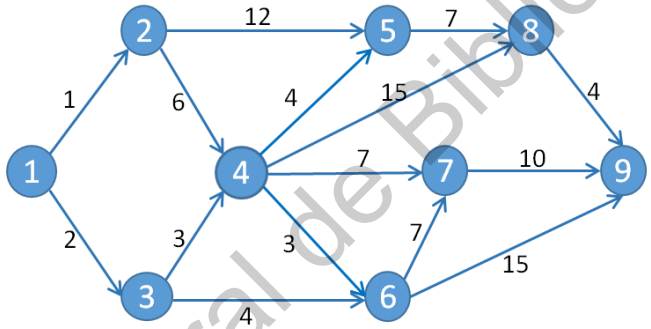
[6] Hernández, M., 2005. Red para el problema 13, ejercicios 2.4. Figura. Introducción a la Teoría de Redes, 2da edición.

Definición 1.3.1.- La matriz de incidencia es una matriz de $n \times m$ donde cada fila corresponde a un nodo y cada columna a una arista, las entradas de la matriz son -1, 0 ó 1 según la condición del arco que entra o sale del nodo. Así, para el arco $a = (i, j) \in A(G)$, la columna de la matriz de incidencia tiene una entrada con +1 si la fila es la correspondiente al nodo i, -1 si es la correspondiente al nodo j y 0 en otro caso.

Es decir;

$$B_{ij} = \begin{cases} 1, si\ v_i, a_j = (u_1, u_2) son\ incidentes\ y\ v_i = u_1 \\ -1, si\ v_i, a_j = (u_1, u_2) son\ incidentes\ y\ v_i = u_2 \\ 0, otro\ caso \end{cases}$$

Por ejemplo, retomemos la red de la figura 1.2.2, la matriz B corresponde a su matriz de incidencia.



$$\boldsymbol{B} = \begin{array}{c} \\ \mathbf{1} \\ \mathbf{2} \\ \mathbf{3} \\ \mathbf{4} \\ \mathbf{5} \\ \mathbf{6} \\ \mathbf{7} \\ \mathbf{8} \\ \mathbf{9} \end{array} \begin{bmatrix} \mathbf{(1,2)} & \mathbf{(1,3)} & \mathbf{(2,4)} & \mathbf{(2,5)} & \mathbf{(3,4)} & \mathbf{(3,6)} & \mathbf{(4,5)} & \mathbf{(4,6)} & \mathbf{(4,7)} & \mathbf{(4,8)} & \mathbf{(5,8)} & \mathbf{(6,7)} & \mathbf{(6,9)} & \mathbf{(7,9)} & \mathbf{(8,9)} \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & -1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & -1 \end{bmatrix}$$

Consideremos ahora el caso más general.

Definición 1.3.2.- Cuando la red es no dirigida, con $V(G) = \{v_1, v_2, \dots, v_n\}$ y $E(G) = \{e_1, e_2, \dots, e_m\}$, la matriz de incidencia se define como:

$$B_{ij} = \begin{cases} 1, si\ v_i\ es\ incidente\ con\ e_j \\ 0, de\ otro\ modo \end{cases}$$

Definición 1.3.3.- Una matriz de adyacencia nodo a nodo es una matriz de $n \times n$ donde n es el número de nodos de la red y muestra la relación entre nodos. Es

importante diferenciar esta matriz en el caso de redes dirigidas y las no dirigidas. En el caso de una red dirigida;

$$A_{ij} = \begin{cases} 1, si \;\; a = (i,j) \in A(G) \\ 0, otro\ caso \end{cases}$$

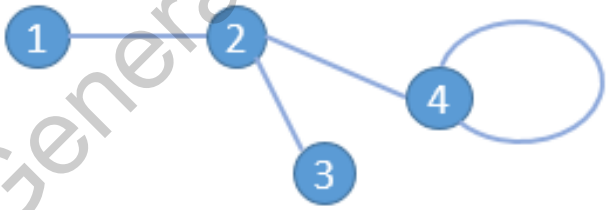
Así, A es la matriz de adyacencia de la red de la figura 1.2.2;

$$\boldsymbol{A} = \begin{array}{c} \\ \mathbf{1} \\ \mathbf{2} \\ \mathbf{3} \\ \mathbf{4} \\ \mathbf{5} \\ \mathbf{6} \\ \mathbf{7} \\ \mathbf{8} \\ \mathbf{9} \end{array} \begin{array}{c} \begin{bmatrix} \mathbf{1} & \mathbf{2} & \mathbf{3} & \mathbf{4} & \mathbf{5} & \mathbf{6} & \mathbf{7} & \mathbf{8} & \mathbf{9} \end{bmatrix} \\ \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

En el caso de una red no dirigida;

$$A_{ij} = \begin{cases} 1, si\ a = (i,j) \in A(G)\ ó\ a = (j,i)\ con\ i \neq j \\ 2, si\ a = (i,i) \in A(G) \\ 0, otro\ caso \end{cases}$$

Ahora, retomando la red de la figura 1.1.6, *C* es su matriz de adyacencia;



$$C = \begin{array}{c} \\ \mathbf{1} \\ \mathbf{2} \\ \mathbf{3} \\ \mathbf{4} \end{array} \begin{array}{c} \begin{bmatrix} \mathbf{1} & \mathbf{2} & \mathbf{3} & \mathbf{4} \end{bmatrix} \\ \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 2 \end{bmatrix} \end{array}$$

Por ejemplo, si reconsideramos la matriz de adyacencia de la red de la figura 1.2.2:

$$M = \begin{array}{c} \\ \mathbf{1} \\ \mathbf{2} \\ \mathbf{3} \\ \mathbf{4} \\ \mathbf{5} \\ \mathbf{6} \\ \mathbf{7} \\ \mathbf{8} \\ \mathbf{9} \end{array} \begin{array}{c} \begin{array}{ccccccccc} \mathbf{1} & \mathbf{2} & \mathbf{3} & \mathbf{4} & \mathbf{5} & \mathbf{6} & \mathbf{7} & \mathbf{8} & \mathbf{9} \end{array} \\ \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

Tenemos que:

$$M^2 = \begin{array}{c} \\ \mathbf{1} \\ \mathbf{2} \\ \mathbf{3} \\ \mathbf{4} \\ \mathbf{5} \\ \mathbf{6} \\ \mathbf{7} \\ \mathbf{8} \\ \mathbf{9} \end{array} \begin{array}{c} \begin{array}{ccccccccc} \mathbf{1} & \mathbf{2} & \mathbf{3} & \mathbf{4} & \mathbf{5} & \mathbf{6} & \mathbf{7} & \mathbf{8} & \mathbf{9} \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 2 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 2 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 3 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

Esta última matriz tiene en sus entradas el número de caminos de longitud 2 entre los nodos, por ejemplo, el número de caminos entre 4 y 7 de longitud 2 es 2, a saber, el camino $4 \to 6 \to 7$ y el $7 \to 6 \to 4$, recordando que para encontrar los caminos se debe considerar a la red como una gráfica no dirigida.

Más propiedades importantes de la red respecto a la matriz de adyacencia, se muestran a continuación.

Definición 1.3.4.- Un isomorfismo entre dos gráficas G y H es una biyección f entre los conjuntos de sus vértices; $f: V(G) \to V(H)$ que preserva la relación de adyacencia. Es decir, cualquier par de vértices u y v de G son adyacentes si y sólo si lo son sus imágenes, $f(u)$ y $f(v)$, en H.

Definición 1.3.5.-La matriz permutación es la matriz cuadrada con todos sus $n \times n$ elementos iguales a 0, excepto uno cualquiera por cada fila y columna, el cual debe ser igual a 1.

Teniendo estas dos definiciones, se establece el siguiente Teorema:

Teorema 1.3.1.- Dos gráficas G y H son isomorfos si y sólo si existe una matriz de permutación P tal que

$$A(H) = P^{-1}A(G)P$$

Demostración.-

$\Rightarrow$)

Supongamos que G y H son isomorfos, luego, siendo $A(H)$ y $A(G)$ las matrices de adyacencia respectivas, una de ellas puede obtenerse de la otra por reordenamiento de filas y posteriormente por reordenamiento de columnas. Reordenar filas de $A(G)$ es equivalente a premultiplicar por una matriz de permutación P obteniendo $PA(G)$. El subsecuente reordenamiento de las columnas correspondientes equivale a multiplicar $PA(G)$ por P^{-1}. (Nótese que cualquier matriz de permutación P es no singular). Luego,

$$A(H) = P^{-1}A(G)P$$

$\Leftarrow$)

Se tiene

$$A(H) = P^{-1}A(G)P$$

De donde $A(H)$ puede ser obtenida de $A(G)$ por reordenamiento de columnas y luego de filas, por tanto H y G son isomorfas.[7] ■

Ejemplo 1.3.1.-

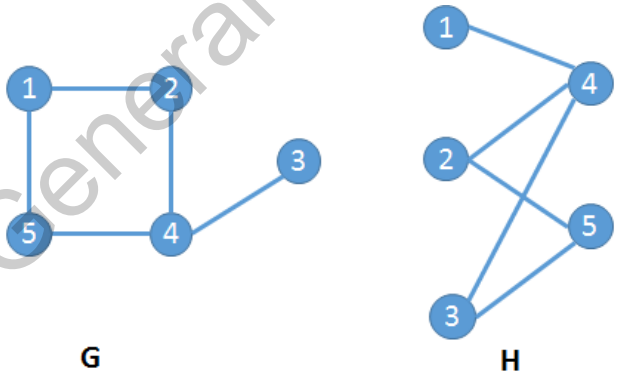


Figura 1.3.1 Gráficas G y H[8]

De las gráficas G y H, se tienen las matrices de adyacencia:

[7] Balakrishnan, V.K., (1997), *Shaum´s outline of theory and problems of Graph Theory*, 1ra edición, New York, Shaum´s outline series, McGRAW-HILL.
[8] Balakrishnan, V.K., (1997). Fig. 1-18. Figura. *Shaum´s outline of theory and problems of Graph Theory*, 1ra edición. New York, Shaum´s outline series, McGRAW-HILL.

$$A(G)=\begin{bmatrix}0&1&0&0&1\\1&0&0&1&0\\0&0&0&1&0\\0&1&1&0&1\\1&0&0&1&0\end{bmatrix}, A(H)=\begin{bmatrix}0&0&0&1&0\\0&0&0&1&1\\0&0&0&1&1\\1&1&1&0&0\\0&1&1&0&0\end{bmatrix}$$

Un isomorfismo f de G a H es $f(1)=5, f(2)=2, f(3)=1, f(4)=4, f(5)=3$. Para obtener $A(H)$ de $A(G)$, se reemplaza la quinta fila de $A(G)$ por su primera fila, la primera fila de $A(G)$ por su tercera fila, y la tercera fila de $A(G)$ por su quinta fila. Supongamos que la matriz resultante es denotada por B. Se reemplaza la quinta columna de B por su primera columna, la primera columna de B por su tercera columna, y la tercera columna de B por su quinta columna. La matriz resultante es $G(H)$. Para obtener la matriz de permutación P de la matriz identidad I de 5×5, se sigue la misma secuencia de operaciones en I que fue realizada para obtener B a partir de $A(G)$:

$$P=\begin{bmatrix}0&0&1&0&0\\0&1&0&0&0\\0&0&0&0&1\\0&0&0&1&0\\1&0&0&0&0\end{bmatrix}$$

De donde se verifica que $PA(G)=B=A(H)P$.

Teorema 1.3.2.- Sea A matriz de adyacencia, el elemento $(A^k)_{ij}$ es igual al número de recorridos (pudiendo repetir vértices y/o aristas) de longitud k entre v_i y v_j.

Demostración.- (Se procederá por inducción)**.**

El caso base es $k=1$. En este caso, $A^k{}_{ij}=A_{ij}$, es decir, el elemento *ij* de la matriz de adyacencia, que por su definición indica el número de recorridos de longitud 1 entre el vértice *i* y el vértice *j*; el caso base se cumple. Con la hipótesis de inducción, suponemos cierto el teorema para $k=m$.

Para el paso inductivo, debe probarse para $k=m+1$. Se tiene;

$$(A^k)_{ij}=(A^{m+1})_{ij}=(A^mA)_{ij}=\sum_{l=1}^{n}(A^m)_{il}(A)_{lj}$$

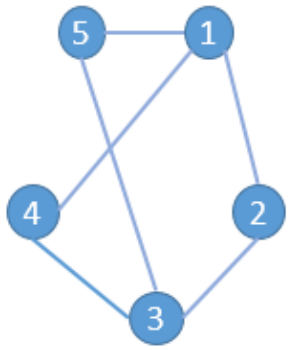
El término $(A^m)_{il}(A)_{lj}$ de la suma anterior sólo es no nulo si $(A^m)_{il}\geq 1$ y $(A)_{lj}=1$, así, si $(A^m)_{il}=p$, por la hipótesis de inducción existen p recorridos de longitud m entre v_i y v_l. Ya que $(A_{lj})=1$, hay p recorridos de v_i a v_j de longitud $m+1$.[9] ■

[9] Comellas, F., Fábrega, J., Sánchez, A., (2001) *Matemática Discreta*, 1ra edición, España, Edicions UPC.

De este teorema se deriva el siguiente corolario:

Corolario 1.3.1.- En una gráfica conexa G, la distancia entre dos vértices v_i y v_j es k si y sólo si k es el menor entero no negativo tal que $(A^k(G))_{ij} \neq 0$.

Para ejemplificar, retomemos la gráfica de la figura 1.1.5;



Su matriz de adyacencia es:

$$A = \begin{bmatrix} 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \end{bmatrix}$$

Calculando A^2

$$A^2 = \begin{bmatrix} 3 & 0 & 3 & 0 & 0 \\ 0 & 2 & 0 & 2 & 2 \\ 3 & 0 & 3 & 0 & 0 \\ 0 & 2 & 0 & 2 & 2 \\ 0 & 2 & 0 & 2 & 2 \end{bmatrix}$$

Así, por ejemplo, $A^2{}_{13} = 3$ implica que existen 3 recorridos de longitud 2 entre el vértice 1 y el vértice 3; $1 \to 4 \to 3$, $1 \to 2 \to 3$ y finalmente $1 \to 5 \to 3$.

Se hace notar que la matriz de adyacencia nodo a nodo de una red no dirigida, como en el ejemplo anterior, es simétrica.

Ya que la matriz de adyacencia y la matriz de incidencia (denotadas en texto como A y B, respectivamente) reflejan la estructura de la red, han de estar relacionadas, como puede verse en el siguiente Teorema.

Teorema 1.3.3.- Sea D la matriz diagonal tal que D_{ii} es el grado $d(v_i)$ del vértice i-ésimo de $V(G)$, luego, $BB^t = A + D$.

Demostración.-

Sea $V(G) = \{v_1, v_2, \dots, v_n\}$ y $E(G) = \{e_1, e_2, \dots, e_n\}$. Si i ≠ j,

$$(BB^t)_{ij} = \sum_{k=1}^{m} b_{ik} b_{jk} = a_{ij}$$

Ya que $b_{ik} b_{jk}$ sólo es diferente de 0 (y vale 1) si $e_k = v_i v_j$.

De otro modo, si $i = j$,

$$(BB^t)_{ii} = \sum_{k=1}^{m} b_{ik} b_{ik} = d(v_i)$$

Ya que, ahora, $\sum_{k=1}^{m} b_{ik} b_{ik}$ cuenta el número de aristas incidentes con v_i.[10] ■

Tomando nuevamente la gráfica de la figura 1.1.5, tenemos que

$$A = \begin{bmatrix} 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \end{bmatrix}; \quad B = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}; \quad B^T = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \end{bmatrix}$$

$$D = \begin{bmatrix} 3 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 2 \end{bmatrix}; \quad BB^T = \begin{bmatrix} 3 & 1 & 0 & 1 & 1 \\ 1 & 2 & 1 & 0 & 0 \\ 0 & 1 & 3 & 1 & 1 \\ 1 & 0 & 1 & 2 & 0 \\ 1 & 0 & 1 & 0 & 2 \end{bmatrix};$$

$$A + D = \begin{bmatrix} 0+3 & 1+0 & 0+0 & 1+0 & 1+0 \\ 1+0 & 0+2 & 1+0 & 0+0 & 0+0 \\ 0+0 & 1+0 & 0+3 & 1+0 & 1+0 \\ 1+0 & 0+0 & 1+0 & 0+2 & 0+0 \\ 1+0 & 0+0 & 1+0 & 0+0 & 0+2 \end{bmatrix} = \begin{bmatrix} 3 & 1 & 0 & 1 & 1 \\ 1 & 2 & 1 & 0 & 0 \\ 0 & 1 & 3 & 1 & 1 \\ 1 & 0 & 1 & 2 & 0 \\ 1 & 0 & 1 & 0 & 2 \end{bmatrix} = BB^T$$

[10] Comellas, F., Fábrega, J., Sánchez, A., Ibid.

2. FLUJO MÁXIMO Y CORTE MÍNIMO

El problema de Flujo Máximo consiste en determinar la máxima cantidad de flujo que puede ser enviada a lo largo de una red dirigida, desde un nodo origen hasta un nodo destino. Es necesario que todo flujo a través de la red se derive del nodo origen y cumpla las restricciones de capacidad de cada uno de los arcos.

2.1 Antecedentes históricos del problema de flujo máximo

Como una primera consideración, podemos rastrear las raíces de los problemas de flujo hasta Gustav Robert Kirchhoff[11] (1824-1887), físico prusiano cuyas principales contribuciones científicas estuvieron en el campo de la óptica, la espectroscopía, la teoría de placas y la radiación del cuerpo negro; fue uno de los pioneros en analizar sistemáticamente circuitos eléctricos. Su trabajo conjunta muchas ideas clave de teoría de flujos y establece redes (gráficas) como objetos matemáticos útiles para representar muchos sistemas físicos. Buscaba responder la pregunta: "¿Si aplicamos un conjunto de voltajes a una red dada, cuál será el flujo resultante?"

Respecto al problema de flujo máximo, en 1930 A.N. Tolstoi publicó *Methods of finding the minimal total kilometrage in cargo-transportation planning in space*[12], donde estudió el problema de transporte y describió aproximaciones de solución. Una de sus descripciones es la (ahora bien conocida) idea que una solución óptima no tiene ningún ciclo con costo negativo en su gráfica residual, siendo él probablemente el primero en observar que ésta es una condición necesaria para la optimalidad. Más aún, él supuso, pero no demostró, que también es condición suficiente para la solución óptima. Los estudios de Tolstoi fueron aplicados en las redes de ferrocarriles de la Unión Soviética.

Por otro lado, en 1954 se publicó el artículo *Flujo Máximo a través de una red* por Ford y Fulkerson, donde se menciona que el problema de flujo máximo fue planteado por T.E. Harris como sigue:

"Considerando una red de carriles que conecta dos ciudades por un número de ciudades intermedias, donde cada arco de la red tiene un número asignado

[11] Gustav Robert Kirchhoff. (s.f). En *Wikipedia*. Recuperado el 31 de marzo de 2017 de https://es.wikipedia.org/wiki/Gustav_Kirchhoff

[12] Tolstoi, A.N. (1930) Methods of finding the minimal total kilometrage in cargo-transportation planning in space, *Transportation Planning, Volumen 1.*

representando su capacidad. Suponiendo un estado estable, encontrar un flujo máximo de una ciudad dada a otra."[13]

En 1962 Ford y Fulkerson publican *Flows in Networks*[14], dando referencias más precisas al origen del problema y donde se menciona un reporte secreto por Harris y Ross (1955) titulado *Fundamentals of a Method for Evaluating Rail Net Capacities*, escrito para la Fuerza Aérea de EUA. El reporte resuelve un problema de flujo máximo relativamente grande originado en la red de ferrocarriles en el occidente de la Unión Soviética y en el oriente de Europa. Vale la pena destacar que, a pesar de lo dicho por Ford y Fulkerson, el interés de Harris y Ross no fue encontrar un flujo máximo, sino un corte mínimo (el problema dual del Flujo Máximo) del sistema Soviético de ferrocarriles, con el objetivo militar de interrumpir la red ferroviaria. Citando:

"El poder aéreo es un medio efectivo de inhabilitación en un sistema ferroviario enemigo, […], como en muchas operaciones militares, sin embargo, el éxito de una inhabilitación depende en gran medida de cuan completa, exacta y oportuna sea la información del comandante, particularmente en cuanto al efecto de los esfuerzos del programa de inhabilitación sobre la capacidad del enemigo para desplazar hombres y suministros."[15]

El reporte Harris-Ross aplica la "flooding technique" descrita en un reporte de la RAND Corporation de agosto de 1955 por A.W. Boldyreff. Ésta equivale a enviar tanto flujo como sea posible a través de la red. Si en algún punto surge un cuello de botella (es decir, más trenes llegan de los que pueden pasar) el exceso es regresado al origen. La técnica no garantiza optimalidad. El reporte Harris-Ross aplica la *flooding technique* al modelo de redes de ferrocarril de la Unión Soviética y al oriente de Europa. Su aplicación es desarrollada paso a paso en el apéndice del reporte.

El bien conocido algoritmo de camino con flujo aumentante de Ford y Flukerson que da garantía de optimalidad, fue publicado en un reporte de la RAND Corporation en 1954.

En el artículo *Application of the Maximum Flow Problem to Sensor Placement on Urban Road Networks for Homeland Security*[16]*,* de Lowell Bruce Anderson publicado

[13] Ford, L.R., Fulkerson, D.R. (10 de Noviembre de 1956) Maximal Flow through a Network, *Research memorandum RM-1400, The RAND Corporation*, *Canadian Journal of Mathematics Volumen 8.*

[14] Ford, L.R., Fulkerson, D.R. (1962) *Flows in Networks*, Princeton, New Jersey, EUA., Princeton Landmarks in Mathematics.

[15] Ford, L.R., Fulkerson, D.R. (10 de Noviembre de 1956) Maximal Flow through a Network, *Research memorandum RM-1400, The RAND Corporation*, *Canadian Journal of Mathematics Volumen 8.*

[16] Anderson, L.B., Atwell, R.J., Barnett, D.S., Bovey, R.L. (Septiembre de 2007) Application of the Maximum Flow Problem to Sensor Placement on Urban Road Networks for Homeland Security, *Homeland Security Affairs, Volumen 3.* No. 3.

en septiembre de 2007, se desarrolla una metodología que encuentra localizaciones óptimas para sensores en la ciudad de Nueva York. Esta metodología usa teoría de gráficas para resolver el problema de flujo máximo e identificar el corte mínimo en redes conteniendo más de un millón de segmentos.

Otra aplicación de resolución del problema de flujo máximo pero en redes dinámicas puede encontrarse en el artículo *Minimize traffic congestion: an application of Maximum Flow in dynamic networks*[17]*.* Se usó la versión del problema de flujo máximo para implementarlo a la congestión de tráfico el cual requiere el uso de redes dinámicas para su representación como flujo.

En relación a aplicaciones del transporte, en el artículo *Identificación de tramos críticos por vulnerabilidad para el traslado de mercancía en la red carretera pavimentada de México*[18]*,* de Luz Gradilla Hernández y Ovidio González Gómez publicado en abril de 2011, se expone un análisis para identificar puntos críticos de la red carretera de acuerdo con los flujos de carga. En este estudio se cuantifica la vulnerabilidad por medio de la afectación en el tiempo total de viaje en la red.

2.2 Representación del Flujo Máximo en Programación Lineal

Definición 2.2.1.- Programación lineal es el área de la optimización que busca maximizar o minimizar una función lineal, la cual es denominada función objetivo, sujeta a un sistema de ecuaciones y/o inecuaciones también lineales llamadas restricciones.

Sea G una red con n aristas y m vértices. Se asocia a cada arista $(i,j) \in A$ una capacidad máxima (upper bound) q_{ij} y se establece $x_{ij} \geq 0$. Se desea encontrar la máxima cantidad de flujo del vértice *1* al vértice m.

Definición 2.2.2.- Formulación del problema de flujo máximo; sea x la cantidad de flujo en la red desde el nodo *1* al nodo m.

Siendo entonces la función objetivo:

[17] Kaanodiya, K.K., Rizwanullah, M. (2012) Minimize traffic congestion: an application of Maximum Flow in dynamic networks. *Journal of the Applied Mathematics, Statistics and Informatics (JAMSI), Volumen 8.* No.1.

[18] Hernández, L.G., González, O. (Abril de 2011) Identificación de tramos críticos por vulnerabilidad para el traslado de mercancía en la red carretera pavimentada de México. *Investigaciones Geográficas, Boletín del Instituto de Geografía, UNAM, Volumen 74.*

$$Max\ Z = x$$

Sujeto a

$$\sum_{j=1}^{m} x_{ij} - \sum_{k=1}^{m} x_{ki} = \begin{cases} x\ si\ i = 1 \\ 0\ si\ i \neq 1, i \neq m \\ -x\ si\ i = m \end{cases}$$
$$x_{ij} \leq q_{ij} \qquad i, j = 1,2, \dots, m$$
$$x_{ij} \geq 0 \qquad i, j = 1,2, \dots, m$$

Esto es llamado la formulación nodo-arco para el problema de flujo máximo dado que la matriz de restricciones es una matriz de incidencia nodo-arco.

Ejemplo 2.2.1.-

Se plantea el problema de flujo máximo de la siguiente red R_1 como programa lineal;

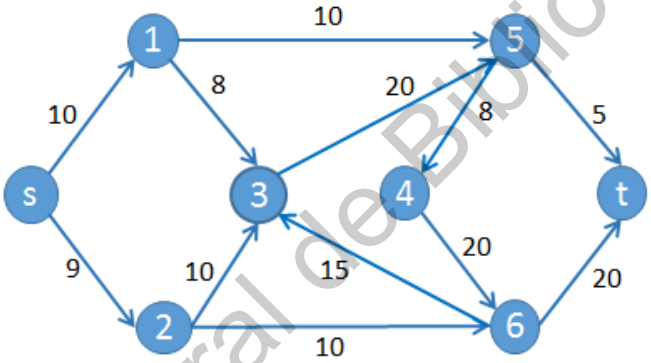


Figura 2.2.1 Red R_1 donde el número asociado a cada arista a corresponde a su capacidad c_a

El programa lineal correspondiente es;

$$Max\ Z = x$$

Sujeto a

$$x = x_{s1} + x_{s2}$$
$$x_{s2} = x_{23} + x_{26}$$
$$x_{54} = x_{46}$$
$$x_{26} + x_{46} = x_{6t} + x_{63}$$
$$0 \leq x_{s1} \leq 10$$
$$0 \leq x_{13} \leq 8$$
$$0 \leq x_{23} \leq 10$$
$$0 \leq x_{35} \leq 20$$
$$0 \leq x_{54} \leq 8$$
$$0 \leq x_{63} \leq 15$$

$$x_{s1} = x_{13} + x_{15}$$
$$x_{13} + x_{23} + x_{63} = x_{35}$$
$$x_{15} + x_{35} = x_{54} + x_{5t}$$
$$x_{6t} + x_{5t} = x$$
$$0 \leq x_{s2} \leq 9$$
$$0 \leq x_{15} \leq 10$$
$$0 \leq x_{26} \leq 10$$
$$0 \leq x_{46} \leq 20$$
$$0 \leq x_{5t} \leq 5$$
$$0 \leq x_{6t} \leq 20$$

Reescribiendo el programa lineal de acuerdo a la definición 2.2.2, la matriz A de restricciones es:

$$\boldsymbol{A} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & -1 & 0 & 1 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & -1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 \end{bmatrix}$$

A corresponde a la matriz de incidencia nodo-arco *B*, vista en la sección anterior:

$$\boldsymbol{B} = \begin{array}{c} \\ \boldsymbol{s} \\ \boldsymbol{1} \\ \boldsymbol{2} \\ \boldsymbol{3} \\ \boldsymbol{4} \\ \boldsymbol{5} \\ \boldsymbol{6} \\ \boldsymbol{t} \end{array} \begin{array}{c} \begin{bmatrix} (\boldsymbol{s},\boldsymbol{1}) & (\boldsymbol{s},\boldsymbol{2}) & (\boldsymbol{1},\boldsymbol{3}) & (\boldsymbol{1},\boldsymbol{5}) & (\boldsymbol{2},\boldsymbol{3}) & (\boldsymbol{2},\boldsymbol{6}) & (\boldsymbol{3},\boldsymbol{5}) & (\boldsymbol{4},\boldsymbol{6}) & (\boldsymbol{5},\boldsymbol{4}) & (\boldsymbol{5},\boldsymbol{t}) & (\boldsymbol{6},\boldsymbol{3}) & (\boldsymbol{6},\boldsymbol{t}) \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & -1 & 0 & 1 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & -1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 \end{bmatrix} \end{array}$$

De donde $A = B$. La estructura de A tiene una propiedad muy particular que se comentará.

Definición 2.2.3.- Una *matriz unimodular* es una matriz cuadrada de enteros con determinante +1 ó -1.

Definición 2.2.4.- Una matriz es *totalmente unimodular*, si cada submatriz cuadrada invertible *B* es también unimodular. En consecuencia, una matriz totalmente unimodular está formada sólo por elementos, -1, 0 y +1.

Teorema 2.2.1.- Sea A una matriz con $a_{ij} \in \{-1,0,1\}\ \forall i,j$. A es totalmente unimodular si:

i) Cada columna de A tiene cuando mucho dos elementos no nulos.
ii) Las filas de A pueden ser particionadas en dos conjuntos, A_1 y A_2 tales que dos entradas no cero de una misma columna están en el mismo conjunto de filas si tienen diferentes signos y en diferente conjunto de filas si tienen el mismo signo.

Para propósitos de lo hasta ahora abordado, es más conveniente enfocarse en la demostración del siguiente Teorema:

Teorema 2.2.2.- La matriz M de incidencia nodo-arco de una gráfica dirigida D es totalmente unimodular.[19]

Demostración.- (Se procederá por inducción)

Sea N una submatriz cuadrada de $t \times t$ de M.

El caso base es $t = 1$, el determinante de la matriz de orden 1x1 tiene tres posibles casos:

Caso 1: $N = [0]$, luego, $\det(N) = 0$

Caso 2: $N = [1]$, luego, $\det(N) = 1$

Caso 3: $N = [-1]$, luego, $\det(N) = -1$

El caso base se cumple; entonces con la hipótesis de inducción, suponemos cierto el teorema para $t = m$, es decir, $\det(N) = -1, 0, \text{ó } 1$.

Para el paso inductivo, se demuestra para $t = m + 1$ a partir de la hipótesis de inducción. Se tienen nuevamente tres casos;

Caso 1: N tiene una columna de ceros, luego, $\det(N) = 0$

Caso 2: N tiene una columna con un solo elemento no nulo: ± 1. El determinante se calcula multiplicando cada elemento a_{ij} de la columna por su cofactor, es decir, el determinante de la matriz obtenida al eliminar la fila y columna de a_{ij} (la matriz menor M_{ij}) y multiplicando por (-1^{i+j}). La suma de estos productos es el determinante. Así, hay m de los $m + 1$ sumandos iguales a cero. Para el sumando del elemento no nulo, se tiene ± 1 multiplicado por el determinante de la matriz de $m \times m$ que por hipótesis de inducción es igual a ± 1 ó 0. Luego, el resultado de los $m + 1$ sumados que calculan el determinante de N es igual a $-1, 1$ ó 0, de donde N es totalmente unimodular.

Caso 3: cada columna de N tiene sólo un elemento 1 y un elemento -1, luego, la suma de todas las filas resulta en el vector nulo y por tanto $\det(N) = 0$.

Se concluye así que la matriz M es totalmente unimodular. ■

A partir de este Teorema, como la matriz de incidencia nodos arcos es totalmente unimodular, también lo es la matriz de restricciones del problema de flujo máximo.

[19] Klerk, E. de. Integer programming and totally unimodular. Delf University of Technology. *Semanticsholar*. Recuperado de: https://pdfs.semanticscholar.org/a5c4/cf30dcbbc130f693dd1b75e0032f42b741bd.pdf

Así, si el vector de capacidades es entero, se concluye que la solución siempre será entera.

A modo de ejemplo, ingresando el problema de la red R_1 en TORA, en el menú de Programación Lineal, se obtiene como resultado:

Title: R1 2
Final Iteration No.: 12
Objective Value (Max) =19.00 -- Alternative solution(s) detected (enter ITERATIONS

Next Iteration | All Iterations | Write to Printer

Variable	Value	Obj Coeff	Obj Val Contrib
x1: Xs1	10.00	1.00	10.00
x2: Xs2	9.00	1.00	9.00
x3: X13	0.00	0.00	0.00
x4: X15	10.00	0.00	0.00
x5: X23	0.00	0.00	0.00
x6: X26	9.00	0.00	0.00
x7: X35	0.00	0.00	0.00
x8: X46	5.00	0.00	0.00
x9: X54	5.00	0.00	0.00
x10: X5t	5.00	0.00	0.00
x11: X63	0.00	0.00	0.00
x12: X6t	14.00	0.00	0.00

Figura 2.2.2 Solución brindada por TORA para el programa lineal de la red R_1

De donde el flujo máximo obtenido es 19, al especificar que el flujo de los arcos iniciales ($s \to 1, s \to 2$) sea igual a la capacidad máxima de éstos (10 y 9). La solución está dada por;

$$x_{s1} = 10 \qquad x_{s2} = 9$$
$$x_{15} = 10 \qquad x_{26} = 9$$
$$x_{46} = 5 \qquad x_{54} = 5$$
$$x_{5t} = 5 \qquad x_{6t} = 14$$

La siguiente es la solución gráfica.

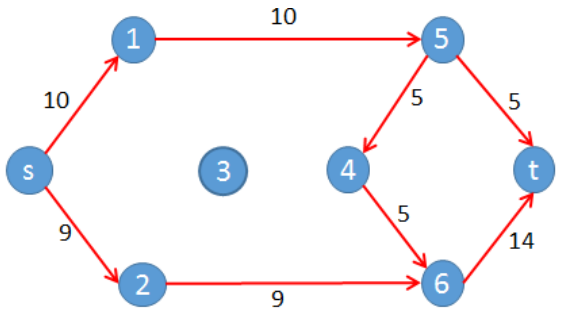


Figura 2.2.3 Solución gráfica del ejemplo 2.2.1

Del resultado de TORA se observa que hay una solución alternativa, la cual se muestra a continuación, obtenida a partir del método de Dos Fases.

Phase 0 (Iter 12						
Basic	Rx14	Rx15	Rx16	Rx17	Rx18	Solution
z (max)	100.00	100.00	100.00	100.00	100.00	19.00
x8	0.00	-1.00	-1.00	-1.00	0.00	5.00
x12	-1.00	-1.00	-1.00	-1.00	-1.00	14.00
x7	0.00	-1.00	0.00	0.00	0.00	8.00
x9	0.00	-1.00	0.00	-1.00	0.00	5.00
x4	0.00	0.00	0.00	0.00	0.00	2.00
x6	-1.00	0.00	0.00	0.00	0.00	9.00
Lower Bound						
Upper Bound						
Unrestr'd (y/n)?						

Se obtiene, por supuesto, 19 como flujo máximo, asignando peso a las variables $x_8, x_{12}, x_7, x_9, x_4, x_6$ que corresponden a las variables $x_{46}, x_{6t}, x_{35}, x_{54}, x_{15}$ y x_{26} originales, respectivamente.

$$x_{s1} = 10 \qquad x_{s2} = 9$$
$$x_{13} = 8 \qquad x_{15} = 2$$
$$x_{26} = 9 \qquad x_{35} = 8$$
$$x_{46} = 5 \qquad x_{54} = 5$$
$$x_{5t} = 5 \qquad x_{6t} = 14$$

Siendo la solución gráfica;

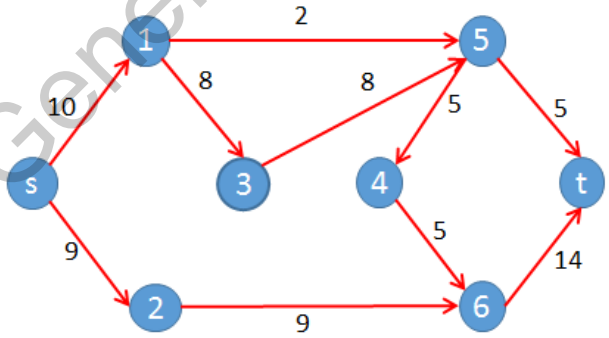


Figura 2.2.4 Solución gráfica alternativa del ejemplo 2.2.1

El valor del flujo es igual a 19, y además, ya que los arcos que salen del nodo fuente (*source*) tienen en conjunto un límite superior de 19, al igual que con la solución óptima original, la conclusión es asignarle a estos arcos todo el flujo posible para obtener el flujo máximo.

2.3 Dualidad en el problema de Flujo Máximo

Asociado a cada problema de programación lineal se encuentra *el* ***problema dual***, el cual posee importantes propiedades con respecto del original (el cual se denomina *problema primal)*. La siguiente consideración es obtenida de George Bernard Dantzing (1914-2005), desarrollador del método simplex y considerado como el "padre de la programación lineal", de su célebre obra *Linear Programming and Extensions* (1963)[20].

> Soluciones factibles al problema dual y al primal pueden parecer tener poca relación entre sí, sin embargo, el óptimo en el área de soluciones básicas factibles es tal que puede utilizarse para obtener el otro fácilmente. […] A menudo es más conveniente usar el dual para resolver un problema de programación lineal que el primal. […] El Teorema de la Dualidad es una declaración sobre el rango de posibles valores Z^* para el primal contra el rango de posibles valores W^* para el dual. (p. 124).

Una forma de expresar el dual convenientemente, requiere que en el primal todas las restricciones sean ecuaciones con el lado derecho no negativo, y variables no negativas. Los siguientes puntos son las ideas clave para construir el problema dual[21]

1) Asignar una variable dual por cada restricción primal
2) Construir una restricción dual por cada variable primal
3) Los coeficientes de restricción y el coeficiente objetivo de la variable primal *j*-ésima definen respectivamente los lados izquierdo y derecho de la restricción dual *j*-ésima.
4) Los coeficientes objetivo duales son iguales a los lados derechos de las ecuaciones de restricción primales.
5) Las restricciones que aparecen en la Tabla 2.3.1 rigen el sentido de optimización, la dirección de las desigualdades y los signos de las variables en el dual.

[20] Dantzing, G. B., (1963) *Linear Programming and Extensions*, a report prepared for United States Air Force Project Rand.
[21] Taha, H.,(2012), *Investigación de Operaciones*, 9na edición, México, PEARSON EDUCACION.

Tabla 2.3.1 Reglas para construir el problema dual[22]

Problema primal	*Problema dual*
Maximizar $Z = \sum_{j=1}^{n} c_j x_j$,	Minimizar $W = \sum_{i=1}^{m} b_i y_i$,
sujeta a	sujeta a
$\sum_{j=1}^{n} a_{ij} x_j = b_i$, para $i = 1, 2, \ldots, m$	$\sum_{i=1}^{m} a_{ij} y_i \geq c_j$, para $j = 1, 2, \ldots, n$
y	y
$x_j \geq 0$, para $j = 1, 2, \ldots, n$.	y_i irrestricta para $i = 1, 2, \ldots, m$.

Objetivo del problema primal[a]	Problema dual		
	Objetivo	*Tipo de restricción*	*Signo de las variables*
Maximización	Minimización	$\geq$	irrestricta
Minimización	Maximización	$\leq$	irrestricta

[a]Todas las restricciones primales son ecuaciones con lado derecho no negativo, y todas las variables son no negativas.

Para ejemplificar lo anterior, tomemos el siguiente programa lineal como programa primal:

$$Max\ Z = x_1 + 2x_2 + 3x_3$$

Sujeto a

$$x_1 + 3x_2 + x_3 \leq 10$$

$$2x_1 + 2x_2 + 3x_2 = 15$$

$$x_1, x_2, x_3 \geq 0$$

El primal cumple el formato de variables no negativas pero las restricciones no son ecuaciones con el lado derecho no negativo, por lo que se define la variable de holgura $x_4 \geq 0$ tal que

$$Max\ Z = x_1 + 2x_2 + 3x_3 + 0x_4$$

Sujeto a

[22] Taha, H.,(2012). TABLA 4.1, Reglas para construir el problema dual. Tabla. Investigación de Operaciones, 9na edición.

$x_1 + 3x_2 + x_3 + x_4 = 10$	*Variables duales*
$2x_1 + 2x_2 + 3x_3 + 0x_4 = 15$	y_1
$x_1, x_2, x_3, x_4 \geq 0$	y_2

Obteniendo el programa primal en forma de ecuación, así, se construye su programa dual con base en la Tabla 2.3.1:

$$Min\ W = 10y_1 + 15y_2$$

Sujeto a

$$y_1 + 2y_2 \geq 1$$

$$3y_1 + 2y_2 \geq 2$$

$$y_1 + 3y_2 \geq 3$$

$$y_1 \geq 0$$

$$y_2\ irrestricta$$

En el problema de flujo máximo, con x la variable de flujo a maximizar y recordando la formulación del problema de flujo máximo, se tiene

$$Max\ Z = x$$

Sujeto a

$$\sum_{j=1}^{m} x_{ij} - \sum_{k=1}^{m} x_{ki} = \begin{cases} x\ si\ i = 1 \\ 0\ si\ i \neq 1, \mathrm{i} \neq\ m \\ -x\ si\ i = m \end{cases}$$

$$x_{ij} \leq q_{ij} \qquad i, j = 1,2, \dots, m$$

$$x_{ij} \geq 0 \qquad i, j = 1,2, \dots, m$$

Denótese por $\boldsymbol{A}$ la matriz de restricciones, al vector de restricciones de capacidad como $\boldsymbol{q}$, y al vector de flujos en los arcos x_{ij} como $\boldsymbol{X}$;

$$\boldsymbol{X} = \begin{bmatrix} x_{12} \\ \vdots \\ x_{ij} \\ \vdots \\ x_{km} \end{bmatrix}$$

Luego, puede reescribirse el problema de flujo máximo en forma matricial como

$$Max\ x$$

Sujeto a

$$(\boldsymbol{e_m} - \boldsymbol{e_1})x + \boldsymbol{AX} = \boldsymbol{0}$$

$$\boldsymbol{X} \leq \boldsymbol{q}$$

$$\boldsymbol{X} \geq \boldsymbol{0}$$

Siendo el programa dual

$$Min \sum_{i=1}^{m} \sum_{j=1}^{m} q_{ij} h_{ij}$$

Sujeto a

$$w_m - w_1 = 1$$

$$w_i - w_j + h_{ij} \geq 0 \qquad i, j = 1,2, \dots, m$$

$$h_{ij} \geq 0 \qquad i, j = 1,2, \dots, m$$

Donde $\boldsymbol{w}$ corresponde a las restricciones primales de conservación y $\boldsymbol{h}$ a las restricciones primales de capacidad $\boldsymbol{X} \leq \boldsymbol{q}$, denotando h_{ij} como la variable dual asociada con la restricción primal que limita al arco (i, j) a la capacidad $q_{ij.}$**.** Nótese que la primera restricción dual está asociada con el flujo primal x cuya columna es $(\boldsymbol{e_m} - \boldsymbol{e_1})$. Una columna A_{ij} de la matriz de incidencia nodo-arco $\boldsymbol{A}$ tiene un +1 en la posición i-ésima y un -1 en la posición j-ésima, lo que lleva a la restricción dual $w_i - w_j + h_{ij} \geq 0$. Las variables duales h_{ij} tienen una interpretación interesante y útil que se explica enseguida.

Definición 2.3.1.- Un *corte* es una partición del conjunto V de nodos en dos subconjuntos, S y $\bar{S} = V - S$, denotado $(S, \bar{S})$. Un corte define al conjunto de arcos que tienen un nodo en S y el otro en $\bar{S}$.

Definición 2.3.2.- Un *corte* $s - t$, respecto a dos nodos s y t, $s \neq t$, es un corte $(S, \bar{S})$ tal que $s \in S$ y $t \in \bar{S}$. Por convención, el nodo s corresponde al nodo fuente y t al nodo destino.

Con estas consideraciones, sea $(S, \bar{S})$ un corte cualquiera y se define

$$w_i = \begin{cases} 0 \ \ si \ \ i \in S \\ 1 \ \ si \ \ i \in \bar{S} \end{cases}$$

(1)

$$h_{ij} = \begin{cases} 1 \ \ si \ \ (i,j) \in (S, \bar{S}) \\ 0 \qquad\quad otro\ caso \end{cases}$$

Esta elección de $\boldsymbol{w}$ y $\boldsymbol{h}$ proporciona una solución factible al problema dual, cuya función objetivo es igual a la capacidad del corte formado por los arcos $(i,j) \in (S, \bar{S})$. Para verlo más claramente, nótese que se tiene una w_i por cada nodo de la red, y de acuerdo a (1) y la definición 2.3.1, $w_1 = 0$, $w_m = 1$, de donde se cumple la primera restricción del programa dual

$$w_m - w_1 = 1$$

La construcción de (1) también implica que para nodos i, j tal que $(i,j) \in A$

$$w_i - w_j + h_{ij} \geq 0 \qquad i,j = 1,2,\dots,m$$

$$h_{ij} \geq 0 \qquad i,j = 1,2,\dots,m$$

Luego, con la solución dual factible en (1), la función objetivo resulta ser la suma de capacidades de aquellos arcos (i,j) que en el problema primal cumplen $(i,j) \in (S, \bar{S})$.

Ahora regresando al programa lineal de la red R_1, se plantea su programa dual.

PROBLEMA PRIMAL

$$Max\ Z = x$$

Sujeto a

$$\begin{gathered} x - x_{s1} - x_{s2} = 0 \\ x_{s2} - x_{23} - x_{26} = 0 \\ -x_{46} + x_{54} = 0 \\ x_{26} + x_{46} - x_{6t} = 0 \\ x_{s1} \leq 10 \\ x_{13} \leq 8 \\ x_{23} \leq 10 \\ x_{35} \leq 10 \\ x_{54} \leq 8 \\ x_{63} \leq 15 \end{gathered} \qquad \begin{gathered} x_{s1} - x_{13} - x_{15} = 0 \\ x_{13} + x_{23} - x_{35} + x_{63} = 0 \\ x_{15} + x_{35} - x_{54} - x_{5t} = 0 \\ x - x_{5t} - x_{6t} = 0 \\ x_{s2} \leq 9 \\ x_{15} \leq 10 \\ x_{26} \leq 10 \\ x_{46} \leq 20 \\ x_{5t} \leq 5 \\ x_{6t} \leq 20 \end{gathered}$$

$$x_{s1}, x_{s2}, x_{13}, x_{15}, x_{23}, x_{26}, x_{35}, x_{46}, x_{54}, x_{5t}, x_{63}, x_{6t} \geq 0$$

PROBLEMA DUAL

$$\begin{aligned} Min\ W = {} & 10w_9 + 9w_{10} + 8w_{11} + 10w_{12} + 10w_{13} + 10w_{14} + 10w_{15} + 20w_{16} + 8w_{17} \\ & + 5w_{18} + 15w_{19} + 20w_{20} \end{aligned}$$

Sujeto a

$$w_1 + w_8 \geq 1$$
$$-w_1 + w_2 + w_{10} \geq 0$$
$$-w_5 + w_7 + w_{12} \geq 0$$
$$-w_2 + w_4 + w_{14} \geq 0$$
$$-w_3 + w_4 + w_{16} \geq 0$$
$$-w_7 - w_8 + w_{18} \geq 0$$
$$-w_4 - w_8 + w_{20} \geq 0$$
$$-w_1 + w_5 + w_9 \geq 0$$
$$-w_5 + w_6 + w_{11} \geq 0$$
$$-w_2 + w_6 + w_{13} \geq 0$$
$$-w_6 + w_7 + w_{15} \geq 0$$
$$w_3 - w_7 + w_{17} \geq 0$$
$$w_6 + w_{19} \geq 0$$

$$w_i \; irrestricta \; (i = 1, \dots 20)$$

2.3.1. El teorema de Flujo Máximo- Corte Mínimo

El Teorema de Flujo Máximo-Corte Mínimo fue establecido en el ya mencionado artículo de Ford y Fulkerson *Flow in Networks*[23]; para abordar la demostración son necesarios antes los siguientes conceptos y lemas.

Definición 2.3.1.1.- Un *camino* en una gráfica dirigida $G = (V, A)$ es una secuencia de nodos y arcos $i_1, a_1, i_2, a_2, \dots, i_{r-1}, a_{r-1}, i_r$, en la que no se repiten nodos, y para todo $1 \leq k \leq r-1$, $a_k = (i_k, i_{k+1}) \in A$ ó $a_k = (i_{k+1}, i_k) \in A$.

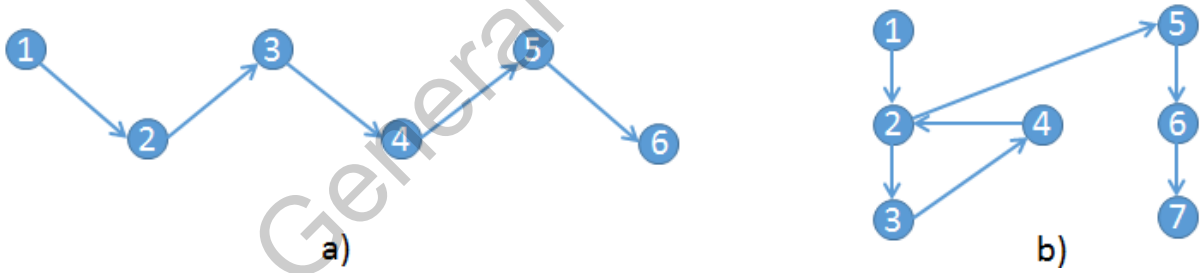


Figura 2.3.1.1 Secuencia de nodos y arcos; **a)** $1, (1,2), 2, (2,3), 3, (3,4), 4, (4,5), 5, (5,6)$ y **b)** $1, (1,2), 2, (2,3), 3, (3,4), 4, (4,2), 2, (2,5), 5, (5,6), 6, (6,7)$

La secuencia de nodos 1-2-3-4-5-6 de **a)** en la figura 2.3.1.1 es un camino, pero la secuencia 1-2-3-4-2-5-6-7 en **b)** no lo es, pues se repite el nodo 2.

Definición 2.3.1.2.-Un *camino dirigido* es aquel donde todos los arcos van en el mismo sentido, es decir, un camino dirigido no cuenta con arcos "hacia atrás".

Definición 2.3.1.3.- Un arco (i, j) con $i \in S$ y $j \in \bar{S}$ se llama *arco de avance* del corte, y a un arco (i, j) con $i \in \bar{S}$ y $j \in S$ se llama *arco de atraso* del corte.

[23] Ford, L.R., Fulkerson, D.R. (10 de Noviembre de 1956) Maximal Flow through a Network, *Research memorandum RM-1400, The RAND Corporation*, *Canadian Journal of Mathematics Volumen 8.*

Por ejemplo, en la figura 2.3.1.2, si $s = 1$ y $t = 6$, el corte indicado es un corte $s - t$.

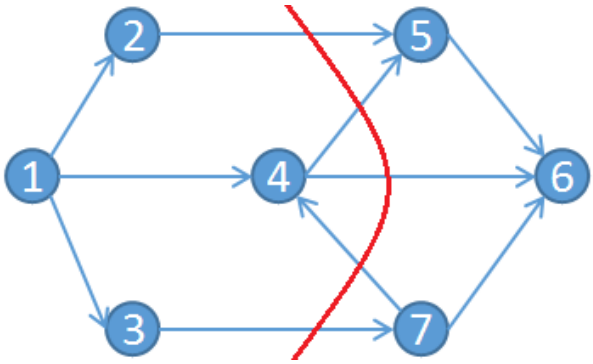


Figura 2.3.1.2 Corte con $S = \{1,2,3,4\}, \bar{S} = \{5,6,7\}$. El conjunto de los arcos en este corte es $\{(2,5),(3,7),(4,5),(4,6),(7,4)\}$

Para calcular el flujo máximo en una red, conviene considerar los posibles aumentos de flujo sobre una solución factible. Esta idea, lleva a definir una red auxiliar llamada *red residual.* La idea intuitiva para construir la red residual es como sigue (Ahuja, Magnati y Orlin, 1993)[24].

> Supóngase que el arco (i,j) transporta x_{ij}° unidades de flujo. Luego, pueden enviarse $q_{ij} - x_{ij}^{\circ}$ unidades de flujo adicional del nodo i al nodo j a través del arco (i,j). También nótese que pueden enviarse x_{ij}° unidades de flujo del nodo j al i, cantidad que cancela el flujo en el arco. […]. Con estas ideas, se define la red residual con respecto a un flujo dado x° como sigue. Se reemplaza cada arco (i,j) de la red original por dos arcos, (i,j) con capacidad $q_{ij} - x_{ij}^{\circ}$ y (j,i) con capacidad x_{ij}°. La red residual consiste sólo de los arcos con capacidad residual positiva. Se usa la notación $G(x^{\circ})$ para representar la red residual correspondiente al flujo x°. (p.44)

Definición 2.3.1.4.- Dada una red con capacidades en los arcos $R = [V, A, q]$ y flujo x, la *red residual de R inducida por* x *es* $G(f) = [V, A', q]$ tal que si $(u, v, q_{uv}) \in A \Longrightarrow (u, v, q_{uv} - x_{uv})$ y $(v, u, x_{vu}) \in A'$.

La capacidad residual r_{ij} tiene dos componentes; $q_{ij} - x_{ij}$, que es la capacidad sin usar del arco (i,j), y el flujo x_{ij}, ya que éste puede cancelarse para modificar los flujos en pos de obtener un flujo máximo. De este modo, el valor del flujo x_{ij} corresponde a x_{ji} en el arco (j,i) en la red residual. En el inciso a) de la figura 2.3.1.3, los paréntesis sobre los arcos indican flujo y capacidad. Por ejemplo, en el arco $(1,3)$ con flujo 5 y capacidad 6, es posible enviar una unidad extra de flujo, lo

[24] Ahuja, R., Magnati, T., Orlín, J. (1993) *Network Flows Theory, Algorithms, and Applications*, 1ra edición,Upper Saddle River; Prentice-Hall Inc.

que genera el arco $(1,3)$ con $q_{ij} - x_{ij} = 1$ en la red residual. Del mismo modo es posible cancelar el flujo de valor 5, lo que se representa en la segunda red con el arco $(3,1)$ con $x_{ij} = 5$.

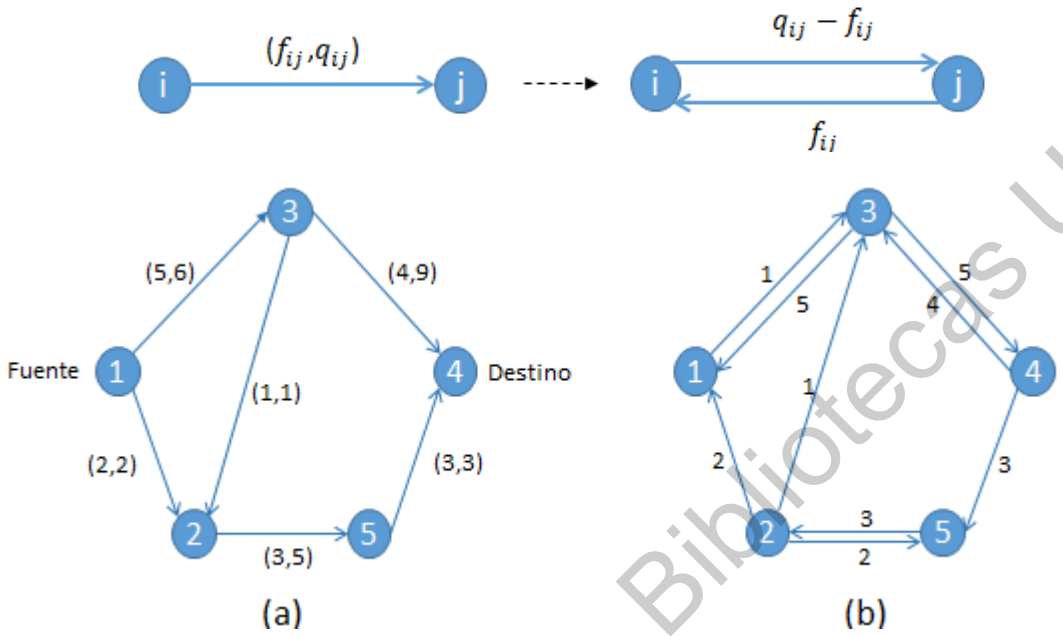


Figura 2.3.1.3 Ilustración de una red residual: **a)** red original con flujo f; **b)** red residual *G(f)*

Definición 2.3.1.5.- Se define la *capacidad* $q(S, \bar{S})$ de un corte $(S, \bar{S})$ como la suma de las capacidades de los arcos de avance en el corte, es decir,

$$q(S, \bar{S}) = \sum_{(i,j) \in (S, \bar{S})} q_{ij}$$

Definición 2.3.1.6.- *Corte mínimo s-t* es aquel cuya capacidad es mínima entre todos los posibles cortes $s - t$.

Lema 2.3.1.1.- El valor de cualquier flujo x es menor o igual que la capacidad de cualquier corte en la red.

$$x \leq q(S, \bar{S})$$

Demostración.-

Sea x un flujo en una red, denotando como x_{ij} el flujo a través del arco $(i,j) \in A$ y el conjunto $S \in V$ a partir de la cual se hará el corte (con $s \in S, t \in \bar{S}$). Por las restricciones para el flujo máximo (definición 2.2.2), para los nodos en S se tiene:

(2) $$x = \sum_{i \in S} \left[\sum_{\{j:(i,j) \in A\}} x_{ij} - \sum_{\{j:(j,i) \in A\}} x_{ji} \right]$$

Puede simplificarse esta expresión notando que si cualesquiera nodos a y b, con $a, b \in S$ y $(a, b) \in A$, la variable x_{ab} cancela la variable $-x_{ba}$. Si por otro lado, $a, b \in \bar{S}$, x_{pq} no aparece en la expresión. Luego;

(3) $$\boldsymbol{x = \sum_{(i,j) \in (S,\bar{S})} x_{ij} - \sum_{(i,j) \in (\bar{S},S)} x_{ij}}$$

La primera suma expresa el flujo de los nodos de S a los nodos en $\bar{S}$, y la segunda el flujo que regresa de los nodos en $\bar{S}$ a S. Luego, el lado derecho de la igualdad denota el flujo neto a través de la corte, lo que implica que el flujo a través de cualquier corte $(S, \bar{S})$ es igual a x. Ya que $x_{ij} \leq q_{ij}$ y $x_{ji} \geq 0$, se tiene que:

(4) $$x = \sum_{(i,j) \in (S,\bar{S})} x_{ij} - \sum_{(i,j) \in (\bar{S},S)} x_{ij} \leq \sum_{(i,j) \in (S,\bar{S})} q_{ij} = q(S, \bar{S})$$

De donde el valor de cualquier flujo es menor o igual a la capacidad de cualquier corte $(S, \bar{S})$ en la red. ■

Intuitivamente, puede pensarse de la siguiente de la siguiente manera: En cualquier $corte\ s - t$ el flujo de s a t debe pasar a través del corte, ya que ésta divide la red en dos componentes disjuntos, luego el valor del flujo no puede exceder la capacidad del corte.

Lema 2.3.1.2.- Para cualquier flujo x en una red, el flujo adicional que puede enviarse del nodo fuente s al nodo destino t es menor o igual a la capacidad residual de cualquier corte $s - t$.

Demostración.-

Sea Δx la capacidad residual del corte $(S, \bar{S})$ en la red. Considerando el flujo $x + \Delta x$, por (4) del lema anterior se tiene que

$$x + \Delta x \leq \sum_{(i,j) \in (S,\bar{S})} q_{ij}$$

Ya que por (3) se tiene que $x = \sum_{(i,j) \in (S,\bar{S})} x_{ij} - \sum_{(i,j) \in (\bar{S},S)} x_{ij}$, restando este término de la ecuación anterior se obtiene:

$$\Delta x \leq \sum_{(i,j)\in(S,\bar{S})} q_{ij} - \left(\sum_{(i,j)\in(S,\bar{S})} x_{ij} - \sum_{(i,j)\in(\bar{S},S)} x_{ij} \right) = \sum_{(i,j)\in(S,\bar{S})} (q_{ij} - x_{ij}) + \sum_{(i,j)\in(\bar{S},S)} x_{ij}$$

Se observa que la primera sumatoria da el posible flujo hacia adelante en la red residual y la segunda sumatoria el posible flujo de retorno; eso constituye a capacidad residual del corte. Luego,

$$\Delta x \leq x = \sum_{(i,j)\in(S,\bar{S})} (q_{ij} - x_{ij} + x_{ji}) = \sum_{(S,\bar{S})} r_{ij} \qquad \blacksquare$$

La demostración del Teorema de flujo máximo y corte mínimo depende en gran medida del algoritmo utilizado para la búsqueda del flujo máximo. Recordando el concepto de red residual, considérese que si se tiene un flujo x_{ij} en el arco (i,j), entonces la capacidad efectiva de ese arco se reduce en x_{ij}, es decir a $q_{ij} - x_{ij}$, ya que ésta es la capacidad residual o maximal de flujo adicional que puede enviarse por la arista. Asimismo, la capacidad efectiva de regreso, en el arco (j,i), se incrementa en x_{ij} a $(q_{ji} + x_{ij})$, pues tiene lugar un pequeño incremento en el flujo de regreso bajo la forma de una reducción en el flujo de ida de ese arco. Una vez hechos estos cambios en las capacidades, se puede encontrar un camino del nodo fuente al nodo destino a la cual asignar flujo adicional. Este camino se conoce como *camino aumentante*, el cual se utiliza en el algoritmo de etiquetado desarrollado por Ford y Fulkerson y se abordará en la sección 3.2.

Teorema de Flujo Máximo-Corte Mínimo.- En una red $R = [V, A, q]$ el valor del flujo máximo es igual a la capacidad de la corte mínimo, es decir

$$\max\{x \mid x\ es\ un\ flujo\ factible\} = \min\{q(S,\bar{S}) \mid (S,\bar{S})\ es\ un\ corte\}$$

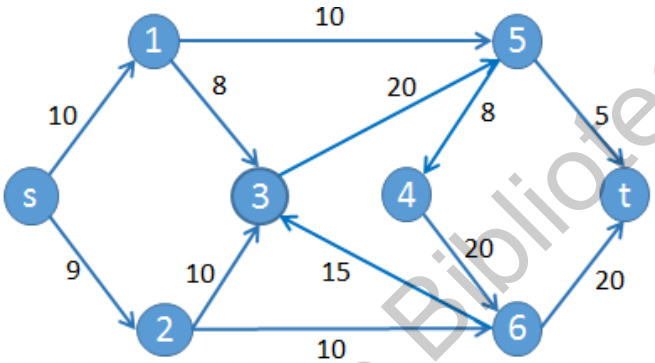
Demostración.-

Dado un flujo factible x, debido al lema 2.3.1.1, x es menor o igual a la capacidad de cualquier corte en la red, en particular, menor o igual al corte mínimo. Luego, basta con mostrar un flujo para el cual se alcance la igualdad. En particular, tómese el flujo maximal obtenido por el algoritmo de etiquetado. Para este flujo, hállense las capacidades residuales de todos los arcos para los cambios de flujo incrementales, y aplíquese el procedimiento de etiquetado, el cual debe terminar antes de etiquetar al nodo destino t, ya que no hay camino de aumento.

Sea S el conjunto formado por los nodos etiquetados, y $\bar{S}$ por los no etiquetados. Así, se ha definido un $corte\ s-t$. Todas las aristas con un extremo en S y otro en $\bar{S}$ tienen una capacidad residual igual a cero, de lo contrario, el nodo en $\bar{S}$ habría sido etiquetado. Esto implica que los arcos del corte quedan saturados por el flujo original,

es decir, el flujo que pasa por ellos es igual a la capacidad. Por otro lado, cualquier arista con un extremo en $\bar{S}$ y otro en S, debe tener flujo cero, ya que, en caso contrario, implicaría una capacidad incremental positiva en la dirección inversa, y se etiquetaría el nodo en $\bar{S}$. Luego, existe un flujo total de S a $\bar{S}$ igual a la capacidad del corte y flujo cero de $\bar{S}$ a S, de donde el flujo del nodo fuente al nodo destino es igual a la capacidad del corte. Así, del lema 2.3.1.2 se concluye que el corte debe ser minimal, y el flujo es maximal. ■

Ejemplo 2.3.1.2.- Considérese nuevamente la red R_1 del ejemplo 2.2.1, se resolverá por programación lineal.



PROBLEMA PRIMAL

$$Max\ Z = x$$

Sujeto a

$$x - x_{s1} - x_{s2} = 0$$
$$x_{s2} - x_{23} - x_{26} = 0$$
$$-x_{46} + x_{54} = 0$$
$$x_{26} + x_{46} - x_{63} - x_{6t} = 0$$
$$x_{s1} \leq 10$$
$$x_{13} \leq 8$$
$$x_{23} \leq 10$$
$$x_{35} \leq 10$$
$$x_{54} \leq 8$$
$$x_{63} \leq 15$$

$$x_{s1} - x_{13} - x_{15} = 0$$
$$x_{13} + x_{23} - x_{35} + x_{63} = 0$$
$$x_{15} + x_{35} - x_{54} - x_{5t} = 0$$
$$x_{5t} + x_{6t} - x = 0$$
$$x_{s2} \leq 9$$
$$x_{15} \leq 10$$
$$x_{26} \leq 10$$
$$x_{46} \leq 20$$
$$x_{5t} \leq 5$$
$$x_{6t} \leq 20$$

$$x_{s1}, x_{s2}, x_{13}, x_{15}, x_{23}, x_{26}, x_{35}, x_{46}, x_{54}, x_{5t}, x_{63}, x_{6t} \geq 0$$

Se obtuvo como resultado de este programa lineal, que el flujo máximo es 19, de acuerdo al Teorema de Flujo Máximo-Corte Mínimo, al encontrar el corte mínimo éste tendrá capacidad igual a 19. Así, resolviendo el programa dual;

PROBLEMA DUAL

$$Min\ W = 10w_9 + 9w_{10} + 8w_{11} + 10w_{12} + 10w_{13} + 10w_{14} + 10w_{15} + 20w_{16} + 8w_{17} + 5w_{18} + 15w_{19} + 20w_{20}$$

Sujeto a

$$\begin{aligned} w_1 + w_8 &\geq 1 \\ -w_1 + w_2 + w_{10} &\geq 0 \\ -w_5 + w_7 + w_{12} &\geq 0 \\ -w_2 + w_4 + w_{14} &\geq 0 \\ -w_3 + w_4 + w_{16} &\geq 0 \\ -w_7 - w_8 + w_{18} &\geq 0 \\ -w_4 - w_8 + w_{20} &\geq 0 \end{aligned} \qquad \begin{aligned} -w_1 + w_5 + w_9 &\geq 0 \\ -w_5 + w_6 + w_{11} &\geq 0 \\ -w_2 + w_6 + w_{13} &\geq 0 \\ -w_6 + w_7 + w_{15} &\geq 0 \\ w_3 - w_7 + w_{17} &\geq 0 \\ w_6 + w_{19} &\geq 0 \end{aligned}$$

$$w_i\ irrestricta\ (i = 1, \dots 20)$$

Ingresando el problema en TORA, en el menú de Programación Lineal, se obtiene como resultado 19, que es efectivamente el valor del problema primal (problema de maximización).

Así, en el resultado del programa dual, las variables duales a partir de la w_7 que resulten contribuir en la solución indican el corte mínimo, ya que de acuerdo a (1), estas variables representan a las capacidades en los arcos y las que resulten valer 1 forman el corte.

En este caso, las variables que nos indican el corte es w_7 y w_8, que corresponden en el programa lineal a las variables x_{s1} y x_{s2}, siendo efectivamente la capacidad de este corte igual a 19.

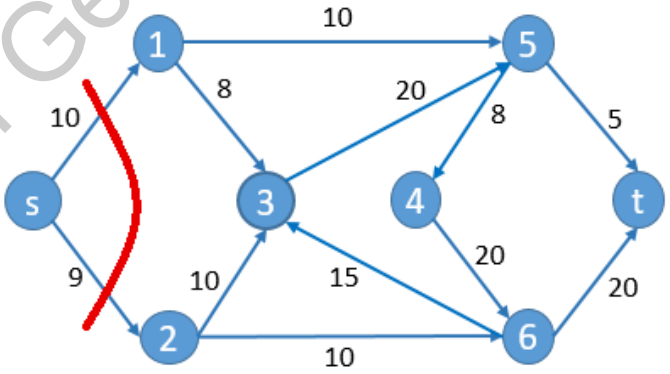


Figura 2.3.1.4 Corte mínimo de capacidad 19 para la red R_1

Title: red 1 dual
Final Iteration No.: 23
Objective Value (Min) =19.00

Next Iteration | All Iterations | Write to Printer

Variable	Value	Obj Coeff	Obj Val Contrib
x1: w1	0.00	0.00	0.00
x2: w2	0.00	0.00	0.00
x3: w3	0.00	0.00	0.00
x4: w4	0.00	0.00	0.00
x5: w5	0.00	0.00	0.00
x6: w6	0.00	0.00	0.00
x7: w7	1.00	10.00	10.00
x8: w8	1.00	9.00	9.00
x9: w9	0.00	8.00	0.00
x10: w10	0.00	10.00	0.00
x11: w11	0.00	10.00	0.00
x12: w12	0.00	10.00	0.00
x13: w13	0.00	10.00	0.00
x14: w14	0.00	20.00	0.00
x15: w15	0.00	8.00	0.00
x16: w16	0.00	5.00	0.00
x17: w17	0.00	15.00	0.00
x18: w18	0.00	20.00	0.00

Figura 2.3.1.5 Solución brindada por TORA para el problema dual de la red R_1

3. MÉTODOS DE SOLUCIÓN

En este capítulo se consideran tres métodos de solución con base en programación lineal: el algoritmo simplex especializado en redes, el método simplex clásico y el método simplex con variables acotadas. Las soluciones correspondientes se obtuvieron con el software WinQSB[25], específicamente con los módulos "Network Modeling" y "Linear and Integer Programming". Posteriormente se aborda el algoritmo de etiquetado de Ford y Fulkerson, seguido de cuatro sugerencias de software libre para la resolución del problema de Flujo Máximo (R, Phyton, QM, TORA), finalizando con dos ejemplos de códigos en línea de libre acceso que permiten al usuario ingresar la red deseada para obtener su flujo máximo así como los pasos intermedios.

3.1 Programación Lineal

Método simplex especializado en redes

Considérese la siguiente red R_2 donde los valores de los arcos indican la capacidad de flujo.

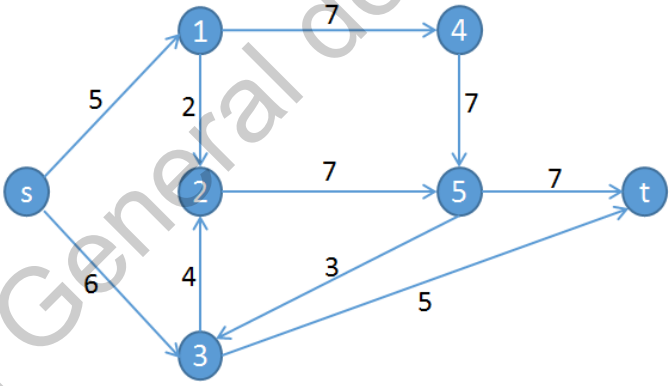


Figura 3.1.1 Red R_2

El programa lineal para flujo máximo en la red R_2 es

$$Max\ Z = x$$

Sujeto a

[25] WinQSB (s.f). En *Swmath.* Recuperado el 10 de Octubre de 2019 de http://www.swmath.org/software/6146

$$x - x_{s1} - x_{s3} = 0 \qquad x_{s,1} - x_{12} - x_{14} = 0$$
$$x_{12} + x_{32} - x_{25} = 0 \qquad x_{s3} + x_{53} - x_{32} - x_{3t} = 0$$
$$x_{14} - x_{45} = 0 \qquad x_{25} + x_{45} - x_{53} - x_{5t} = 0$$
$$x - x_{3t} - x_{5t} = 0 \qquad x_{s1} \leq 5$$
$$x_{s3} \leq 6 \qquad x_{12} \leq 2$$
$$x_{14} \leq 7 \qquad x_{25} \leq 7$$
$$x_{32} \leq 4 \qquad x_{3\mathrm{t}} \leq 5$$
$$x_{45} \leq 7 \qquad x_{53} \leq 3$$
$$x_{5t} \leq 7 \qquad x_{ij} \geq 0$$

Luego, el programa dual es

$$Min\ W = 5y_8 + 6y_9 + 2y_{10} + 7y_{11} + 7y_{12} + 4y_{13} + 5y_{14} + 7y_{15} + 3y_{16} + 7y_{17}$$

Sujeto a

$$y - 5y_8 - 6y_9 - 2y_{10} - 7y_{11} - 7y_{12} - 4y_{13} - 5y_{14} - 7y_{15} - 3y_{16} - 7y_{17} = 0$$

$$y_1 + y_7 \geq 1 \qquad -y_1 + y_2 + y_8 \geq 0$$
$$-y_1 + y_4 + y_9 \geq 0 \qquad -y_2 + y_3 + y_{10} \geq 0$$
$$-y_2 + y_5 + y_{11} \geq 0 \qquad -y_3 + y_6 + y_{12} \geq 0$$
$$y_3 - y_4 + y_{13} \geq 0 \qquad -y_4 - y_7 + y_{14} \geq 0$$
$$-y_5 + y_6 + y_{15} \geq 0 \qquad y_4 - y_6 + y_{16} \geq 0$$
$$-y_6 - y_7 + y_{17} \geq 0 \qquad y_i\ irrestricta$$

Utilizando el módulo “Network Modeling” de WinQSB, la solución con algoritmo simplex especializado en redes muestra un flujo máximo igual a 11.

Este algoritmo sigue los mismos pasos del método simplex, es decir, busca la variable entera entrante, determina la variable básica que sale para así obtener una nueva y mejor solución básica factible, sin embargo, este método ejecuta estos pasos aprovechando la estructura particular de la red y así prescindir de una tabla simplex.

Definición 3.1.1.- Un *árbol* de una gráfica *G* es una subgráfica conexa y sin ciclos.

Definición 3.1.2.- Un *árbol de expansión* de una gráfica *G* es una subgráfica generadora que es un árbol.

El árbol de expansión para el problema de flujo máximo consiste en dos subárboles de la red G unidos por el arco (t, s). Sean T_s y T_t los árboles que contienen a *s* y *t* respectivamente.

Las figuras 3.1.2 y 3.1.3 muestran la introducción y solución de “Network Modeling” con el número de iteraciones y el tiempo de CPU para la red R_2. También, se muestra la solución gráfica. En WinQSB, el límite de nodos a introducir son 2,043.

07-24-2019	From	To	Net Flow		From	To	Net Flow
1	NodoS	Nodo1	5	5	Nodo3	Nodo2	1
2	NodoS	Nodo3	6	6	Nodo3	NodoT	5
3	Nodo1	Nodo4	5	7	Nodo4	Nodo5	5
4	Nodo2	Nodo5	1	8	Nodo5	NodoT	6
Total	Net Flow	From	NodoS	To	NodoT	=	11

Figura 3.1.2 Introducción de los datos de la red R_2 en el módulo de "Network Modeling"

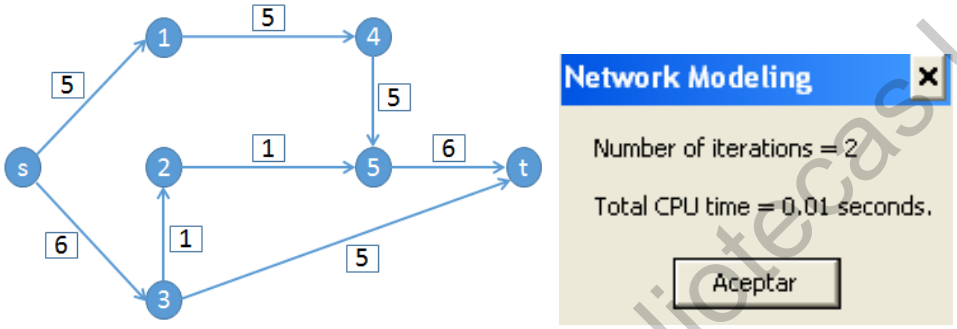


Figura 3.1.3 Solución al problema de flujo máximo de la red R_2 (simplex especializado en redes)

Método simplex clásico

El siguiente es el programa lineal introducido en WinQSB en su módulo "Linear and Integer Programming", que utiliza el método simplex clásico.

Variable	F	FlujoS1	FlujoS3	Flujo12	Flujo14	Flujo25	Flujo32	Flujo3T	Flujo45	Flujo53	Flujo5T	Direction	R. H. S.
Maximize	1	0	0	0	0	0	0	0	0	0	0		
NodoS	1	-1	-1	0	0	0	0	0	0	0	0	=	0
Nodo1	0	1	0	-1	-1	0	0	0	0	0	0	=	0
Nodo2	0	0	0	1	0	-1	1	0	0	0	0	=	0
Nodo3	0	0	1	0	0	0	-1	-1	0	1	0	=	0
Nodo4	0	0	0	0	1	0	0	0	-1	0	0	=	0
Nodo5	0	0	0	0	0	1	0	0	1	-1	-1	=	0
NodoT	-1	0	0	0	0	0	0	1	0	0	1	=	0
ArcoS1	0	1	0	0	0	0	0	0	0	0	0	<=	5
ArcoS3	0	0	1	0	0	0	0	0	0	0	0	<=	6
Arco12	0	0	0	1	0	0	0	0	0	0	0	<=	2
Arco14	0	0	0	0	1	0	0	0	0	0	0	<=	7
Arco25	0	0	0	0	0	1	0	0	0	0	0	<=	7
Arco32	0	0	0	0	0	0	1	0	0	0	0	<=	4
Arco3T	0	0	0	0	0	0	0	1	0	0	0	<=	5
Arco45	0	0	0	0	0	0	0	0	1	0	0	<=	7
Arco53	0	0	0	0	0	0	0	0	0	1	0	<=	3
Arco5T	0	0	0	0	0	0	0	0	0	0	1	<=	7
LowerBou	0	0	0	0	0	0	0	0	0	0	0		
UpperBou	M	M	M	M	M	M	M	M	M	M	M		
VariableT	nuous	tinuous	ntinuous	ıtinuous	tinuous	ıtinuous	tinuous	ıtinuous	tinuous	ntinuous	ntinuous		

Figura 3.1.4 Introducción del programa lineal de la red R_2 en el módulo de programación lineal de WinQSB

La solución obtenida, con el número de iteraciones y el tiempo de CPU es:

	Decision Variable	Solution Value	Unit Cost or Profit c(j)	Total Contribution	Reduced Cost	Basis Status	Allowable Min. c(j)	Allowable Max. c(j)
1	F	11.0000	1.0000	11.0000	0	basic	0	M
2	FlujoS1	5.0000	0	0	0	basic	-1.0000	M
3	FlujoS3	6.0000	0	0	0	basic	-1.0000	M
4	Flujo12	2.0000	0	0	0	basic	0	M
5	Flujo14	3.0000	0	0	0	basic	-1.0000	0
6	Flujo25	3.0000	0	0	0	basic	0	0
7	Flujo32	1.0000	0	0	0	basic	-1.0000	0
8	Flujo3T	5.0000	0	0	0	basic	0	M
9	Flujo45	3.0000	0	0	0	basic	-1.0000	0
10	Flujo53	0	0	0	0	at bound	-M	0
11	Flujo5T	6.0000	0	0	0	basic	-1.0000	0
	Objective	Function	(Max.) =	11.0000	(Note:	Alternate	Solution	Exists!!)
	Constraint	Left Hand Side	Direction	Right Hand Side	Slack or Surplus	Shadow Price	Allowable Min. RHS	Allowable Max. RHS
1	NodoS	0	=	0	0	1.0000	0	M
2	Nodo1	0	=	0	0	0	0	3.0000
3	Nodo2	0	=	0	0	0	0	3.0000
4	Nodo3	0	=	0	0	0	0	1.0000
5	Nodo4	0	=	0	0	0	0	3.0000
6	Nodo5	0	=	0	0	0	0	6.0000
7	NodoT	0	=	0	0	0	0	M
8	ArcoS1	5.0000	<=	5.0000	0	1.0000	2.0000	6.0000
9	ArcoS3	6.0000	<=	6.0000	0	1.0000	5.0000	7.0000
10	Arco12	2.0000	<=	2.0000	0	0	0	5.0000
11	Arco14	3.0000	<=	7.0000	4.0000	0	3.0000	M
12	Arco25	3.0000	<=	7.0000	4.0000	0	3.0000	M
13	Arco32	1.0000	<=	4.0000	3.0000	0	1.0000	M
14	Arco3T	5.0000	<=	5.0000	0	0	4.0000	6.0000
15	Arco45	3.0000	<=	7.0000	4.0000	0	3.0000	M
16	Arco53	0	<=	3.0000	3.0000	0	0	M
17	Arco5T	6.0000	<=	7.0000	1.0000	0	6.0000	M

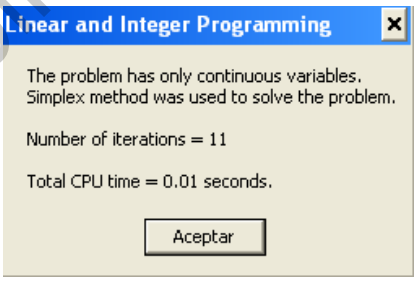


Figura 3.1.5 Solución al problema de flujo máximo de la red R_2 (simplex clásico)

Siendo la solución con algoritmo simplex un flujo máximo igual a 11.

Una gran ventaja de la solución por el método simplex clásico, es que al final se obtiene información que se utiliza en el análisis post óptimo.

En el listado de solución mostrado arriba, en la columna “slack or surplus” (variables de holgura), los valores mayores a cero indican las variables, en este caso flujos que aún no saturan al arco. Por ejemplo, en la solución mostrada, en el arco $(2,5)$ con capacidad 7, el flujo es igual a 3, de donde la columna slack or surplus muestra una holgura igual a 4.

Los valores de la columna “shadow prices,” son los precios sombra asociados a las restricciones del programa lineal en la solución óptima. Para las restricciones de capacidad de los arcos, estos precios sombra identifican a los arcos que al aumentar (disminuir) su capacidad aumentan (disminuyen) el objetivo óptimo del flujo máximo.

El precio sombra es de suma importancia en la teoría económica; corresponde a la tasa de cambio del valor objetivo óptimo Z^* respecto a la j-ésima restricción b_j. Fácilmente se verifica que dicho valor es justamente la variable dual j-ésima y_j:

$$y_j^* = \frac{\delta Z^*}{\delta b_j}$$

Los precios sombra indican los correspondientes aumentos al valor óptimo actual por cada unidad que aumente la capacidad del arco, por ejemplo, en el caso de la restricción del arco $(s,1)$ el precio sombra es igual a 1, lo que implica que disponer de una unidad más de capacidad en ese arco aumenta en 1 el resultado óptimo.

Estas consideraciones sólo tienen validez en los intervalos generados en el análisis de sensibilidad de la solución, mostrados en las columnas “allowable Min. RHS” y “allowable Max. RHS”, que indican el máximo y mínimo valor permitido para las restricciones tal que se conserve la base óptima.

Por ejemplo, la restricción del arco $(s,3)$ puede variar en el intervalo $[6,7]$ sin que la optimalidad de la solución cambie.

El listado de salida de programación lineal indica una solución alterna, éste y la solución gráfica se muestran en la figura 3.1.6 y 3.1.7:

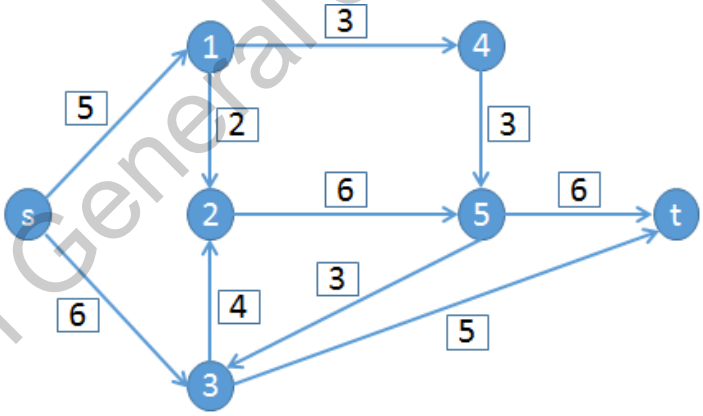


Figura 3.1.6 Solución gráfica alternativa para el problema de flujo máximo de la red R_2

	Decision Variable	Solution Value	Unit Cost or Profit c(j)	Total Contribution	Reduced Cost	Basis Status	Allowable Min. c(j)	Allowable Max. c(j)
1	F	11.0000	1.0000	11.0000	0	basic	0	M
2	FlujoS1	5.0000	0	0	0	basic	-1.0000	M
3	FlujoS3	6.0000	0	0	0	basic	-1.0000	M
4	Flujo12	2.0000	0	0	0	basic	0	M
5	Flujo14	3.0000	0	0	0	basic	-1.0000	0
6	Flujo25	6.0000	0	0	0	basic	0	0
7	Flujo32	4.0000	0	0	0	basic	0	M
8	Flujo3T	5.0000	0	0	0	basic	0	M
9	Flujo45	3.0000	0	0	0	basic	-1.0000	0
10	Flujo53	3.0000	0	0	0	basic	0	1.0000
11	Flujo5T	6.0000	0	0	0	basic	-1.0000	0
	Objective	Function	(Max.) =	11.0000	(Note:	Alternate	Solution	Exists!!)
	Constraint	Left Hand Side	Direction	Right Hand Side	Slack or Surplus	Shadow Price	Allowable Min. RHS	Allowable Max. RHS
1	NodoS	0	=	0	0	1.0000	0	M
2	Nodo1	0	=	0	0	0	0	3.0000
3	Nodo2	0	=	0	0	0	0	6.0000
4	Nodo3	0	=	0	0	0	0	0
5	Nodo4	0	=	0	0	0	0	3.0000
6	Nodo5	0	=	0	0	0	0	6.0000
7	NodoT	0	=	0	0	0	0	M
8	ArcoS1	5.0000	<=	5.0000	0	1.0000	2.0000	6.0000
9	ArcoS3	6.0000	<=	6.0000	0	1.0000	6.0000	7.0000
10	Arco12	2.0000	<=	2.0000	0	0	0	3.0000
11	Arco14	3.0000	<=	7.0000	4.0000	0	3.0000	M
12	Arco25	6.0000	<=	7.0000	1.0000	0	6.0000	M
13	Arco32	4.0000	<=	4.0000	0	0	1.0000	4.0000
14	Arco3T	5.0000	<=	5.0000	0	0	4.0000	5.0000
15	Arco45	3.0000	<=	7.0000	4.0000	0	3.0000	M
16	Arco53	3.0000	<=	3.0000	0	0	3.0000	M
17	Arco5T	6.0000	<=	7.0000	1.0000	0	6.0000	M

Figura 3.1.7 Solución alternativa para el problema de flujo máximo de la red R_2

El programa dual asociado al problema de flujo máximo anterior es como sigue:

Variable	NodoS	Nodo1	Nodo2	Nodo3	Nodo4	Nodo5	NodoT	ArcoS1	ArcoS3	Arco12	Arco14	Arco25	Arco32	Arco3T	Arco45	Arco53	Arco5T	Direction	R. H. S.
Minimize								5	6	2	7	7	4	5	7	3	7		
F	1						-1											>=	1
FlujoS1	-1	1						1										>=	0
FlujoS3	-1			1					1									>=	0
Flujo12		-1	1							1								>=	0
Flujo14		-1			1						1							>=	0
Flujo25			-1			1						1						>=	0
Flujo32			1	-1									1					>=	0
Flujo3T				-1			1							1				>=	0
Flujo45					-1	1									1			>=	0
Flujo53				1		-1										1		>=	0
Flujo5T						-1	1										1	>=	0
LowerBo	-M	-M	-M	-M	-M	-M	-M	0	0	0	0	0	0	0	0	0	0		
UpperBo	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M		
Variable	tinuous	tinuous	inuous	tinuous	inuous	inuous	inuous	tinuous	tinuous	tinuous	tinuous	tinuous	tinuous	tinuous	tinuous	tinuous	tinuous		

Figura 3.1.8 Dual del programa lineal de flujo máximo de la red R_2 (con WinQSB)

La solución del dual, con el método simplex clásico se muestra en la figura 3.1.9 donde se incluyen el número de iteraciones y el tiempo de CPU empleado. En la figura 3.1.10 se muestra la solución dual, que es el corte mínimo con valor igual a 11.

	Decision Variable	Solution Value	Unit Cost or Profit c[j]	Total Contribution	Reduced Cost	Basis Status	Allowable Min. c[j]	Allowable Max. c[j]
1	NodoS	1.0000	0	0	0	basic	0	0
2	Nodo1	0	0	0	0	basic	0	0
3	Nodo2	0	0	0	0	basic	0	0
4	Nodo3	0	0	0	0	basic	0	0
5	Nodo4	0	0	0	0	basic	0	0
6	Nodo5	0	0	0	0	basic	0	0
7	NodoT	0	0	0	0	at bound	0	M
8	ArcoS1	1.0000	5.0000	5.0000	0	basic	2.0000	6.0000
9	ArcoS3	1.0000	6.0000	6.0000	0	basic	5.0000	7.0000
10	Arco12	0	2.0000	0	0	basic	0	5.0000
11	Arco14	0	7.0000	0	4.0000	at bound	3.0000	M
12	Arco25	0	7.0000	0	4.0000	at bound	3.0000	M
13	Arco32	0	4.0000	0	3.0000	at bound	1.0000	M
14	Arco3T	0	5.0000	0	0	basic	4.0000	6.0000
15	Arco45	0	7.0000	0	4.0000	at bound	3.0000	M
16	Arco53	0	3.0000	0	3.0000	at bound	0	M
17	Arco5T	0	7.0000	0	1.0000	at bound	6.0000	M
	Objective	Function	(Min.) =	11.0000	(Note:	Alternate	Solution	Exists!!)

	Constraint	Left Hand Side	Direction	Right Hand Side	Slack or Surplus	Shadow Price	Allowable Min. RHS	Allowable Max. RHS
1	F	1.0000	>=	1.0000	0	11.0000	0	M
2	FlujoS1	0	>=	0	0	5.0000	-1.0000	M
3	FlujoS3	0	>=	0	0	6.0000	-1.0000	M
4	Flujo12	0	>=	0	0	2.0000	0	M
5	Flujo14	0	>=	0	0	3.0000	-1.0000	0
6	Flujo25	0	>=	0	0	3.0000	0	0
7	Flujo32	0	>=	0	0	1.0000	-1.0000	0
8	Flujo3T	0	>=	0	0	5.0000	0	M
9	Flujo45	0	>=	0	0	3.0000	-1.0000	0
10	Flujo53	0	>=	0	0	0	-M	0
11	Flujo5T	0	>=	0	0	6.0000	-1.0000	0

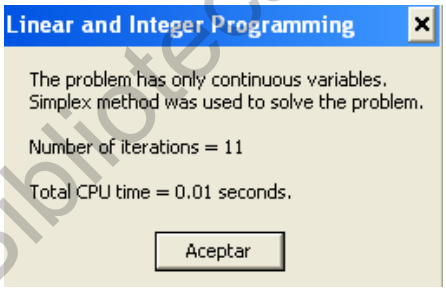


Figura 3.1.9 Solución óptima del dual; corte mínimo en arcos (s,1) y (s,3)

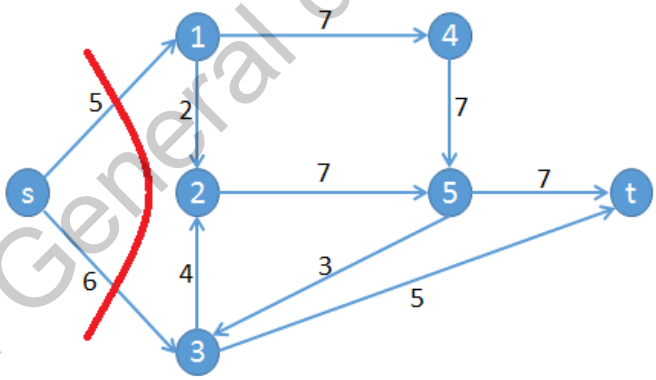


Figura 3.1.10 Corte mínimo de la red R_2, con capacidad igual a 11

Método simplex con variables acotadas

En el programa lineal de flujo máximo, tener variables con cota superior sugiere usar el algoritmo simplex con variables acotadas. No tratar a las variables como restricciones conlleva ventajas computacionales, pues en los criterios de entradas y salidas de variables a la base, los límites de las variables acotadas se manejan implícitamente. En este método las cotas superiores no se deben sobrepasar cuando la variable básica entrante se incrementa hasta alcanzar un nuevo vértice.

La figura 3.1.11 muestra la entrada de datos del problema de flujo máximo anterior, con variables acotadas.

Variable –>	F	FlujoS1	FlujoS3	Flujo12	Flujo14	Flujo25	Flujo32	Flujo3T	Flujo45	Flujo53	Flujo5T	Direction	R. H. S.
Maximize	1	0	0	0	0	0	0	0	0	0	0		
NodoS	1	-1	-1	0	0	0	0	0	0	0	0	=	0
Nodo1	0	1	0	-1	-1	0	0	0	0	0	0	=	0
Nodo2	0	0	0	1	0	-1	1	0	0	0	0	=	0
Nodo3	0	0	1	0	0	0	-1	-1	0	1	0	=	0
Nodo4	0	0	0	0	1	0	0	0	-1	0	0	=	0
Nodo5	0	0	0	0	0	1	0	0	1	-1	-1	=	0
NodoT	-1	0	0	0	0	0	0	1	0	0	1	=	0
LowerBound	0	0	0	0	0	0	0	0	0	0	0		
UpperBounc	M	5	6	2	7	7	4	5	7	3	7		
VariableTyp	inuous	ntinuous	ntinuous	ntinuous	ntinuous	ntinuous	ntinuous	ntinuous	tinuous	tinuous	ntinuous		

Figura 3.1.11 Problema de flujo máximo de la red R_2 con variables acotadas

Ahora se presenta la resolución por simplex con variables acotadas, y enseguida el número de iteraciones con el tiempo de CPU empleado.

Decision Variable	Solution Value	Unit Cost or Profit c[j]	Total Contribution	Reduced Cost	Basis Status	Allowable Min. c[j]	Allowable Max. c[j]
F	11.0000	1.0000	11.0000	0	basic	0	M
FlujoS1	5.0000	0	0	0	basic	-1.0000	M
FlujoS3	6.0000	0	0	0	basic	-1.0000	M
Flujo12	2.0000	0	0	0	basic	0	M
Flujo14	3.0000	0	0	0	basic	-1.0000	0
Flujo25	3.0000	0	0	0	basic	0	0
Flujo32	1.0000	0	0	0	basic	-1.0000	0
Flujo3T	5.0000	0	0	0	basic	0	M
Flujo45	3.0000	0	0	0	basic	-1.0000	0
Flujo53	0	0	0	0	at bound	-M	0
Flujo5T	6.0000	0	0	0	basic	-1.0000	0
Objective	Function	(Max.) =	11.0000	(Note:	Alternate	Solution	Exists!!)

Constraint	Left Hand Side	Direction	Right Hand Side	Slack or Surplus	Shadow Price	Allowable Min. RHS	Allowable Max. RHS
NodoS	0	=	0	0	1.0000	0	M
Nodo1	0	=	0	0	0	0	3.0000
Nodo2	0	=	0	0	0	0	3.0000
Nodo3	0	=	0	0	0	0	1.0000
Nodo4	0	=	0	0	0	0	3.0000
Nodo5	0	=	0	0	0	0	6.0000
NodoT	0	=	0	0	0	0	M

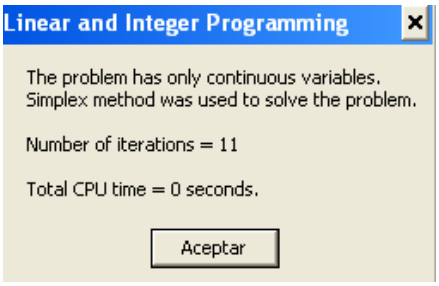


Figura 3.1.12 Solución de problema de flujo máximo de la red R_2 por método simplex con variables acotadas

Puesto que el software WinQSB reporta número de iteraciones y tiempo de CPU al resolver un problema de flujo máximo, esto permite comparar la eficiencia computacional de los métodos mostrados.

Nótese que en el caso de usar el algoritmo simplex con variables acotadas el tiempo computacional de resolución es más eficiente. En cuanto a iteraciones, el método simplex especializado en redes resultó el más eficiente

3.2 Algoritmo de Ford y Fulkerson

El algoritmo de etiquetado de Ford y Fulkerson es el primer algoritmo propuesto para resolver el problema de flujo máximo, y se basa en los conceptos de gráfica residual y caminos aumentantes.

Algoritmo de etiquetado de Ford y Fulkerson.- La etiquetas se asignan a cada nodo y tienen forma (k, c_i), donde k denota el nodo precursor y c_i es el flujo que puede enviarse de la fuente al nodo i por el camino creado con el etiquetado y examen en cada iteración. El procedimiento es el siguiente:

Paso 0. Iguálese $f = 0$

Paso 1. Etiquétese el nodo s con $(-, \infty)$. El resto de los nodos están sin etiquetar.

Paso 2. Selecciónese cualquier nodo etiquetado i, con etiqueta (k, c_i). Para todos los nodos no etiquetados j, tales que $f_{ij} < q_{ij}$, asígnese la etiqueta (i, c_j), donde $c_j = \min\{c_i, q_{ij} - f_{ij}\}$. Para todos los nodos j no etiquetados tales que (j, i) sea una arista con $f_{ji} > 0$, asígnese la etiqueta (i, c_j), donde $c_j = \min\{c_i, f_{ji}\}$.

Paso 3. Repítase el paso 2 hasta que el nodo t esté etiquetado o no se puedan asignar más etiquetas. En este último caso, la solución actual es optimal.

Paso 4. (Aumento) Si el nodo t se etiqueta (i, c_t), entonces increméntense f y el flujo del arco (i, t) en c_t. Prosiga regresivamente por el camino de aumento determinado por los nodos, incrementando el flujo de cada arco del camino en c_t. Vuelva al paso1.

Al terminar el algoritmo, el nodo t no tiene etiqueta, luego, el corte mínimo está dado por el conjunto de arcos:

$$(S, \bar{S}) = \{(i, j) \in A \mid i \text{ tiene etiqueta}, j \text{ no tiene etiqueta}\} \quad (1)$$

Para justificar (1), considérese a los arcos tal que al terminar el algoritmo tienen en un extremo un vértice etiquetado y en el otro, uno sin etiquetar. Este conjunto forma un corte de capacidad igual al flujo logrado al final del algoritmo, el cual es máximo. Nótese la similitud con la demostración del teorema de flujo máximo- corte mínimo.

Ejemplo 3.2.1.-

La figura 3.2.1(a) representa la red R_2 original con las capacidades indicadas en los arcos y un flujo inicial de 0. En el inciso (b) los nodos muestran las etiquetas obtenidas por el algoritmo. El nodo t obtuvo la etiqueta $(3,5)$, lo que indica que el flujo en el arco $(5, t)$ debe incrementarse en 5. Así, siguiendo las etiquetas, el

camino aumentante (en 3 unidades) es $s \rightarrow 3 \rightarrow t$, el camino se remarca en negro, los números en recuadros indican el flujo total a través del arco.

El inciso (c) muestra las etiquetas actualizadas, de donde la etiqueta $(5,1)$ en el nodo t muestra el camino aumentante en una unidad $s \rightarrow 3 \rightarrow 2 \rightarrow 5 \rightarrow t$. El flujo a través de los arcos también ha sido actualizado.

En el inciso (d), reiniciando el etiquetado, nótese que el nodo 3 no puede etiquetarse desde el nodo s, ya que no hay capacidad sin usar en esta dirección, es decir $f_{s3} \not< q_{s3}$. Sin embargo, el nodo 3 sí puede etiquetarse desde el nodo 2, ya que el flujo existente (de valor 1) proporciona una capacidad inversa de una unidad (de acuerdo al algoritmo, para el nodo 3 se cumple que $f_{32} > 0$, se asigna la etiqueta $(2, c_j) = (2,1)$. El nodo destino ha sido nuevamente etiquetado y se encuentra un nuevo camino aumentante de valor 2, $s \rightarrow 1 \rightarrow 2 \rightarrow 5 \rightarrow t$. Se vuelve al paso 2) hasta etiquetar nuevamente al nodo destino, encontrando nuevamente un camino aumentante de valor 3, el cual se muestra en el inciso (e) de la figura. En el inciso (e) se muestra el flujo total de valor 11 resultado de los caminos aumentantes.

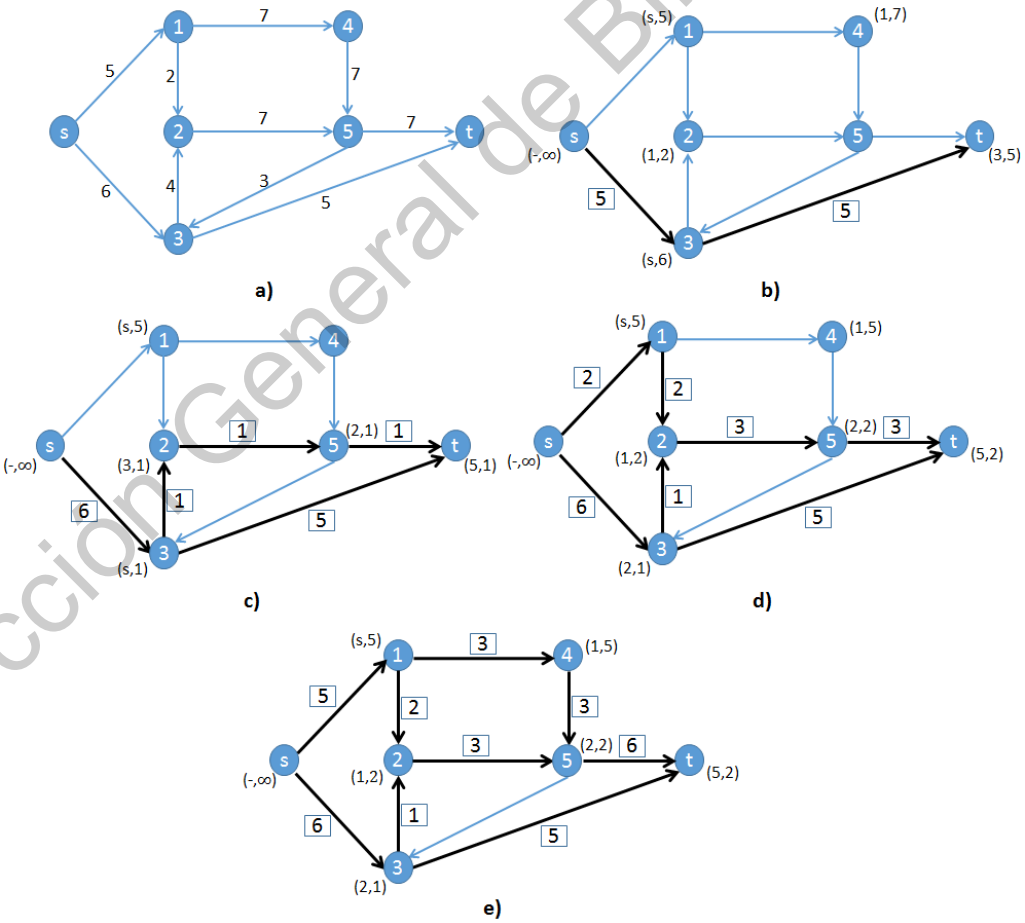


Figura 3.2.1 Aplicación del algoritmo de Ford y Fulkerson a la red R_2

El flujo encontrado es maximal, ya que reiniciando el algoritmo sólo puede etiquetarse el nodo fuente. Recuérdese que cuando el algoritmo no etiqueta al nodo destino, termina el proceso, lo que genera una partición de los nodos entre los etiquetados y no etiquetados, de donde se genera el corte mínimo. Así, al reiniciar el algoritmo se etiqueta al nodo fuente (s, ∞) y el resto de los nodos queda sin etiquetar, siendo la partición $[S, \bar{S}] = \{(s,1), (s,3)\}$.

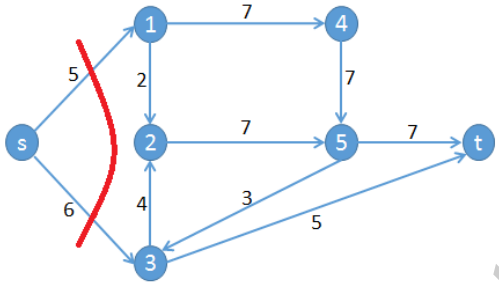


Figura 3.2.2 *Corte mínimo de la red* R_2

Ejemplo 3.2.2.-

La figura 3.2.3 muestra el algoritmo de etiquetado para la red R_1. El inciso (a) muestra que el nodo t tiene la etiqueta inicial $(6,9)$, de donde se genera el camino aumentante $s \to 2 \to 6 \to t$ de valor 9. En el inciso (b), se etiquetó el nodo t como $(5,5)$, con camino aumentante $s \to 1 \to 5 \to t$ de valor 5. En el inciso (c), la etiqueta en t es $(6,5)$ y genera el camino aumentante $s \to 1 \to 5 \to 4 \to 6 \to t$ con valor 5. Al reiniciar el algoritmo, el único nodo que puede ser etiquetado es el nodo fuente s, de donde el flujo máximo es igual a 19.

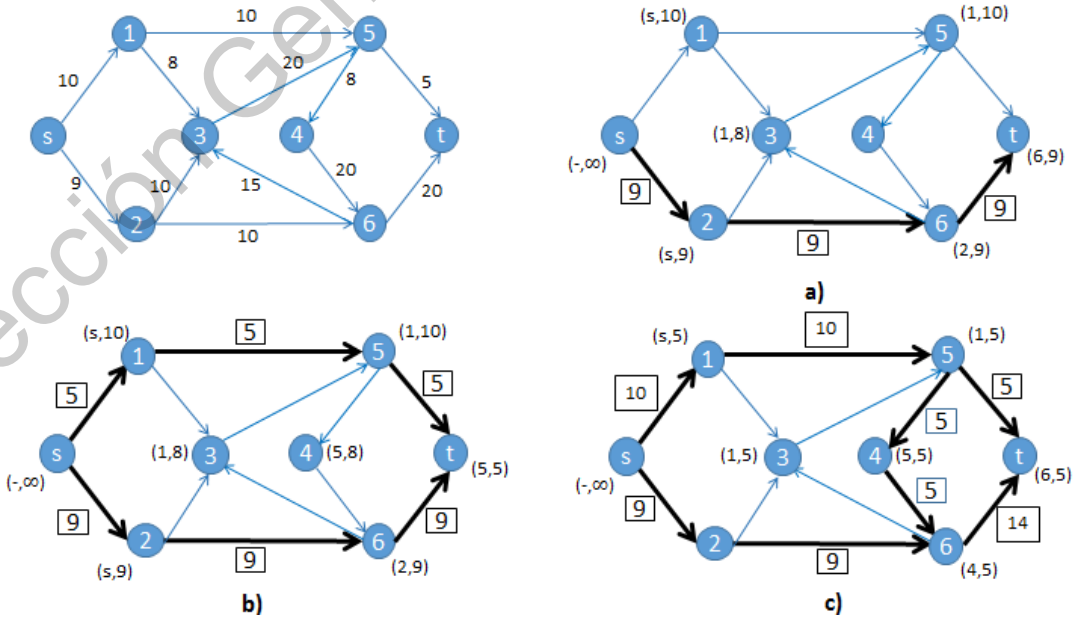


Figura 3.2.3 Algoritmo de etiquetado aplicado a la red R_1

En este caso, al reiniciar el algoritmo por cuarta vez, se etiqueta al nodo fuente con $(-,\infty)$, con el resto de los nodos sin etiquetar. En el paso 2, se selecciona a s (el único nodo etiquetado), pero no hay nodos no etiquetados j tal que $f_{sj} < q_{sj}$, puesto que, debido a la iteración anterior del algoritmo, $f_{s1} = 10 = q_{s1}$ y $f_{s2} = 9 = q_{s2}$. No se pueden asignar más etiquetas, luego, el flujo encontrado actual es optimal.

El único nodo etiquetado es el nodo fuente, de donde por (1), el corte está dado por los arcos $(s,1)$ y $(s,2)$.

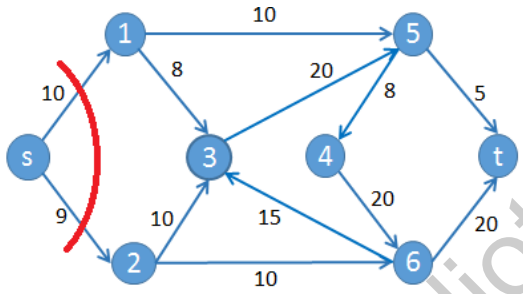


Figura 3.2.4 Corte mínimo de la red R_1

En cuanto a complejidad computacional se refiere, se aceptan tres enfoques básicos para la medida de la bondad de un algoritmo:

1.-Análisis del caso más favorable: La rapidez que posee el algoritmo bajo las mejores condiciones.

2.-Análisis del caso medio: Rendimiento medio del algoritmo considerando todos los posibles conjuntos de datos iniciales. Normalmente se asume que todos son igualmente probables, o se elige una distribución de probabilidad. Este caso suele ser complicado de calcular y puede ser poco significativo si muchos de los conjuntos de datos afectan poco o son improbables en la práctica.

3.-Análisis del caso más desfavorable: La rapidez que posee el algoritmo cuando los datos de entrada son tales que generan el peor rendimiento. Este análisis provee una cota superior del número de pasos que un algoritmo ejecuta para la realización del problema.

El estudio del caso más desfavorable suele ser el más utilizado. Así, tomando el algoritmo de etiquetado de Ford y Fulkerson en una red R con n vértices y m arcos, en cada iteración del algoritmo se etiqueta algún nodo i al menos una vez, inspeccionando cada arco (i,j). Luego, en el proceso de etiquetado se examina cada arco al menos una vez, lo que requiere $O(m)$ computaciones. Tómese Q una cota superior para las capacidades de los arcos, luego, la capacidad de un corte $s-t$ en R es a lo más Qn, lo que acota también el flujo máximo. Ya que el algoritmo de etiquetado incrementa el flujo en al menos una unidad en cada iteración, se

necesitan Qn iteraciones para obtener dicho valor de flujo, de donde el algoritmo tiene una complejidad de $O(mQn)$.

Es evidente que el algoritmo pierde rendimiento cuanto más grande sea el valor de Q, sin embargo existe una refinación del algoritmo hecha en 1972 por el matemático canadiense Jack Edmonds y el científico de la computación estadounidense Richard Manning Karp[26]. Edmonds y Karp demostraron que si en cada iteración se selecciona el camino de aumento más corto, luego, hay a lo más nm caminos de aumento. Encontrar este camino mínimo requiere de la llamada búsqueda de anchura (algoritmo de búsqueda en gráficas), la cual requiere $O(m)$ cómputos. Luego, el algoritmo tiene una complejidad $O(n^2m)$, lo cual en términos generales es mucho más eficiente que $O(mQn)$.

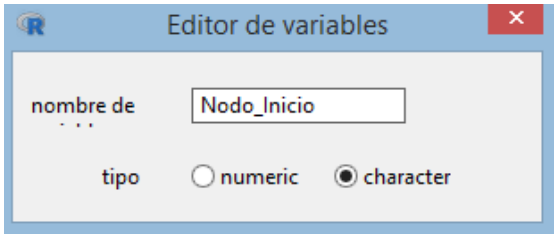
3.3 Rutinas de software libre

El surgimiento del movimiento de software libre en 1983, impulsado por Richard Stallman y que condujo a la fundación de la *Free Software Foundation*[27], ha permitido el desarrollo de lenguajes de programación de uso libre como R y Python, que a lo largo del tiempo y con trabajo colaborativo han generado gran cantidad de bibliotecas con rutinas especializadas para la solución de problemas en diversos campos de las matemáticas y la ingeniería. Dentro de estas rutinas desarrolladas y de uso libre están las correspondientes al cálculo del flujo máximo y del corte mínimo en una gráfica conexa, las cuales se describen y ejemplifican a continuación.

R es un entorno y lenguaje de programación enfocado al análisis estadístico, desarrollado en sus inicios por Robert Gentleman y Ross Ihaka del Departamento de Estadística de la Universidad de Auckland[28]. Retomando la red R_2, R también ofrece una función para resolver el problema de flujo máximo usando “igraph”, la cual es una biblioteca para el análisis de redes. El objetivo principal de esta biblioteca es proporcionar un conjunto de tipos de datos y funciones para la implementación de algoritmos de gráficas y el manejo rápido de gráficas grandes.

Debido al tamaño de la red R_2, los datos pueden introducirse directamente en R usando la función *edit(data.frame ()).* También es posible modificar el nombre de las columnas en el editor de datos, se puede verificar imprimiendo el data frame, en este caso llamado “datos” como se indica en la figura 3.3.1.

[26] Cook, W.J., Cunningham, W.H., Pulleyblank, W.R., Schrijver, A. (1998) *Combinatorial Optimization,* Canadá, John Wiley & Sons, Inc. p. 44.
[27] Free Software Foundation. (s.f). En *Wikipedia*. Recuperado el 30 de agosto de 2019 de https://es.wikipedia.org/wiki/Free_Software_Foundation
[28] R (lenguaje de programación). (s.f). En *Wikipedia*. Recuperado el 15 de septiembre de 2019 de https://es.wikipedia.org/wiki/R_(lenguaje_de_programacion)



Editor de datos

	Nodo_Inic>	Nodo_Final	Capacidad	var4
1	1	2	5	
2	1	4	6	
3	2	3	2	
4	2	5	7	
5	3	6	7	
6	4	3	4	
7	4	7	5	
8	5	6	7	
9	6	4	3	
10	6	7	7	
11				

```
> datos<-edit(data.frame())
> datos
   Nodo_Inicio Nodo_Final Capacidad
1            1          2         5
2            1          4         6
3            2          3         2
4            2          5         7
5            3          6         7
6            4          3         4
7            4          7         5
8            5          6         7
9            6          4         3
10           6          7         7
```

Figura 3.3.1 Ejemplificación del uso de la función *edit(data.frame())*

Se agrega la librería, se le asigna al objeto *G* la función *graph.data.frame*, la cual crea gráficas *igraph* de uno o dos marcos de datos, *class(G)* declara la clase del objeto, *V(G)$name* nombres de los vértices y *E(G)$Capacidad* la capacidad de las aristas. Se declara un vector w el cual contiene las capacidades.

```
> library("igraph")
> G<-graph.data.frame(datos,directed=TRUE)
> class(G)
[1] "igraph"
> V(G)$name
[1] "1" "2" "3" "4" "5" "6" "7"
> E(G)$Capacidad
 [1] "5" "6" "2" "7" "7" "4" "5" "7" "3" "7"
> w<-c(datos$Capacidad)
```

Se continúa con la función *max_flow(graph,source,target,capacity=NULL)*, la cual utiliza el algoritmo de etiquetado. Los argumentos de la función son:

Graph	La gráfica de entrada
Source	Nodo fuente
Target	Nodo destino
Capacity	Vector que contiene la capacidad de los arcos

```
> max_flow(g,"1","7",capacity=w)
$value
[1] 11

$flow
 [1] 5 6 0 5 0 0 5 5 1 6

$cut
[1] 1 2

$partition1
+ 1/7 vertex, named, from 42409fc:
[1] 1

$partition2
+ 6/7 vertices, named, from 42409fc:
[1] 2 3 4 5 6 7

$stats
$stats$nopush
[1] 6

$stats$norelabel
[1] 1

$stats$nogap
[1] 0
```

Figura 3.3.2 Resultado del problema de Flujo Máximo para la red R_2 por el software R

Nótese que, dado que los nodos deben tener valor numérico al ser introducidos a *R*, el nodo destino "1" corresponde al nodo s original, y el nodo destino "7", al nodo t.

De los resultados, *value* denota el valor del flujo máximo y el vector *flow* el valor de flujo en los arcos de acuerdo a la solución dada:

$s \to 1$	5	$3 \to 2$	1
$s \to 3$	6	$3 \to t$	5
$1 \to 2$	0	$4 \to 5$	5
$1 \to 4$	5	$5 \to 3$	0
$2 \to 5$	1	$5 \to t$	6

Cut denota los arcos del corte mínimo $(S, \bar{S})$, siendo estos los arcos 1 y 2, correspondientes a $(s, 1)$ y $(s, 3)$. *Partition1* indica los nodos pertenecientes a S del corte $(S, \bar{S})$, en este caso $s \in S$. Análogamente, *partition2* muestra los nodos de $\bar{S}$, siendo éstos $1,2,3,4,5$ y t.

Los últimos cinco datos obtenidos son algunas estadísticas del algoritmo push-relabel (algoritmo de reetiquetado). De esta solución se retoman los resultados anteriores; el flujo máximo es igual a 11 y el corte mínimo está dado por los arcos $(s, 1), (s, 3)$.

Por otro lado, Python es un lenguaje de programación creado a finales de los ochenta por Guido van Rossum en el Centro para las Matemáticas y la Informática (CWI, Países Bajos), que posee una licencia de código abierto y administrado por la *Python Software Foundation*.[29] A continuación, se presenta un código en este lenguaje para encontrar el flujo máximo y corte mínimo de la red R_2, haciendo uso de OR-Tools, el cual es un paquete de software de Google para C++, C#, Phyton y Java, enfocado a resolver problemas de optimización. (Ver figura 3.3.3)

Se invocan las funciones de los respectivos paquetes, y se define la red mediante tres arreglos, para los nodos de inicio, los nodos finales y para las capacidades respectivamente. Para cada arco $i \in A$, éste va de start_nodes[i] a end_nodes[i], con capacidad dada por capacities[i]. Posteriormente se usa el SimpleMaxFlow solver, el cual será la clase asignada al objeto max_flow. Para cada nodo inicial y final, se crea un arco con la capacidad dada, usando el método (función) AddArcWithCapacity() que ya está definido dentro de la clase. Con los arcos definidos, se invoca al método Solve(), suministrando un nodo fuente (0) y un nodo destino (6), mostrando así el flujo en cada arco.

[29] Python. (s.f). En *Wikipedia*. Recuperado el 15 de septiembre de 2019 de https://es.wikipedia.org/wiki/Python

```
>>> from ortools.graph import pywrapgraph
>>> from __future__ import print_function
def main():
    start_nodes=[0,0,1,1,2,3,3,4,5,5]
    end_nodes=[1,3,2,4,5,2,6,5,3,6]
    capacities=[5,6,2,7,7,4,5,7,3,7]
    max_flow=pywrapgraph.SimpleMaxFlow()
    for i in range(0,len(start_nodes)):
            max_flow.AddArcWithCapacity(start_nodes[i], end_nodes[i],capacities[i])
    if max_flow.Solve(0,6)==max_flow.OPTIMAL:
            print('Flujo Máximo:', max_flow.OptimalFlow())
            print('')
            print(' Arco     Flujo/Capacidad')
            for i in range (max_flow.NumArcs()):
                    print('%ls -> %ls    %3s  /  %3s' % (
                            max_flow.Tail(i),
                            max_flow.Head(i),
                            max_flow.Flow(i),
                            max_flow.Capacity(i)))
            print('Vértices de lado de la fuente en el min-cut:', max_flow.GetSourceSideMinCut())
            print('Vértices de lado del destino en el min-cut:', max_flow.GetSinkSideMinCut())
    else:
            print('Hubo un problema con la introducción de la red para flujo máximo.')
```

Figura 3.3.3 Código en Python para calcular el flujo máximo y corte mínimo para la red R_2

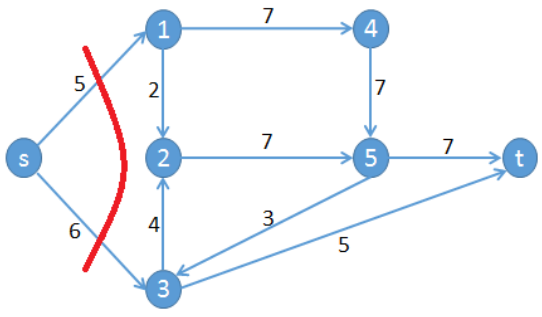
Así, se obtiene:

```
Flujo Máximo: 11

 Arco     Flujo/Capacidad
0 -> 1      5  /     5
0 -> 3      6  /     6
1 -> 2      2  /     2
1 -> 4      3  /     7
2 -> 5      3  /     7
3 -> 2      1  /     4
3 -> 6      5  /     5
4 -> 5      3  /     7
5 -> 3      0  /     3
5 -> 6      6  /     7
Vértices de lado de la fuente en el min-cut: [0]
Vértices de lado del destino en el min-cut: [6, 5, 2, 4, 3, 1]
```

Figura 3.3.4 Resultado del problema de Flujo Máximo para la red R_2 por el software Python

Obteniendo nuevamente un flujo máximo de valor 11, y un corte mínimo $(S, \bar{S}) = \{(s,1),(s,2)\}$.



3.3.1 Software de práctica en textos universitarios

Algunos textos universitarios de investigación de operaciones ofrecen algún software de práctica para el estudiante. Un caso es el software libre QM para Windows v.4[30], del desarrollador Prentice-Hall Inc, orientado a usarse con el texto *Introduction to Management Science* de Bernard W. Taylor, el cual proporciona análisis matemáticos para actividades como la Gestión de Operaciones, Métodos Cuantitativos, o Ciencia de la Administración. Cuenta con métodos de cálculo para PERT/CPM (*Project Evaluation and Review Techniques/Critical Path Method*), programación lineal, análisis de decisiones, problema de transporte, funciones estadísticas, teoría de juegos, etc. En este paquete, se selecciona el módulo "networks", seguido de "maximal flow". En este caso, el límite de nodos a ingresar en QM es de 101, con a lo más 100 arcos en la red. La solución del flujo máximo para la red R_2 se muestra enseguida.

Source	Sink
1	7

R2

Branch name	Start node	End node	Capacity	Reverse capacity
Branch 1	1	2	5	0
Branch 2	1	4	6	0
Branch 3	2	3	2	0
Branch 4	2	5	7	0
Branch 5	3	6	7	0
Branch 6	4	3	4	0
Branch 7	4	7	5	0
Branch 8	5	6	7	0
Branch 9	6	4	3	0
Branch 10	6	7	7	0

Figura 3.3.1.1 Entrada de datos para la red R_2 en QM

Iteration	Path	Flow	Cumulative Flow
1	1-> 2-> 5-...	5	5
2	1-> 4-> 7	5	10
3	1-> 4-> 3-...	1	11

[30] QM (s.f). En *Prenhall.* Recuperado el 10 de Octubre de 2019 de https://wps.prenhall.com/bp_weiss_software_1

R2 Solution					
Branch name	Start node	End node	Capacity	Reverse capacity	Flow
Maximal Network Flow	11				
Branch 1	1	2	5	0	5
Branch 2	1	4	6	0	6
Branch 3	2	3	2	0	0
Branch 4	2	5	7	0	5
Branch 5	3	6	7	0	1
Branch 6	4	3	4	0	1
Branch 7	4	7	5	0	5
Branch 8	5	6	7	0	5
Branch 9	6	4	3	0	0
Branch 10	6	7	7	0	6

Figura 3.3.1.2 Resultado del problema de flujo máximo para la red R_2 proporcionado por QM

Como se ve en la figura 3.3.1.2, se obtiene nuevamente el valor 11 como el flujo máximo a enviar desde el nodo fuente al nodo destino.

También para Windows, TORA[31] es un software de práctica que tiene por objeto usarse con muchas de las técnicas presentadas en el libro Investigación de Operaciones de Hamdy A. Taha, publicado a través de la editorial Prentice Hall.

Una primera opción es resolver el problema de flujo máximo ingresando su programa lineal; seleccionando en el Menú el módulo “Linear Programming”. La versión estudiantil de TORA cuenta con un límite de 100 variables (flujo en los arcos) y 100 restricciones.

También es posible resolver el problema seleccionando en el Menú la sección “Network models” seguido de “Maximal Flow”. Igualmente, el límite de nodos a ingresar es 100. Nótese que en este caso, si es posible nombrar a los nodos como *s* y *t*. La solución del flujo máximo con TORA para la red R_2 se muestra enseguida.

		N1	N2	N3	N4	N5	N6	N7
	Node Name	s	1	2	3	4	5	t
N1	s		5.00	0.00	6.00	0.00	0.00	0.00
N2	1	0.00		2.00	0.00	7.00	0.00	0.00
N3	2	0.00	0.00		0.00	0.00	7.00	0.00
N4	3	0.00	0.00	4.00		0.00	0.00	5.00
N5	4	0.00	0.00	0.00	0.00		7.00	0.00
N6	5	0.00	0.00	0.00	3.00	0.00		7.00
N7	t	0.00	0.00	0.00	0.00	0.00	0.00	

*Figura 3.3.1.3 I*ngreso de datos de la red R_2 en la sección de “Maximal Flow”

[31] Investigación de Operaciones 1. (s.f). En *Investigación de Operaciones IND-331.* Recuperado el 10 de Octubre de 2019 de https://investigaciondeoperacionesind331.blogspot.com/p/programacion-entera.html

		Maximum flow in network = 11.00 (4 iterations)						
		N1	N2	N3	N4	N5	N6	N7
		s	1	2	3	4	5	t
N1	s		5.00	0.00	6.00	0.00	0.00	0.00
N2	1	0.00		0.00	0.00	5.00	0.00	0.00
N3	2	0.00	0.00		0.00	0.00	1.00	0.00
N4	3	0.00	0.00	1.00		0.00	0.00	5.00
N5	4	0.00	0.00	0.00	0.00		5.00	0.00
N6	5	0.00	0.00	0.00	0.00	0.00		6.00
N7	t	0.00	0.00	0.00	0.00	0.00	0.00	

Figura 3.3.1.4 Solución brindada por TORA para la red R_2, la cual muestra el flujo solución y el número de iteraciones.

Asimismo se cuenta con el software QSB (por sus siglas en inglés, Sistemas cuantitativos para empresas)[32], del cual su versión para Windows, WinQSB, se ha usado en varias ocasiones a lo largo de este texto. QSB fue creado y desarrollado por Yih-Long Chang y Kiran Desai, y licenciado por John Wiley & Sons, editorial conocida por la publicación de material científico e información técnica. WinQSB se fundamenta en la llamada administración científica y el uso de los métodos cuantitativos de la Investigación de Operaciones en la toma de decisiones empresariales. Una nota importante es que el software necesita ser corrido en Windows XP debido a que no ha sido actualizado a versiones más recientes.

Puede resolverse el problema de flujo máximo mediante el ingreso del programa lineal correspondiente en el apartado “Linear and Integer Programming”, la cual cuenta con un límite de 181 variables con 181 restricciones. Como se vio con más detalle en la sección 3.1, es posible escoger resolver el programa lineal por simplex clásico o por simplex con variables acotadas. La figuras 3.1.4 y 3.1.5 muestran la introducción del programa lineal de la red R_2 y la solución del mismo por simplex clásico, así como la figura 3.1.7 muestra solución alterna encontrada. Las figuras 3.1.11 y 3.1.12 muestran, para el caso de método simplex por variables acotadas, la introducción de datos del programa lineal así como la solución arrojada, respectivamente.

Igualmente es posible usar “Network Modeling”, y luego seleccionar el submenú “Maximal Flow Problem”, con el límite de nodos igual a 2,043. De la sección 3.1, las figuras 3.1.2 y 3.1.3 muestran la introducción de datos de la red R_2 y la solución por método simplex especializado en redes.

[32] WinQSB (s.f). En *Swmath.* Recuperado el 10 de Octubre de 2019 de http://www.swmath.org/software/6146

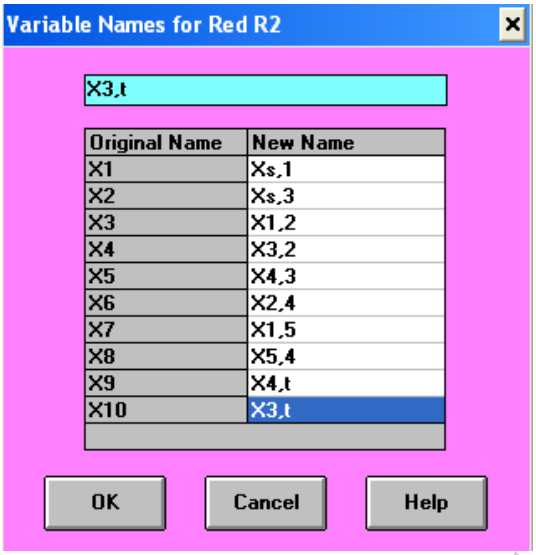
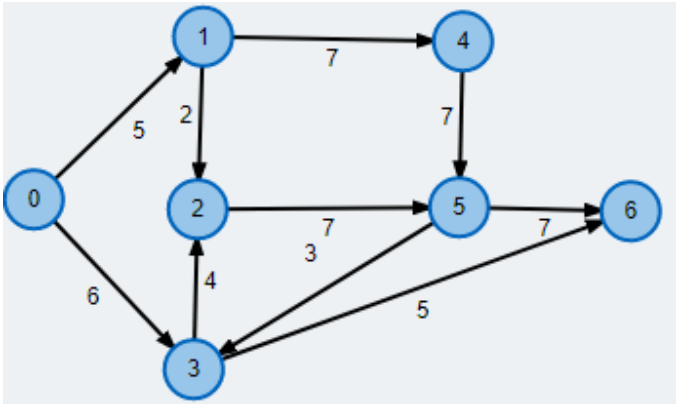


Figura 3.3.1.5 Edición de los nombres de las variables en WinQSB

3.4 Algoritmos en línea de libre acceso

En cuanto a la posibilidad de resolver problemas de flujo máximo online, el Departamento de Geometría Aplicada y Matemáticas Discretas de la Universidad Técnica de Munich, cuenta con la página *http://www-m9.ma.tum.de/Allgemeines*, siendo una de los algoritmo presentados el de Ford and Fulkerson.

En la pestaña “create a graph”, se edita la gráfica dada como ejemplo en la página. Para crear un nodo, se da doble clic en el área de dibujo, para crear un arco, se selecciona el nodo inicial y posteriormente el nodo final, se crea un arco con capacidad cero, la cual se modifica dando doble click en ella, para borrar un nodo o arco se presiona click derecho. Al correr el programa, se selecciona el nodo fuente y el nodo destino. Seleccionando “next”, pueden verse gráficamente los pasos del algoritmo de Ford and Fulkerson hasta llegar al flujo máximo.



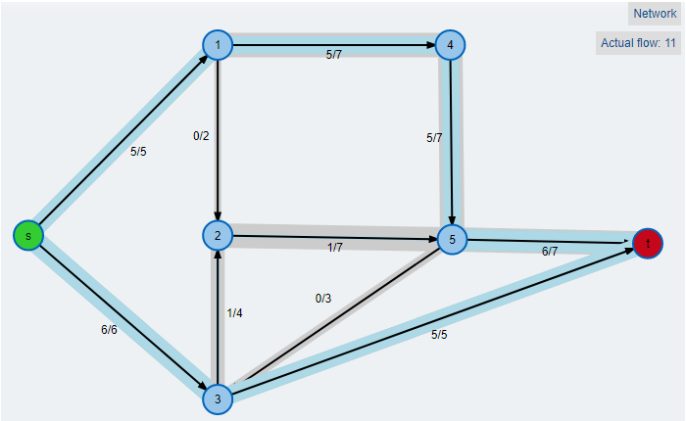
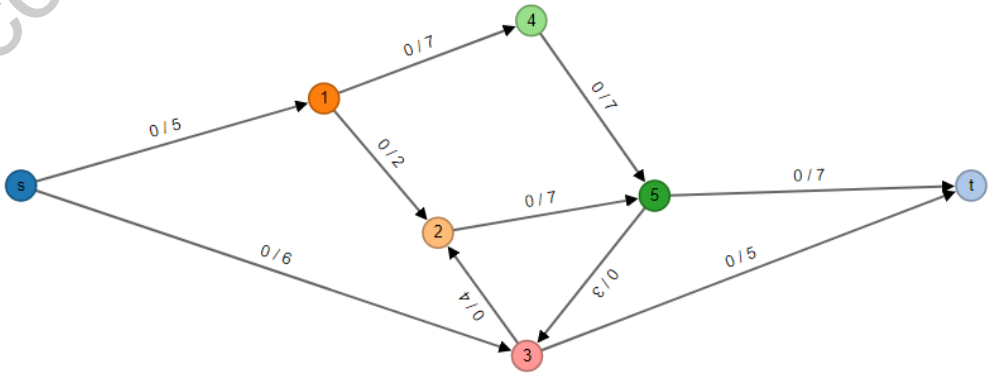


Figura 3.4.1 Solución brindada para la red R_2 por el software libre de *http://www-m9.ma.tum.de/Allgemeines* con flujo máximo igual a 11

También se cuenta con la página bl.ocks.org, el cual es un visor simple para compartir ejemplos de código, dirigido por Michael "Mike" Bostock, informático estadounidense y especialista en visualización de datos. La página que en este caso compete, es *https://bl.ocks.org/estk/9629395*, donde el usuario "Estk", ex estudiante de la Universidad de California en Santa Cruz, presenta la teoría básica de flujo en redes y con el algoritmo de Ford and Fulkerson se resuelve el problema de flujo máximo de una red a introducir.

Al ingresar en la página, se muestra una red por defecto, la cual puede modificarse o borrar en su totalidad. Click izquierdo en una parte vacía del área de dibujo crea un nodo, seleccionar un nodo y arrastrar el cursor hasta otro crea un arco, Alt+arrastar permite mover un nodo para controlar el diseño de la red, puede borrarse tanto un arco como un nodo seleccionándolos y presionando "delete". Al seleccionar un nodo, puede asignársele un nombre escribiéndolo en "New Node Name", del mismo modo, al seleccionar un arco, en "Nex Max capacity" se le asigna la capacidad. Posteriormente, "solve" proporciona la solución del problema de flujo máximo de la red, mientras que en "Show Steps" pueden verse las redes residuales y caminos aumentativos de cada paso del algoritmo.



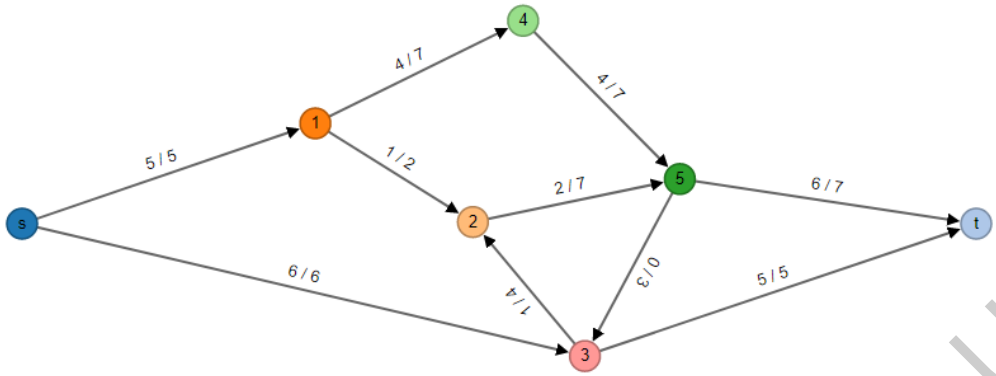


Figura 3.4.2 Red R_2 ingresada en *bl.ocks.org*, con solución de flujo máximo igual a 11 unidades

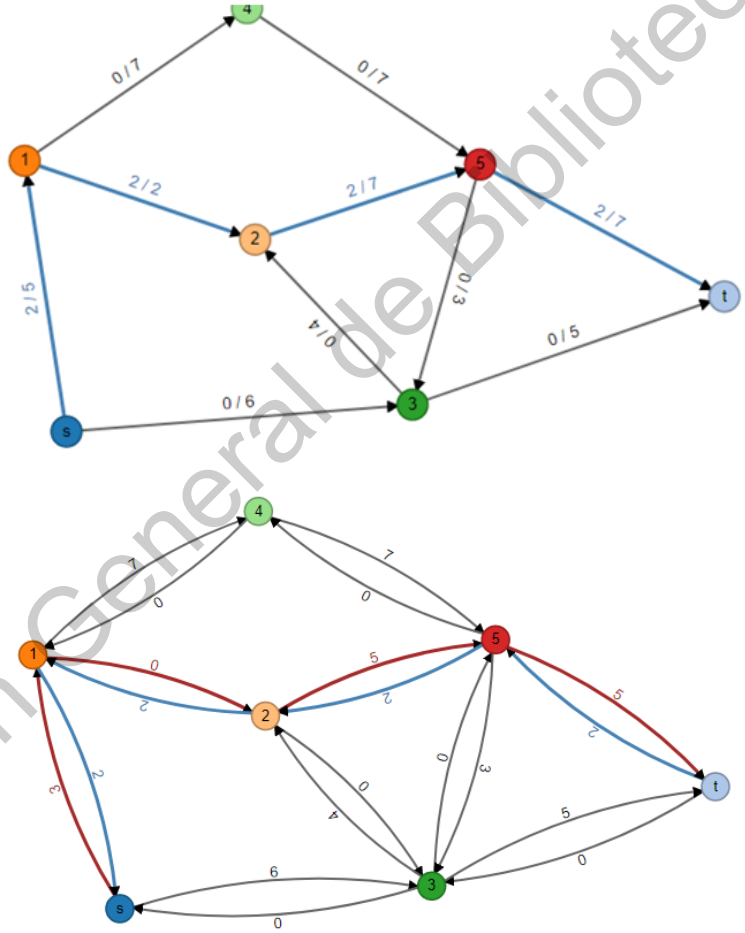


Figura 3.4.3 Primer camino aumentativo y red residual proporcionados por bl.ocks.org para la red R_2

Una visión breve de esta revisión de software disponible para resolver el problema de flujo máximo, se muestra en la siguiente tabla, con los distintos productos encontrados y sus características principales.

Tabla 3.4.1 Cuadro resumen de rutinas de software libre

Entorno/Lenguaje	**Creado por**	**Desarrollador**	**Capacidades**	
R	Departamento de Estadística de la Universidad de Auckland, Robert Gentleman y Ross Ihaka	R Development Core Team	-	-
Python	Centro para las Matemáticas y la Informática (CWI, Países Bajos), Guido van Rossum	Python Software Foundation	-	-
TORA	Investigación de Operaciones, Hamdy A. Taha	Prentice-Hall Inc	100 variables	100 restricciones
QM	Introduction to Management Science, Bernard W. Taylor	Prentice-Hall Inc	100 nodos	100 arcos
WinQSB v.2.0	Yih-Long Chang y Kiran Desai	John Wiley & Sons	**Linear Programming** 181 variables **Network Modeling** 100 nodos	 181 restricciones 100 arcos

La revisión de estos productos de software para resolver el problema de flujo máximo y de corte mínimo muestra que hay buena disponibilidad en diversas fuentes, y que es posible manejar problemas de tamaño pequeño y mediano en tiempos razonables de respuesta.

En el siguiente capítulo se examinarán aplicaciones del problema de flujo máximo, ilustrando sus soluciones con algunos de estos productos de software ya mencionados.

4. EXTENSIONES Y APLICACIONES

4.1 Modificación para múltiples fuentes y destinos

Considérese una red en la que haya más de un vértice de origen y más de un vértice destino. Para remediar esta situación y que sea posible aplicar el algoritmo de flujo máximo, que funciona con una sola fuente y destino, simplemente es necesario crear dos nuevos vértices S y T. Se une el vértice S a cada vértice fuente original $s_1, s_2, \dots, s_n$ por los arcos $(S, s_1), (S, s_2), \dots, (S, s_n)$ con capacidad infinita. Igualmente, se une cada vértice destino original $t_1, t_2, \dots, t_n$ al vértice T por un arco $(t_1, T), (t_2, T), \dots, (t_n, T)$ con capacidad infinita.

Cualquier flujo en la nueva red de S a T corresponde a un flujo en la red original de las fuentes originales a los destinos originales, y viceversa. Más aún, un flujo máximo de la gráfica extendida lo es igualmente en la gráfica original. Por tanto, el algoritmo de flujo máximo puede ser aplicado a la gráfica extendida, y el flujo máximo generado otorga un flujo máximo en la gráfica original.

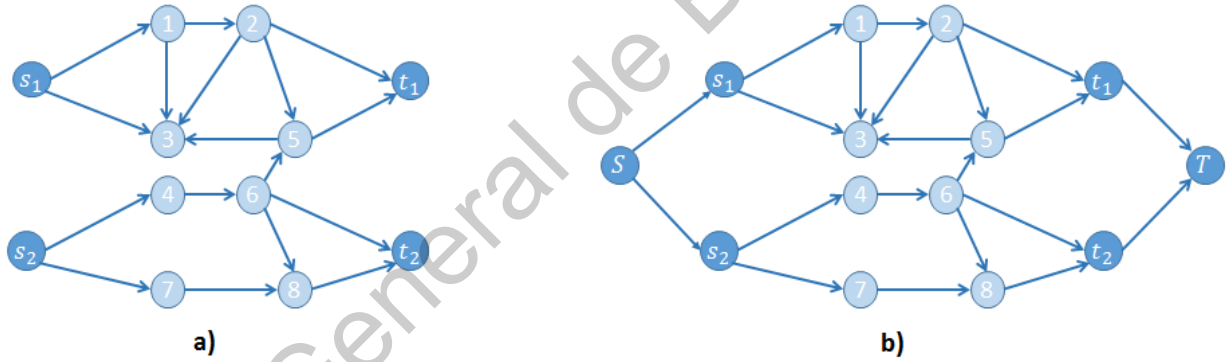


Figura 4.1.1 Gráfica con múltiples fuentes y destinos; **a)** gráfica original, **b)** gráfica extendida

Como ejemplo de lo anterior, se dará una aplicación real. Una cuestión de relevancia para el subsector del transporte de carga por ferrocarril, tanto para las empresas ferroviarias como para la SCT, es la identificación de cortes mínimos en la red ferroviaria al considerar los flujos de importación desde Estados Unidos que entran por la frontera norte hacia el centro del país.

El siguiente mapa ilustra el sistema ferroviario mexicano, siendo las líneas de interés Kansas City Southern de México (KCSM), Ferrocarril Mexicano (Ferromex) y la línea Coahuila-Durango.

Figura 4.1.2 Sistema ferroviario de México[33]

Las importaciones entran a México por medio de dos líneas ferroviarias que llegan a la frontera norte, Ferromex y KCSM, las cuales se conectan con vías ferroviarias estadounidenses.

Al representar el sistema como red, se hace evidente que no hay un nodo fuente definido, ya que, tomando los nodos de cruce en la frontera norte, se tendrían 8 puntos que son fuente de flujos: San Diego, Mexicali, Nogales, El Paso, Ojinaga, Piedras Negras, Nuevo Laredo y Matamoros.

Para tener un solo nodo fuente, que represente el flujo total de carga que entra a México, se considera un único nodo al que se conectan el total de 8 nodos de entrada de carga y que aparece en la gráfica como el nodo fuente 1, representado como USA en la tabla 4.1.1.

[33] AsoMexFFCC (2014). *Mapa del Sistema Ferroviario de México*. [Mapa] Recuperado de https://commons.wikimedia.org/wiki/File:130503_Mapa_Ferroviario.pdf

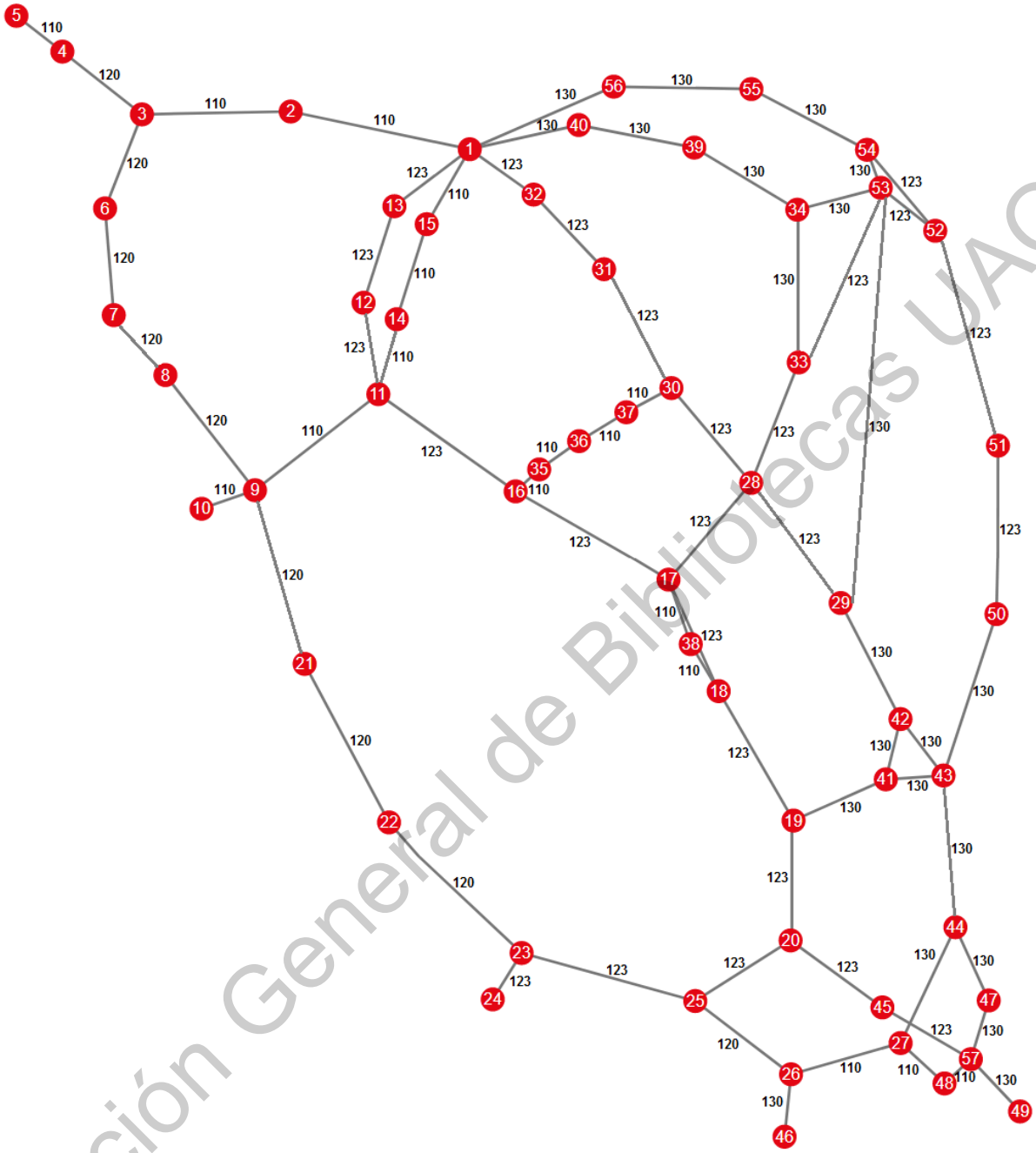


Figura 4.1.3 Representación en red del sistema ferroviario de México, líneas Ferromex, KCSM y Coahuila-Durango

La ciudad de México se elige como nodo destino (nodo 57). Las cantidades en los arcos representan las capacidades de vía, que es el tonelaje máximo que acepta la vía para un carro de ferrocarril de cuatro ejes que circule por ella[34]. Esto no es lo mismo que flujo en *toneladas/día*, sin embargo, no se cuenta con información real de los flujos que se miden de esta manera o en *trenes/día*, así que no puede

[34] SCT. www.sct.gob.mx

considerarse directamente como el “flujo máximo” al encontrado en este ejercicio como el tonelaje máximo que puede circular por la red desde la frontera norte hasta el centro. Es complicado conseguir datos reales de la circulación de redes de carga, pues las empresas ferroviarias consideran estos datos como confidenciales.

Más bien, la *capacidad de vía* es una variable *proxy* (variable aproximada a las variables objeto de análisis), que sirve para dar idea de cómo ocurre el flujo real ferroviario en toneladas/día, ya que mientras más grande es la capacidad de la vía, mayores flujos circulan sobre ella. Lo útil de utilizar esta variable proxy es que permite identificar los cortes mínimos en la red.

La siguiente tabla muestra las estaciones, representadas por nodos.

Tabla 4.1.1 Nodos de la Red Ferroviaria

1. USA	20. Irapuato	39. Nuevo Laredo
2. Mexicali	21. Culiacán	40. Laredo
3. Benjamín Hill	22. Tepic	41. Salinas
4. Nogales	23. Guadalajara	42. Laguna Seca
5. Nacozari	24. Manzanilla	43. San Luis Potosí
6. Hermosillo	25. Pénjamo	44. Ing. Buchanan López
7. Guaymas	26. Ajuno	45. La Griega
8. Cd. Obregón	27. Acámbaro	46. Lázaro Cárdenas
9. Sufragio	28. Paredón	47. San Nicolás
10. Topolobampo	29. R. Arizpe	48. Toluca
11. Chihuahua	30. Monclova	49. Veracruz
12. Cd. Juárez	31. Piedras Negras	50. Tampico
13. El Paso	32. Eagle Pass	51. Altamira
14. Ojinanga	33. Chipinque	52. P.C. Morales
15. Presidio	34. Salinas Victoria	53. Monterrey
16. Escalón	35. Hércules	54. Lobos
17. Torreón	36. Avante Ahmsa	55. Matamoros
18. Felipe Pescador	37. El Rey	56. Brownsville
19. Chicalote	38. Durango	57. C.D.M.X.

El sistema ferroviario comunica la mayor parte del territorio nacional, conecta a las poblaciones más importantes, así como los principales puertos y fronteras, de allí la importancia de analizar la red a fin de determinar el flujo máximo posible a través de

ella, así como la búsqueda de un corte que impediría la conexión entre el nodo fuente (USA) y el nodo destino (Cd. De México). Este análisis puede hacerse de diferentes formas, por ejemplo, planteando el programa lineal consistente en 63 arcos, es decir, 63 variables, y 46 restricciones. El programa lineal puede encontrarse en este texto en el Anexo 1.

Se procede a resolver el problema de flujo máximo para la red del sistema ferroviario en WinQSB:

	Decision Variable	Solution Value	Unit Cost or Profit c(j)	Total Contribution	Reduced Cost	Basis Status	Allowable Min. c(j)	Allowable Max. c(j)
1	X1,2	110.0000	1.0000	110.0000	0	basic	1.0000	M
2	X2,3	110.0000	0	0	0	basic	0	M
3	X3,6	110.0000	0	0	0	basic	0	M
4	X6,7	110.0000	0	0	0	basic	0	M
5	X7,8	110.0000	0	0	0	basic	0	M
6	X8,9	110.0000	0	0	0	basic	0	M
7	X11,9	10.0000	0	0	0	basic	0	0
8	X9,21	120.0000	0	0	0	basic	0	M
9	X21,22	120.0000	0	0	0	basic	0	M
10	X22,23	120.0000	0	0	0	basic	0	M
...	...	...	...	...	...	...	...	...
57	X51,50	123.0000	0	0	0	basic	0	M
58	X50,43	123.0000	0	0	0	basic	0	M
59	X42,43	7.0000	0	0	0	basic	0	0
60	X43,41	0	0	0	0	basic	0	0
61	X43,44	130.0000	0	0	0	basic	-1.0000	0
62	X42,41	0	0	0	0	at bound	-M	0
63	X41,19	0	0	0	0	at bound	-M	0
	Objective	Function	(Max.) =	363.0000	(Note:	Alternate	Solution	Exists!!)

Figura 4.1.4 Solución brindada por WinQSB para la red del sistema ferroviario

De la solución brindada, se observa que el flujo máximo es de 363 toneladas. Ahora bien, resolviendo el problema dual en WinQSB, las variables duales que corresponden a las restricciones de los arcos de la red son a partir de la no. 47, de donde el corte mínimo está dado por las variables duales positivas a partir de ésta, es decir, UB_X20,45, UB_X27,48 y UB_X44_47. Nótese que se reporta la existencia de una solución alterna.

	Decision Variable	Solution Value	Unit Cost or Profit c(j)	Total Contribution	Reduced Cost	Basis Status	Allowable Min. c(j)	Allowable Max. c(j)
1	C1	1.0000	0	0	0	basic	0	110.0000
2	C2	1.0000	0	0	0	basic	0	110.0000
3	C3	1.0000	0	0	0	basic	0	110.0000
4	C4	1.0000	0	0	0	basic	0	110.0000
5	C5	1.0000	0	0	0	basic	0	0
6	C6	1.0000	0	0	0	basic	0	0
7	C7	1.0000	0	0	0	basic	0	0
8	C8	1.0000	0	0	0	basic	0	0
9	C9	1.0000	0	0	0	basic	0	0
10	C10	1.0000	0	0	0	basic	0	0
...	...	...	...	...	...	...	...	...
46	C46	1.0000	0	0	0	basic	0	0
47	UB_X1,2	0	110.0000	0	110.0000	at bound	0	M
48	UB_X2,3	0	110.0000	0	110.0000	at bound	0	M
49	UB_X3,6	0	120.0000	0	120.0000	at bound	0	M
50	UB_X6,7	0	120.0000	0	120.0000	at bound	0	M
51	UB_X7,8	0	120.0000	0	120.0000	at bound	0	M
52	UB_X8,9	0	120.0000	0	120.0000	at bound	0	M
53	UB_X11,9	0	110.0000	0	0	basic	110.0000	120.0000
54	UB_X9,21	0	120.0000	0	10.0000	at bound	110.0000	M
55	UB_X21,22	0	120.0000	0	10.0000	at bound	110.0000	M
56	UB_X22,23	0	120.0000	0	10.0000	at bound	110.0000	M
57	UB_X23,25	0	123.0000	0	13.0000	at bound	110.0000	M
58	UB_X25,20	0	123.0000	0	123.0000	at bound	0	M
59	UB_X25,26	0	123.0000	0	13.0000	at bound	110.0000	M
60	UB_X26,27	0	110.0000	0	0	basic	110.0000	110.0000
61	UB_X19,20	0	123.0000	0	0	at bound	123.0000	M
62	UB_X20,45	1.0000	123.0000	123.0000	0	basic	0	123.0000
63	UB_X45,57	0	123.0000	0	0	at bound	123.0000	M
64	UB_X44,27	0	130.0000	0	130.0000	at bound	0	M
65	UB_X27,48	1.0000	110.0000	110.0000	0	basic	110.0000	110.0000
66	UB_X48,57	0	110.0000	0	0	at bound	110.0000	M
67	UB_X44,47	1.0000	130.0000	130.0000	0	basic	123.0000	130.0000
...	...	...	...	...	...	...	...	...
107	UB_X43,44	0	130.0000	0	0	at bound	130.0000	M
108	UB_X42,41	0	130.0000	0	130.0000	at bound	0	M
109	UB_X41,19	0	130.0000	0	130.0000	at bound	0	M
	Objective	Function	(Min.) =	363.0000	(Note:	Alternate	Solution	Exists!!)

Figura 4.1.5 Solución brindada por WinQSB para el problema dual de la red del sistema ferroviario

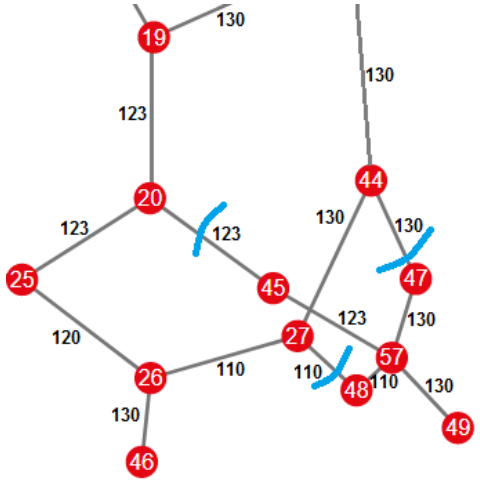


Figura 4.1.6 Corte mínimo de la red del sistema ferroviario

La suma de las capacidades de este corte es:

$$123 + 110 + 130 = 363$$

que es efectivamente igual al valor del flujo máximo encontrado.

El corte corresponde a interrumpir el flujo entre las estaciones de Irapuato (Gto.) y La Griega (El Marqués, Qro.), Acámbaro (Gto.) y Toluca, y entre Ing. Buchanam López (San Luis de la Paz, Gto.) y San Nicolás (Qro.).

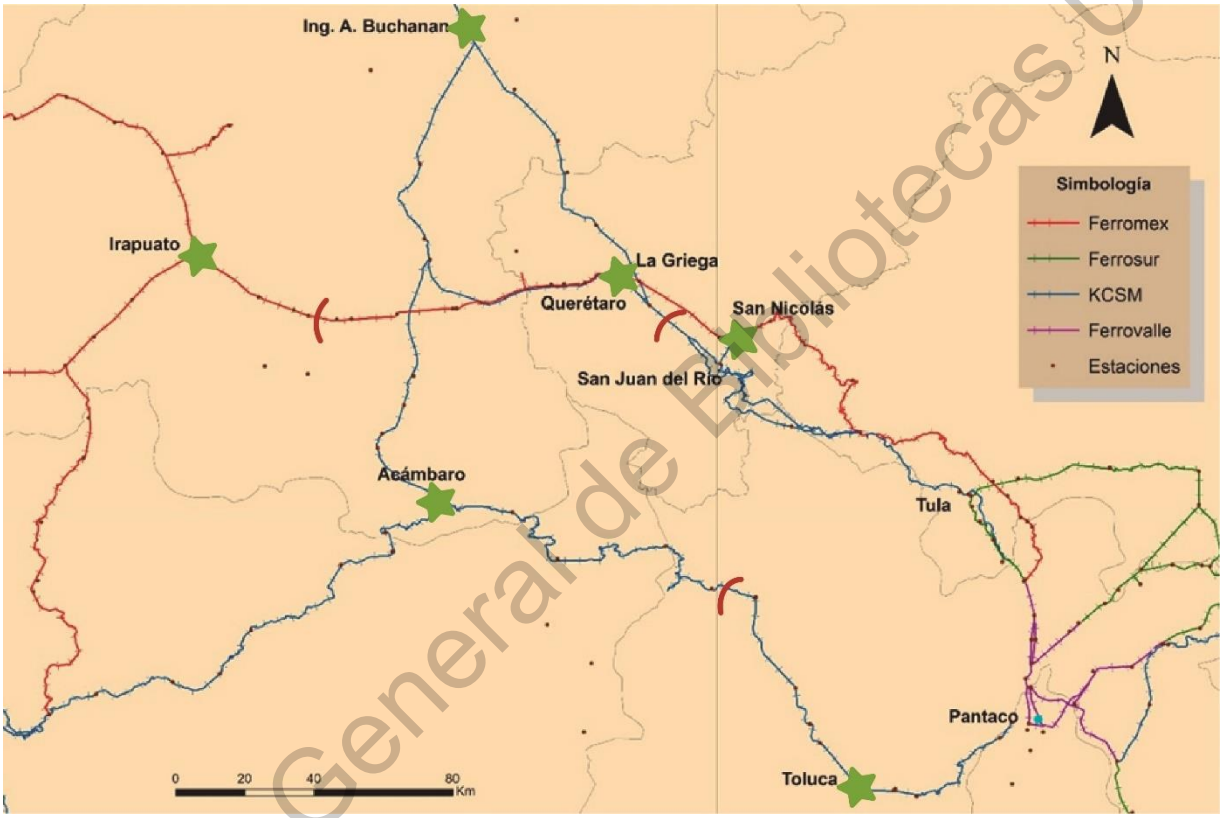


Figura 4.1.7 Corte mínimo mostrado en el mapa ferroviario[35]

Como se describió en el capítulo 3, el software estadístico R, a partir de los datos aportados en una tabla de Excel, también ofrece una función para resolver el problema.

[35]Mapa ferroviario. Elaboración propia del IMT por la M. en Geo. Gabriela García. 2019.

```
> w<-c(data$weight)
> max_flow(g,"1","57",capacity = w)
$value
[1] 363

$flow
 [1]  10 100   0   0 123 130  10  10
 [9]  10  10  10   0 110 100   0 100
[17] 100   0   0   0   0   0   0  13
[25]   0   0  13   0   0   0 123   0
[33]   0   0 123 110 110 110   0 110
[41] 110   0 110   0   0   0   0   0
[49] 130   0   0   0   0   0   0   0
[57]   0   0 123   0   0   0   0   0
[65]   0   0   0 123 123 123   0   0
[73] 123   7   0   0 130   0 130 123
[81] 130 110   0 123 123 123 130   0
[89]   0   7   0  14 116 130 130

$cut
[1] 80 81 82

$partition1
+ 50/52 vertices, named, from c31926d:
 [1] 1  2  3  6  7  8  9  11 12 13 14
[12] 15 16 17 18 19 20 21 22 23 25 26
[23] 27 28 29 30 31 32 33 34 35 36 37
[34] 38 39 40 41 42 43 44 45 47 48 50
[45] 51 52 53 54 55 56

$partition2
+ 2/52 vertices, named, from c31926d:
[1] 49 57

$stats
$stats$nopush
[1] 145

$stats$norelabel
[1] 75

$stats$nogap
[1] 10

$stats$nogapnodes
[1] 26

$stats$nobfs
[1] 2


> min_cut(g,"1","57",capacity=w)
[1] 363
```

Figura 4.1.8 Solución brindada por R para la red ferroviaria

De donde se corrobora el valor de flujo máximo encontrado (363), aunque hay una diferencia en el corte mínimo, siendo que en este caso el corte reportado proviene de

las restricciones 80, 81 y 82, correspondientes los arcos $x_{45,57}$, $x_{47,57}$ y $x_{48,57}$. La capacidad de este corte es:

$$110 + 123 + 130 = 363$$

Concordando con la solución ya vista. El nuevo corte corresponde a impedir el flujo entre La Griega (El Marqués, Querétaro) y CDMX, San Nicolás (Qro.) y CDMX, y entre Toluca y CDMX.

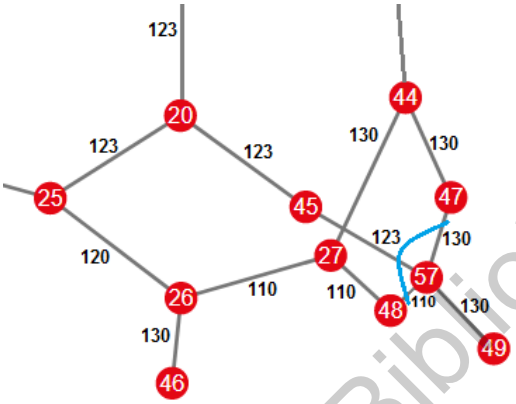


Figura 4.1.9 Corte mínimo alternativo de la red ferroviaria

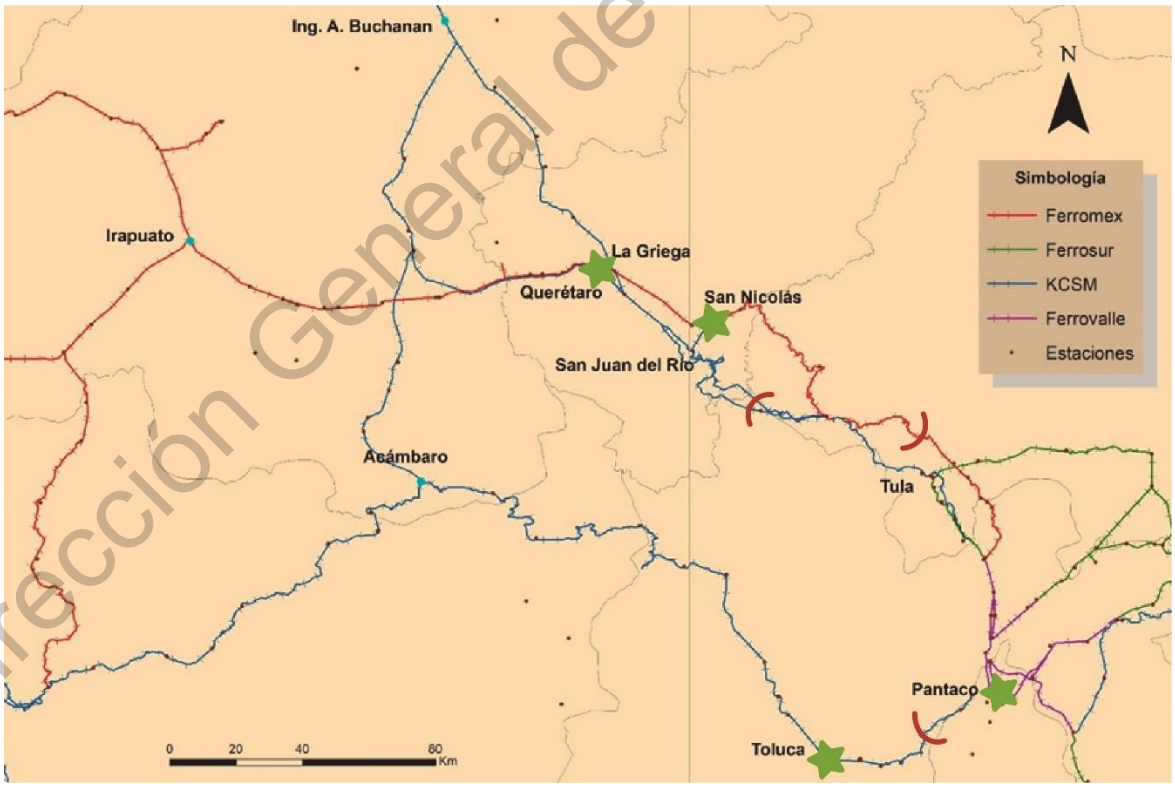


Figura 4.1.10 Corte mínimo alternativo en mapa ferroviario[36]

[36] Mapa ferroviario. Elaboración propia del IMT por la M. en Geo. Gabriela García. 2019.

4.2 Modificación para capacidades en los nodos, y para cotas inferiores en los arcos

En algunas situaciones, los nodos de una red pueden tener capacidades asociadas. Una transformación que permite el uso del algoritmo de flujo máximo es separar el nodo en dos vértices unidos por una arista cuya capacidad será la capacidad del nodo original. Nótese que las aristas que entran al vértice k, lo hacen en el vértice $k´$, mientras que las que salen de k, lo hacen de $k´´$.

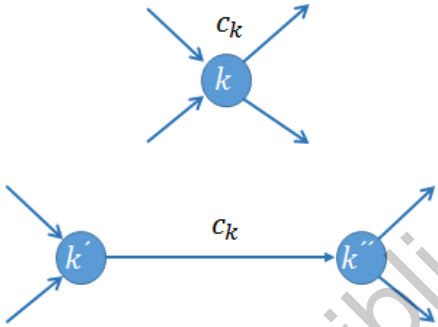


Figura 4.2.1 Transformación para nodos con capacidades

Ejemplo 4.2.1-

Se tienen cuatro plantas de producción cuyas mercancías deben llevarse a uno de los dos almacenes, cada arco de una planta a un almacén cuenta con una capacidad. Asimismo, el primer depósito tiene un límite de 1,200 de almacenamiento, y el segundo un límite de 1,500. Se genera un nodo fuente con un arco con capacidad suficientemente grande a cada una de las plantas, se aplica la transformación antes explicada, creando así los nodos “Alma1B” y “Alma2B”, y se agregan arcos con capacidad suficientemente grande de estos nuevos nodos a un nodo destino. Se obtiene así la red $R´_3$ de la cuál puede calcularse su flujo máximo.

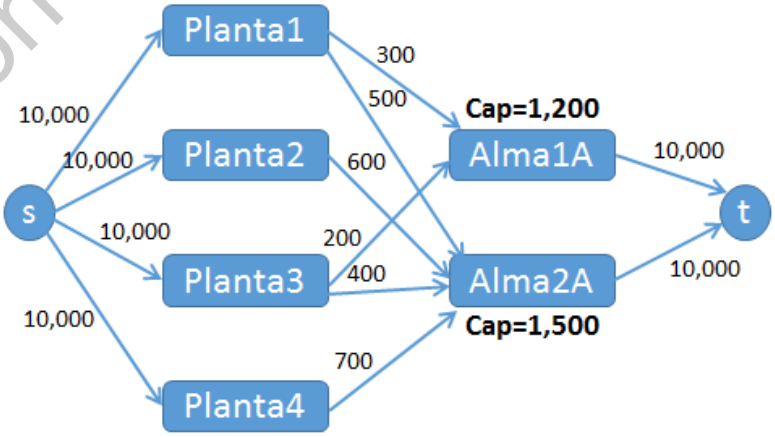


Figura 4.2.2 Red R_3

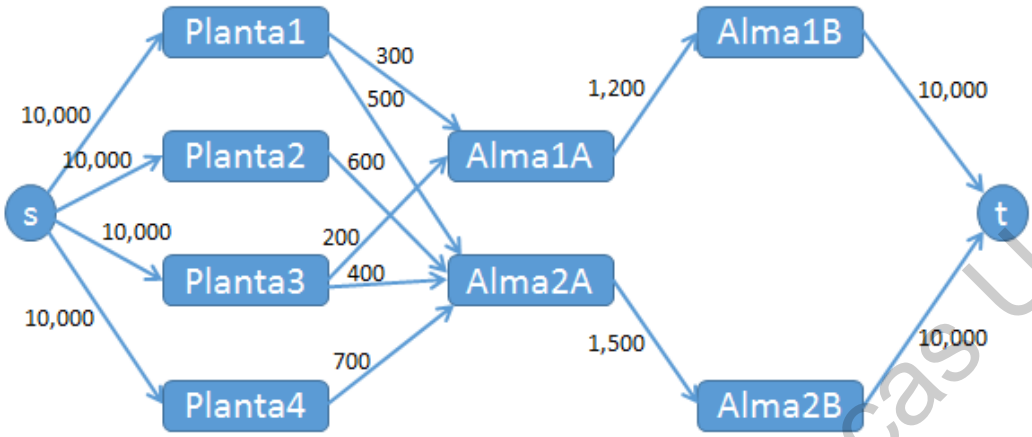


Figura 4.2.3 Red R'_3

10-10-2019	From	To	Net Flow		From	To	Net Flow
1	S	Planta1	300	8	Planta3	Alma2A	400
2	S	Planta2	400	9	Planta4	Alma2A	700
3	S	Planta3	600	10	Alma1A	Alma1B	500
4	S	Planta4	700	11	Alma2A	ALma2B	1500
5	Planta1	Alma1A	300	12	Alma1B	T	500
6	Planta2	Alma2A	400	13	ALma2B	T	1500
7	Planta3	Alma1A	200				
Total	Net Flow	From	S	To	T	=	2000

Figura 4.2.4 Solución brindada por "Network Modeling" de WinQSB para el problema de flujo máximo de la red R'_3

La solución encontrada es un flujo máximo igual a 2,000.

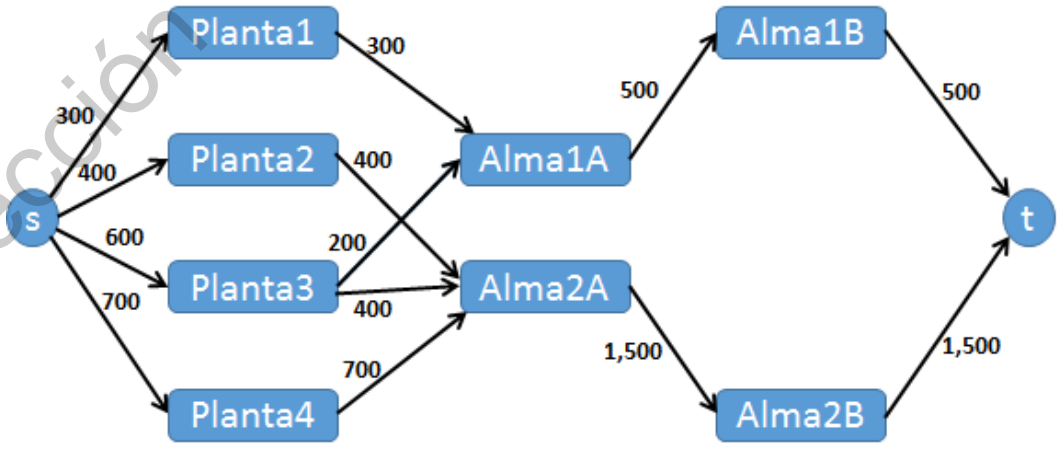


Figura 4.2.5 Flujo máximo para la red R'_3

Cuando hay cotas inferiores al flujo en los arcos de la red, se requiere que al menos l_{ij} unidades deben fluir del nodo i al nodo j. La primera complicación surge al desconocer si existe un flujo factible que cumpla la restricción, condición importante ya que el algoritmo de flujo máximo requiere de un flujo factible para su realización. Para proseguir en este caso, es necesario crear dos nodos adicionales, u y w, así como un arco del nodo destino al nodo fuente con capacidad infinita. Si se tiene una arista $(i,j) \in A$ tal que $l_{ij} > 0$, se adiciona un arco (u,j) con capacidad l_{ij}, y un arco (i,w) con capacidad igual a la suma de las cotas inferiores de los arcos (i,k) con $l_{ik} > 0$. Se reemplaza la capacidad original $c(i,j)$ por $c_{ij} - l_{ij}$. Esta nueva red tiene todas las cotas inferiores de los arcos igual a cero, así que el algoritmo de flujo máximo puede ser aplicado buscando el flujo máximo de u a w.

Ejemplo 4.2.2.-

Tómese la red R_4 de la figura 4.2.6. Para el flujo inicial, considérese que existen dos arcos con cota inferior, $(2,3)$ con cota igual a 2 y $(2,4)$ con cota 1, de donde deben llegar al menos 3 unidades de flujo al nodo 2. Del nodo 3 se deriva el arco $(3,4)$ con capacidad 3, ya que 4 es el nodo destino y el arco $(1,2)$ no tiene cota inferior y su capacidad es 6, un flujo igual a 3 es factible. De la red se tiene $l_{24} = 1$ y $l_{23} = 2$, se crean los nodos u y w. Luego, de acuerdo con lo anterior se crean las aristas $(u,4)$ y $(2,w)$ con capacidad 1, y las aristas $(u,3)$ y $(2,w)$ con capacidad 2, de donde la arista $(2,w)$ es en realidad de capacidad 3. La capacidad del arco $(2,3)$ pasa a ser $c_{23} - l_{23} = 5 - 2 = 3$ y la del arco $(2,4)$ se vuelve $c_{24} - l_{24} = 5 - 1 = 4$.

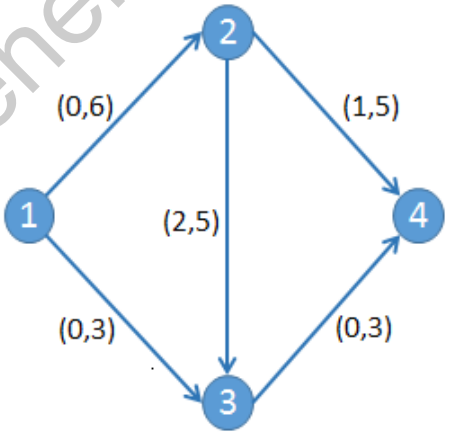


Figura 4.2.6 Red R_4 donde cada arista tiene una cota mínima y una cota máxima (capacidad)[37]

[37] Evans, J.R. & Minieka, E., 1992. Fig. 6.16. Figura. Optimization Algorithms for Networks and Graphs, 2da edición. CRC Press.

La figura 4.2.7 muestra un flujo inicial factible de valor 3 para la red R_4, de manera que pueda aplicarse el algoritmo de Flujo Máximo-Corte mínimo.

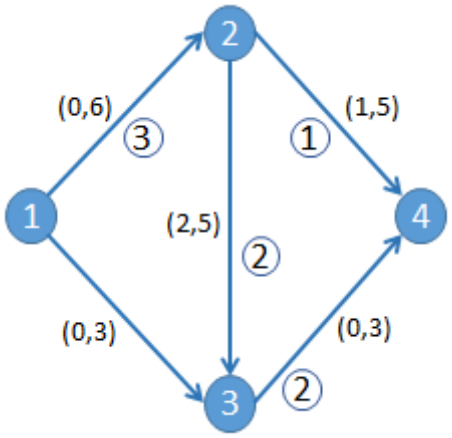


Figura 4.2.7 Flujo inicial factible para la red R_4[38]

Aplicando la transformación antes explicada, se obtiene la red extendida R'_4 mostrada en la figura 4.2.8, con flujo igual a 3 unidades.

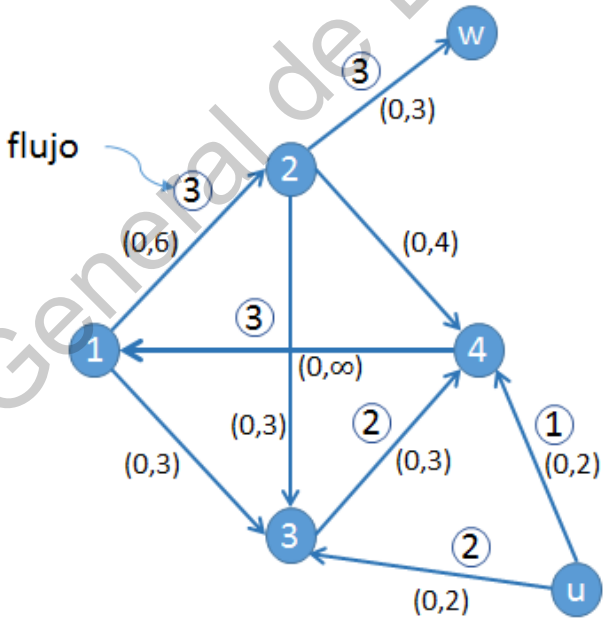


Figura 4.2.8 Red extendida R'_4 correspondiente a la red R_4[39]

[38] Evans, J.R. & Minieka, E., 1992. Fig. 6.18. Figura. Optimization Algorithms for Networks and Graphs, 2da edición. CRC Press.
[39] Evans, J.R. & Minieka, E., 1992. Fig. 6.17. Figura. Optimization Algorithms for Networks and Graphs, 2da edición. CRC Press.

En la práctica la capacidad infinita del arco $(4,1)$ es cambiado por una capacidad suficientemente grande. En este caso, al introducir los datos de la red en WinQSB se establece 50 como la capacidad del arco $(4,1)$.

10-11-2019	From	To	Net Flow		From	To	Net Flow
1	NodeU	Node3	1	4	Node2	NodeW	3
2	NodeU	Node4	2	5	Node3	Node4	1
3	Node1	Node2	3	6	Node4	Node1	3
Total	Net Flow	From	NodeU	To	NodeW	=	3

Figura 4.2.9 Resultado arrojado por WinQSB en su módulo “Network Modeling”

Se encuentra que el flujo máximo posible para la red es de 3 unidades.

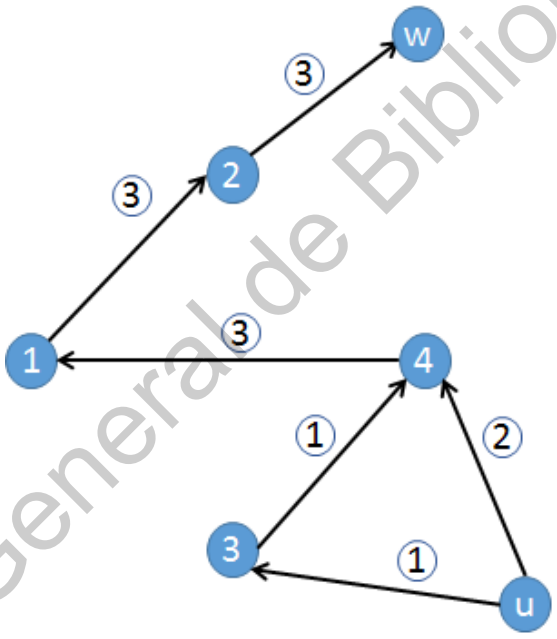


Figura 4.2.10 Solución gráfica para el problema de flujo máximo para la red R'_4

4.3 Flujo Máximo a Costo Mínimo

El problema de *flujo a costo mínimo*, al igual que el problema de flujo máximo, toma en cuenta una red con capacidades en los arcos. Adicionalmente, considera un costo de flujo a través de un arco. Para este modelo, es necesario que al menos uno de los nodos sea un nodo fuente, teniendo así nodos que *ofertan*, y que exista al menos un nodo de *demanda*, siendo el resto *nodos de transbordo*, siendo el objetivo **minimizar**

el costo total de enviar el suministro disponible a través de la red para satisfacer la demanda dada.

El problema de flujo a costo mínimo cuenta con importantes aplicaciones como en la red de distribución de una compañía, operaciones de una red de suministros, coordinación de mezclas de productos en plantas o la administración de flujo de efectivo.

La formulación del programa lineal de este problema es

$$Min\ Z\ = \sum_{(i,j)\in A} c_{ij} f_{ij}$$

Sujeta a

$$\sum_{j=1}^{m} f_{ij} - \sum_{k=1}^{m} f_{ki} = b_i$$

$$\sum b_i = 0; \quad f_{ij} \leq q_{ij} \quad i,j = 1,2,\dots,m$$

La primera suma de las restricciones indica el flujo neto generado en el nodo i, es decir, el flujo que sale del nodo menos el flujo que entra. La segunda suma indica que el flujo total generado por los nodos que ofertan es igual al flujo total absorbido por los nodos con demanda.

Del valor de b_i depende la naturaleza del nodo i;

$$b_i > 0 \qquad \text{si } i \text{ es un nodo fuente}$$

$$b_i < 0 \qquad \text{si } i \text{ es un nodo demanda}$$

$$b_i = 0 \qquad \text{si } i \text{ es un nododе trasbordo}$$

Para el caso del problema de flujo máximo, la red ya cuenta con un nodo de oferta (nodo fuente s), un nodo de demanda (nodo destino t) y el resto nodos de trasbordo, al igual que arcos con capacidades. Para que este problema quede en formato del problema de costo mínimo, se establece $c_{ij} = 0$ para todos los arcos de la red, $b_i = 0$ para los nodos de trasbordo, se escoge una cantidad $\bar{F}$ que sea una cota superior del flujo factible máximo a través de la red y se define $b_s = \bar{F}$ y $b_t = \bar{F}$.

Ejemplo 4.3.1.-

Se cuenta con la siguiente red R_5, con capacidades y costos en los arcos, así como el nodo fuente 1 y el nodo destino t.

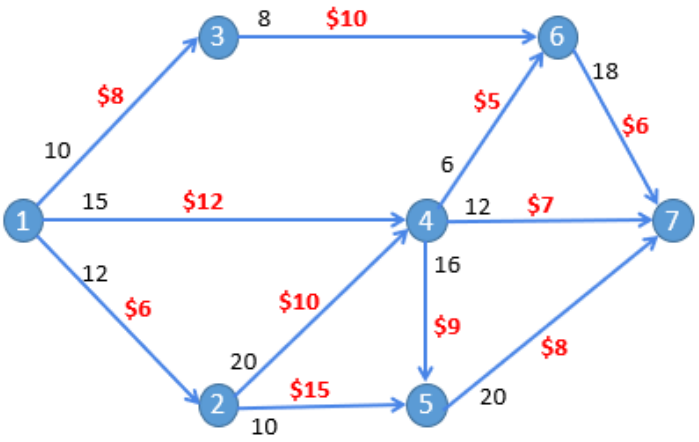


Figura 4.3.1 Red R_5 para el problema de flujo máximo a costo mínimo

Se resuelve el problema puro de flujo máximo de la red R_5, es decir, sin considerar los costos de los arcos.

	Decision Variable	Solution Value	Unit Cost or Profit c(j)	Total Contribution	Reduced Cost	Basis Status	Allowable Min. c(j)	Allowable Max. c(j)
1	F	35.0000	1.0000	35.0000	0	basic	0	M
2	X12	12.0000	0	0	0	basic	-1.0000	M
3	X13	8.0000	0	0	0	basic	-1.0000	M
4	X14	15.0000	0	0	0	basic	-1.0000	M
5	X24	7.0000	0	0	0	basic	0	1.0000
6	X25	5.0000	0	0	0	basic	-1.0000	0
7	X36	8.0000	0	0	0	basic	-1.0000	M
8	X45	16.0000	0	0	0	basic	0	M
9	X46	6.0000	0	0	0	basic	0	M
10	X47	5.0000	0	0	0	basic	-1.0000	0
11	X57	16.0000	0	0	0	basic	0	M
12	X67	14.0000	0	0	0	basic	0	M
	Objective	Function	(Max.) =	35.0000	(Note:	Alternate	Solution	Exists!!)

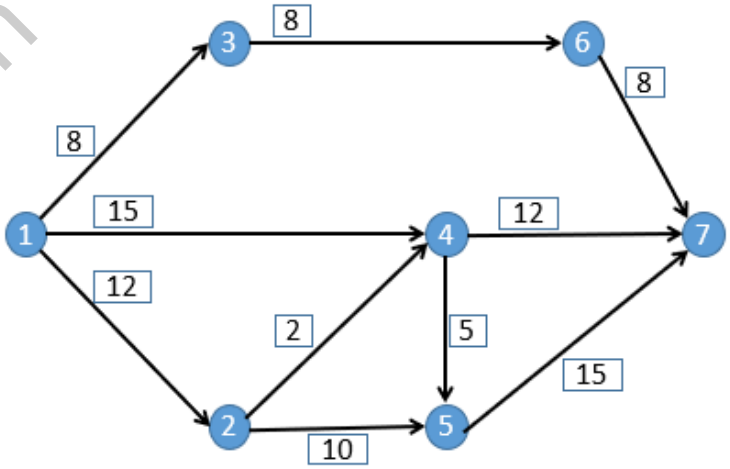


Figura 4.3.2 Solución del problema puro de flujo máximo para la red R_5

De la solución por programación lineal de WinQSB, se encontró el FLUMAX=35 en dos soluciones alternas.

	Decision Variable	Solution Value	Unit Cost or Profit c[j]	Total Contribution	Reduced Cost	Basis Status	Allowable Min. c[j]	Allowable Max. c[j]
1	F	35.0000	1.0000	35.0000	0	basic	0	M
2	X12	12.0000	0	0	0	basic	-1.0000	M
3	X13	8.0000	0	0	0	basic	-1.0000	M
4	X14	15.0000	0	0	0	basic	-1.0000	M
5	X24	2.0000	0	0	0	basic	-1.0000	0
6	X25	10.0000	0	0	0	basic	0	M
7	X36	8.0000	0	0	0	basic	-1.0000	M
8	X45	1.0000	0	0	0	basic	-1.0000	0
9	X46	6.0000	0	0	0	basic	0	M
10	X47	10.0000	0	0	0	basic	0	M
11	X57	11.0000	0	0	0	basic	-1.0000	0
12	X67	14.0000	0	0	0	basic	0	M
	Objective	Function	(Max.) =	35.0000	(Note:	Alternate	Solution	Exists!!)

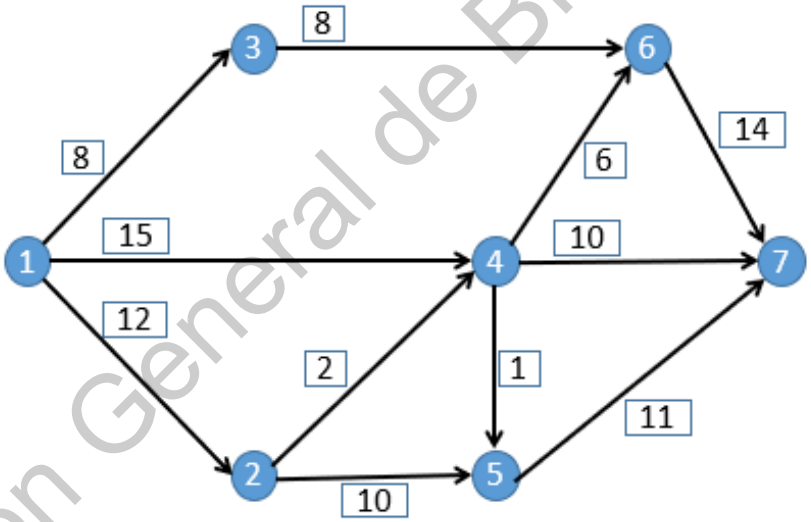


Figura 4.3.3 Solución óptima alterna para el problema puro de flujo máximo de la red R_5

Ahora, para obtener el flujo máximo a costo mínimo, se establece el siguiente programa lineal donde la oferta y demanda del problema se hace igual a 35, el flujo máximo encontrado en la primera etapa de cálculo.

$$Min\ Z = 6x_{12} + 8x_{13} + 12x_{14} + 10x_{24} + 15x_{25} + 10x_{36} + 9x_{45} + 5x_{46} + 7x_{47} + 8x_{57} + 6x_{67}$$

Sujeto a

$$x_{12} + x_{13} + x_{14} = 35 \qquad -x_{12} + x_{24} + x_{25} = 0$$

$$-x_{13} + x_{36} = 0$$
$$-x_{45} - x_{25} + x_{57} = 0$$
$$-x_{47} - x_{57} - x_{67} = -35$$
$$x_{13} \leq 10$$
$$x_{24} \leq 20$$
$$x_{36} \leq 8$$
$$x_{46} \leq 6$$
$$x_{57} \leq 20$$

$$-x_{14} - x_{24} + x_{45} + x_{46} + x_{47} = 0$$
$$-x_{36} - x_{46} + x_{67} = 0$$
$$x_{12} \leq 12$$
$$x_{14} \leq 15$$
$$x_{25} \leq 10$$
$$x_{45} \leq 16$$
$$x_{47} \leq 12$$
$$x_{67} \leq 18$$

Donde se observa que ahora el "peso" de cada variable (flujo en los arcos) corresponde al costo asociado a cada arco, y el objetivo es minimizar el costo total del flujo de valor 35. Esta última condición se establece en la primera y última restricción, donde se indica que el flujo saliente del nodo 1 debe ser igual a 35, y el flujo entrante al nodo 7 igual a 35.

Variable	X1,2	X1,3	X1,4	X2,4	X2,5	X3,6	X4,5	X4,6	X4,7	X5,7	X6,7	irectio	R. H.
Minimize	6	8	12	10	15	10	9	5	7	8	6		
C1	1	1	1									=	35
C2	-1			1	1							=	0
C3		-1				1						=	0
C4			-1	-1			1	1	1			=	0
C5					-1		-1			1		=	0
C6						-1		-1			1	=	0
C7									-1	-1	-1	=	-35
LowerBou	0	0	0	0	0	0	0	0	0	0	0		
UpperBou	12	10	15	20	10	8	16	6	12	20	18		
VariableT	nteger	nteger	nteger	nteger	nteger	nteger	nteger	nteger	nteger	nteger	nteger		

Figura 4.3.4 Planteamiento del programa lineal para el problema de flujo máximo a costo mínimo de la red R_5

	Decision Variable	Solution Value	Unit Cost or Profit c(j)	Total Contribution	Reduced Cost	Basis Status	Allowable Min. c(j)	Allowable Max. c(j)
1	X1,2	12.0000	6.0000	72.0000	0	basic	2.0000	M
2	X1,3	8.0000	8.0000	64.0000	0	basic	0	13.0000
3	X1,4	15.0000	12.0000	180.0000	0	basic	-M	16.0000
4	X2,4	3.0000	10.0000	30.0000	0	basic	6.0000	12.0000
5	X2,5	9.0000	15.0000	135.0000	0	basic	13.0000	19.0000
6	X3,6	8.0000	10.0000	80.0000	0	basic	-M	15.0000
7	X4,5	0	9.0000	0	4.0000	at bound	5.0000	M
8	X4,6	6.0000	5.0000	30.0000	0	basic	-M	7.0000
9	X4,7	12.0000	7.0000	84.0000	0	basic	-M	13.0000
10	X5,7	9.0000	8.0000	72.0000	0	basic	6.0000	M
11	X6,7	14.0000	6.0000	84.0000	0	basic	-M	8.0000
	Objective	Function	(Min.) =	831.0000				

Figura 4.3.5 Solución brindada por WinQSB para el problema de flujo máximo a costo mínimo de la red R_5

De la solución del módulo “Linear and Integer Programming” se observa que el flujo máximo a costo mínimo corresponde a enviar 35 unidades del nodo 1 al nodo 7 con un costo total de $831.00

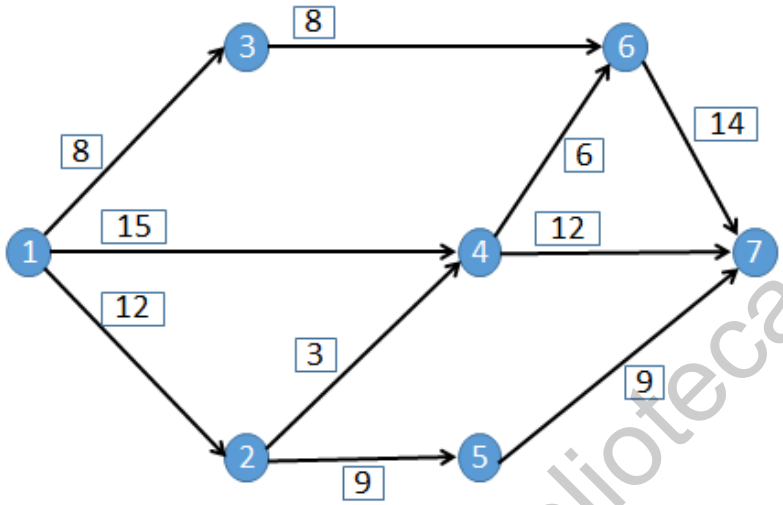


Figura 4.3.6 Solución gráfica para el problema de flujo máximo a costo mínimo para la red R_5

4.4 Aplicaciones encontradas en la literatura

Los siguientes tres ejemplos han sido extraídos de la obra de 1992 de J.R. Evans y E. Mineka; *Optimization Algorithms for Networks and graphs*[40].

Ejemplo 4.4.1.-

(*Máximo flujo a través de un oleoducto*) El petróleo debe ser transportado de la refinería (la fuente, designada nodo *s*) a los almacenes (destino, designado nodo *t*) a través de los arcos de la red. Cada arco cuenta con una capacidad que limita la cantidad de flujo que puede pasar por el arco. Se busca determinar la cantidad máxima de flujo (se denotará v) que puede ser enviado desde *s* hasta *t* sin sobrepasar el límite de capacidad de los arcos.

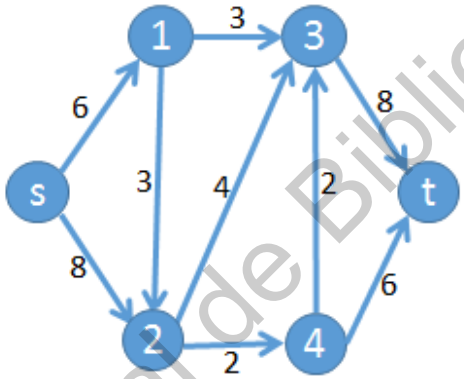


Figura 4.4.1 Red R_6 de un oleoducto[41]

Aplicando el principio de conservación de flujo y considerando las capacidades de las aristas, se tiene:

$$Max\ Z = x$$

Sujeto a

$$x - x_{s1} - x_{s2} = 0 \qquad x_{s1} - x_{13} - x_{12} = 0$$
$$x_{s2} + x_{12} - x_{23} - x_{24} = 0 \qquad x_{13} + x_{23} + x_{43} - x_{3t} = 0$$
$$x_{24} - x_{4t} = 0 \qquad x_{s1} \leq 6$$
$$x_{s2} \leq 8 \qquad x_{12} \leq 3$$
$$x_{13} \leq 3 \qquad x_{23} \leq 4$$
$$x_{24} \leq 2 \qquad x_{43} \leq 2$$
$$x_{3t} \leq 8 \qquad x_{4t} \leq 6$$

[40] Evans, J.R. & Minieka, E. Ibid.
[41] Evans, J.R. & Minieka, E., 1992. Fig. 6.1. Figura. Optimization Algorithms for Networks and Graphs, 2da edición. CRC Press.

Ingresando el programa lineal en el software WinQSB, se obtiene como resultado $x_{s1} = 3$, $x_{s2} = 6$, siendo así el flujo máximo de valor 9.

	21:33:28		Monday	November	12	2018		
	Decision Variable	Solution Value	Unit Cost or Profit c(j)	Total Contribution	Reduced Cost	Basis Status	Allowable Min. c(j)	Allowable Max. c(j)
1	X1	3,0000	1,0000	3,0000	0	basic	0	1,0000
2	X2	6,0000	1,0000	6,0000	0	basic	1,0000	M
3	X3	0	0	0	0	at bound	-M	0
4	X4	3,0000	0	0	0	basic	-1,0000	M
5	X5	4,0000	0	0	0	basic	-1,0000	M
6	X6	2,0000	0	0	0	basic	-1,0000	M
7	X7	0	0	0	0	at bound	-M	0
8	X8	7,0000	0	0	0	basic	-1,0000	0
9	X9	2,0000	0	0	0	basic	-1,0000	M
	Objective	Function	(Max.) =	9,0000				
	Constraint	Left Hand Side	Direction	Right Hand Side	Slack or Surplus	Shadow Price	Allowable Min. RHS	Allowable Max. RHS
1	C1	0	=	0	0	1,0000	-3,0000	3,0000
2	C2	0	=	0	0	1,0000	-6,0000	2,0000
3	C3	0	=	0	0	0	-1,0000	7,0000
4	C4	0	=	0	0	0	-4,0000	2,0000
5	C5	3,0000	<=	6,0000	3,0000	0	3,0000	M
6	C6	6,0000	<=	8,0000	2,0000	0	6,0000	M
7	C7	0	<=	3,0000	3,0000	0	0	M
8	C8	3,0000	<=	3,0000	0	1,0000	0	4,0000
9	C9	4,0000	<=	4,0000	0	1,0000	0	5,0000
10	C10	2,0000	<=	2,0000	0	1,0000	0	4,0000
11	C11	0	<=	2,0000	2,0000	0	0	M
12	C12	7,0000	<=	8,0000	1,0000	0	7,0000	M
13	C13	2,0000	<=	6,0000	4,0000	0	2,0000	M

Figura 4.4.2 Solución del programa lineal correspondiente a la red R_6

Asimismo, la solución proporcionada por WinQSB del programa dual es la siguiente:

	00:18:59		Tuesday	November	13	2018		
	Decision Variable	Solution Value	Unit Cost or Profit c[j]	Total Contribution	Reduced Cost	Basis Status	Allowable Min. c[j]	Allowable Max. c[j]
1	C1	1,0000	0	0	0	basic	0	3,0000
2	C2	1,0000	0	0	0	basic	0	5,0000
3	C3	0	0	0	0	basic	0	7,0000
4	C4	0	0	0	0	basic	0	2,0000
5	C5	0	6,0000	0	0	basic	3,0000	6,0000
6	C6	0	8,0000	0	5,0000	at bound	3,0000	M
7	C7	0	3,0000	0	0	at bound	3,0000	M
8	C8	1,0000	3,0000	3,0000	0	basic	3,0000	4,0000
9	C9	1,0000	4,0000	4,0000	0	basic	1,0000	5,0000
10	C10	1,0000	2,0000	2,0000	0	basic	0	6,0000
11	C11	0	2,0000	0	2,0000	at bound	0	M
12	C12	0	8,0000	0	1,0000	at bound	7,0000	M
13	C13	0	6,0000	0	4,0000	at bound	2,0000	M
	Objective	Function	(Min.) =	9,0000	(Note:	Alternate	Solution	Exists!!)
	Constraint	Left Hand Side	Direction	Right Hand Side	Slack or Surplus	Shadow Price	Allowable Min. RHS	Allowable Max. RHS
1	X1	1,0000	>=	1,0000	0	6,0000	1,0000	M
2	X2	1,0000	>=	1,0000	0	3,0000	0	1,0000
3	X3	0	>=	0	0	3,0000	0	1,0000
4	X4	0	>=	0	0	3,0000	-1,0000	M
5	X5	0	>=	0	0	4,0000	-1,0000	M
6	X6	0	>=	0	0	2,0000	-1,0000	M
7	X7	0	>=	0	0	0	-M	0
8	X8	0	>=	0	0	7,0000	-1,0000	0
9	X9	0	>=	0	0	2,0000	-1,0000	0

Figura 4.4.3 Solución del programa dual correspondiente a la red R_6

En el programa primal, las restricciones de la quinta en adelante corresponden a las capacidades de las aristas, y el corte mínimo está dado por las variables duales que resulten positivas a partir de la w_5, siendo éstas w_8, w_9 y w_{10}. Luego, el corte mínimo está dado por las aristas x_{13}, x_{23} y x_{24}.

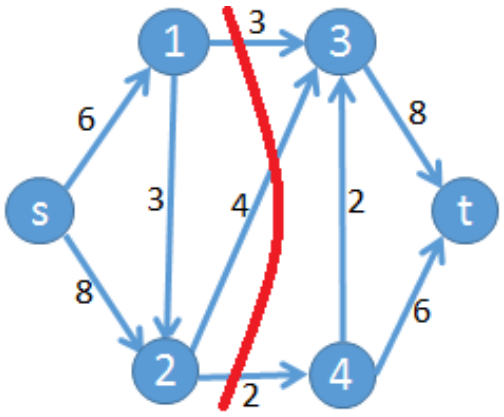


Figura 4.4.4 Corte mínimo de la red R_6

Ejemplo 4.4.2.-

(Selección de lugares) Considérese el problema de seleccionar sitios para un sistema de transmisión de mensajes electrónicos. Cualquier número de sitios pueden ser elegidos de un conjunto finito de lugares potenciales. Se saben los costos c_i de establecer el sitio i y el ingreso r_{ij} generado entre los sitios i y j, si ambos son seleccionados. ¿Qué configuración de sitios maximizaría el ingreso neto?

Supóngase que se tienen cuatro sitios potenciales, 1, 2, 3 y 4. Los costos y ganancias generadas están dados en la siguiente tabla.

Tabla 4.4.1 Tabla de costos e ingresos del problema de selección de lugares

			r_{ij}			
I	c_i	j:	1	2	3	4
1	6		—	7	5	1
2	5			—	6	2
3	4				—	1
4	5					—

La figura 4.4.5 muestra una representación gráfica. Para esta red, se desea elegir sitios formando una subred tal que la suma de los ingresos de las aristas menos el costo de los vértices sea tan grande como sea posible. Claramente, una manera de resolver este problema es enumerar todas las posibles combinaciones de subconjuntos de sitios. Para problemas de tamaño moderado, esto se vuelve computacionalmente infactible.

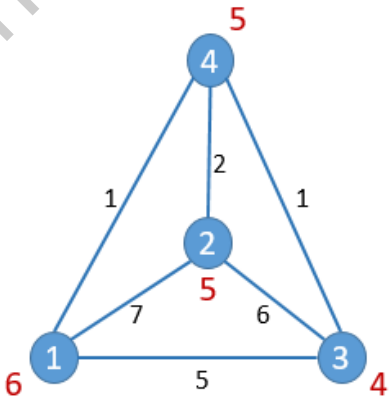


Figura 4.4.5 Representación gráfica del problema de selección de lugares[42]

Se convertirá entonces este problema en una "red lógica" que aborde el objetivo. Debe tenerse en cuenta que si se incluye la arista (i,j) que produce la ganancia r_{ij},

[42] Evans, J.R. & Minieka, E., 1992. Fig. 6.2. Figura. Optimization Algorithms for Networks and Graphs, 2da edición. CRC Press.

los sitios i y j deben incluirse con un costo de $c_i + c_j$. Se presenta cada arista en la red física mediante el componente de red lógica que se muestra en figura 4.4.6.

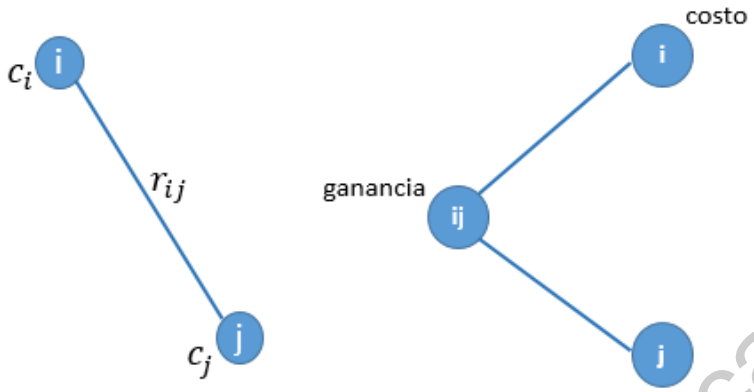


Figura 4.4.6 Construcción de una red lógica para la selección de lugares

Posteriormente se crea un nodo de origen s y una arista de s a cada nodo de ingresos ij con una capacidad de r_{ij}, y un nodo destino t con una arista hacia cada nodo de costo i con una capacidad de c_i. Todos los demás arcos tienen una capacidad grande, en este caso capacidad 100. La red completa para este ejemplo se muestra en la figura 4.4.7.

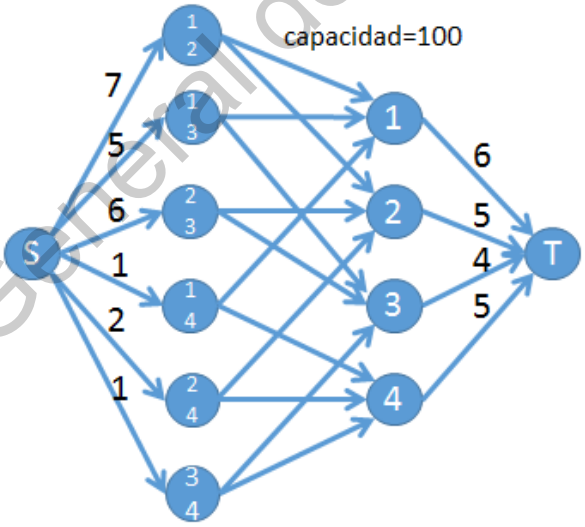


Figura 4.4.7 Red lógica R_7 para el ejemplo de selección de lugares[43]

La solución a este problema implica encontrar un tipo particular de corte. En particular, interesan los cortes simples que separan a la red en dos componentes tales que s y t están en componentes diferentes. Se busca el corte con capacidad

[43] Evans, J.R. & Minieka, E., 1992. Fig. 6.3. Figura. Optimization Algorithms for Networks and Graphs, 2da edición. CRC Press.

mínima. Para la red R_7, cualquier posible corte de capacidad mínima debe incluir los nodos de costos asociados con los nodos de ingresos elegidos, ya que los arcos de conexión tienen una capacidad grande (100 en este caso) y no podrían estar en un corte mínimo. Por ejemplo, en la figura 4.4.8, si $R = \{23,24,34\}$ se encuentra en el mismo lado que s en un corte de capacidad mínima, luego $C = \{2,3,4\}$ debe también estar en el mismo lado del corte. Se observa que la capacidad del corte es

$$r_{12} + r_{13} + r_{14} + c_2 + c_3 + c_4 + 100(6)$$

$$= 100(6) + \sum r_{ij} - (r_{23} + r_{24} + r_{34} - c_2 - c_3 - c_4)$$

$$= constante - (ingreso - costo)$$

En la figura 4.4.8 se aprecia en verde el conjunto de arcos $\mathrm{R} = \{23, 24, 34\}$ y sus conexiones con los nodos 2, 3, y 4, todos del mismo lado que el nodo S. Con color rojo se aprecia el corte correspondiente de los arcos que separan el conjunto R de las conexiones con el nodo T.

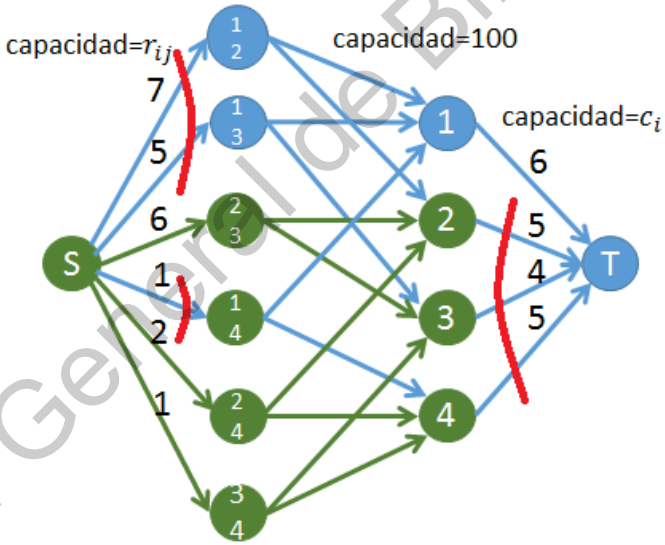


Figura 4.4.8 Ejemplo de un corte en la red de selección de lugares[44]

El programa lineal para esta red es el siguiente

$$Max\ Z = x_{S(12)} + x_{S(13)} + x_{S(23)} + x_{S(14)} + x_{S(24)} + x_{S(34)}$$

Sujeto a

[44]Evans, J.R. & Minieka, E., 1992. Fig. 6.4. Figura. Optimization Algorithms for Networks and Graphs, 2da edición. CRC Press.

$$x - x_{S(12)} - x_{S(13)} - x_{S(23)} - x_{S(14)} - x_{S(24)} - x_{S(34)} = 0$$
$$x_{S(13)} - x_{(13)1} - x_{(13)3} = 0$$
$$x_{S(14)} - x_{(14)1} - x_{(14)4} = 0$$
$$x_{S(34)} - x_{(34)3} - x_{(34)4} = 0$$
$$-x_{(12)2} - x_{(23)2} - x_{(24)2} + x_{2T} = 0$$
$$-x_{(14)4} - x_{(24)4} - x_{(34)4} + x_{4T} = 0$$
$$x_{S(13)} \leq 5$$
$$x_{S(14)} \leq 1$$
$$x_{S(34)} \leq 1$$
$$x_{(12)2} \leq 100$$
$$x_{(13)3} \leq 100$$
$$x_{(23)3} \leq 100$$
$$x_{(14)4} \leq 100$$
$$x_{(34)3} \leq 100$$
$$x_{1T} \leq 6$$
$$x_{3T} \leq 4$$
$$x_{ij} \geq 0$$

$$x_{S(12)} - x_{(12)1} - x_{(12)2} = 0$$
$$x_{S(23)} - x_{(23)2} - x_{(23)3} = 0$$
$$x_{S(24)} - x_{(24)2} - x_{(24)4} = 0$$
$$-x_{(12)1} - x_{(13)1} - x_{(14)1} + x_{1T} = 0$$
$$-x_{(13)3} - x_{(23)3} - x_{(34)3} + x_{3T} = 0$$
$$x_{S(12)} \leq 7$$
$$x_{S(23)} \leq 6$$
$$x_{S(24)} \leq 2$$
$$x_{(12)1} \leq 100$$
$$x_{(13)1} \leq 100$$
$$x_{(23)2} \leq 100$$
$$x_{(14)1} \leq 100$$
$$x_{(24)2} \leq 100$$
$$x_{(34)4} \leq 100$$
$$x_{2T} \leq 5$$
$$x_{4T} \leq 5$$

Ingresando el problema en TORA, se obtiene el valor de flujo máximo igual a 19.

Title: balinski primal
Final Iteration No.: 21
Objective Value (Max) =19.00 -- Alternative solution(s) detected (enter ITERATIONS mode to

Next Iteration | All Iterations | Write to Printer

Variable	Value	Obj Coeff	Obj Val Contrib
x1: Xs12	7.00	1.00	7.00
x2: Xs13	5.00	1.00	5.00
x3: Xs23	3.00	1.00	3.00
x4: Xs14	1.00	1.00	1.00
x5: Xs24	2.00	1.00	2.00
x6: Xs34	1.00	1.00	1.00
x7: X12,1	5.00	0.00	0.00
x8: X12,2	2.00	0.00	0.00
x9: X13,1	1.00	0.00	0.00
x10: X13,3	4.00	0.00	0.00
x11: X23,2	3.00	0.00	0.00
x12: X23,3	0.00	0.00	0.00
x13: X14,1	0.00	0.00	0.00
x14: X14,4	1.00	0.00	0.00
x15: X24,2	0.00	0.00	0.00
x16: X24,4	2.00	0.00	0.00
x17: X34,3	0.00	0.00	0.00
x18: X34,4	1.00	0.00	0.00
x19: X1T	6.00	0.00	0.00
x20: X2T	5.00	0.00	0.00
x21: X3T	4.00	0.00	0.00
x22: X4T	4.00	0.00	0.00

Figura 4.4.9 Resultado de TORA para el programa lineal del problema de selección de lugares

Siendo el programa dual;

$$Min\ W = 7w_{11} + 5w_{12} + 6w_{13} + w_{14} + 2w_{15} + w_{16} + 100w_{17} + 100w_{18} + 100w_{19} + 100w_{20} + 100w_{21} + 100w_{22} + 100w_{23} + 100w_{24} + 100w_{25} + 100w_{26} + 100w_{27} + 100w_{28} + 6w_{29} + 5w_{30} + 4w_{31} + 5w_{32}$$

Sujeto a

$$y - 7w_{11} - 5w_{12} - 6w_{13} - w_{14} - 2w_{15} - w_{16} - 100w_{17} - 100w_{18} - 100w_{19} - 100w_{20} - 100w_{21} - 100w_{22} - 100w_{23} - 100w_{24} - 100w_{25} - 100w_{26} - 100w_{27} - 100w_{28} - 6w_{29} - 5w_{30} - 4w_{31} - 5w_{32} = 0$$

$$\begin{aligned}
w_1 + w_{11} &\geq 1; & w_2 + w_{12} &\geq 1;\\
w_3 + w_{13} &\geq 1; & w_4 + w_{14} &\geq 1;\\
w_5 + w_{15} &\geq 1; & w_6 + w_{16} &\geq 1;\\
-w_1 - w_7 - w_{17} &\geq 0; & -w_1 - w_8 + w_{18} &\geq 0;\\
-w_2 - w_7 + w_{19} &\geq 0; & -w_2 - w_9 + w_{20} &\geq 0;\\
-w_3 - w_8 + w_{21} &\geq 0; & -w_3 - w_9 + w_{22} &\geq 0;\\
-w_4 - w_7 + w_{23} &\geq 0; & -w_4 - w_{10} + w_{24} &\geq 0;\\
-w_5 - w_8 + w_{25} &\geq 0; & -w_5 - w_{10} + w_{26} &\geq 0;\\
-w_6 - w_9 + w_{27} &\geq 0; & -w_6 - w_{10} + w_{28} &\geq 0;\\
w_7 + w_{29} &\geq 0; & w_8 + w_{30} &\geq 0;\\
w_9 + w_{31} &\geq 0; & w_{10} + w_{32} &\geq 0;
\end{aligned}$$

$$w_1, \dots, w_{10}\ irrestrictas, w_{11}, \dots, w_{32} \geq 0$$

Ingresando este programa dual en TORA, el resultado es que efectivamente el valor mínimo es igual a 19, de donde éste es el valor del corte mínimo.

Title: balinski dual
Final Iteration No.: 60
Objective Value (Min) =19.00

Next Iteration | All Iterations | Write to Printer

Variable	Value	Obj Coeff	Obj Val Contrib
x1: W1	1.00	0.00	0.00
x2: W2	1.00	0.00	0.00
x3: W3	1.00	0.00	0.00
x4: W4	0.00	0.00	0.00
x5: W5	0.00	0.00	0.00
x6: W6	0.00	0.00	0.00
x7: W7	-1.00	0.00	0.00
x8: W8	-1.00	0.00	0.00
x9: W9	-1.00	0.00	0.00
x10: W10	0.00	0.00	0.00
x11: W11	0.00	7.00	0.00
x12: W12	0.00	5.00	0.00
x13: W13	0.00	6.00	0.00
x14: W14	1.00	1.00	1.00
x15: W15	1.00	2.00	2.00
x16: W16	1.00	1.00	1.00
x17: W17	0.00	100.00	0.00
x18: W18	0.00	100.00	0.00
x19: W19	0.00	100.00	0.00
x20: W20	0.00	100.00	0.00
x21: W21	0.00	100.00	0.00
x22: W22	0.00	100.00	0.00
x23: W23	0.00	100.00	0.00
x24: W24	0.00	100.00	0.00
x25: W25	0.00	100.00	0.00
x26: W26	0.00	100.00	0.00
x27: W27	0.00	100.00	0.00
x28: W28	0.00	100.00	0.00
x29: W29	1.00	6.00	6.00
x30: W30	1.00	5.00	5.00
x31: W31	1.00	4.00	4.00
x32: W32	0.00	5.00	0.00

Figura 4.4.10 Resultado de TORA para el programa lineal dual del problema de selección de lugares

En el primal las capacidades de las aristas corresponden a las restricciones de la décimo primera (11) en adelante, luego, el corte mínimo está dado por las variables duales positivas a partir de w_{11}, es decir $w_{14}, w_{15}, w_{16}, w_{29}, w_{30}, w_{31}$, de donde el corte mínimo está compuesto por $x_{S,14}, x_{S,24}, x_{S,34}, x_{1T}, x_{2T}, x_{3T}$. La suma de las capacidades de este corte es efectivamente 19;

$$1 + 2 + 1 + 6 + 5 + 4 = 19$$

El corte mínimo de la red lógica de la figura 4.5.11 indica que los nodos a elegir en la red física de la figura 2.3.14 son precisamente 1, 2 y 3 que tienen un costo total para establecerlos de 6 + 5 + 4 = 15, y dan un ingreso total de 7 + 6 + 5 = 18, con ingreso neto de 18 – 15 = 3.

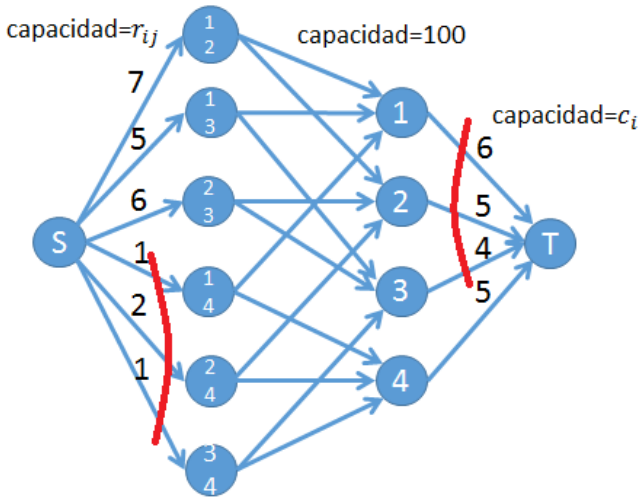


Figura 4.4.11 Corte mínimo para la red de selección de lugares

Ejemplo 4.4.3.-

(Flujos dinámicos en sistemas de manejo de materiales) Un sistema de manejo de materiales, tal como un sistema de bandas transportadoras, puede ser representado por una red dirigida en que cada arista representa el potencial (capacidad) para el transporte de artículos de un punto a otro sobre un camino fijo. Una restricción de capacidad restringe el número de elementos que pueden fluir a través del arco en cada intervalo de tiempo. También se puede asociar un costo a cada elemento (producto) que fluye a lo largo de un arco. Dado que elementos discretos se mueven dentro de la red, debe incluirse la dimensión del tiempo en el modelado de tales situaciones.

También se puede asociar un costo a cada elemento que fluye a lo largo de un arco. Para modelar el comportamiento dinámico de los elementos que fluyen a través de la red, asignamos un entero positivo que denota el número de periodos de tiempo requeridos por una unidad de flujo para viajar a través de cada arista. Un flujo dinámico de s a t en una red es cualquier flujo que cumpla con todos los requisitos de capacidad de arco en todo momento. Un ejemplo de una red de flujo dinámico es mostrado en la figura 4.4.12 con la red R_8. Los parámetros de las aristas representan capacidad y tiempo de viaje.

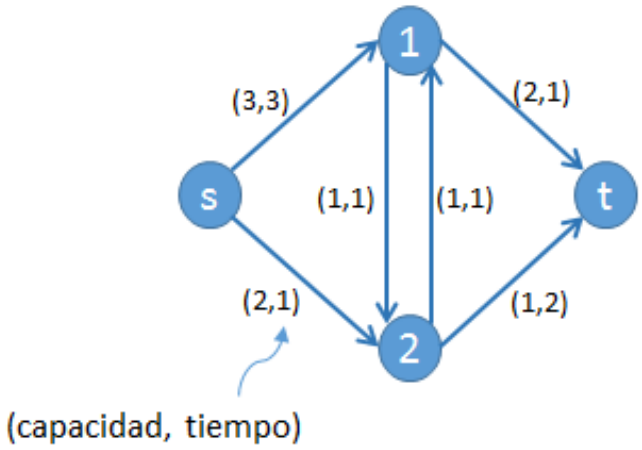


Figura 4.4.12 Ejemplo de red R_8 de flujo dinámico[45]

Dicha red puede transformarse en una red estática expandida en el tiempo al representar los intervalos de tiempo horizontalmente y los nodos de la red dinámica y los nodos de la red dinámica original verticalmente, uno para cada intervalo de tiempo. La figura 4.4.13 muestra la red R'_8 de tiempo expandido correspondiente a la red R_8.

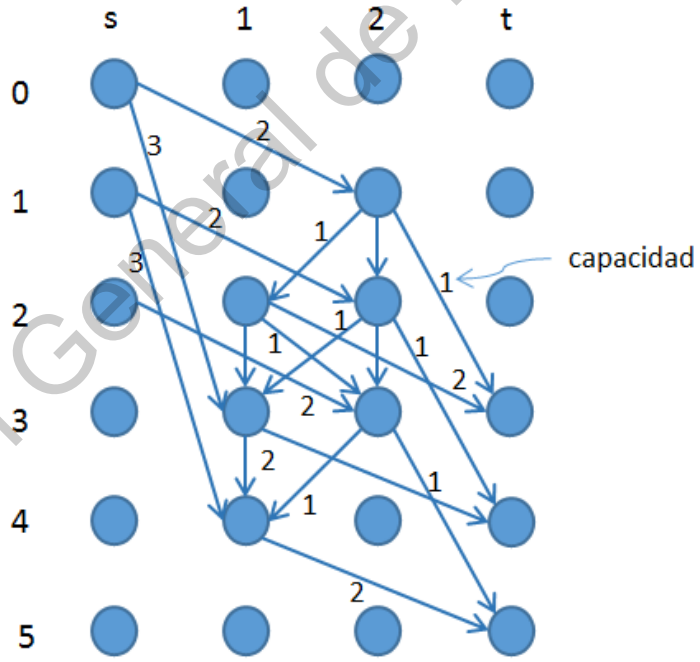


Figura 4.4.13 Red estática R'_8 con tiempo extendido para la red R_8[46]

[45] Evans, J.R. & Minieka, E., 1992. Fig. 6.5. Figura. Optimization Algorithms for Networks and Graphs, 2da edición. CRC Press.
[46] Evans, J.R. & Minieka, E., 1992. Fig. 6.6. Figura. Optimization Algorithms for Networks and Graphs, 2da edición. CRC Press.

El problema de flujo máximo también se ha usado para resolver el Problema de Apareamiento Maximal Bipartita, el cual se describe a continuación, iniciando con algunas definiciones necesarias para la discusión.

Definición 4.4.1.- Una gráfica $G = (V, A)$ es *bipartita* con clases de vértices V_1 y V_2 si $V(G) = V_1 \cup V_2$, $V_1 \cap V_2 = \emptyset$ y cada arista $a \in A(G)$ une a un vértice de V_1 con un vértice de V_2.

Un ejemplo de gráfica bipartita con siete nodos se muestra en la figura 4.4.14

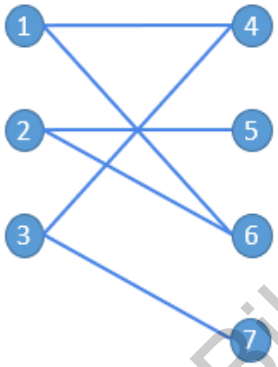


Figura 4.4.14 Gráfica G bipartita con clases $V_1 = \{1,2,3\}$ y $V_2 = \{4,5,6,7\}$

Definición 4.4.2.- Un *apareamiento M* en una gráfica bipartita $G = (V, A)$ es un subconjunto de *aristas independientes*, es decir, si $(a, b), (c, d) \in M$, luego $\{a, b\} \cap \{c, d\} = \emptyset$. Si $(a, b) \in M$, se dice que a y b son *vértices apareados*.

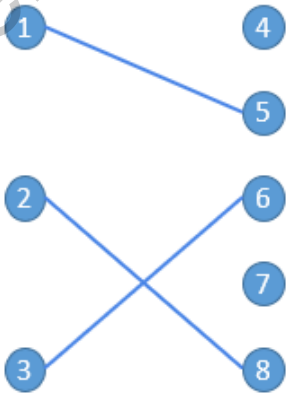


Figura 4.5.15 Gráfica bipartita G donde $M = \{(1,5), (2,8), (3,6)\}$

Definición 4.4.3.- Sea *M* un apareamiento en una gráfica bipartita $G = (V_1 \cup V_2, A)$. Si *M* aparea todos los vértices de V_1 con V_2, se dice que *M* es un *apareamiento completo* de V_1 en V_2 y cumple $|M| = |V_1|$. Si $|V_1| = |V_2|$, el apareamiento completo también es *perfecto.*

Definición 4.4.4.- Un *apareamiento maximal* en una gráfica bipartita es un apareamiento de cardinalidad máxima.

El problema de apareamiento maximal puede resolverse como un problema de flujo máximo. Para ello, a la gráfica bipartita original se agrega un nodo fuente s que se conecta a todos los nodos tipo V_1 y un nodo destino t, al cual se conectan todos los nodos tipo V_2, y se asigna capacidad unitaria a todos los arcos. Así se obtiene una gráfica modificada donde se calcula el flujo máximo $s - t$. El siguiente ejemplo ilustra esta técnica.

Ejemplo 4.4.4.- Un rescatista tiene seis animales que desea dar en adopción a cinco de sus conocidos. La siguiente tabla indica la posibilidad de adoptar a determinada mascota de acuerdo a cada persona. Se busca el máximo número de adopciones posibles. La gráfica bipartita $adoptante - mascota$ más los nodos fuente y destino se muestra en la figura 4.4.16.

Adoptante	Perro	Hámster	Gato	Cuyo	Loro	Conejo
1	X					
2		X	X			
3		X		X		X
4				X		X
5	X				X	

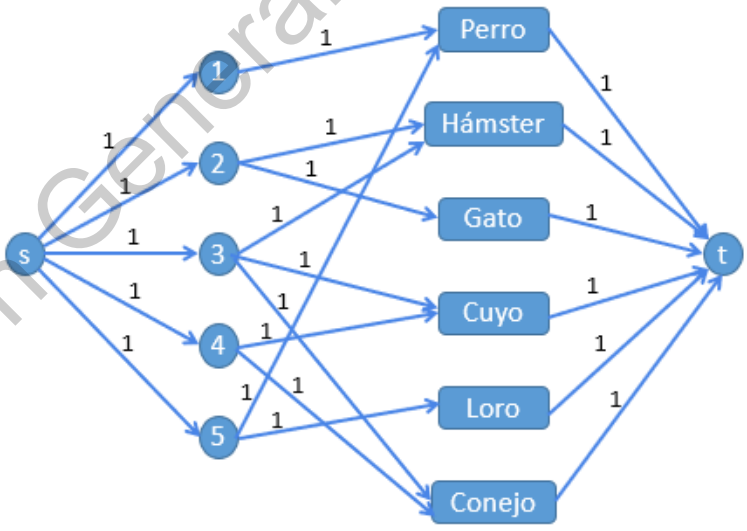


Figura 4.4.16 Red R_9 correspondiente al ejemplo 4.4.4

Se obtiene la siguiente solución con TORA; la formulación del problema como programa lineal detecta soluciones alternas para el problema.

Title: mascotas
Final Iteration No.: 25
Objective Value (Max) =5.00 -- Alternative solution(s) detected (enter ITERATIONS mode to determine such solutions)

Next Iteration | All Iterations | Write to Printer

Variable	Value	Obj Coeff	Obj Val Contrib
x1: Xs,1	1.00	1.00	1.00
x2: Xs,2	1.00	1.00	1.00
x3: Xs,3	1.00	1.00	1.00
x4: Xs,4	1.00	1.00	1.00
x5: Xs,5	1.00	1.00	1.00
x6: X1,p	1.00	0.00	0.00
x7: X2,h	0.00	0.00	0.00
x8: X2,g	1.00	0.00	0.00
x9: X3,h	1.00	0.00	0.00
x10: X3,c	0.00	0.00	0.00
x11: X3,cn	0.00	0.00	0.00
x12: X4,c	1.00	0.00	0.00
x13: X4,cn	0.00	0.00	0.00
x14: X5,p	0.00	0.00	0.00
x15: X5,L	1.00	0.00	0.00
x16: Xp,t	1.00	0.00	0.00
x17: Xh,t	1.00	0.00	0.00
x18: Xg,t	1.00	0.00	0.00
x19: Xc,t	1.00	0.00	0.00
x20: XL,t	1.00	0.00	0.00
x21: Xcn,t	0.00	0.00	0.00

Figura 4.4.17 Solución brindada por TORA para el problema de flujo máximo del ejemplo 4.4.4

De donde el flujo máximo es igual a 5, es decir, todas las personas adoptan, y el conejo queda sin adoptar ya que $x_{3cn} = x_{4cn} = 0$.

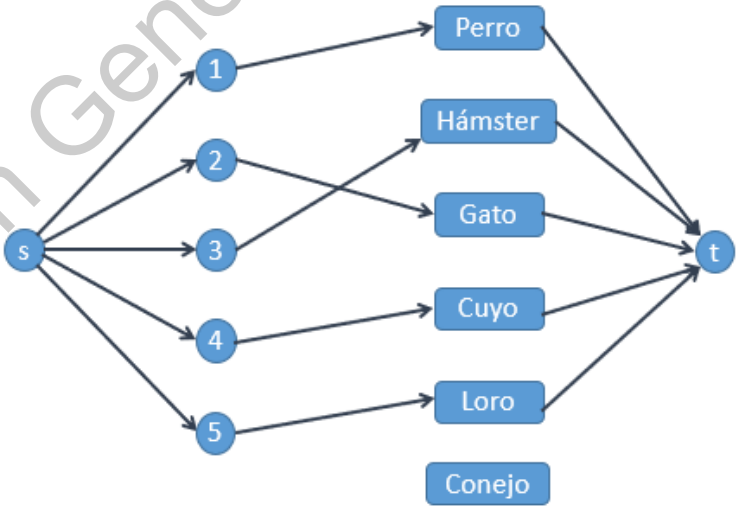


Figura 4.4.18 Asignación de cardinalidad máxima para la red R_9

Del resultado de TORA se observa que existe la siguiente solución óptima alterna, donde el gato es la mascota que queda sin adoptar.

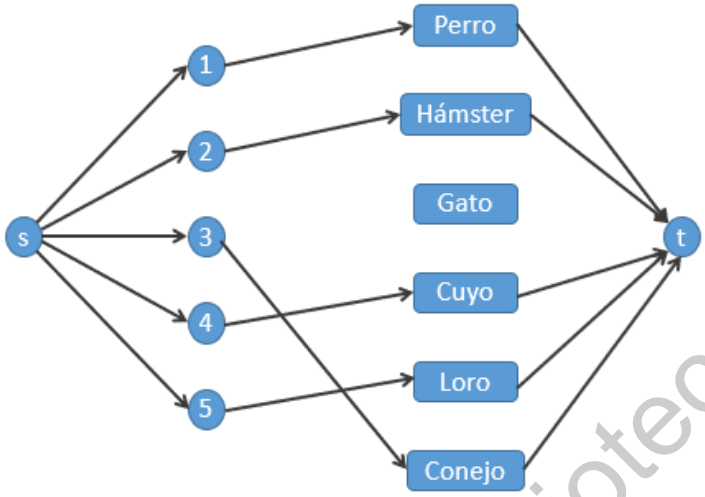


Figura 4.4.19 Asignación alterna de cardinalidad máxima

A continuación, se abordan algunas aplicaciones del apareamiento maximal en el campo del transporte.

La modalidad de *vehículo compartido* ha aparecido en el transporte urbano y suburbano como una alternativa al transporte privado y al público. Viajes específicos a un destino se anuncian en redes sociales y aplicaciones para smartphone ofreciendo compartir un vehículo con cierto número de lugares disponibles y compartiendo los gastos de viaje. En la práctica se tiene un cierto número de vehículos compartidos que ofrecen lugares y un número de pasajeros interesados, los cuales tienen preferencias para elegir la oferta de vehículo compartido. En México, aplicaciones como *BlaBlaCar, DiDi, Aventones*, ofrecen viajes compartidos que se eligen en sus plataformas de Internet o en un teléfono celular.

El problema a resolver entonces es encontrar el máximo apareamiento bipartita que permite a los pasajeros la elección con sus preferencias.

Ejemplo 4.4.5.-

Se cuenta con 5 vehículos y 15 pasajeros, cada automóvil tiene un número de asientos disponibles, por ejemplo, el vehículo 1 cuenta con tres asientos, denotados en los nodos V_{11}, V_{12} y V_{13}. La figura 4.4.20 plantea este problema en forma de red, con todos los arcos con capacidad igual a 1.

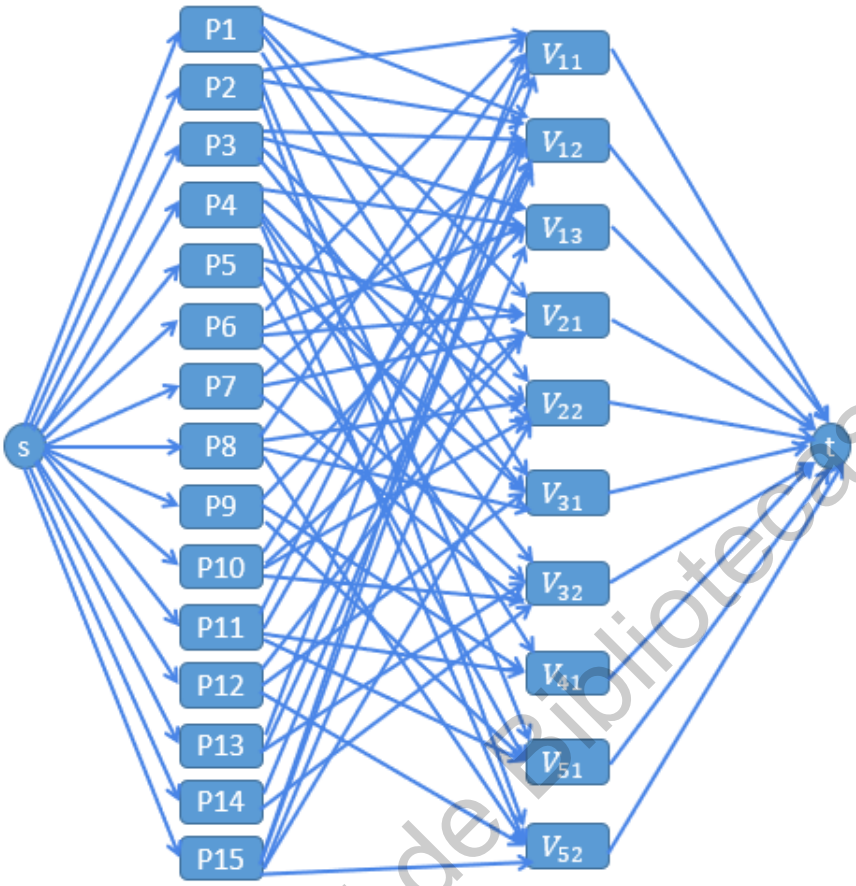


Figura 4.4.20 Red R_{10}, con capacidad igual a 1 para todos los arcos

From \ To	Fuente	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15	V11	V12	V13	V21	V22	V31	V32	V41	V51	V52	Sumidero
Fuente		1	1	1	1	1	1	1	1	1	1	1	1	1	1	1											
P1																		1		1	1	1					
P2																	1	1							1	1	
P3																		1	1		1	1					
P4																			1		1		1	1		1	
P5																				1		1			1		
P6																	1		1	1			1				
P7																		1		1			1				
P8																	1				1	1				1	
P9																			1					1	1		
P10																		1		1	1		1				
P11																	1	1						1	1		
P12																				1		1				1	
P13																	1	1					1				
P14																		1					1				
P15																	1	1	1		1					1	
V11																											1
V12																											1
V13																											1
V21																											1
V22																											1
V31																											1
V32																											1
V41																											1
V51																											1
V52																											1
Sumidero																											

Figura 4.4.21 Planteamiento de la red R_{10} por el módulo Network Modeling en WinQSB

From	To	Net Flow		From	To	Net Flow
Fuente	P1	1	16	P9	V13	1
Fuente	P2	1	17	P11	V41	1
Fuente	P3	1	18	P12	V21	1
Fuente	P6	1	19	P14	V12	1
Fuente	P8	1	20	P15	V11	1
Fuente	P9	1	21	V11	Sumidero	1
Fuente	P11	1	22	V12	Sumidero	1
Fuente	P12	1	23	V13	Sumidero	1
Fuente	P14	1	24	V21	Sumidero	1
Fuente	P15	1	25	V22	Sumidero	1
P1	V31	1	26	V31	Sumidero	1
P2	V51	1	27	V32	Sumidero	1
P3	V22	1	28	V41	Sumidero	1
P6	V32	1	29	V51	Sumidero	1
P8	V52	1	30	V52	Sumidero	1
Net Flow	From	Fuente	To	Sumidero	=	10

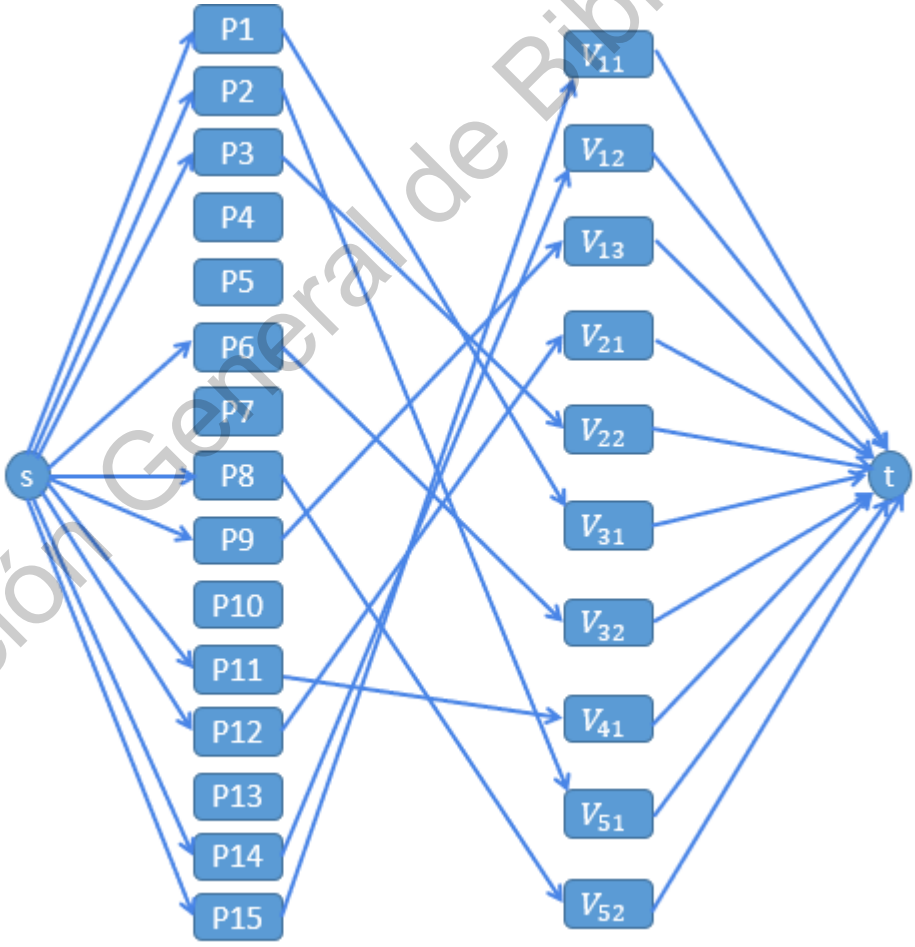


Figura 4.4.22 Solución brindada por WinQSB para la red R_{10}

La solución arroja que 10 de los 15 pasajeros obtienen un lugar en alguno de los 5 autos; en la solución óptima los pasajeros P4, P5, P7 P10 y P13 no obtienen lugar.

Como ejemplos del uso del apareamiento maximal bipartita en problemas de transporte, están los dos siguientes trabajos; el primero en relación al uso de coche compartido y el segundo en relación con el transporte ferroviario de pasajeros.

En el artículo *Mobility Sharing as a Preference Matching Problem*[47], se aborda la congestión de tráfico desde el uso de vehículos ocupados sólo para el conductor. Los avances tecnológicos y de comunicación permiten el crecimiento de los viajes compartidos e tiempo real para mejorar la eficiencia en el sistema, pero la mayoría de los algoritmos de viaje optimizan el emparejamiento de los pasajeros en función a criterios como tiempo total mínimo del vehículo o distancia recorrida por el conductor, y muy pocos consideran la preferencia de los pasajeros como el objetivo del emparejamiento. El artículo hace un modelo general de pasajeros en una red de carreteras e lo aplica analizando 301,430 viajes en Manhattan.

Asimismo, en la tesis *A Graph Approach to Distribute Waiting Rooms for Passengers*[48] se analiza el problema de salas de espera en estaciones de tren, de manera que se optimice la asignación de sala de espera a cada pasajero considerando restricciones tales como; el plan de uso de la sala debe satisfacer la demanda y entregar viajes en tren, los pasajeros que lleguen temprano deben tener acceso a la sala de espera, todos los pasajeros que esperen el mismo tren deben ser asignados a la misma sala, que debe ser lo suficientemente grande y el esquema de espera no se puede cambiar una vez que se realiza. Así se generan dos principales problemas, la asignación de la sala de espera y encontrar el plan óptimo de la sala de espera. Mediante teoría de gráficas, la tesis resuelve el primer problema mediante apareamiento maximal en una gráfica bipartita, y el segundo con un emparejamiento óptimo de una gráfica bipartita ponderada.

[47] Zhang, H., Zhao, J., (Julio de 2019). Mobility Haring as a Preference Matching Problem. *IEEE Xplore Digital Library. Volumen 20.* (Número 7).
[48] Liu, X., Zhu, Z., (26 de Abril de 2012) A Graph Approach to Distribute Waiting Rooms for Passengers. *ASCE Library.* (American Society of Civil Engineers).

5. CONCLUSIONES

A lo largo del presente texto, con ayuda de la introducción de los conceptos básicos de teoría de gráficas y teoría de redes, se ha presentado la utilidad de modelar diversos problemas en estructuras de redes, la importancia del problema de flujo máximo y su dual, la búsqueda del corte mínimo, se han mostrado diversos ejemplos acordes a cada tópico y se obtuvo una aplicación real al modelar parte de la red ferroviaria mexicana obteniendo un flujo máximo representativo de la carga ferroviaria que se mueve de los nodos en la frontera con Estados Unidos al centro de México, así como dos cortes mínimos que desconectan la red creada.

A partir de un repaso a los antecedentes del problema de flujo máximo, salta a la vista el hecho de que los primeros estudios formales surgieron para resolver una problemática de transporte, siendo ésta la aplicación más intuitiva del modelo, más no la única. Gracias a la estructura propia de las gráficas, y por tanto, de las redes como aquí se han definido, el modelo de flujo máximo puede ajustarse a gran variedad de áreas, problemas y sub-problemas. Como se mencionó en la sección 4.5, un tipo especial de red (gráfica bipartita) es ampliamente ventajoso en los problemas de asignación llamados Problema de Apareamiento Maximal Bipartita.

El presente también buscó demostrar la importancia del dual del problema de Flujo Máximo; la búsqueda de Corte Mínimo, el cual cuenta con muchas aplicaciones por sí misma en problemas de optimización.

Asimismo, para la resolución de los problemas de Flujo Máximo-Corte Mínimo se cuentan con diversos métodos, lo que facilita su estudio y aplicación. Una primera instancia es el uso de la programación lineal para describir la función de flujo a maximizar en la red, creando así un problema primal y a partir de éste un problema dual que provee el corte mínimo, resolviéndose por método simplex. Como se describió en la sección 3.1, la solución por simplex otorga un análisis post óptimo importante para una mayor comprensión del problema tratado y la toma de decisiones, ya que muestra detalles de las variables de holgura, los intervalos comprendidos donde el análisis de sensibilidad tiene validez y los precios sombra, los cuales son de suma importancia en aplicaciones económicas al describir la tasa de cambio del valor objetivo óptimo respecto a las restricciones. También se cuenta con el algoritmo basto conocido de Ford y Fulkerson. La resolución de estos problemas puede darse con el uso de los software descritos tales como TORA, WinQSB, R, Phyton o QM, así como diversos códigos en línea.

Por supuesto, los tópicos abordados pueden extenderse para estudios o trabajos futuros relacionados, como los ejemplos a continuación, que por restricciones de tiempo para el presente trabajo ya no se examinaron con más detalle.

Retomando el tema de matriz adyacencia usada en los capítulos 1 y 2, una *matriz Laplaciana* es una matriz de representación de una gráfica, en el sentido de que está definida como $L = D - A$, donde D es la matriz de grado de la gráfica, y A es la matriz de adyacencia de la misma. Una propiedad importante que provee la matriz Laplaciana es que el número de componentes en una gráfica conexa (árboles de expansión) es igual al determinante de la matriz Laplaciana.

Asimismo, una parte fundamental de este texto fue la presentación e importancia del corte mínimo en una gráfica G, la cual la separa en dos partes al hacer una partición del conjunto V de vértices en los subconjuntos S y $\bar{S}$. Existe una generalización consistente en la separación de la gráfica en k partes; el problema de $k-cortes\ mínimos$ que requiere encontrar el corte k de peso mínimo. Este problema encuentra sus aplicaciones en campos como minería de datos, comunicación en cómputo paralelo y elementos finitos.

Siguiendo con el corte mínimo, se cuenta con el algoritmo de Karger, el cual es un algoritmo aleatorio con enfoque en la simulación. Karger utiliza el concepto de contracción de una arista (i, j) en un solo nodo ij, pero conservando las aristas que estaban conectadas con i o j. El algoritmo iterativamente elige aristas al azar y "reduce" la gráfica hasta que queden únicamente dos nodos, los que representan un corte en el gráfico original. Al realizar el algoritmo un número suficiente de veces, se puede encontrar un corte mínimo con alta probabilidad.

Otro problema importante en la teoría de redes y ampliamente estudiado es encontrar el *flujo a costo mínimo* en una red R donde los arcos tienen capacidad limitada y costos asociados. Este problema fue formulado en la sección 4.3, y puede resolverse con un enfoque de flujo máximo, usando en particular el algoritmo de Ford y Fulkerson para encontrar caminos aumentantes[49].

Supóngase que la función de flujo a costo mínimo fuera remplazada por

$$Max\ \{pf - \sum_{(i,j)\in A} c_{ij} f_{ij}\}$$

donde p es un número tan grande como se quiera, por ejemplo, más grande que el costo máximo total del flujo al que se pudiera incurrir al pasar de la fuente al destino. De este modo, interpretando p como la ganancia recibida por cada unidad enviada de s a t, la ecuación anterior se aclara como la mejor ganancia neta después de los costos de transporte del flujo. De esta interpretación, se sigue que cualquier flujo que maximice esta expresión también minimizará la función objetivo original, y viceversa.

[49] Evans, J.R. & Minieka, E. Ibid.

El algoritmo de flujo a costo mínimo primero envía tantas unidades como sea posible de s a t tal que se incurra en un costo total de 0 por la jornada entera de s a t. Luego, este algoritmo envía tantas unidades como sea posible de s a t tal que el costo incremental del flujo sea igual a 1. El algoritmo se detiene cuando un flujo total de f ha sido enviado de s a t, o cuando más unidades no pueden ser enviadas, lo que sea que ocurra primero. En otras palabras, el algoritmo resuelve el problema para $p = 0$, luego para $p = 1$, luego para $p = 2$, etc.

Supóngase que tantas unidades como sea posible con un costo incremental total de $p - 1$ o menos han sido enviadas por el algoritmo de s a t. ¿Cómo el algoritmo determina cómo enviar unidades de flujo de s a t de manera que el costo total incremental sea de p? Para hacer esto, el algoritmo debe localizar un camino de flujo aumentante de s a t con la propiedad de que el costo total incremental de enviar una unidad “a través del camino” sea igual a p.

Se insta entonces a continuar con los estudios del Teorema de Flujo Máximo-Corte Mínimo así como sus derivados, ya que con los campos de estudio emergentes es evidente que sus aplicaciones no acaban sino por comenzar.

6. ANEXOS

6.1 Anexo 1. Programa lineal para la red del sistema ferroviario mexicano

$$\boldsymbol{Max\ Z = x_{1,2} + x_{1,13} + x_{1,15} + x_{1,32} + x_{1,40} + x_{1,56}}$$

$$x_{1,2} - x_{2,3} = 0 \qquad x_{2,3} - x_{3,6} = 0$$

$$x_{3,6} - x_{6,7} = 0 \qquad x_{7,8} - x_{8,9} = 0$$

$$x_{8,9} + x_{11,9} - x_{9,21} = 0 \qquad x_{9,21} - x_{21,22} = 0$$

$$x_{21,22} - x_{22,23} = 0 \qquad x_{22,23} - x_{23,25} = 0$$

$$x_{23,25} - x_{25,20} - x_{25,26} = 0 \qquad x_{25,26} - x_{26,27} = 0$$

$$x_{25,20} + x_{19,20} - x_{20,45} = 0 \qquad x_{20,45} - x_{45,57} = 0$$

$$x_{26,27} + x_{44,27} - x_{27,48} = 0 \qquad x_{27,48} - x_{48,57} = 0$$

$$x_{44,47} - x_{47,57} = 0 \qquad x_{1,13} - x_{13,12} = 0$$

$$x_{13,12} - x_{12,11} = 0 \qquad x_{1,15} - x_{15,14} = 0$$

$$x_{15,14} - x_{14,11} = 0 \qquad x_{12,11} + x_{14,11} - x_{11,9} - x_{11,16} = 0$$

$$x_{1,32} - x_{32,31} = 0 \qquad x_{32,31} - x_{31,30} = 0$$

$$x_{31,30} - x_{30,28} = 0 \qquad x_{11,16} - x_{16,17} = 0$$

$$x_{28,17} + x_{16,17} - x_{17,18} - x_{17,38} = 0 \qquad x_{17,38} - x_{38,18} = 0$$

$$x_{17,18} + x_{38,18} - x_{18,19} = 0 \qquad x_{30,28} + x_{33,28} - x_{28,17} - x_{28,29} = 0$$

$$x_{1,40} - x_{40,39} = 0 \qquad x_{40,39} - x_{39,34} = 0$$

$$x_{39,34} - x_{34,33} - x_{34,53} = 0 \qquad x_{53,33} - x_{34,33} - x_{33,28} = 0$$

$$x_{1,56} - x_{56,55} = 0 \qquad x_{56,55} - x_{55,54} = 0$$

$$x_{55,54} - x_{54,53} - x_{54,52} = 0 \qquad x_{34,53} + x_{54,53} - x_{53,33} - x_{53,29} - x_{53,52} = 0$$

$$x_{28,29} + x_{53,29} - x_{29,42} = 0 \qquad x_{54,52} + x_{53,52} - x_{52,51} = 0$$

$$x_{52,51} - x_{51,50} = 0 \qquad x_{51,50} - x_{50,43} = 0$$

$$x_{42,43} + x_{50,43} - x_{43,41} - x_{43,44} = 0 \qquad x_{29,42} - x_{42,41} - x_{42,43} = 0$$

$$x_{42,41} + x_{43,41} - x_{41,19} = 0$$

$$x_{18,19} + x_{41,19} - x_{19,20} = 0$$

$$x_{6,7} - x_{7,8} = 0$$

$$x_{43,44} - x_{44,27} - x_{44,47} = 0$$

$$x_{1,2} \leq 110$$

$$x_{2,3} \leq 110$$

$$x_{3,6} \leq 120$$

$$x_{6,7} \leq 120$$

$$x_{7,8} \leq 120$$

$$x_{8,9} \leq 120$$

$$x_{11,9} \leq 110$$

$$x_{9,21} \leq 120$$

$$x_{21,22} \leq 120$$

$$x_{22,23} \leq 120$$

$$x_{23,25} \leq 123$$

$$x_{25,20} \leq 123$$

$$x_{25,26} \leq 123$$

$$x_{26,27} \leq 110$$

$$x_{19,20} \leq 123$$

$$x_{20,45} \leq 123$$

$$x_{45,57} \leq 123$$

$$x_{44,27} \leq 130$$

$$x_{27,48} \leq 110$$

$$x_{48,57} \leq 110$$

$$x_{44,47} \leq 130$$

$$x_{47,57} \leq 130$$

$$x_{13,12} \leq 123$$

$$x_{12,11} \leq 123$$

$$x_{1,15} \leq 110$$

$$x_{15,14} \leq 110$$

$$x_{14,11} \leq 110$$

$$x_{11,16} \leq 123$$

$$x_{1,13} \leq 123$$

$$x_{1,32} \leq 123$$

$$x_{32,31} \leq 123$$

$$x_{31,30} \leq 123$$

$$x_{30,28} \leq 123$$

$$x_{16,17} \leq 123$$

$$x_{17,18} \leq 123$$

$$x_{17,38} \leq 110$$

$$x_{38,18} \leq 110$$

$$x_{18,19} \leq 123$$

$$x_{33,28} \leq 123$$

$$x_{28,17} \leq 123$$

$$x_{28,29} \leq 123$$

$$x_{1,40} \leq 130$$

$$x_{40,39} \leq 130$$

$$x_{39,34} \leq 130$$

$$x_{34,33} \leq 130$$

$$x_{34,53} \leq 130$$

$$x_{53,33} \leq 123$$

$$x_{1,56} \leq 130$$

$x_{56,55} \leq 130$

$x_{55,54} \leq 130$

$x_{54,53} \leq 130$

$x_{54,52} \leq 123$

$x_{53,29} \leq 130$

$x_{53,52} \leq 123$

$x_{29,42} \leq 130$

$x_{52,51} \leq 123$

$x_{51,50} \leq 123$

$x_{50,43} \leq 130$

$x_{42,43} \leq 130$

$x_{43,41} \leq 130$

$x_{43,44} \leq 130$

$x_{42,41} \leq 130$

$x_{41,19} \leq 130$

7. REFERENCIAS BIBLIOGRÁFICAS

Anderson, L.B., Atwell, R.J., Barnett, D.S., Bovey, R.L. (Septiembre de 2007) Application of the Maximum Flow Problem to Sensor Placement on Urban Road Networks for Homeland Security, *Homeland Security Affairs, Volumen 3.* No. 3.

Ahuja, R., Magnati, T., Orlín, J. (1993) *Network Flows Theory, Algorithms, and Applications*, 1ra edición, Upper Saddle River; Prentice-Hall Inc.

Balakrishnan, V.K., (1997), *Shaum´s outline of theory and problems of Graph Theory*, 9na edición, New York, Shaum´s outline series, McGRAW-HILL.

Comellas, F., Fábrega, J., Sánchez, A., (2001) *Matemática Discreta*, 1ra edición, España, Edicions UPC.

Cook, W.J., Cunningham, W.H., Pulleyblank, W.R., Schrijver, A. (1998) *Combinatorial Optimization,* Canadá, John Wiley & Sons, Inc.

Dantzing, G. B., (1963) *Linear Programming and Extensions*, a report prepared for United States Air Force Project Rand.

Evans, J.R. & Minieka, E., (1992) *Optimization Algorithms for Networks and Graphs*, 2da edición. CRC Press.

Ford, L.R., Fulkerson, D.R. (10 de Noviembre de 1956) Maximal Flow through a Network, *Research memorandum RM-1400, The RAND Corporation*, *Canadian Journal of Mathematics Volumen 8.*

Ford, L.R., Fulkerson, D.R. (1962) *Flows in Networks*, Princeton, New Jersey, EUA., Princeton Landmarks in Mathematics.

Free Software Foundation. (s.f). En *Wikipedia*. Recuperado el 30 de agosto de 2019 de https://es.wikipedia.org/wiki/Free_Software_Foundation

Gustav Robert Kirchhoff. (s.f). En *Wikipedia*. Recuperado el 31 de marzo de 2017 de https://es.wikipedia.org/wiki/Gustav_Kirchhoff.

Hernández, L.G., González, O. (Abril de 2011) Identificación de tramos críticos por vulnerabilidad para el traslado de mercancía en la red carretera pavimentada de México. *Investigaciones Geográficas, Boletín del Instituto de Geografía, UNAM, Volumen 74.*

Investigación de Operaciones 1. (s.f). En *Investigación de Operaciones IND-331.* Recuperado el 10 de Octubre de 2019 de https://investigaciondeoperacionesind331.blogspot.com/p/programacion-entera.html

Kaanodiya, K.K., Rizwanullah, M. (2012) Minimize traffic congestion: an application of Maximum Flow in dynamic networks. *Journal of the Applied Mathematics, Statistics and Informatics (JAMSI), Volumen 8.* No.1.

Klerk, E. de. Integer programming and totally unimodular. Delf University of Technology. *Semanticsholar*. Recuperado de: https://pdfs.semanticscholar.org/a5c4/cf30dcbbc130f693dd1b75e0032f42b741bd.pdf

Liu, X., Zhu, Z., (26 de Abril de 2012) A Graph Approach to Distribute Waiting Rooms for Passengers. *ASCE Library.* (American Society of Civil Engineers)

Mapa ferroviario. Elaboración propia del IMT por la M. en Geo. Gabriela García. 2019.

Python. (s.f). En *Wikipedia*. Recuperado el 15 de septiembre de 2019 de https://es.wikipedia.org/wiki/Python

QM (s.f). En *Prenhall.* Recuperado el 10 de Octubre de 2019 de https://wps.prenhall.com/bp_weiss_software_1

R (lenguaje de programación). (s.f). En *Wikipedia.* Recuperado el 15 de septiembre de 2019 de https://es.wikipedia.org/wiki/R_(lenguaje_de_programacion)

Taha, H., (2012), *Investigación de Operaciones*, 9na edición, México, PEARSON EDUCACION.

Tolstoi, A.N. (1930) Methods of finding the minimal total kilometrage in cargo-transportation planning in space, *Transportation Planning, Volumen 1.*

WinQSB (s.f). En *Swmath.* Recuperado el 10 de Octubre de 2019 de http://www.swmath.org/software/6146

Zhang, H., Zhao, J., (Julio de 2019). Mobility Haring as a Preference Matching Problem. *IEEE Xplore Digital Library. Volumen 20.* (Número 7).