

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
FACULTAD DE INGENIERÍA



FACULTAD DE INGENIERÍA

APUNTES TEORÍA DE REDES

Idalia Flores de la Mota

DIVISIÓN DE ESTUDIOS DE POSGRADO
DEPARTAMENTO DE SISTEMAS

APUNTE
198-B

FACULTAD DE INGENIERIA UNAM.



612685

G.- 612685

FLORES DE LA MOTA, Idalia. *Apuntes teoría de redes.*
México, UNAM, Facultad de Ingeniería, División de
Estudios de Posgrado, 1999, 181 p.



FACULTAD DE INGENIERIA

Apuntes teoría de redes

Prohibida la reproducción o transmisión total o parcial de esta obra por cualquier medio o sistema electrónico o mecánico (incluyendo el fotocopiado, la grabación o cualquier sistema de recuperación y almacenamiento de información), sin consentimiento por escrito del editor.

Derechos reservados.

© 1999, Facultad de Ingeniería, Universidad Nacional Autónoma de México.

Ciudad Universitaria, 04510, México, D.F.

ISBN 968-36-9076-9

Primera edición, 1999.

Impreso y hecho en México.

SOBRE LA AUTORA

Idalia Flores De La Mota

Estudió el Doctorado en Investigación de Operaciones en la Facultad de Ingeniería de la UNAM. Obtuvo mención honorífica por su tesis de maestría así como la medalla Gabino Barreda por el mejor promedio de su generación. Ha asistido a varios congresos nacionales e internacionales, y ha sido miembro del Sistema Nacional de Investigadores. Ha publicado en memorias de congresos así como apuntes para las materias de Matemáticas Aplicadas y Programación Entera. Ha impartido cursos en universidades del extranjero, en la Facultad de Economía de la Universidad de la Habana, Cuba así como en la Universidad Nacional Autónoma de Nicaragua. Es profesora de tiempo completo de la Facultad de Ingeniería desde 1990 y actualmente es la Jefa de la Sección de Investigación de Operaciones de la División de Estudios de Posgrado de la Facultad de Ingeniería.

G- 612685

Como parte de las actividades del Departamento de Sistemas de la División de Estudios de Posgrado de la Facultad de Ingeniería UNAM, nos hemos propuesto el desarrollo de una serie de apuntes que sirvan de apoyo a los diferentes cursos que se imparten y, desde luego, como material de referencia o de lectura para quienes así lo deseen.

En dichos apuntes se busca mantener un alto nivel, de manera que contribuyan a una sólida formación teórica del alumno, pero al mismo tiempo se intenta acercarlo a las aplicaciones de tales conocimientos, como es en síntesis el objetivo central del posgrado de ingeniería.

La estructura de estos apuntes consiste de siete capítulos, en los cuales se desarrolla cada uno de los temas con unas notas históricas al final de cada capítulo.

Como autora de estos apuntes quiero expresar mi agradecimiento a mis ayudantes, Ivonne Peña Galeana y Leonardo Niño Ojeda por su colaboración en la elaboración de los mismos, así como a la Lic. Guadalupe Castro por la revisión y comentarios que hicieron mejorar la calidad de este trabajo. Finalmente quiero agradecer al Jefe de la DEPFI, Dr. Gabriel de las Nieves Sánchez Guerrero por su interés en la publicación de los mismos y las facilidades prestadas para su impresión.

PRÓLOGO

La Teoría de Redes es una área dentro de la Investigación de Operaciones cuyo origen está vinculado a la Programación Lineal, el gran desarrollo de la computación ha provocado un gran desarrollo de los algoritmos propios de esta teoría lo que ha permitido un gran avance en la teoría algorítmica. Su versatilidad y la representación gráfica de los problemas hace de ésta una de las áreas más usadas en el campo de las aplicaciones, cuyo rango es tan amplio que abarca la química, la física, redes de computadora, muchas ramas de la ingeniería, manufactura, política pública y sistemas sociales, tráfico urbano, telecomunicaciones y transporte por citar solo algunos. El problema central de resolver un problema de flujo a costo mínimo, conlleva a que los problemas de flujo máximo, ruta más corta, asignación, transporte y circulación se puedan ver ligados y formando parte de este problema central.

Estos apuntes tienen como objetivo tener un apoyo didáctico para la clase, así como un complemento de la bibliografía sugerida para el curso. Y a que por su amplitud no es posible cubrir todos los tópicos de redes que están contenidos en un libro de texto, se da una visión sencilla y general del tema. Los grandes temas que se abordan son flujo máximo y corte mínimo, ruta más corta, árbol de expansión mínima y por último el problema de flujo a costo mínimo. Se exponen diferentes algoritmos de un mismo problema para compararlos y ver su complejidad computacional, así como una variedad de ejemplos desarrollados paso a paso para facilitar su comprensión. Finalmente se expone una serie de notas históricas al final de cada capítulo, con el objetivo de que el lector tenga una visión más amplia sobre cada tema.

ÍNDICE

CAPÍTULO I

FORMULACIÓN DE PROBLEMAS DE REDES TÍPICOS

1.1 Introducción	1
1.2 Tipos básicos de problemas de redes	7
1.3 Redes de tránsito urbano	7
1.4 Planteamiento de modelo de redes	8
1.5 El problema de la ruta más corta	8
1.6 El problema del flujo máximo	10
1.7 El árbol de expansión mínima	12
1.8 Planteamiento de modelo de redes como problemas de programación lineal	17
1.9 Notas históricas	31

CAPÍTULO 2

CONCEPTOS BÁSICOS DE REDES

2.1 Introducción	32
2.2 Representación matricial de una red	32
2.3 Redes especiales	34
2.4 Flujo y divergencia en una red	37
2.5 Circulaciones	42
2.6 Potenciales y tensión	44
2.7 Flujos óptimos y potenciales	47
2.8 Notas históricas	48

CAPÍTULO 3

TRAYECTORIAS Y CORTES

3.1 Introducción	49
3.2 El problema de determinar una trayectoria	52
3.3 Algoritmo de enrutamiento	57
3.4 Aplicaciones a conectividad	64
3.5 Notas históricas	66

CAPÍTULO 4

FLUJOS Y CAPACIDADES

4.1 Introducción	67
4.2 Principio fundamental de divergencia	67
4.3 El Problema de flujo máximo de una red	71
4.4 El problema de factibilidad de flujo	79
4.5 Notas históricas	100

CAPÍTULO 5

ÁRBOL DE EXPANSIÓN MÍNIMA

5.1 Introducción	101
5.2 Árbol de expansión mínima	107
5.3 Algoritmo de Kruskal	108
5.4 Algoritmo de PRIM	112
5.5 Algoritmo de Sollin	115
5.6 Análisis de sensibilidad	119
5.7 Notas históricas	123

CAPÍTULO 6

PROBLEMA DE RUTA MÁS CORTA

6.1 Introducción	124
6.2 Caracterización de una arborescencia	127
6.3 Métodos de solución y justificación	129
6.4 Problema de ruta más corta con programación dinámica	148
6.5 Notas históricas	153

CAPÍTULO 7

PROBLEMA DE FLUJO A COSTO MÍNIMO

7.1 Introducción	154
7.2 Formulación de programación lineal	155
7.3 Algoritmo simple para redes con capacidades en los arcos	160
7.4 Método basado en la eliminación de circuitos negativos	167
7.5 Método basado en ruta más corta	174
7.6 Notas históricas	180

BIBLIOGRAFÍA

CAPÍTULO 1

FORMULACIÓN DE PROBLEMAS DE REDES TÍPICOS

1.1 INTRODUCCIÓN

Cualquier red está formada por tres componentes (1) nodos, (2) arcos y (3) flujo en los arcos como se ilustra en la figura 1.1.

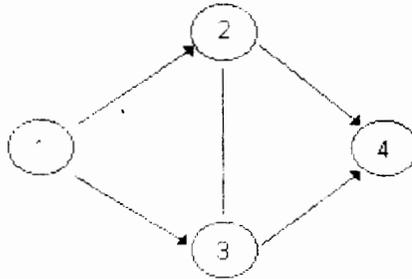


Figura 1.1

Los círculos son los nodos y están unidos por arcos, observe que en la figura hay dos tipos de arcos, dirigidos y no dirigidos. Un arco dirigido es aquel sobre el cual puede moverse el flujo en una sola dirección, y uno no dirigido es aquel sobre el cual puede moverse el flujo en cualquier sentido. En la figura 1.1, el arco que une los nodos 1 y 2 es un arco dirigido y el que une los nodos 2 y 3 es no dirigido.

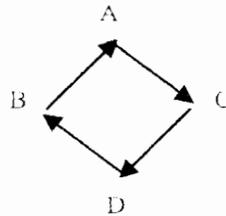
Por lo general los nodos se enumeran, tal como se ha mostrado en la figura 1.1 y los arcos se denotan por los nodos que unen. Por ejemplo, el arco que une los nodos 1 y 2 se identifica como el arco $j_1 \sim (1,2)$.

El flujo que pasa de un nodo a otro a través de un arco es un factor desconocido en la red y se le denota como X_{ij} para el flujo entre los nodos i y j .

El flujo de una red puede constar de muchos bienes o productos distintos. Unos cuantos ejemplos serían: gas natural en un gasoducto, distribución de artículos de mayoristas a detallistas o entre fábricas y almacenes. El costo unitario del flujo para cada arco se denota como C_{ij} para los nodos i y j . En el caso de una red carretera, las ciudades son los nodos y las rutas de transporte entre las ciudades son los arcos. El costo por automóvil para cada ruta es el costo del flujo. En algunos problemas pueden existir capacidades para cada arco que limiten la cantidad de flujo.

Puede entonces definirse una red como un conjunto de nodos, arcos y flujos que pasan de un nodo a otro a través de los arcos.

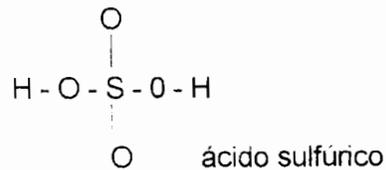
Un buen dibujo es mejor que una larga lectura (o una imagen dice más que mil palabras). El lenguaje de las gráficas trata de poner ésta idea a tal efecto. De hecho, muy a menudo es un reflejo natural que nos hace abstraer una situación dada, dibujando en una hoja de papel puntos que pueden representar individuos.



Lugares

	A	C	
	B	D	
	E	F	
	G	H	

componentes quimicos:



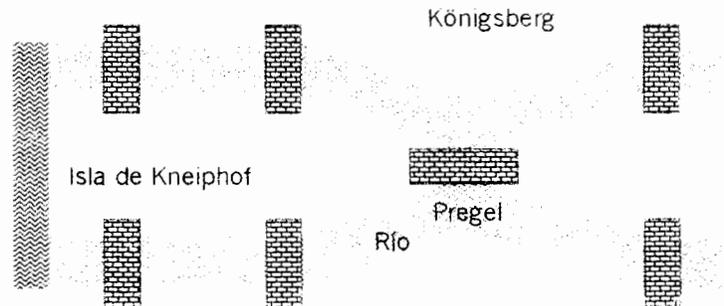
y así conectando con líneas o flechas que simbolizan una cierta relación.

Esta representación de la realidad por figuras tiene las siguientes ventajas:

1. De esta forma podemos expresar la estructura profunda de una situación dada.

- Desde un punto de vista práctico, presenta una visión completa del problema, lo que representa una valiosa guía para la intuición y el razonamiento.

El primer documento en gráficas "solutio problematis ad geometriam situs pertinentis", lo escribió Euler en 1736. Comenzó con una curiosidad matemática: cómo puede uno caminar una y sólo una vez a través de cada uno de los siete puentes de Königsberg. Dichos puentes se muestran en la figura siguiente:



Desde 1946 la teoría de las gráficas ha sido ampliamente desarrollada, bajo la inspiración de muchos especialistas en investigación de operaciones, motivados por problemas actuales. En forma paralela, un esfuerzo importante de síntesis ha prevalecido.

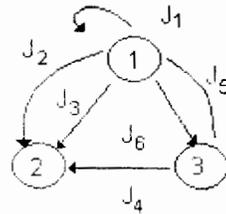
Es a partir de 1960, que ayudados por la aparición de las primeras computadoras electrónicas, hemos visto una explosión real de investigación y de aplicaciones.

Una red G consiste de 3 componentes:

Un conjunto N de nodos (vértices o puntos) que se representan con círculos pequeños, un conjunto A de arcos (líneas, flechas o cadenas) que se representan con flechas, la dirección de éstas da una orientación al arco.

Y una función $\rho : A \rightarrow N \times N$ que asigna a cada $J \in A$ una pareja $(i, i') \in N \times N$ tal que $i \neq i'$, se supone $N \neq \emptyset$. Si $i = i'$ se trata de un rizo.

EJEMPLO 1.1



$$N = \{1,2,3\}$$

$$A = \{J_1, J_2, J_3, J_4, J_5, J_6\}$$

$$\rho : A \rightarrow N \times N$$

o bien

$$\rho(J_1) = (1,1)$$

$$\rho(J_2) = (1,2)$$

$$\rho(J_3) = (1,2)$$

$$\rho(J_4) = (3,2)$$

$$\rho(J_5) = (1,3)$$

$$\rho(J_6) = (1,3)$$

los arcos j tal que $\rho(j) = (i,i)$ como ya se ha dicho anteriormente se denominarán rizos.

Si no existen arcos paralelos es normal hacer

$$j \sim (i_1, i_2)$$

o bien

$$(i_1, i_2)$$

y se dice que i_1 es el nodo inicial de j , e i_2 es el nodo final. También se dice que i_1, i_2 son nodos adyacentes y que j incide en i_1, i_2 .

Una red donde no existen arcos en paralelo se conoce como una gráfica dirigida o digráfica y el conjunto A se identifica como un subconjunto de $N \times N$. El término gráfico se usa para una estructura similar donde los arcos no están orientados.

EJEMPLO 1.2

El gerente de programación de la General Motors, de la región oriental de México, está interesado en elaborar un plan semanal para enviar automóviles de su puerto de entrada a diversas distribuidoras regionales. Para elaborar el plan, ha recopilado datos sobre costos de transporte por automóvil en todo el país, necesidades mensuales de automóviles de cada distribuidor y llegadas mensuales de automóviles a cada puerto de entrada. Los automóviles se pueden enviar en forma directa a cada distribuidor o puede enviarse un conjunto de ellos a un distribuidor, descargar algunos, y enviar el resto a algún otro distribuidor. En el mapa de la figura 1.2 se muestran la localización de los distribuidores y de los puertos de entrada.



Figura 1.2

En la tabla 1 y en la figura 1.3 se muestran en pesos por automóvil, los costos de transporte entre los puertos de entrada, las ciudades de transbordo y las ciudades que son destinos finales. En los casos en los que no existe relación directa entre un par de ciudades no se muestran los costos.

TABLA 1

DE:CIUDAD	A:CIUDAD			
	México	Villahermosa	Puebla	Jalapa
Tampico	-	\$450	\$500	-
Veracruz	-	-	\$250	\$150
Puebla	\$150	\$400	-	-
Jalapa	\$200	-	\$200	-

USO DE LA REPRESENTACIÓN EN REDES

Este problema consiste en determinar la forma de enviar los automóviles de manera que se minimicen los costos, y al mismo tiempo, que se satisfagan las necesidades de todos los distribuidores.

En la gráfica o red siguiente (figura 1.3) los círculos representan a cada uno de los puertos de entrada y los distribuidores y las flechas, las relaciones entre ellos. Los números que están fuera de los círculos son las disponibilidades y las demandas según sea el caso de cada ciudad.

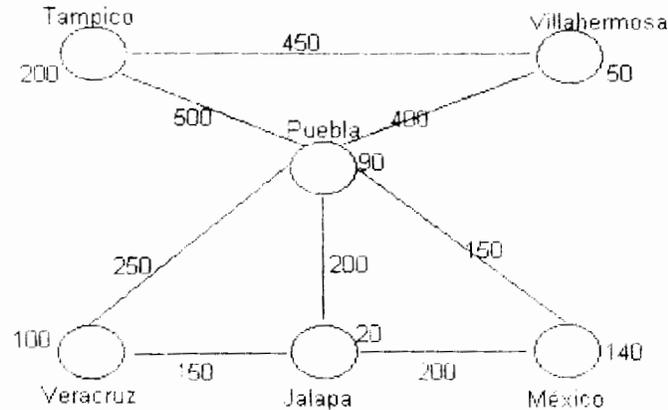


Figura 1.3

El gerente necesita determinar cuántos automóviles se deben enviar de Tampico y Veracruz a cada una de las otras ciudades, para que sean vendidos allí o transferidos a alguna otra ciudad. Se da cuenta de que no es difícil encontrar un plan de envíos que satisfaga las necesidades de todas las ciudades, como se muestra en la figura 1.4, pero... ¿será éste el plan más económico?

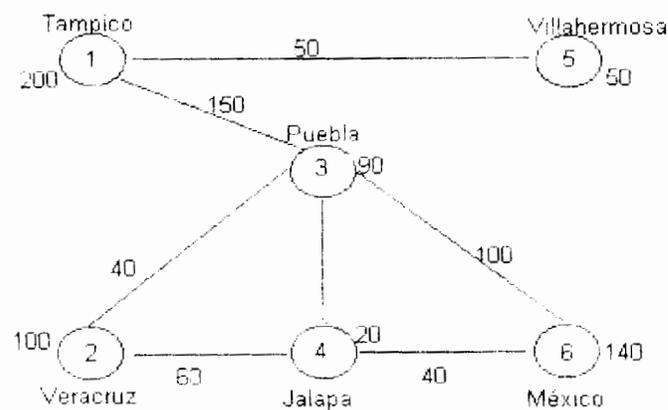


Figura 1.4

1.2 TIPOS BÁSICOS DE PROBLEMAS DE REDES

Consideramos cuatro tipos importantes de problemas de redes:

1. El problema de transbordo
2. El problema de la ruta más corta
3. El problema de flujo máximo
4. El problema de transporte

El problema de la General Motors es un ejemplo típico de problema de transbordo, puesto que se tienen disponibilidades y/o demandas y algunas ciudades al mismo tiempo pueden recibir y enviar automóviles a otras. Los otros tipos de problemas se ilustran más adelante. Algo común a estos es que los cuatro se pueden plantear en forma similar a los que se denominan problemas de flujo de costo mínimo; pero antes, es necesario analizar un poco la terminología de los problemas de redes.

1.3 REDES DE TRÁNSITO URBANO

El sistema de transporte consiste de redes que representan los modos disponibles (automóviles, autobuses, etc.) La descripción de la red es una abstracción de lo que actualmente existe en el campo, y como tal, no incluye todas las calles locales o las calles colectoras en el área. Se desarrolla una descripción de la red para representar los viajes en automóvil y camión, con una reseña separada para el transporte público, si se incluye este medio. Estas descripciones pueden abarcar la geometría del sistema de transporte.

La geometría de la red incluye la numeración de las intersecciones (llamadas nodos para propósitos de asignación). La numeración de los nodos permite identificar los segmentos entre ellos (llamados tramos). En las redes de transporte se pueden identificar grupos de tramos por los que pasan rutas específicas (llamadas líneas). Esta descripción geométrica de la red de transporte muestra todas las posibles vías por las que puedan realizarse los viajes entre puntos del área.

En la descripción de la red, se identifican los centroides de zona (centros de actividad); éstos se conectan a los nodos por medio de los tramos imaginarios denominados correctores de centroides. Los centroides se utilizan como los puntos en los cuales se efectúa la "carga" a la red.

Una vez que la red de transporte se ha descrito en términos de la manera como se pueden conectar los puntos, es necesario cuantificar la facilidad con la que se realizan estas conexiones.

Las velocidades de viaje y la capacidad de un tramo en una vía rápida, probablemente serían mayores que en una calle arterial; esta diferencia de nivel de servicio debe cuantificarse e incluirse como parte de la descripción del sistema de transporte.

Para la descripción de la red vehicular, se deben reunir puntos específicos para determinar el nivel de servicio en cada tramo. Estos requerimientos incluyen aspectos físicos tales como la longitud del tramo y el número de carriles, ya sea que el tipo de instalación bajo consideración sea vía rápida, arterial, etc. Así como la localización del tramo en el área urbana.

El costo de recorrido en un tramo específico se conoce como impedancia y tiene implícitas las mediciones de tiempo y costo.

1.4 PLANTEAMIENTO DE MODELOS DE REDES

Problemas de transbordo

Si un problema de redes se refiere a la minimización de los costos del flujo de algún producto entre nodos, en donde cada nodo puede ser un punto de abastecimiento, un punto de demanda, o ambos, entonces se considera que el problema de redes es un problema de transbordo.

El problema de la General Motors es un excelente ejemplo de un problema de transbordo. Existen tres tipos de nodos en un problema de transbordo. Si un nodo actúa al mismo tiempo como receptor y emisor de flujo se le denomina nodo de transbordo. En la figura 1.4 los nodos 3 y 4 son nodos de transbordo. El problema de transbordo es el más general de los problemas de redes, dado que cada nodo puede tener al mismo tiempo oferta y demanda y no existen restricciones sobre los flujos o sobre los tipos de nodos.

1.5 EL PROBLEMA DE LA RUTA MÁS CORTA

Si se define una red de manera que los coeficientes de cada arco sean no negativos (tal como medidas de distancia); entonces podríamos estar interesados en encontrar la ruta más corta entre dos nodos de la red. A este problema se le denomina problema de la ruta más corta.

Como ejemplo del problema de ruta más corta considere el siguiente:

EJEMPLO 1.3

La compañía de muebles "El Mueble Moderno" quiere transportar unos comedores de su planta en Naucalpan, Estado de México a una Distribuidora que se encuentra en Tlalpan, en el sur de la ciudad de México, en el menor tiempo posible.

Las rutas que enlazan estas dos instalaciones forman la red que se muestra en la figura 1.5

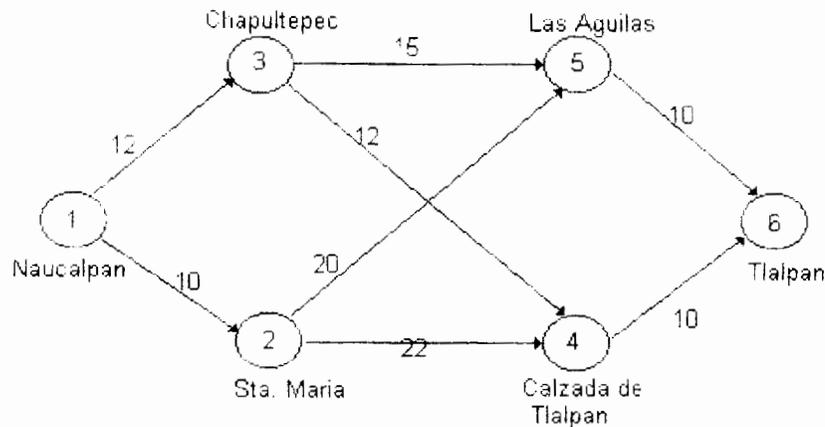


Figura 1.5

En donde las "distancias" son los tiempos de viaje del camión dados en minutos y considerando que se traslada a las 6 de la mañana y por lo tanto el tránsito es rápido, a una velocidad media de 50 kph.

Observe que con el objeto de plantear este problema en forma de problema de costo mínimo, puede elegirse en cualquier nodo sólo el camino a través de un arco. Lo anterior implica que es necesario tener los siguientes flujos en los arcos:

$$X_{ij} = \begin{cases} 1 & \text{si se viaja a través de la ruta entre la instalación } i \text{ y la instalación } j \\ 0 & \text{de otra manera} \end{cases}$$

Si se recorre la ruta (i,j), esto significa que no puede utilizarse ninguna otra ruta que parta de la instalación i.

Por ejemplo, si se viaja por la ruta de Naucalpan a Chapultepec entonces el flujo entre esos puntos será igual a 1 y X_{13} será cero.

Es posible satisfacer estas condiciones utilizando en la red un flujo imaginario de una unidad. Este flujo parte del nodo de salida u **origen** y llega a un nodo final o **terminal**. En otras palabras, existe un **suministro** de una unidad en el origen y una **demanda** de una unidad en el nodo terminal. En nuestro ejemplo, habría un suministro (u oferta) de una unidad en Naucalpan y una demanda de esa misma unidad en Tlalpan

La última pregunta que permanece pendiente es: ¿ Qué costos deben utilizarse en este problema de costo mínimo? Para responder a ésta pregunta observe que si $X_{ij} = 1$, entonces será necesario viajar de los nodos i a j . Si se denotan estas distancias mediante d_{ij} y se utiliza la ruta entre i y j , entonces el costo para esa ruta se convierte en $d_{ij} X_{ij}$. Dado que X_{ij} es cero o uno, el costo para cualquier ruta será d_{ij} o cero. Por esto, podemos utilizar las distancias, d_{ij} como los costos para el problema de flujo a costo mínimo.

De acuerdo con la figura 1.5, para salir de Naucalpan se puede ir por dos caminos, por Chapultepec o por Santa María, pero no por ambas. Para los puntos Chapultepec, Santa María, Las Águilas y Calzada de Tlalpan se requiere que el flujo que llega a esos nodos sea igual al flujo que sale, puesto que no existe demanda en ninguno de ellos. La restricción para Tlalpan exige llegar a ese punto ya sea por Las Águilas o por Calzada de Tlalpan, obligando a que la suma de los flujos sea igual a 1.

En un problema de ruta más corta puede haber arcos dirigidos y arcos no dirigidos. En nuestro ejemplo, sólo se tienen arcos dirigidos. Para el caso de arcos no dirigidos sería necesario tener una variable de i a j y otra de j a i . No sería difícil modificar el planteamiento para manejar las variables en ambas direcciones.

Finalmente, cabe mencionar que existen diversas variaciones del problema de la ruta más corta, y que el que se ha presentado aquí es sólo una de ellas. Las dos variaciones que existen son:

1. Encontrar la ruta más corta entre algún nodo y cada uno de los otros nodos de la red.
2. Encontrar la ruta más corta entre cualquier par de nodos de la red.

1.6 EL PROBLEMA DE FLUJO MÁXIMO

En el problema anterior estábamos interesados en los valores que se generan a través de cierto flujo que pasa por una red. Este valor puede estar dado en términos de dinero, distancia, tiempo o alguna otra medida. Existen problemas en los que el valor del flujo no es tan importante como la **cantidad de flujo** que pasa a través de la red. Los gasoductos, tráfico de automóviles y las líneas de transmisión de electricidad son ejemplos de esta situación. Los problemas en los que interesa

determinar el flujo máximo que pasa a través de una red se denominan **problemas de flujo máximo**.

Para estudiar este tipo de problemas, es necesario suponer que existen **restricciones de capacidad** en los arcos. Si no fuera así, el flujo máximo que pasaría a través de la red sería infinito. Considere el siguiente ejemplo:

EJEMPLO 1.4

Suponga que la Compañía Nacional de Subsistencias Populares (CONASUPO) tiene un programa anual de costalera. Esta se compra de dos fábricas, una en Mérida con capacidad de producción máxima de 10 millones de costales al año, y otra en Saltillo con capacidad de producción máxima de 7 millones de costales al año. Los excedentes en la fábrica de Mérida pueden transferirse a la planta de Saltillo.

La disponibilidad de transporte entre las dos fábricas permite un máximo de 8 millones de costales por año. Hay tres centros almacenadores: en la ciudad de México, Guadalajara y Oaxaca. la tabla 2 proporciona la capacidad máxima anual de transporte de las fábricas a los centros almacenadores.

TABLA 2

De: Ciudad	A: Ciudad		
	México	Guadalajara	Oaxaca
Saltillo	4	8	-
Mérida	3	2	3

Los excedentes de Guadalajara y Oaxaca pueden transferirse a la ciudad de México. la capacidad máxima anual es de 3 y 4 millones de costales respectivamente.

Una vez en los centros almacenadores, los costales se entregan a los ejidatarios de la región. La capacidad máxima anual de entrega es de 4 millones en la región almacenadora de Guadalajara, 7 millones en la región del Distrito Federal y 5 millones en la región de Oaxaca.

La pregunta es ¿Cuál es el flujo máximo anual de costales nuevos que pueden circular en este sistema?. El problema se puede representar gráficamente en la red de la figura 1.6

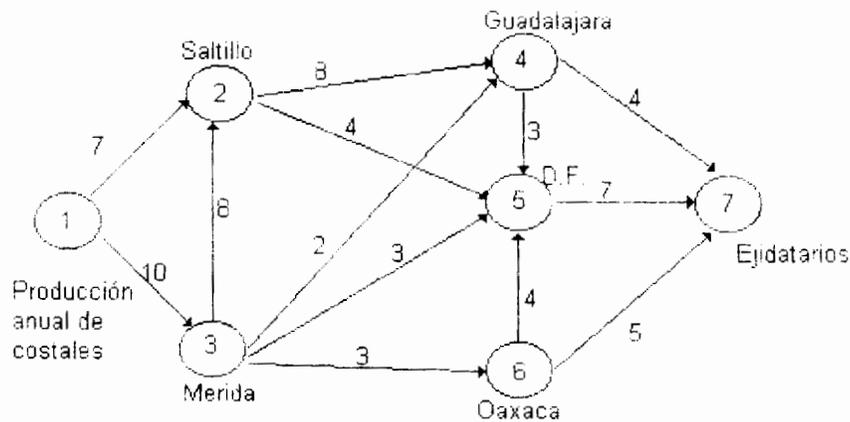


Figura 1.6

Se puede observar en la red que los números en los arcos se refieren a las capacidades máximas de transporte, no se pusieron las capacidades mínimas por ser éstas todas iguales a cero.

1.7 EL ÁRBOL DE EXPANSIÓN MÍNIMA

Antes de especificar en que consiste este problema cabe mencionar lo que en redes se entiende por un árbol. Un **árbol** es un subconjunto de los arcos de la red original que conecta a todos los nodos sin formar ningún circuito.

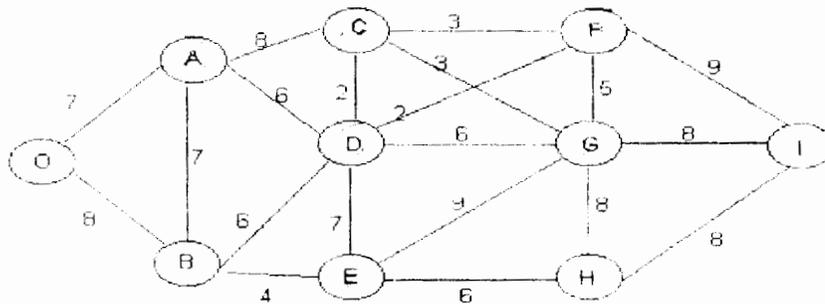
En este problema se conocen los costos o distancias entre diferentes nodos en una red. Sin embargo, los arcos **no** se especifican, y lo que se trata de encontrar es un árbol que comunique a todos los nodos de la red, pero cuyo costo o distancia total sea mínima.

Este tipo de problemas se ubica en las redes de comunicación eléctrica, telefónica, telegráfica, carretera, ferrocarrilera, aérea, marítima, etc., donde los nodos representan, por ejemplo, puntos de consumo eléctrico, teléfonos, telégrafos, terminales de autobuses, trenes, aeropuertos, puertos marítimos, etc., y los arcos podrían ser, las líneas de alta tensión eléctrica, líneas telefónicas y telegráficas, carreteras y vías de ferrocarril, rutas aéreas y marítimas entre otras. A continuación se presenta un ejemplo de este caso.

EJEMPLO 1.5

Suponga que en la red que se muestra en la figura 1.7, los nodos son centros de consumo eléctrico, y los números en los arcos son distancias en kilómetros. Se trata de encontrar el árbol que, con una longitud total mínima, comunica a todos los nodos. Como el costo de tendido de cable eléctrico es proporcional a la distancia, se habrá encontrado, con la distancia mínima, también el costo mínimo.

Figura 1.7



Después de usar algún paquete de cómputo el árbol mínimo de comunicación, que no necesariamente es el único, se muestra en la figura 1.8

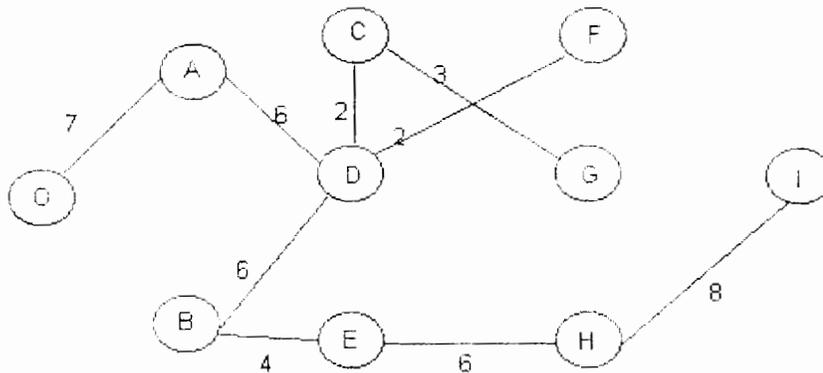


Figura 1.8

El siguiente ejemplo muestra los tres casos de problemas de redes que hemos visto hasta aquí.

EJEMPLO 1.6

En fecha reciente se ha reservado el parque La Marquesa para pasear y acampar. No se permite la entrada de automóviles al parque, pero existe un sistema de caminos angostos para tranvías y jeeps conducidos por los guardabosques.

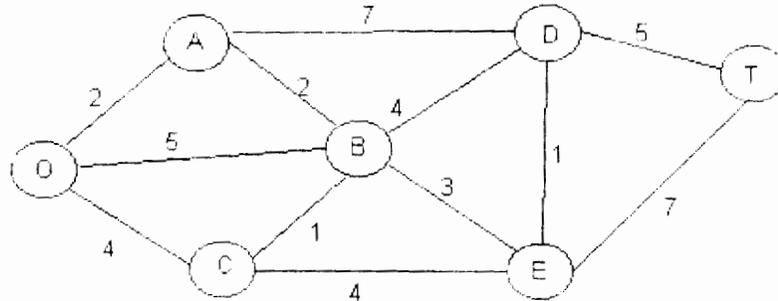


Figura 1.9

En la figura 1.9 se muestra este sistema de caminos (sin las curvas), en donde O es la localización de la entrada al parque; las otras letras designan la localización de estaciones de guardabosques (y otras instalaciones). Los números dan las distancias de estos caminos sinuosos en kilómetros.

El parque contiene un paisaje maravilloso en la estación T. Se usa un número pequeño de tranvías para transportar visitantes de la entrada del parque a la estación T, y de regreso, para quienes desean contemplar este paisaje sin tener que caminar.

La dirección del parque está encarando en este momento tres problemas. Uno es determinar cuál ruta de la entrada del parque a la estación T tiene la menor distancia total, para la operación de los tranvías.

Un segundo problema se refiere a la instalación de líneas telefónicas debajo de los caminos, para establecer comunicación de este tipo entre todas las estaciones, incluyendo la entrada al parque. Puesto que la instalación es cara y, a la vez, perturba el medio ambiente natural, se instalarán las líneas sólo debajo del número suficiente de caminos para proveer cierta conexión entre todo par de estaciones. La cuestión es dónde deben colocarse las líneas para realizar esto con un número total mínimo de kilómetros de línea instalada.

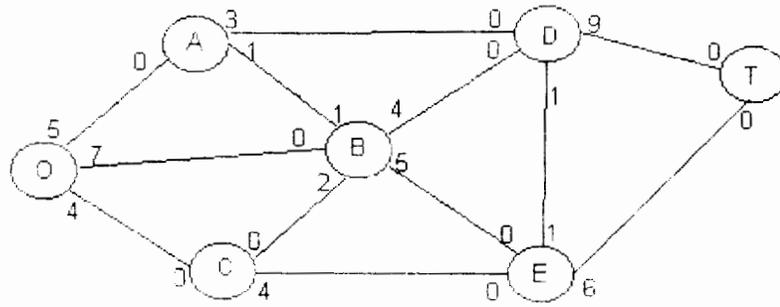


Figura 1.10

El tercer problema es que, durante la temporada pico, mas personas desean el viaje en tranvía desde la entrada del parque a la estación T que las que pueden ser acomodadas. Con el fin de evitar la perturbación indebida de la ecología y la vida salvaje de la región, se han racionado estrictamente el número de viajes al día del tranvía, que pueden llevarse a cabo sobre cada uno de los caminos, estos límites son diferentes para los diferentes caminos como se muestra en la figura 1.10. Por lo tanto durante la temporada pico, podrían seguirse diversas rutas, sin importar la distancia, para incrementar el número de viajes por tranvía que pueden efectuarse diariamente. La cuestión consiste en trazar las rutas de los diversos viajes para maximizar el número de viajes que pueden efectuarse diariamente, sin violar los límites de cada camino. Para resolver el problema de ruta más corta considere la siguiente tabla 3:

TABLA 3

n	Nodos resueltos conectados a nodos no resueltos	Su nodo no resuelto conectado más próximo	Distancia total involucrada	n_ésimo nodo más cercano	Su distancia mínima	Su ultima conexión
1	O	A	2	A	2	OA
2	O A	C B	4 2 + 2 = 4	C B	4 4	OC AB
3	A B C	D E E	2 + 7 = 9 4 + 3 = 7 4 + 4 = 8	E	7	BE
4	A B E	D D D	2 + 7 = 9 4 + 4 = 8 7 + 1 = 8	D D	8 8	BE ED
5	D E	T T	8 + 5 = 13 7 + 7 = 14	T	13	DT

Ahora puede recorrerse hacia atrás la ruta más corta, del destino al origen, a través de la última columna de la tabla, como a $T \rightarrow D \rightarrow E \rightarrow B \rightarrow A \rightarrow O$ o bien, $T \rightarrow D \rightarrow B \rightarrow A \rightarrow O$. Por lo tanto, se han identificado las dos alternativas para la ruta más corta del origen al destino como $O \rightarrow A \rightarrow B \rightarrow E \rightarrow D \rightarrow T$ y $O \rightarrow A \rightarrow B \rightarrow D \rightarrow T$, con una distancia total de 13 millas sobre cualquiera de las dos.

Como se podrá observar los problemas 1, 2 y 3 son respectivamente problema de ruta más corta, problema de árbol de expansión mínima y problema de flujo máximo, dicho problema se resolverá posteriormente.

EJEMPLO 1.7

En este ejemplo se muestra la forma en que se puede resolver un problema práctico de tránsito. Para concretar más, empezamos con un mapa que muestra una parte del centro de la ciudad de México.

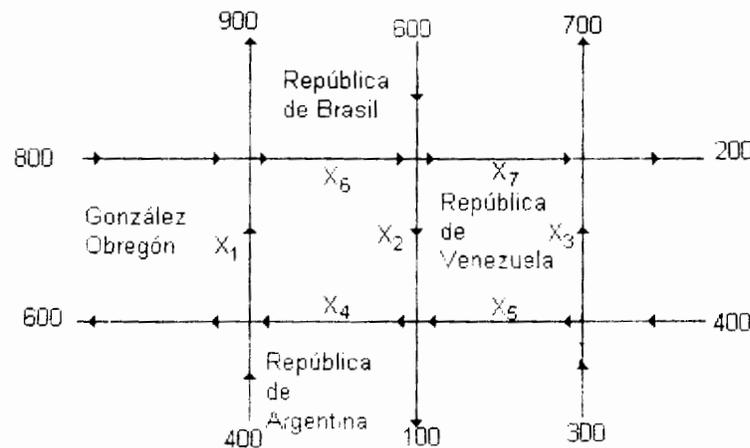


Figura 1.11

En el mapa se indicó el flujo de tránsito que entra o sale a cada calle, en unidades de vehículos por hora (vph). Ya que el flujo de tránsito varía considerablemente durante el día, supondremos que los números mostrados representan el flujo de tránsito promedio a la hora de máximo flujo que se da aproximadamente entre las 3 y las 5 de la tarde.

Suponga ahora que un grupo político está planeando una manifestación en República de Argentina, entre República de Venezuela y González Obregón a las 4 de la tarde del miércoles. La policía puede, hasta cierto punto, controlar el flujo de tránsito reajustando los semáforos, colocando policías en los cruces clave, o cerrando la calle crítica al tránsito de vehículos. Si se disminuye el tránsito por

República de Argentina, aumentará el de las calles adyacentes. La cuestión es minimizar el tránsito por República de Argentina (entre República de Venezuela y González Obregón) sin ocasionar congestionamientos en las otras calles. Para resolver este problema de minimización le agregamos marcas a nuestro mapa, ver figura 1.11.

Aquí se han marcado las seis intersecciones A hasta F y se ha denotado el flujo de tránsito entre las intersecciones adyacentes por las variables x_1 hasta x_7 . El problema consiste ahora en minimizar x_4 sujeta a las restricciones del problema.

Para encontrar estas restricciones veamos, por ejemplo, la intersección B. El tránsito que fluye a la intersección B es, según el mapa, $x_2 + x_5$; mientras que el tránsito que sale de la intersección B es $x_4 + 100$. Suponiendo que el tránsito no se acumula en la intersección B, el tránsito de "entrada" debe ser igual al tránsito de "salida". Así se obtiene la ecuación

$$x_2 + x_5 = x_4 + 100$$

o bien

$$x_2 - x_4 + x_5 = 100$$

A partir de este análisis en cada intersección, se obtiene el siguiente sistema de seis ecuaciones en siete incógnitas:

en A	x_1	$-x_4$		$= -200$
en B	x_2	$-x_4 + x_5$		$= 100$
en C	x_3	$+x_5$		$= 700$
en D	x_1		$-x_6$	$= 100$
en E	x_2		$-x_6 + x_7$	$= 600$
en F	x_3		$+x_7$	$= 900$

1.8 PLANTEAMIENTO DE MODELOS DE REDES COMO PROBLEMAS DE PROGRAMACIÓN LINEAL

Consideraremos cuatro tipos de problemas de redes: problemas de transbordo, de ruta más corta, de flujo máximo y de transporte. Analizaremos también el planteamiento de problemas de PERT/CPM y problemas de asignación de personal como problemas de redes. El tema más importante de esta sección es que *todos* estos problemas de flujo pueden plantearse de manera similar como problemas de flujo de costo mínimo y pueden resolverse como problemas de programación lineal. Para varios de estos problemas existen algoritmos especiales que pueden ser más eficientes que el método simplex de programación lineal, pero debido a que se hará hincapié en los planteamientos, aquí no se analizan esos algoritmos.

Enseguida se presentan y analizan cada uno de los problemas de redes y después se plantea cada uno de ellos.

PROBLEMA DE TRANSBORDO

Como segundo ejemplo de un problema de transbordo, considere el siguiente: la Compañía Petrolera Alianza tiene un solo campo petrolero desde donde envía todo el petróleo, a través de un oleoducto, a uno de dos centro de embarque, en donde se almacena en buques tanque para su envío a refinerías de Estados Unidos.

La oferta diaria en el campo es de 2,000 barriles. Deben considerarse los costos del oleoducto, los costos de embarque y las cantidades de petróleo que pueden enviarse a través de los oleoductos. Los costos del oleoducto y las capacidades diarias de éste se muestran en la tabla 4.

TABLA 4
Compañía Petrolera Alianza. Costos y capacidades de los ductos

Instalación de envío	Costo por barril	Capacidad del oleoducto (en barriles)
1	\$0.20	1000
2	\$0.15	500

En la tabla 5 se presentan los costos de embarque de cada estación de embarque a cada refinería y las demandas diarias de las refinerías. En la figura 1.12 se plantea este problema en forma de red.

TABLA 5
Compañía Petrolera Alianza. Costos de transporte y demandas.

Refinería Número de ubicación	Costo de transporte por barril		Demanda diaria
	Del centro 1	Del centro 2	
1	\$0.10	\$0.15	600
2	\$0.20	\$0.25	800

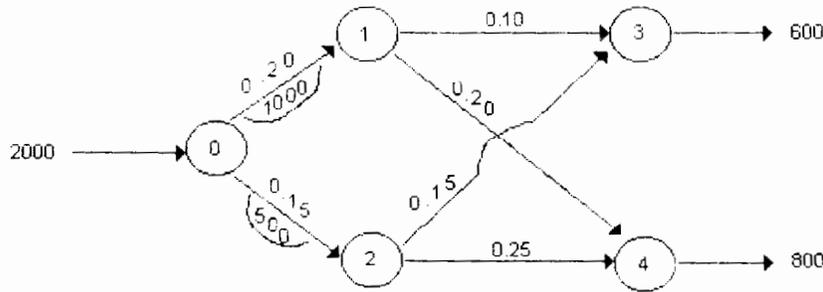


Figura 1.12. Red de la Compañía Petrolera Alianza

En el planteamiento de red, el nodo 0 es el campo petrolero, los nodos 1 y 2 son los nodos de embarque y los nodos 3 y 4 representan las refineries de Nuevo León e Hidalgo respectivamente. Las capacidades de los oleoductos se muestran en los arcos encerradas en semicírculos y los costos se ilustran encima de los arcos.

Este problema puede plantearse en forma de programación lineal si

$$x_{ij} = \text{barriles enviados del nodo } i \text{ al nodo } j$$

Entonces, el problema es

$$\text{Min } Z = 0.2x_{01} + 0.15x_{02} + 0.10x_{13} + 0.2x_{14} + 0.15x_{23} + 0.25x_{24}$$

SUJETO A:

$$\begin{array}{rcccccccc} x_{01} & + & x_{02} & & & & & & \leq 2000 \\ x_{01} & & & - & x_{13} & - & x_{14} & & = 0 \\ & & x_{02} & & & & - & x_{23} & - & x_{24} & = 0 \\ & & & & x_{13} & & & + & x_{23} & & = 600 \\ & & & & & & x_{14} & & & + & x_{24} & = 800 \\ x_{01} & & & & & & & & & & \leq 1000 \\ & & x_{02} & & & & & & & & \leq 500 \\ & & & & & & & & & & \text{todas las } x_{ij} \geq 0 \end{array}$$

Este problema difiere un poco del de General Motors ya que existen capacidades en algunos de los arcos.

Estas capacidades resultan ser restricciones de cota superior. Las restricciones de la oferta aparecen como una restricción de menor o igual, en vez de ser una restricción de igualdad, puesto que la demanda total es inferior a la oferta disponible.

PROBLEMA DE LA RUTA MÁS CORTA

Si se define una red de manera que los coeficientes de cada arco sean no negativos (tal como medidas de distancia), entonces podríamos estar interesados en encontrar la ruta más corta entre dos nodos de la red. A este problema se le denomina *problema de la ruta más corta*.

Como ejemplo del problema de ruta más corta, consideremos el problema de viajar en automóvil desde Nueva Orleans a Atlanta en el tiempo más corto, dentro de lo legal. Las carreteras que enlazan estas dos ciudades forman la red que se muestra en la figura 1.13 en donde las "distancias" son los tiempos de viaje en automóvil dados en minutos.

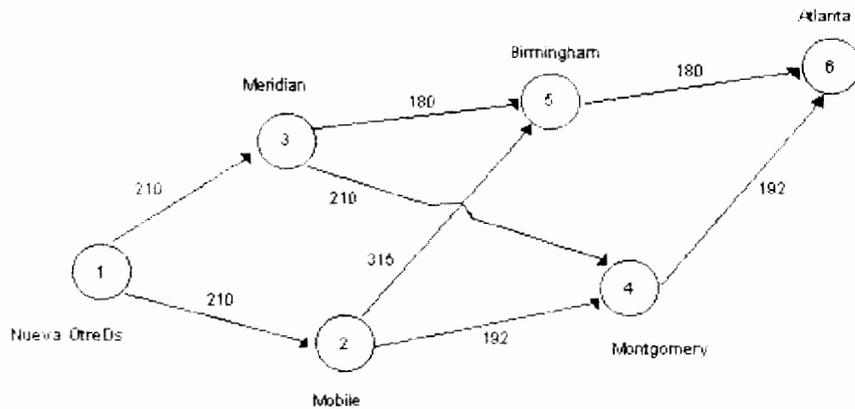


Figura 1.13. Rutas del manejo de Nueva Orleans a Atlanta.

Observe que con el objeto de plantear este problema en forma de problema de costo mínimo, puede elegirse en cualquier nodo sólo el camino a través de un arco.

Lo anterior implica que es necesario tener los siguientes flujos en los arcos:

$$x_{ij} = \begin{cases} 1 & \text{si se viaja a través de la carretera entre la ciudad } i \text{ y la ciudad } j \\ 0 & \text{de otra manera} \end{cases}$$

Atlanta exige llegar a esta ciudad ya sea de Birmingham o de Montgomery, obligando a que la suma de los flujos sea igual a 1.

En un problema de ruta más corta puede haber arcos dirigidos y arcos no dirigidos. En nuestro ejemplo sólo se tienen arcos dirigidos. Para el caso de arcos no dirigidos, sería necesario tener una variable de i a j y otra de j a i . No sería difícil modificar el planteamiento para manejar las variables en ambas direcciones.

Puesto que ya hemos formulado el problema de la ruta más corta como problema de flujo de costo mínimo, podemos entonces utilizar un programa de computación especial de programación lineal, para resolver el problema.

El lector debe percatarse de que existen diversas variaciones del problema de la ruta más corta, y que la que se ha presentado aquí es sólo una de ellas. Las dos variaciones que existen son: (1) Encontrar la ruta más corta entre algún nodo y cada uno de los otros nodos de la red; (2) Encontrar la ruta más corta entre cualquier par de nodos de la red. Al igual que con la versión que se presenta aquí, es posible formular como problemas de flujo de costo mínimo estos otros problemas de ruta más corta. Existen también algoritmos de aplicación especial para las otras versiones.

PROBLEMA DE PERT/CPM

El objeto de estos modelos es encontrar la secuencia de actividades o tareas que requieren del *mayor* tiempo para su terminación. A esta secuencia de trabajo se le denomina *ruta crítica*. Como ejemplo de un análisis de red de actividades, considérese la siguiente situación.

La Facultad de Administración de la UAM está planeando una conferencia para ejecutivos de negocios. En la tabla 6, se listan las actividades que deben llevarse a cabo antes de la conferencia y también se presentan los tiempos estimados para terminar cada una de las labores.

TABLA 6

Actividad	Descripción	Tiempo Estimado. (semanas)	Actividades precedentes
A	Elaborar el programa	3	-
B	Recopilar lista de asistentes	5	-
C	Contactar a oradores para que asistan	9	A
D	Elaborar folleto para el programa	2	B
E	Hacer arreglos físicos	4	A
F	Enviar los folletos	4	D
G	Preparar programa de las conferencias	3	E
H	Detalles de último minuto	1	C, G, F

Este problema de planeación también se muestra en un formato de redes (figura 1.14) y se ilustran en forma directa en los arcos los símbolos que representan a las actividades y el tiempo estimado para terminar la actividad; también se enumeraron los nodos.

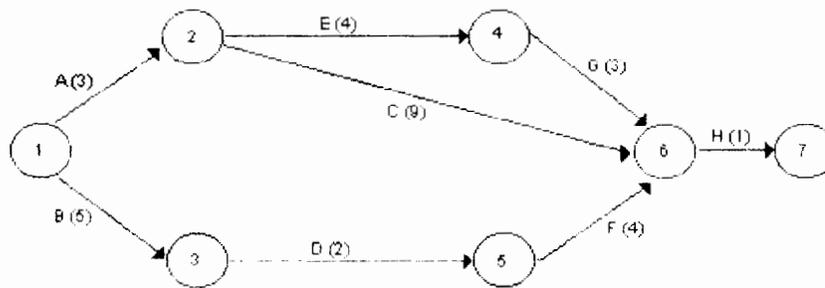


Figura 1.14. Red de planeación de conferencias.

Dado que estamos intentando descubrir cuál es la ruta más larga en esta red, tenemos la situación opuesta al problema de la ruta más corta. No obstante, este problema puede planearse en forma similar. En este caso, los tiempos estimados para cada arco se utilizan como "distancias", que van a maximizarse en vez de minimizarse. De nuevo se incluye un suministro imaginario de una unidad de flujo en el primer nodo y una demanda imaginaria de una unidad en el último. El planteamiento de programación lineal de este problema es:

En esta figura se muestra también una cantidad desconocida de flujo, f , que entra en el gasoducto en el nodo 1 (el campo de gas) y que sale del gasoducto en el nodo 5 (la terminal de Chiapas). Utilizando este flujo, f , puede plantearse este problema de la siguiente manera:

MAXIMIZAR: f

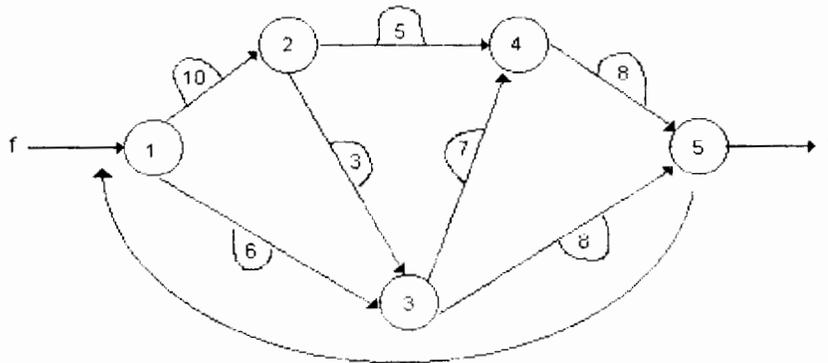
SUJETO A:

$$\begin{array}{rcccccccc}
 x_{12} & + & x_{13} & & & & & & = & f \\
 x_{12} & & & - & x_{23} & - & x_{24} & & & = & 0 \\
 & & x_{13} & + & x_{23} & & & - & x_{34} & - & x_{35} & = & 0 \\
 & & & & & & x_{24} & + & x_{34} & & & - & x_{45} & = & 0 \\
 & & & & & & & & & & x_{35} & + & x_{45} & = & f \\
 x_{12} \leq 10, & x_{13} \leq 6, & x_{23} \leq 3, & x_{24} \leq 5, & x_{34} \leq 7, & x_{35} \leq 8, & x_{45} \leq 8 \\
 x_{ij} \geq 0 & \text{para toda } i & \text{y toda } j
 \end{array}$$

Este planteamiento no se ajusta a nuestra formulación estándar de programación lineal de flujo de costo mínimo, puesto que el flujo que se desconoce, f , aparece tanto como variable de la función objetivo, como en forma de valor de lado derecho de las restricciones. Si se plantea de esta manera no es posible utilizar el método de flujo de costo mínimo para resolverlo.

Para evitar esta dificultad, en primer lugar se elimina el flujo f y se introduce un arco artificial o ficticio que conecta los nodos 5 y 1. El objetivo se convierte entonces en maximizar el flujo que pasa por este arco ficticio. Maximizar el flujo que regresa del nodo 5 al nodo 1, por un arco ficticio que no tiene capacidad, dará la cantidad de flujo que va del nodo 1 al nodo 5 a lo largo de la red de capacidades. En la figura 1.16 se muestra la red de gasoductos, incluyendo el arco de regreso.

Figura 1.16. Red modificada de los gasoductos



Ahora, utilizando esta nueva red de gasoductos, se tiene un planteamiento modificado, en donde el objetivo es maximizar x_{51} :

MAXIMIZAR: x_{51}

SUJETO A :

$$\begin{array}{rcccccccc}
 x_{51} & - & x_{12} & - & x_{13} & & & & = & 0 \\
 & & x_{12} & & & - & x_{23} & - & x_{24} & = & 0 \\
 & & & & x_{13} & + & x_{23} & & - & x_{34} & - & x_{35} & = & 0 \\
 & & & & & & & & x_{24} & + & x_{34} & & - & x_{45} & = & 0 \\
 - & x_{51} & & & & & & & & & + & x_{35} & + & x_{45} & = & 0 \\
 x_{12} \leq 10, & x_{13} \leq 6, & x_{23} \leq 3, & x_{24} \leq 5, & x_{34} \leq 7, & x_{35} \leq 8, & x_{45} \leq 8 \\
 x_{ij} \geq 0 & \text{para toda } i & \text{y toda } j
 \end{array}$$

Ahora queda planteado el problema de flujo máximo en forma estándar de programación lineal de redes, excepto que no existen ofertas ni demandas. Los problemas de este tipo se denominan *redes circulares*.

PROBLEMA DE TRANSPORTE

El denominado *problema de transporte* es un caso especial del problema de transbordo, en el que todos los nodos son o fuentes (nodos de oferta) o destinos (nodos de demanda). En un problema de transporte no existen nodos de transbordo. Dado que es posible dividir el problema de transporte en dos conjuntos diferentes de nodos, es un problema de red *bipartita*. Para continuar analizando los problemas de transporte utilizaremos de nuevo un ejemplo.

Considere el caso de la Cervecería Modelo. Esta empresa elabora un cerveza que se distribuye a nivel nacional a partir de dos fábricas de cerveza, una en cada una de las dos costas de México (en Jalisco y Veracruz). La cerveza se envía a cuatro mayoristas que se encargan de la distribución subsecuente (en Chihuahua, Guerrero, Nuevo León y Yucatán), por lo que a la Modelo le ocupa sólo la distribución a los mayoristas. Los costos de distribución, por conjuntos de 100 cajas que se envían a cada mayorista, se presentan en la tabla 7 junto con la oferta mensual de cada fábrica y la demanda mensual de cada mayorista.

TABLA 7
Costos de distribución para la Modelo

Fábrica de Cerveza	Yucatán	N. León	Chih	Gro.	Oferta (en Cientos de cajas)
Jalisco	\$21	\$15	\$18	\$9	550
Veracruz	\$10	\$14	\$16	\$23	650
Demanda (en cientos de cajas)	200	250	400	350	

Si se representa este problema en forma de red, aparecerá según se muestra en la figura 1.17. Dado que el flujo de un nodo de oferta sólo va a un nodo de demanda, se codificará el sistema común de numeración de los nodos para numerar los nodos de oferta en forma independiente de los nodos de demanda.

Observe que esto dará como resultado que haya nodos que se denotan como x_{11} , pero dado que los nodos de oferta están conectados sólo a los nodos de demanda, no existe ambigüedad con respecto a qué arco identifica a x_{11} .

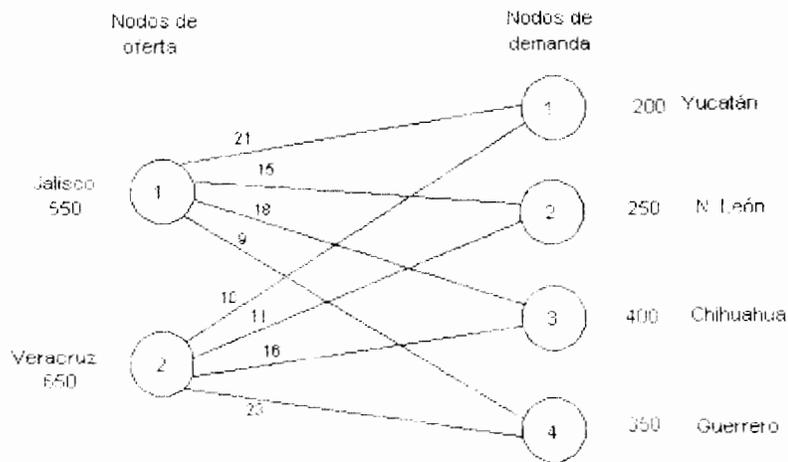


Figura 1.17. Red de transporte para la Modelo

El planteamiento de programación lineal del problema de transporte es muy similar al de los problemas de transbordo que se analizaron al inicio del capítulo.

De Jalisco a Yucatán	0
De Jalisco a Nuevo León	20 (cientos de cajas)
De Jalisco a Chihuahua	0
De Jalisco a Guerrero	350 (cientos de cajas)
De Veracruz a Yucatán	200 (cientos de cajas)
De Veracruz a Nuevo León	50 (cientos de cajas)
De Veracruz a Chihuahua	400 (cientos de cajas)
De Veracruz a Guerrero	0

No es necesario utilizar un paquete de programación lineal de redes para resolver un problema de transporte tan pequeño como ejemplo. Para ello se ha adaptado una versión del método simplex que se denomina **método del cruce del arroyo** (o de la piedra de paso: stepping stone) y se analiza en detalle en otro capítulo.

PROBLEMA DE ASIGNACIÓN

El ejemplo final de problemas de redes que consideramos aquí es el *problema de asignación de personal* (o en forma más simple, el problema de asignación). Este problema se refiere a la asignación de agentes a trabajos en forma tal que se minimicen los costos de esa asignación. En calidad de ejemplo, considérese la situación que debe manejar el entrenador Ricardo Martínez del Equipo de Natación de la Acuática Nelson Vargas.

El entrenador está intentando organizar el mejor equipo de relevo de mujeres para los 200 metros. Tiene 4 muchachas en el equipo: Luisa, Julieta, Maria y Rocío. Luisa sólo nada el estilo libre, por lo que no hay problema respecto de esa parte del equipo. Cada una de las otras tres chicas puede nadar en cualquiera de los otros tres estilos: mariposa, dorso y pecho. Entonces la cuestión es cual de ellas debe nadar en qué estilo. Los tiempos de cada una de las nadadoras en cada uno de los estilos se muestran en la tabla 8.

TABLA 8
Tiempos por estilo del equipo de la Nelson Vargas

Nadadora	Estilo		
	Mariposa	Dorso	Pecho
Julieta	33 seg.	35 seg.	37 seg.
Maria	33 seg.	37 seg.	37 seg.
Rocío	33 seg.	36 seg.	39 seg.

Este problema puede plantearse en forma de red considerando tres nodos fuente, cada uno con un suministro de una unidad, y tres nodos destino, cada uno con una demanda de una unidad. Los nodos fuente representan los agentes disponibles (nadadoras) y los nodos destino, los trabajos (estilos). La red para este ejemplo se muestra en la figura 1.18.

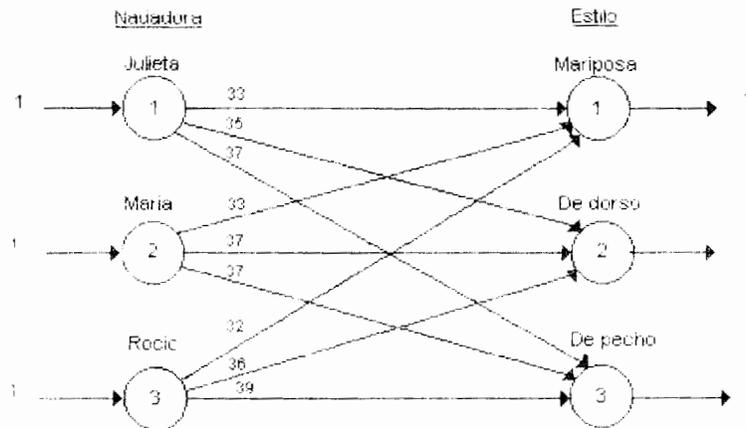


Figura 1.18. Red de asignación para el equipo de natación de la Nelson Vargas.

Es fácil de observar que el problema de asignación no es otra cosa que un caso especial de red de transporte en el que todas las ofertas o suministros y las demandas son iguales a 1 y en la que el número de nodos de oferta es igual al número de nodos de demanda. Puesto que esto es cierto, el procedimiento de programación lineal es muy similar:

$$\text{Min } Z = 33x_{11} + 35x_{12} + 37x_{13} + 33x_{21} + 37x_{22} + 37x_{23} + 32x_{31} + 36x_{32} + 39x_{33}$$

SUJETO A :

$$\begin{array}{rccccccc} x_{11} & + & x_{12} & + & x_{13} & & & & = & 1 \\ & & & & & x_{21} & + & x_{22} & + & x_{23} & = & 1 \\ & & & & & & & & & & x_{31} & + & x_{32} & + & x_{33} & = & 1 \\ x_{11} & & & & & + & x_{21} & & & & + & x_{31} & & & & = & 1 \\ & & x_{12} & & & & & + & x_{22} & & & & + & x_{32} & & = & 1 \\ & & & & & & & & & & & & & & & x_{ij} \geq 0 & \text{para toda } i \text{ y toda } j \end{array}$$

Se ha omitido la última restricción de este planteamiento puesto que sería redundante, al igual que en el planteamiento de transporte. Una característica especial de este problema es que $x_{ij} = 0$ ó 1 .

En otras palabras:

$$x_{ij} = \begin{cases} 1 & \text{si a la nadadora } i \text{ se le asigna el trabajo (estilo) } j \\ 0 & \text{de otra manera} \end{cases}$$

Estamos seguros de obtener este resultado porque en cada nodo de oferta existe sólo una unidad que pueda enviarse a lo largo de varios arcos.

1.9 NOTAS HISTÓRICAS

El estudio de los modelos de redes de flujo antecede al desarrollo de la programación lineal. Los primeros estudios en esta área los realizaron Kantorovich (1939), Hitchcock (1941) y Koopmans (1947) considerando el problema de transporte como un caso especial del problema de flujo a costo mínimo. Esos estudios proporcionaron una nueva luz en la estructura del problema y dieron nuevos métodos y algoritmos. El interés por las redes de flujo creció con el advenimiento del método SIMPLEX desarrollado por Dantzig en 1947, quien incluso especializó este algoritmo para el problema de transporte (ver Dantzig, 1951).

Durante los años 50's, las investigaciones comenzaron a mostrara un creciente interés en el problema de flujo a costo mínimo y sus especializaciones: el problema de la ruta más corta, el problema de flujo máximo y el problema de asignación; esto se debió principalmente a la importante aplicación de estos modelos en la vida real. Algunos autores desarrollaron algoritmos para resolver estos problemas. Dantzig, Ford y Fulkerson fueron pioneros en este aspecto. Mientras Dantzig se enfocó en los métodos basados en el SIPLEX, Ford y Fulkerson desarrollaron algoritmos combinatorios primales duales. Los libros de Dantzig (1962) y de Ford y Fulkerson (1962) están presentes a través de la discusión de estas primeras contribuciones.

En los años siguientes, los problemas de flujo en red y sus generalizaciones emergieron como tópicos de gran interés en cientos de artículos y numerosos textos, así como en libros de referencia.

CAPÍTULO 2 CONCEPTOS BÁSICOS DE REDES

2.1 INTRODUCCIÓN

En este capítulo se presentan conceptos fundamentales para poder desarrollar diferentes algoritmos de redes. La representación matricial de una red que nos permite encontrar árboles de expansión y que además muestra la relación entre el teorema fundamental del Algebra Lineal y las estructuras de red. Asimismo, la dualidad entre potenciales y tensión y su relación con la programación lineal, que nos introducirá a otro concepto dual: el de trayectorias y cortes que se verá en el capítulo 3.

2.2 REPRESENTACIÓN MATRICIAL DE UNA RED

¿Cómo podemos representar numéricamente a una red?

Sea G una red, la matriz de incidencias nodos - arcos E es una matriz de $m \times n$ donde m es el número de nodos y n es el número de arcos cuyos elementos $e(i,j)$ son:

Una forma de recordar los signos de esta definición es que "una flecha siempre va

$$e(i, j) = \begin{cases} +1 & \text{si el arco } j \text{ sale del nodo } i \\ -1 & \text{si el arco } j \text{ entra al nodo } i \\ 0 & \text{en otro caso} \end{cases}$$

desde donde está hasta donde no está".

EJEMPLO 2.1

Construya la matriz nodos - arcos asociada a la red de la figura 2.1.

$$E = \begin{matrix} & \begin{matrix} J_1 & J_2 & J_3 & J_4 \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \end{matrix} & \begin{bmatrix} 1 & 1 & 0 & 1 \\ -1 & 0 & 1 & 0 \\ 0 & -1 & -1 & -1 \end{bmatrix} \end{matrix}$$

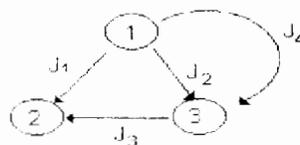


Figura 2.1

Si existen rizados la matriz nodos - arcos no sirve.

J_1, J_2, J_3 son linealmente dependientes ya que $J_1 + J_3 - J_2 = 0$.

¿Que significado tiene esto en la red?

Si se cambia de sentido un arco significa que se multiplica por -1 la columna, entonces en este caso que los vectores sean linealmente dependientes significa que se forma un circuito.

Los árboles son aquellas estructuras que no tienen circuitos, por lo tanto son linealmente independientes.

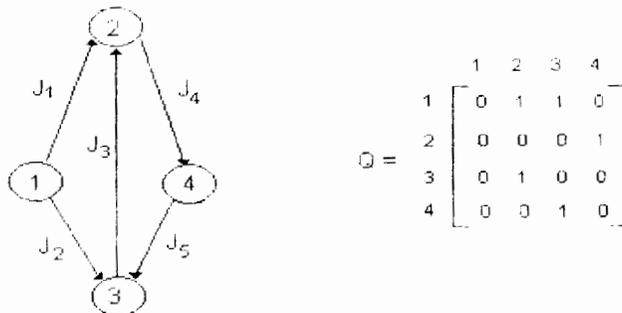
Además de la matriz de incidencia (o función de incidencia). Existe una matriz de adyacencia, Q de orden $m \times m$ definida en $N \times N$ de la siguiente manera:

$$q(i_1, i_2) = \begin{cases} 1 & \text{si existe un arco que parte de } i_1 \text{ y llegue a } i_2 \\ 0 & \text{en caso contrario} \end{cases}$$

Por supuesto se tiene que G es una digráfica, esto es $A \subset N \times N$

EJEMPLO 2.2

Encuentre la matriz de adyacencia nodos-nodos de la siguiente red:



$$Q = \begin{matrix} & \begin{matrix} 1 & 2 & 3 & 4 \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \end{matrix} & \begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{matrix}$$

Figura 2.2

Una matriz de $m \times m$ puede ser la matriz adyacente de una red con m nodos si consiste sólo de 0 y 1 y si no contiene rizos debe tener ceros en la diagonal principal y se define como se muestra en la figura 2.2.

El grado exterior de un nodo i es el número de arcos que salen de i por ejemplo:

$$\begin{array}{ll} g_e(1) = 2 & g_e(3) = 1 \\ g_e(2) = 1 & g_e(4) = 1 \end{array}$$

El grado interior de un nodo i es el número de arcos que entran al nodo i .

$$\begin{array}{l} g_i(1) = 0 \\ g_i(2) = 2 \\ g_i(3) = 2 \\ g_i(4) = 1 \end{array}$$

la suma de los grados de todos los arcos es par

$$\sum g_e + \sum g_i = 5 + 5 = 10$$

2.3 REDES ESPECIALES

Una red G es **bipartita** si el conjunto de nodos N puede partitionarse en N_1, N_2 con $N_1 \cap N_2 = \emptyset$ de manera que todo arco $j \sim (i_1, i_2)$ es tal que $i_1 \in N_1, i_2 \in N_2$ o $i_2 \in N_1, i_1 \in N_2$

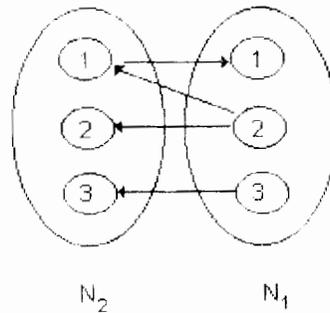


Figura 2.3

Nota:

1. Las matrices nodos-nodos para gráficas no dirigidas son simétricas.
2. La matriz nodos-nodos de una red bipartita es de la forma:

$$q = \begin{matrix} & I_1 & I_2 \\ I_1 & 0 & \# \\ I_2 & \# & 0 \end{matrix}$$

que es simétrica.

LEMA DE HANDSHAKING

En una gráfica, la suma de los grados de todos los nodos es igual al doble del número de arcos.

Demostración

Ya que cada arco tiene dos extremos, debe contribuir exactamente dos veces a la suma de los grados.

Este lema apareció por primera vez aunque en forma diferente en un artículo de Leonhard Euler (1707-1783). Titulado *Solutio Problematis ad Geometriam Situs Pertinentis* (la solución de un problema referido a la geometría de posición) este artículo data de 1736 y es ampliamente conocido como el primer artículo en la teoría de gráficas, donde también da la solución de los puentes de Königsberg.

Para gráficas que no son dirigidas simplemente se habla del grado de un vértice como el número de arcos que "llegan a él" (pueden entrar o salir)

REDES SIMPLES

Es una gráfica que no contiene rizados ni arcos en paralelo con un nodo sumidero y un nodo fuente.

Un nodo **fuelle** es un nodo de donde únicamente salen arcos.

Un nodo **sumidero** es un nodo a donde únicamente entran arcos.

REDES CIRCULATORIAS

Una red es circulatoria si no existen nodos fuente o sumideros.

CONECTIVIDAD

Una **cadena** de longitud q (cardinalidad q) es una sucesión de q arcos.

$$L = \{j_1, \dots, j_q\}$$

tal que cada arco j_r de la sucesión ($2 \leq r \leq q-1$) tiene un punto final en común con el arco j_{r-1} ($j_{r-1} \neq j_r$) y un segundo punto final común con el arco j_{r+1} ($j_{r+1} \neq j_r$).

El punto final i de j_1 que no es adyacente a j_2 y el punto final i' de j_q que no es adyacente a j_{q-1} son llamados los puntos finales de la cadena L . Decimos que la cadena L une los nodos i, i' . Así en la siguiente red:

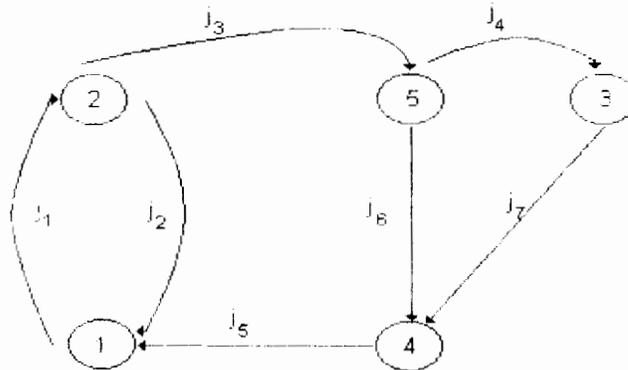


Figura 2.4

$L = \{j_2, j_5, j_6, j_4\}$ es una cadena del nodo 2 al nodo 3.

Una cadena es elemental si no se pasa 2 veces por un mismo nodo.

Un **ciclo** es una cadena cuyos puntos finales coinciden. Una **trayectoria** de longitud q (cardinalidad q) es una sucesión de q arcos $P = \{j_1, \dots, j_q\}$ con $j_1 = (i_0, i_1)$, $j_2 = (i_1, i_2), \dots, j_q = (i_{q-1}, i_q)$. Es decir, una trayectoria es una cadena cuyos arcos están todos dirigidos en el mismo sentido.

El nodo i_0 es el punto inicial de la trayectoria P y el nodo i_q es el punto terminal de la trayectoria P .

Así para la gráfica anterior $P = \{j_1, j_3, j_4, j_7\}$ es una trayectoria del nodo 1 al nodo 4, y se puede describir como la sucesión de nodos $\{1, 2, 5, 3, 4\}$.

Un **circuito** es una trayectoria cuyos nodos finales coinciden. Una **gráfica** se llama **conectada** si para cualquier par de nodos (i_1, i_2) existe una cadena que une i_1 con i_2 . La relación

$$i_1 R i_2 = \begin{cases} i_1 = i_2 \\ \text{o existe una cadena que une } i_1 \text{ con } i_2 \end{cases}$$

Es una relación de equivalencia (reflexiva, simétrica y transitiva).

2.4 FLUJO Y DIVERGENCIA EN UNA RED

Un flujo en una red es una función real definida en A (donde A es el conjunto de arcos de una red G).

$$x : A \rightarrow \mathbb{R}$$

El valor $x(j)$ es el flujo en el arco j , y se interpreta como la cantidad de material que fluye en el arco j .

EJEMPLO 2.3

Sea la red siguiente:

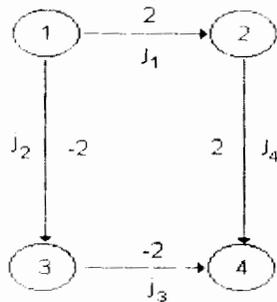


Figura 2.5

El tipo de material es el mismo para todos los arcos. Para entender en forma más clara los conceptos es útil entender j como un canal y $x(j)$ como el número de litros de agua por segundo pasando por cada punto de j un flujo estacionario del nodo inicial i al nodo final i' . La cantidad que entra en i es la misma que sale por i' , pero puede ser positivo, negativo o cero dependiendo de la dirección física del flujo.

Para un orden fijo de arcos $A = \{j_1, \dots, j_n\}$ un flujo x se puede representar como un vector (x_1, \dots, x_n) así para la figura anterior el flujo se puede representar como

$$x = (2, -2, -2, 2)$$

Así x es un "vector" indexado por el conjunto A (es decir un elemento del espacio R^A)

Existe otra notación para flujos que está vinculada con la representación de una red por su matriz de adyacencia. Como más tarde se requerirá que la red sea una digráfica (gráfica dirigida) cada arco se puede identificar por un par de nodos (i, i') . La idea es simplemente escribir $x(i, i')$ en lugar de $x(j)$ tomando $x(i, i') = 0$ en los casos en que no exista arco entre i, i' .

Los flujos en G se representan como funciones en $N \times N$ que pueden representarse matricialmente como:

$$x = \begin{matrix} & \begin{matrix} 1 & 2 & 3 & 4 \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \end{matrix} & \begin{bmatrix} 0 & 2 & -2 & 0 \\ 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & -2 \\ 0 & 0 & 0 & 0 \end{bmatrix} \end{matrix}$$

DIVERGENCIA

Las matrices de incidencia vuelven a entrar en escena cuando tratamos de analizar qué sucede a un flujo x en un nodo i , particularmente las entradas y salidas o pérdidas. El nodo i está representado en la matriz de incidencia E por el renglón i .

Las cantidades de material que salen de i están asociadas con los arcos j tales que $x(j) > 0$ y $e(i,j) = 1$, o tales que $x(j) < 0$ y $e(i,j) = -1$.

¿Qué significa esto?

EJEMPLO 2.4

Sea la red de la figura 2.6.

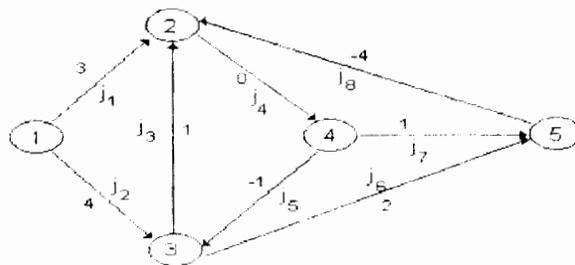


Figura 2.6

Para el nodo 3 se tiene en la matriz de incidencia:

$$E = \begin{matrix} & j_1 & j_2 & j_3 & j_4 & j_5 & j_6 & j_7 & j_8 \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{matrix} & \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & -1 & 1 & 0 & 0 & 0 & -1 \\ 0 & -1 & 1 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & -1 & 1 \end{bmatrix} \end{matrix}$$

el arco j_3 sale del nodo 3 y tiene asociado un 1 es decir $e(3,3) = 1$ (recuerde que cada entrada de la matriz está representada por $e(i, j)$) si se tiene un flujo $x(j=3) = 1$ (cantidad de material que sale del nodo 3) significa que el flujo "pasa" por el arco j_3 en el sentido que lleva el arco y por lo tanto $x(j) e(i,j) > 0$. Si $x(j) < 0$ por ejemplo $x(j) = -4$ significa que el flujo recorre en sentido contrario el arco y por lo tanto $e(i,j) = -1$ como es el caso de $x(8) = -4$ y $e(2,8) = -1$ lo que implica que $x(j) e(i,j) > 0$.

Así la cantidad total que sale desde i es la suma de todos los términos de la forma $e(i,j) x(j)$ que son positivos.

En forma semejante las cantidades que llegan a i corresponden a los casos donde $x(j) > 0$ y $e(i,j) = -1$ por ejemplo:

En el nodo 5: $x(6) = 2$ es decir la cantidad de flujo que llega al nodo 5 por el arco j_6 es de 2 y $e(5,6) = -1$ en la matriz de incidencia por lo tanto $x(6) e(5,6) < 0$

O $x(j) < 0$ y $e(i,j) = 1$ como en el nodo 4, $x(5) = -1$ es decir por el arco j_5 pasa un flujo que va al nodo 4 con -1 y $e(4,5) = 1$.

La suma de todos los términos de la forma $e(i,j) x(j)$ que son negativos es el negativo de la cantidad de flujo que llega a i .

De esta forma la suma de $e(i,j) x(j)$ sobre todos los arcos j nos da el total de salidas de i menos el total de llegadas a i .

Esta cantidad es la divergencia del flujo en el nodo i , y se denota $y(i)$, así

$$y(i) = \sum_{j \in E} e(i, j) x(j) = \text{divergencia de } x \text{ en } i$$

Por ejemplo, de la figura anterior:

en el nodo 3 llegan 4 unidades por j_2
 salen 2 j_6
 salen 1 j_5
 salen 1 j_3

Este nodo tiene la particularidad de que:

$$\# \text{ total de llegadas} = \# \text{ total de salidas.}$$

Así $y(3) = 0$ en el nodo 5 se tiene $y(5) = -7$ esto significa que llegan 7 unidades más de las que salen y en general se dice que:

- Un nodo es fuente para el flujo x si $y(i) > 0$
- Un nodo es sumidero para el flujo x si $y(i) < 0$
- Si el flujo se conserva $y(i)=0$

Se llama y a la función de divergencia (vector) asociada con x y se escribe:

$$y = Ex = \text{div } x$$

En este caso para el flujo de la figura anterior usamos la matriz de incidencia y se calcula

$$y = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & -1 & 1 & 0 & 0 & 0 & -1 \\ 0 & -1 & 1 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & -1 & 1 \end{bmatrix} \begin{bmatrix} 3 \\ 4 \\ 1 \\ 0 \\ -1 \\ 2 \\ 1 \\ 4 \end{bmatrix} = \begin{bmatrix} 7 \\ 0 \\ 0 \\ 0 \\ -7 \end{bmatrix}$$

Así i_1 es una fuente, i_5 es sumidero, y el flujo se conserva en i_2, i_3 e i_4

El hecho de que las 7 unidades creadas en el nodo fuente i_1 sean las 7 que se destruyen en el sumidero no es un accidente. La intuición física sugiere, y el álgebra lo confirma que la cantidad de flujo que se crea en la fuente es igual a la cantidad que se destruye en el sumidero.

Esto se expresa como el **principio de la divergencia total**:

$$\sum_{i \in N} y(i) = 0 \quad \text{para } y = \text{div } x$$

Esto se verifica de la siguiente forma:

$$\sum_{i \in N} y(i) = \sum_{i \in N} \sum_{j \in A} e(i, j)x(j) = \sum_{j \in A} \sum_{i \in N} e(i, j)x(j)$$

Pero

$$\sum_{i \in N} e(i, j) = 0 \quad \text{para toda } j \in A$$

ya que cada columna de la matriz de incidencia contiene exactamente un 1 y un -1 y la suma da 0.

OPERACIONES VECTORIALES

Dos flujos x y x^1 se pueden sumar o superponer para producir un flujo resultante x'' :
 $x''(j) = x(j) + x^1(j)$ para toda $j \in A$.

Así mismo, un flujo se puede multiplicar por un escalar $x' = \lambda x$ lo que significa que
 $x'(j) = \lambda x(j)$ para toda $j \in A$.

¿Que está sucediendo físicamente?

Considerando la red del ejemplo 2.4 se tiene:

El flujo x^1 está representado en esta red, como una unidad de flujo pasando desde el nodo 1 hasta el nodo 5 pasando por los nodos 2 y 3.

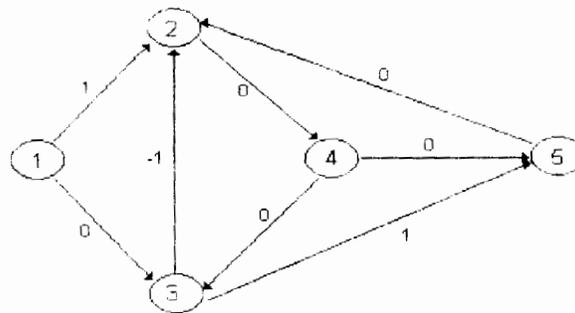


Figura 2.7

Haciendo la suma se tiene:

$x'' = x + x'$ y se tienen 8 unidades en el nodo fuente y llegan 8 unidades al nodo sumidero.

Note que el arco j_3 es usado por x y x' pero no por x'' .

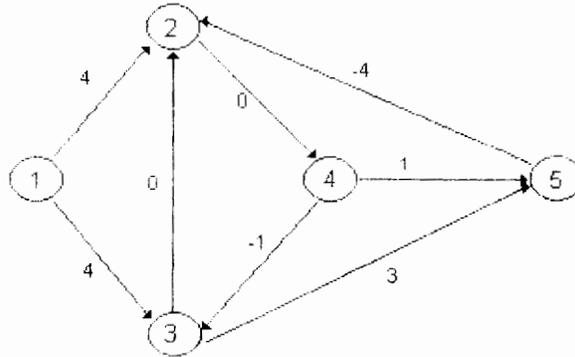


Figura 2.8

El flujo $2x'$ podría representar dos unidades moviéndose en el mismo sentido que x' de $i_1 \rightarrow i_2 \rightarrow i_3 \rightarrow i_5$, mientras $-x'$ podría representar una unidad moviéndose en forma inversa. Trivialmente se tienen las reglas

$$\text{div} (x + x') = \text{div} x + \text{div} x'$$

$$\text{div} (\alpha x) = \alpha \text{div} x$$

2.5 CIRCULACIONES

Un papel especial lo constituyen los flujos x en una red G tales que:

$$\text{div} x = 0$$

Esto es que x se conserva en cada nodo. Tales flujos se conocen como circulaciones. La suma y el producto por un escalar de una circulación es nuevamente una circulación. Así el conjunto de todas las circulaciones forma un subespacio de \mathbb{R}^A ; el espacio de las circulaciones C , claramente C es el espacio nulo de la matriz de incidencia E .

Una de las razones de porqué las circulaciones son importantes es que las discusiones teóricas a menudo pueden simplificarse en términos de ellas, esto debido a que todo flujo en una red G se puede identificar como una circulación de una red más grande. En el caso del flujo x del ejemplo 2.4 la idea se ilustra en la figura 2.9.

Formamos de una red G una nueva red \underline{G} aumentando un nuevo nodo i^1 (el nodo de distribución) y un arco $j_i \sim (i^1, i)$ (un arco de distribución) para cada uno de los nodos viejos i . Para esta nueva red los conjuntos de nodos y arcos se denotan \underline{N} y \underline{A} . Para cada flujo x en G le corresponde un flujo \underline{x} en \underline{G} definido por:

$$\begin{aligned} \underline{x}(j) &= x(j) \text{ para todos los arcos viejos} \\ \underline{x}(j_i) &= y(i) \text{ para todos los arcos de distribución} \end{aligned}$$

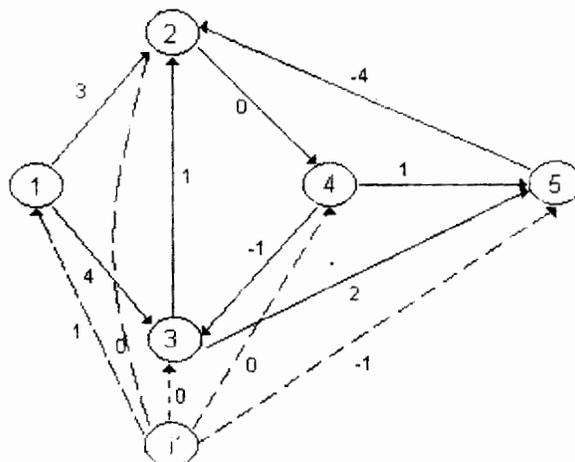


Figura 2.9

Así \underline{x} es una circulación en \underline{G} . Inversamente, cada circulación en \underline{G} corresponde de esta forma a un flujo en G .

También se tiene el caso en que se tienen identificados un nodo fuente y un nodo sumidero por lo cual sólo se agrega un arco como se ilustra en la figura siguiente:

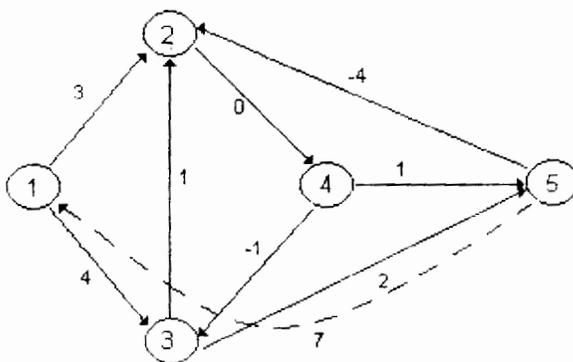


Figura 2.10

El nuevo arco j se conoce como un arco de suministro.

2.6 POTENCIALES Y TENSIÓN

Un potencial u en una red G es una función real definida en el conjunto de los nodos N .

$$u: N \rightarrow \mathbb{R}$$

El valor $u(i)$ es llamado el **potencial** en el nodo i . Con un arco $j \sim (i, i')$ se asocia la diferencia de potenciales

$$v(j) = u(i') - u(i) = \text{tensión a través de } j$$

El signo de la diferencia depende de la orientación del arco. Así se define la **función tensión v** en A , y se denomina el diferencial del potencial u .

La tensión $v(j)$ se puede escribir como

$$v(j) = - \sum_{i \in N} u(i) e(i, j)$$

o

$$v = -uE = \Delta u$$

Un ejemplo de potenciales y tensiones se muestra en la figura 2.11 donde los números en los nodos son los potenciales y los números en los arcos son las tensiones correspondientes.

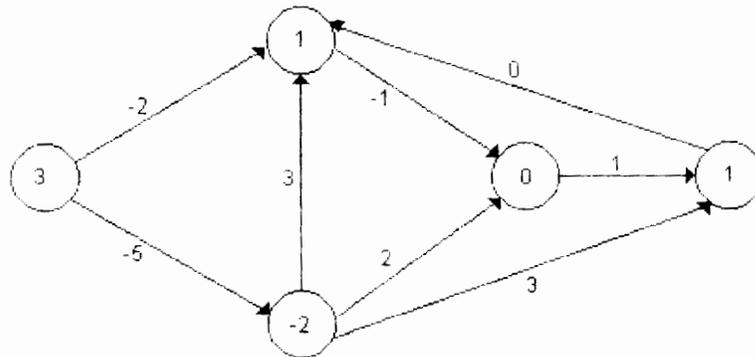


Figura 2.11

En general, llamamos a v un diferencial en G si $v = \Delta u$ para algún potencial u . El conjunto de todos los diferenciales se preserva bajo la suma y la multiplicación por un escalar, y así como el espacio de las circulaciones forma un subespacio de \mathbb{R}^A , y

se llama el espacio de los diferenciales y se denota por D . De esta forma D es el espacio de los renglones en la matriz de incidencia E , el rango de la transformación lineal $u \rightarrow -uE$ de $\mathbb{R}^N \rightarrow \mathbb{R}^A$.

Es natural usar la notación siguiente de producto punto para elementos de \mathbb{R}^N y \mathbb{R}^A .

$$u \cdot y = \sum_{i \in N} u(i)y(i) \quad v \cdot x = \sum_{j \in A} v(j)x(j)$$

y se obtiene de esta manera la fórmula de conversión:

$$v \cdot x = -u \cdot y \quad \text{sí } y = \text{div } x, v = \Delta u$$

la validez de esta fórmula parte del hecho de que ambos lados se reducen a la expresión:

$$v(j) = -\sum u(i) e(i,j)$$

entonces

$$v \cdot x = - \sum_{i \in N, j \in A} u(i)e(i,j)x(j) = -uEx = -u \cdot y$$

Una consecuencia inmediata de la fórmula de conversión es el hecho de que

$$v \cdot x = 0 \quad \text{para todos } v \in D, x \in C, \text{ ya que:}$$

1. $v \cdot x = -u \cdot y$ como y está asociada a un flujo x que es circulación, entonces $y(i) = 0$ para todo i , lo que implica que $v \cdot x = 0$
2. Si $x \in C, v \in D$ y C es el espacio nulo de E y D es el espacio renglón de E por la segunda parte del Teorema Fundamental de Algebra Lineal que dice:

Dada una matriz de $m \times n$ se tiene

$$N(A) = (R(A^T))^{\perp} \\ R(A^T) = N(A)^{\perp}$$

además

$$N(A^T) = (R(A))^{\perp} \\ R(A) = (N(A^T))^{\perp}$$

y $\dim(\text{espacio fila}) + \dim(\text{espacio nulo}) = \text{número de columnas}$.

Entonces C y D son complementos ortogonales, lo que significa que el producto entre 2 elementos cualesquiera de ellos es igual a cero además:

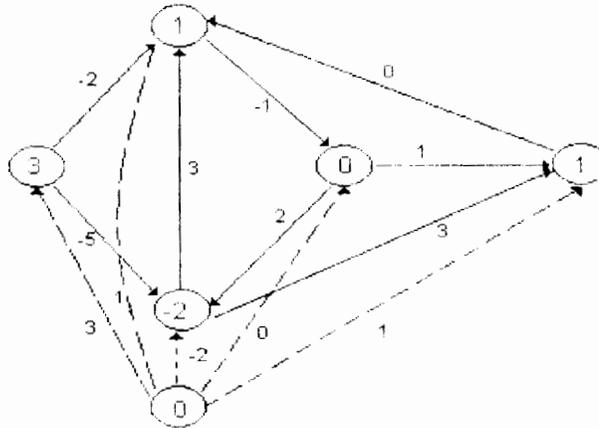
$$\dim C + \dim D = |A|$$

$$D = C^\perp = \{v \in R^A \mid \forall x = 0 \quad \forall x \in C\}$$

$$C = D^\perp = \{x \in R^A \mid \forall x = 0 \quad \forall v \in D\}$$

Así como un flujo en G se puede ver como parte de una circulación de la red aumentada \underline{G} , se puede también ver cada potencial en G como parte de un diferencial en \underline{G} : como se muestra en la figura siguiente:

Figura 2.12



Entonces dado cualquier potencial u en G con una tensión asociada v se define el potencial \underline{u} en \underline{G} como:

$$\underline{u}(i) = \begin{cases} u(i) & \text{para toda } i \text{ tal que } i \text{ es "nodo viejo"} \\ u(i) = 0 & \text{para el nodo de distribución } i \end{cases}$$

La tensión $\underline{v} = \Delta \underline{u}$ en \underline{G} satisface

$$\underline{v}(j) = v(j) \text{ para toda } j \text{ donde } j \text{ son "arcos viejos"}$$

$$\underline{v}(j_i) = u(i) \text{ para los arcos de distribución } j_i$$

En términos de dualidad los variables definidas sobre los arcos se convierten en variables definidas sobre los nodos.

La tabla siguiente define un par de sistemas lineales duales.

<i>Primal</i>	<i>Dual</i>
Flujos - variables definidas sobre los arcos	potencial - variables definidas sobre los nodos
divergencias - variables definidas sobre los nodos	tensión - variables definidas sobre los arcos

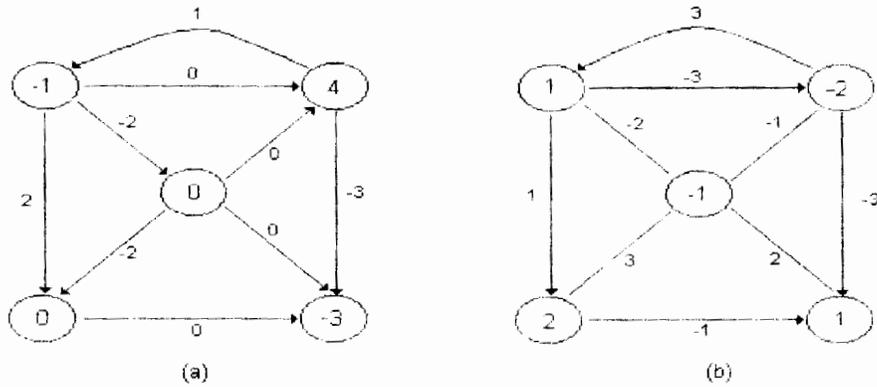
Los potenciales y las tensiones se pueden sumar o multiplicar por escalares, dos mismos potenciales pueden tener la misma tensión. En particular si:

$$u'(i) = u(i) + k \text{ donde } k \text{ es una constante, para toda } i \text{ en } N$$

Ejercicio

Considere las redes de la figura 2.13 (a y b). En la figura a se muestran los flujos y se pide calcular las divergencias. En la figura b se muestran las potenciales y se pide calcular las tensiones en los arcos.

Figura 2.13 a y b.



2.7 FLUJOS ÓPTIMOS Y POTENCIALES

Para cada arco $j \in A$ existe un intervalo $c(j) \subset \mathbb{R}$ y una función $f_j: c(j) \rightarrow \mathbb{R}$ de tal forma que para cada nodo $i \in N$ existe un intervalo $c(i) \subset \mathbb{R}$ y una función $f_i: c(i) \rightarrow \mathbb{R}$. El problema de flujo óptimo es a grandes rasgos:

$$\min \sum_{j \in A} f_j(x(j)) + \sum_{i \in N} f_i(y(i)) \text{ sobre todos los flujos}$$

$$x \in \mathbb{R}^A \text{ satisface } x(j) \in c(j) \quad \forall j \in A$$

$$y(i) \in c(i) \quad \forall i \in N \text{ donde } y = \text{div } x.$$

En forma análoga se tiene el problema de potencial óptimo

$$\min \sum g_i(u(i)) + \sum g_j(u(j)) \quad \forall u \in \mathbb{R}^N$$

que satisfacen

$$u(i) \in D(i) \quad \forall i \in N \quad \text{y} \quad v(j) \in D(j) \quad \forall j \in A$$

2.8 NOTAS HISTÓRICAS

Aunque las **redes eléctricas** se han estudiado por mucho tiempo, la utilidad de los flujos y potenciales de las redes en la modelación de problemas en economía e investigación de operaciones no fue reconocida hasta los cincuenta. El libro de L.R. Ford y D.R. Fulkerson (1962) ha jugado un papel realmente significativo al estimular el crecimiento y aplicaciones de la teoría de redes en esas nuevas áreas. La notación y terminología en ese libro son naturales para la gente familiarizada con la programación lineal y han sido ampliamente aceptadas en la Investigación de Operaciones.

La noción de **sistemas lineales duales** corresponde a la programación lineal. Ha sido desarrollada en muchas direcciones interesantes por A.W. Tucker (1960), (1963) y prueba ser especialmente útil al hacer generalizaciones de los problemas de optimización en redes a otros de programación separable.

Los términos **flujo** y **divergencia** no fueron usados en teoría de redes antes de Rockafellar y pueden no estar completamente ligados a la ingeniería eléctrica debido a otro uso conectado con magnetismo. Sin embargo son simples, naturales y cubren una necesidad definida. El "nodo de distribución" en la red aumentada corresponde a la "tierra" en teoría eléctrica.

Los flujos x que pertenecen al espacio de circulaciones C se dice que también satisfacen la **Ley de Corriente de Kirchhoff**, mientras que las tensiones v en el espacio de diferenciales D satisfacen la **Ley de Voltaje de Kirchhoff**. El uso de esta terminología es en honor al trabajo pionero de G Kirchhoff en 1847. Una discusión de esas condiciones en la terminología de topología combinatoria se encuentra en Slepian (1968). El hecho de que C y D son ortogonales es también conocido como el **Teorema de Tellegen**.

CAPÍTULO 3 TRAYECTORIAS Y CORTES

3.1 INTRODUCCIÓN

El objetivo de este capítulo es describir dos conceptos duales asociados con nodos y arcos: la trayectoria y el corte; así como el resultado central que los relaciona: el teorema de la red pintada y su equivalencia, el Lema de Minty.

DEFINICIÓN 3.1

En una red G se dice que P es una trayectoria, si consiste de una sucesión finita de la forma:

$$i_0, j_1, i_1, j_2, \dots, j_r, i_r.$$

donde $r > 0$ y cada i_k es un nodo mientras que j_k es un arco tal que

$$j_k \sim (i_{k-1}, i_k) \text{ o } j_k \sim (i_k, i_{k-1})$$

Los nodos i_0 e i_r reciben el nombre de nodos **inicial** y **final** de la trayectoria, respectivamente. Si $i_0 = i_r$, se dice que P es un **circuito**.

Un arco j_k en una trayectoria P se dice que se recorre positivamente (negativamente) si $j_k \sim (i_{k-1}, i_k)$ (si $j_k \sim (i_k, i_{k-1})$). Si todos los arcos de P se recorren positivamente, se dice que P es una **trayectoria positiva** o **circuito positivo**. De manera semejante P es una trayectoria negativa o circuito negativo si todos sus arcos se recorren negativamente.

Si G es una digráfica, la trayectoria P se puede denotar como:

$$P: i_0 \rightarrow i_1 \leftarrow i_2 \rightarrow i_3$$

Pues a cada arco le corresponde un par de nodos en forma única.

EJEMPLO 3.1

En la figura 3.1:

- a) un circuito es: 1, J_1 , 2, J_3 , 3, J_2 , 1
- b) una trayectoria 2, J_5 , 5, J_6 , 3

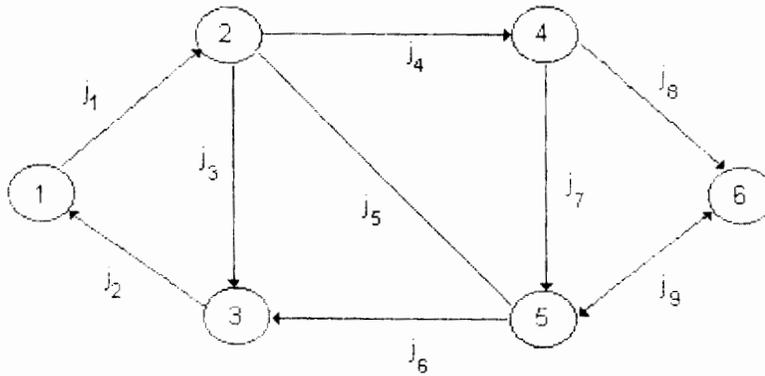


Figura 3.1

es usual escribir si G es digráfica

a) P: 1→2→3→1

b) P: 2→5→3

Una trayectoria P puede recorrer más de una vez un arco, si esto sucede se dice que P tiene **multiplicidades** o que es una **trayectoria con multiplicidades**.

Una trayectoria **elemental** o **simple** es aquella en donde cada arco y nodo se recorren una sola vez.

Sea P una trayectoria sin multiplicidades. Denotaremos por: P+ el conjunto de arcos que se recorren positivamente y por P- el conjunto de arcos que se recorren negativamente.

Se define la **función de incidencia** de la trayectoria elemental P como:

$$e_p(j) = e(j, p) = \begin{cases} +1 & \text{si } j \in P^+ \\ -1 & \text{si } j \in P^- \\ 0 & \text{si } j \notin P \end{cases}$$

Note que esta función además de indicar la orientación en el arco puede interpretarse como un flujo "unitario" en la red para el ejemplo 3.1

a) $e_p = [1, -1, 1, 0, 0, 0, 0, 0, 0]$

b) $e_p = [0, 0, 0, 0, 1, -1, 1, 0, 0]$

EJEMPLO 3.2

Dada la siguiente red:

- a) ¿ Existe una trayectoria positiva del nodo 5 al 2? No
- b) ¿ Existe un circuito positivo que contenga al arco J_6 ? No

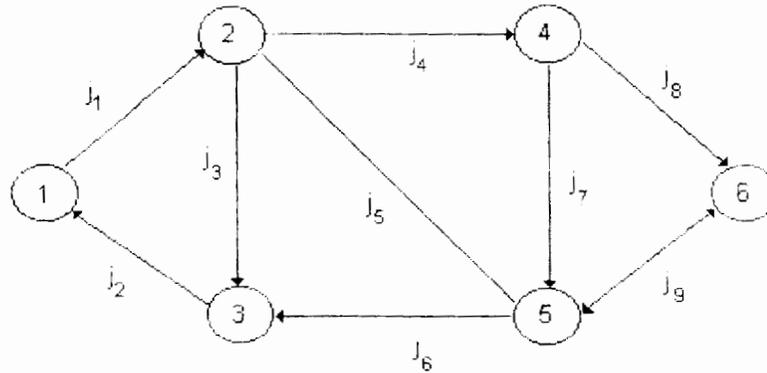


Figura 3.2

CONECTIVIDAD

Una red G se dice **conexa** si para cada par de nodos distintos s y s^1 existe una trayectoria $P: s \rightarrow s^1$, esto es, una trayectoria P que tiene como nodo inicial s y como nodo final s^1 .

Si G no es conexa podemos particionarla en k redes $G_i(N_i, A_i)$ $i=1,2,\dots,k$ donde los subconjuntos de nodos N_i forman una partición de N y son tales que dos nodos s y s^1 pertenecen al mismo conjunto N_i si y sólo si existe una trayectoria $P: s \rightarrow s^1$ o bien $s = s^1$.

Los subconjuntos de arcos A_i se definen como los arcos cuyos extremos pertenecen a N_i y también forman una partición de A .

Una red **fuertemente conexa** es aquella para la cual existe una trayectoria positiva $p: s \rightarrow s^1$ para cada par de nodos distintos s y s^1 .

EJEMPLO 3.3

Una red conexa

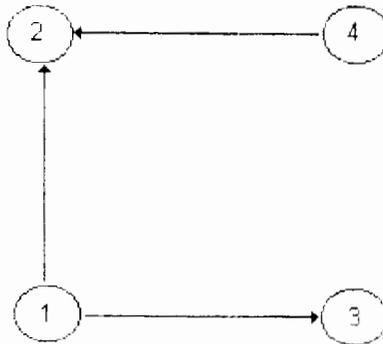


Figura 3.3

Una red fuertemente conexa

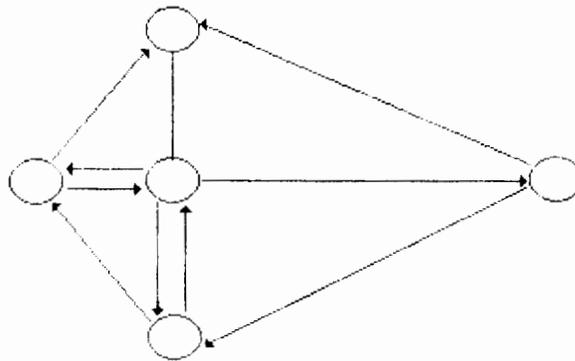


Figura 3.4

3.2 EL PROBLEMA DE DETERMINAR UNA TRAYECTORIA

Un aspecto básico de la definición de conexidad de una red es la existencia de trayectorias de un nodo a otro con el propósito de analizar este problema, que se repetirá continuamente, diremos que los arcos pueden recorrerse de acuerdo con las siguientes reglas:

- Arcos que se recorren en cualquier sentido (verdes)
- Arcos que únicamente se recorren en el sentido del arco (blancos)
- Arcos que únicamente se recorren en sentido inverso (negros)
- Arcos que no pueden recorrerse (rojos)

Para el ejemplo 3.2 dado el siguiente coloreado

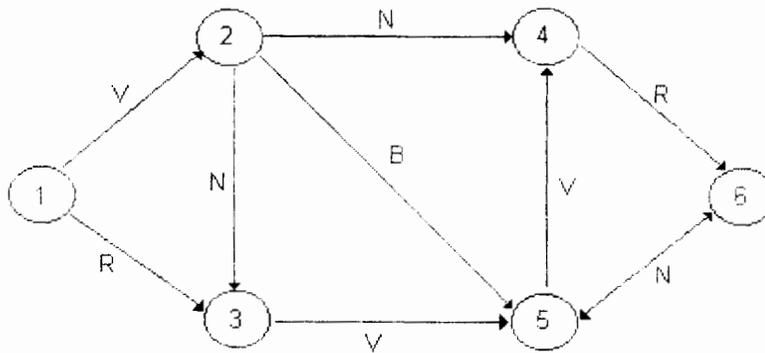


Figura 3.5

Encuentre una trayectoria del nodo 5 al nodo 2 y otra del nodo 1 al nodo 6.

PROBLEMA DE LA TRAYECTORIA PINTADA

Sea G una red y N^+ y N^- conjuntos ajenos de nodos. Suponga que hemos efectuado un pintado de la red. (Colores: verde, blanco, negro y rojo).

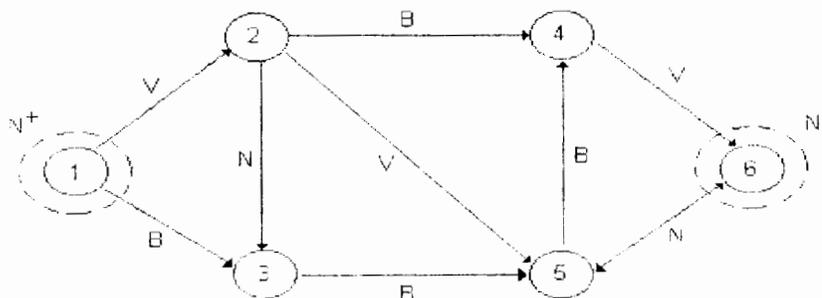
Determine una trayectoria $P: N^+ \rightarrow N^-$; esto es, una trayectoria que parte de N^+ y termina en N^- tal que cada arco de P^+ es verde o blanco y cada arco de P^- es verde o negro.

Una trayectoria P con las propiedades anteriores se dice **compatible** con la coloración o bien el problema anterior equivale a encontrar una trayectoria compatible de N^+ a N^- .

EJEMPLO 3.4

En la siguiente red encuentre una trayectoria compatible con la coloración de N^+ a N^-

Figura 3.6



Se propone la siguiente trayectoria P: $1 \rightarrow 2 \rightarrow 4 \rightarrow 6$.

Es importante hacer notar que en el pintado, cada arco de la red tiene exactamente uno de los colores que se han dado, aunque puede haber un color no usado. Por ejemplo para probar conexidad todos los arcos se pintan de verde mientras que para probar que una red es fuertemente conexa se pintarian de blanco o negro como se verá más adelante.

CORTES

Un concepto dual a una trayectoria, es el concepto de corte o cortadura que podemos formularlo como sigue:

Sean S y S' dos subconjuntos de nodos de una red G y se definen los conjuntos de arcos:

$$[S, S']^+ = \{j \in A \mid j \sim (i, i') \ i \in S, i' \in S'\}$$

$$[S, S']^- = \{j \in A \mid j \sim (i', i) \ i \in S, i' \in S'\}$$

el caso más usual es que $S' = N \setminus S$, es decir que S' es el complemento de S en N.

En este caso se dice que un conjunto de la forma $[S, N \setminus S]$ es un corte $Q = [S, N \setminus S]$ tal que:

$$Q^+ = [S, N \setminus S]^+ \quad \text{y} \quad Q^- = [S, N \setminus S]^-$$

o también

$$Q^+ = \{j \in A \mid j \sim (i, i'), \ i \in S, i' \in S'\}$$

$$Q^- = \{j \in A \mid j \sim (i', i), \ i \in S, i' \in S'\}$$

La palabra corte para Q proviene de la idea de que cualquier trayectoria P con nodo inicial en S y nodo terminal en $N \setminus S$ debe, al menos en una etapa, atravesar uno de los arcos en Q, al borrar arcos en Q se cortaría tal trayectoria. El hecho de que P debe usar un arco de Q se establece formalmente de la siguiente manera:

Sea i el primero de los nodos en P que no está en S, tal nodo existe ya que P va de S a $N \setminus S$. El arco de P inmediatamente anterior i, llamado j, que une al nodo de S con un nodo de $N \setminus S$, y pertenece a Q, entonces $j \in P^+ \cap Q^+$ ó $j \in P^- \cap Q^-$

Por ejemplo, en la red anterior

$$Q = [S, N \setminus S] \quad \text{Si} \quad S = \{1, 2, 3\}$$

$$Q^+ = \{(2, 4), (2, 5), (3, 5)\} \quad Q^- = \emptyset$$

EJEMPLO 3.6

Sea la red, con la siguiente coloración (ver figura 3.7).

Sea $Q = [S, N \setminus S]$ si $S = \{2, 5\}$

$$Q^+ = \{(2, 4), (2, 3), (5, 4)\}$$

$$Q^- = \{(1, 2), (3, 5), (6, 5)\}$$

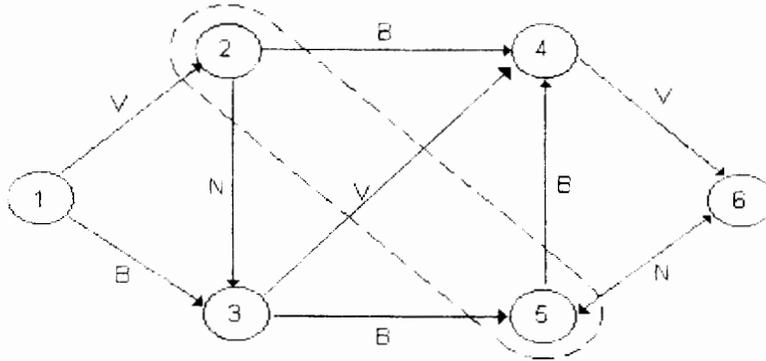


Figura 3.7

EJEMPLO 3.7

En la red del ejemplo 3.6 especifique el corte Q considerando que los elementos de Q^+ sean blancos o verdes y los de Q^- sean negros o verdes.

Sea $S = \{2, 5\}$ con $Q = [S, N \setminus S]$

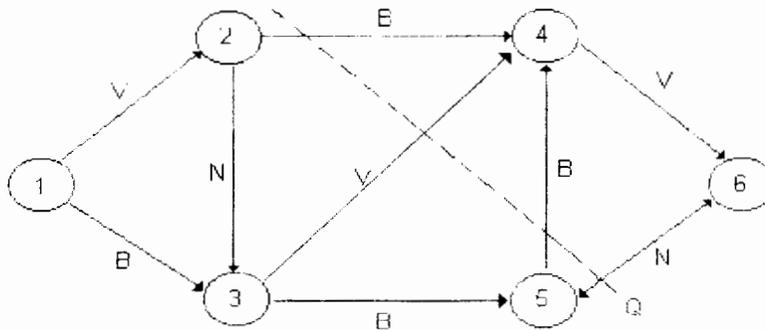


Figura 3.8

La función de incidencia del corte Q se define como:

$$e_Q(j) = \begin{cases} 1 & \text{si } j \in Q^+ \\ -1 & \text{si } j \in Q^- \\ 0 & \text{en otro caso} \end{cases} \quad e_Q \in E = e_Q$$

Es interesante observar que e_Q , como una función en el conjunto de los arcos, se puede considerar como un diferencial; sin embargo, ésta es la tensión correspondiente al negativo del potencial

$$e_S(i) = \begin{cases} 1 & \text{si } i \in S \\ 0 & \text{si } i \notin S \end{cases}$$

Donde S es tal que $Q=[S, N \setminus S]$.

En general, $e_Q = \Delta u$ con

$$u(i) = \begin{cases} 1 & \text{si } i \in N \setminus S \\ 0 & \text{si } i \in S \end{cases}$$

EJEMPLO 3.8

Sea la red

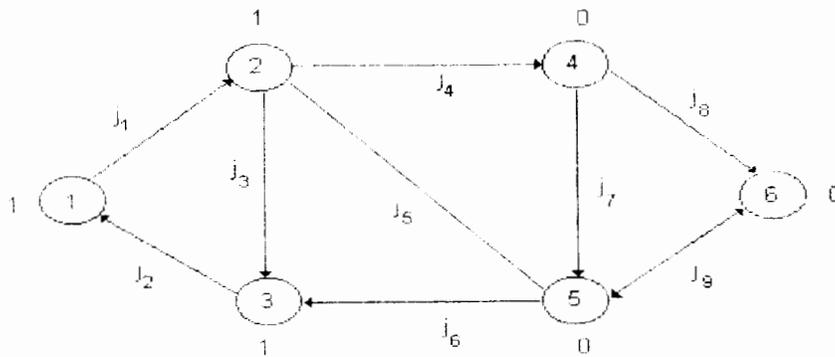


Figura 3.9

Sea $Q=[S, N \setminus S]$ $S=\{1,2,3\}$

$$Q^+ = \{(2,4), (2,5)\}$$

$$Q^- = \{(5,3)\}$$

$$e_Q = [0,0,0,1,1,-1,0,0,0]$$

Un corte se dice **elemental** si al quitar estos arcos de la red, el número de componentes de la red aumenta en uno.

3.3 ALGORITMO DE ENRUTAMIENTO

Se usa para determinar una solución al problema de las trayectorias compatible con una coloración dada o bien un corte.

La idea del algoritmo es combinar un conjunto de nodos $S \subseteq N^+$ y una función $\Theta: SN^+ \rightarrow A$ que recibe el nombre de enrutamiento (ó Θ -enrutamiento) de S con base en N^+ tal que etiqueta cada nodo $i \in SN^+$, con un arco $j \in A$, dicha etiqueta servirá para representar las trayectorias construidas.

- a) Para cada $i \in SN^+$, $\Theta(i)$ es un arco que une i con algún nodo de S .
- b) Se genera la sucesión $i_0, \Theta(i_0), i_1, \Theta(i_1), \dots, i_k, \Theta(i_k)$, donde i_k es el nodo final del arco $\Theta(i_{k-1})$ y eventualmente se llega a N^- .

ALGORITMO DE ENRUTAMIENTO

Propósito: Determinar una trayectoria de N^+ a N^- compatible con una coloración dada.

Descripción

- PASO 1 Sea $S = N^+$ y Θ vacío
- PASO 2 Determine el corte $Q = [S, N \setminus S]$
- 2.1 Si existe $j \in Q^+$ verde o blanco o si $j \in Q^-$ es verde o negro ir a 3
 - 2.2 Si no existe j terminar. No hay solución al problema
- PASO 3 Sea $\Theta(i) = j$ con $i \notin S$ hacer $S := S \cup \{i\}$ (enrutamiento compatible con la coloración).
- 3.1 Si $i \in N^-$ terminar. Θ contiene una trayectoria compatible $P: N^+ \rightarrow N^-$
 - 3.2 Si $i \notin N^-$ se tiene $S \cap S^- = \emptyset$ ir a 2

EJEMPLO 3.9

Dada la red con $N^+ = \{s\}$ y $N^- = \{s'\}$ encontrar una trayectoria compatible con el pintado

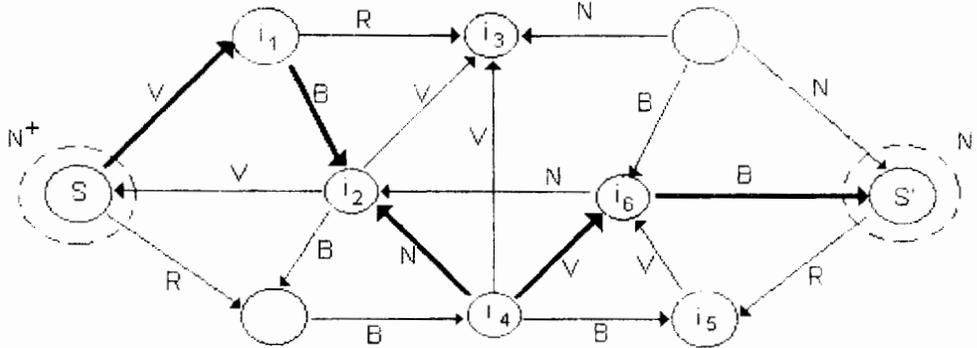


Figura 3.10

el nodo i_k se aumenta a S en la k-ésima iteración

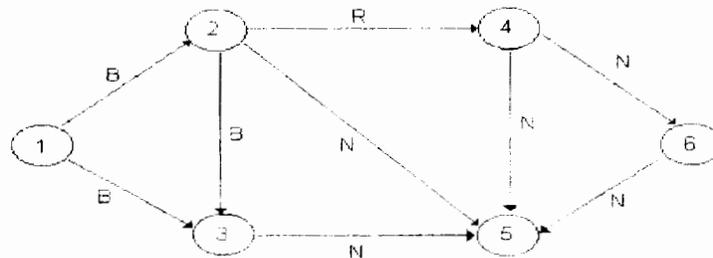
$$P: s \rightarrow i_1 \rightarrow i_2 \rightarrow i_4 \leftarrow i_6 \leftarrow s'$$

EJEMPLO 3.10

Determine si existe una trayectoria $P N^+ \rightarrow N^-$ en la siguiente red.

$$N^+ = \{1\} \quad N^- = \{6\}$$

Figura 3.11



Definimos un corte que sólo contenga al nodo 1, nos fijamos en sus arcos, si el arco que va al nodo 2 es blanco extendemos s a ese nodo, lo mismo para el nodo 3, el conjunto de estados alcanzables es 2,3. Se define otro corte que abarca a los nodos 1,2,3 y se vuelve a empezar.

EJEMPLO 3.11

Dada la siguiente red (figura 3.12), encuentre una trayectoria compatible con la coloración o un corte.

PASO 1

Comenzamos con $S_0 = N^+ = \{i_1\}$ y Θ vacío.

PASO 2

$Q_1 = \{j_1, j_2, j_3\}$; j_1, j_2, j_3 son compatibles con el pintado.

PASO 3

Sea

$$\Theta(i_2) = j_1$$

$$\Theta(i_3) = j_2$$

$$\Theta(i_4) = j_3$$

$$S_1 = S_0 \cup \{i_2, i_3, i_4\} = \{i_1, i_2, i_3, i_4\}$$

$$S_1 \cap N^- = \emptyset, \text{ Ir al paso 2.}$$

Iteración 1

$Q_2 = \{j_4, j_5, j_7, j_9, j_{10}, j_{11}\}$; j_4, j_5 son compatibles con el pintado.

VER PASO 3

Sea $\Theta(i_7) = j_4$

$$\Theta(i_5) = j_5$$

$$S_2 = S_1 \cup \{i_5, i_7\} = \{i_1, i_2, i_3, i_4, i_5, i_7\}$$

$$S_2 \cap N^- = \emptyset, \text{ Ir al paso 2.}$$

Iteración 2

$$Q_3 = \{j_9, j_{10}, j_{11}, j_{13}, j_{14}, j_{17}, j_{18}\}.$$

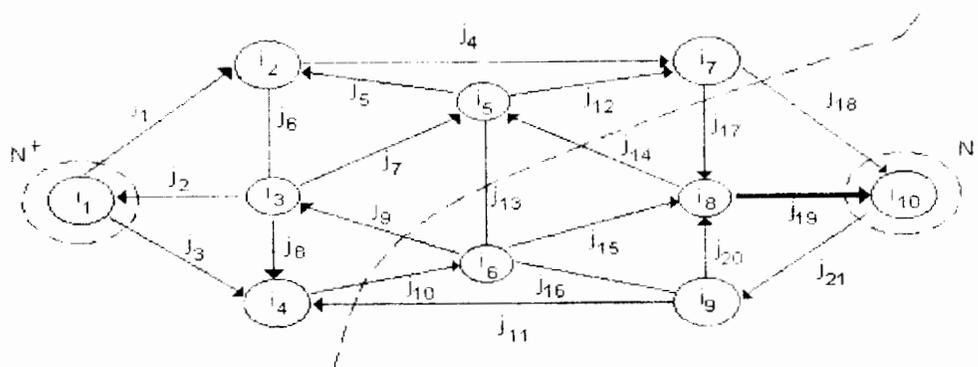


Figura 3.12

No existen arcos compatibles con la coloración. Por lo cual no existe solución al problema.

NOTA: El pintado es:

<u>Color</u>	<u>Arco</u>
Verde	$J_1, J_3, J_4, J_{12}, J_{15}, J_{21}$
Blanco	$J_9, J_{11}, J_{13}, J_{14}$
Negro	$J_2, J_5, J_6, J_7, J_8, J_{16}, J_{18}, J_{20}$
Rojo	J_{10}, J_{17}, J_{19}

DUALIDAD DE TRAYECTORIAS Y CORTES

El problema complementario al de encontrar una trayectoria compatible con una coloración dada es el corte compatible. Se dice que el **corte Q separa N^+ de N^-** si es de la forma $[S, N \setminus S]$, para algún $S \subseteq N$, tal que $N^+ \subseteq S$ y $N^- \cap S = \emptyset$. Se denotará Q con $N^+ \downarrow N^-$.

Problema del corte coloreado (o pintado) Sean N^+ y $N^- \subseteq N$ tales que $N^+ \cap N^- = \emptyset$. Sea una coloración en la red G con los colores verde, blanco, negro y rojo.

El problema es determinar un corte $Q: N^+ \downarrow N^-$ tal que todo arco de Q^+ sea rojo o negro mientras que todo arco de Q^- sea rojo o blanco.

Un corte que cumple las restricciones de color se dice **compatible** con la coloración y si además separa N^+ de N^- constituye la solución al problema del corte coloreado. Observe la dualidad entre las restricciones de color para trayectorias y cortes.

Teorema (de la red coloreada) Sea $N^+ \subseteq N$, tales que $N^+ \cap N^- = \emptyset$. Entonces, para toda coloración de la red G con los colores verde, blanco, negro y rojo, una y sólo una de las siguientes afirmaciones es válida.

- El problema de la trayectoria coloreada tiene solución P.
- El problema del corte coloreado tiene solución Q.

Demostración:

Sea G una red coloreada y $N^+ \subseteq N$ y $N^+ \cap N^- = \emptyset$ aplique el algoritmo de enrutamiento, entonces el algoritmo tiene 2 terminaciones posibles y excluyentes, si termina en el paso 2 se ha construido un corte Q compatible con la coloración, la existencia de este corte garantiza la no existencia de una trayectoria compatible; o bien, en el paso 3, donde se obtiene una trayectoria coloreada P la que a su vez excluye la posibilidad de encontrar un corte Q compatible con la coloración

Existe otro resultado fuertemente relacionado con el teorema de la red coloreada. Este resultado es el lema de Minty que utiliza el concepto de corte elemental paralelo al de trayectoria elemental.

LEMA DE MINTY

Considere una red G con pintado de arcos verde, blanco, negro o rojo. Dado cualquier arco j blanco o negro uno y sólo uno de los siguientes postulados es cierto:

- Existe un corte Q (elemental) compatible con la coloración que usa j .
- Existe un circuito (elemental) compatible con la coloración que usa j .

Demostración:

Sea $J=(i_1, i_2)$ un arco blanco o negro, si J es blanco considere $N^+ = \{i_2\}$ y $N^- = \{i_1\}$. Si J es negro considere $N^+ = \{i_1\}$ y $N^- = \{i_2\}$.

Apliquemos el algoritmo del enrutamiento para encontrar una trayectoria de N^+ a N^-

Observe que el arco J siempre pertenece al corte Q del paso 2 del algoritmo, pero debido a su coloración y la forma de selección de N^+ y N^- J no es compatible con la coloración.

Ahora el algoritmo sólo tiene dos terminaciones excluyentes en el paso 2: que es el determinar un corte que contendría a J o bien en el paso 3 al encontrar una trayectoria de N^+ a N^- que no contiene a J y como J es compatible con una trayectoria de N^- a N^+ por su forma de selección se forma un circuito.

PROPOSICIÓN

El teorema de la red coloreada y el lema de Minty son equivalentes.

Demostración: Use Minty para probar el Teorema.

Ya vimos que se demuestra el lema de Minty usando el teorema de la red coloreada, para demostrar que son equivalentes basta demostrar el teorema de la red coloreada usando el Lema de Minty.

Entonces procedemos a demostrar el Teorema de la red coloreada.

Demostración:

Sea G una red coloreada con los cuatro colores de costumbre y sean N^+ , N^- dos subconjuntos ajenos del conjunto N de nodos de la red. Sea G' una red construida agregando a G el arco $J'=(s', s')$ y los arcos (s, i) para todo $i \in N^+$ y (k, s') para toda $k \in N^-$ como se muestra en la figura siguiente:

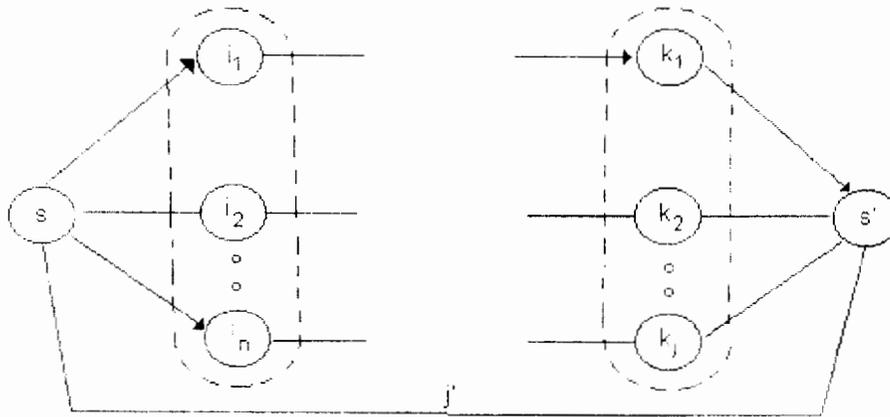


Figura 3.13

Se colorean todos los nuevos arcos de blanco. La aplicación del lema de Minty para el arco J' lleva dos posibilidades:

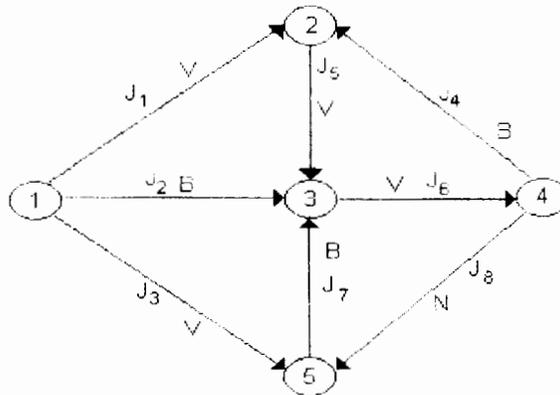
- i) Existe un circuito elemental P compatible con la coloración que contiene al arco J' . Puesto que J' es blanco, este circuito debe ser de la forma $P: s', J', s, j_1, P', j_2, s'$ en donde $j_1 \sim (s, i)$ con $i \in N^+$, $j_2 \sim (k, s')$ con $k \in N^-$ y P' es una trayectoria de i a k compatible con la coloración. Con esto se establece una correspondencia biunívoca de los circuitos P con estas características y las trayectorias $P': i \rightarrow k$ con $i \in N^+, k \in N^-$ de donde se deriva la primera condición del teorema de la trayectoria coloreada.
- ii) Existe un corte elemental Q compatible con la coloración que contiene el arco J' . En este caso Q no puede contener ningún arco nuevo distinto de J' puesto que, siendo estos blancos, no satisfarían las restricciones de color (nótese que $j \in Q$). De aquí que los demás arcos de Q son arcos de la red original G y por tanto constituyen un corte compatible con la coloración en ella. De nuevo, esta correspondencia de cortes es biunívoca por lo que se deriva la segunda alternativa del teorema de la red coloreada.

Conviene señalar que el Lema de Minty en términos algebraicos es equivalente al Lema de Farkas; que no es otra cosa que un resultado de separación de dos conjuntos convexos (poliedros).

EJEMPLO 3.12

Determine si el arco $j \sim (i_4, i_5)$ pertenece a un circuito elemental compatible o a un corte elemental compatible con la coloración definida en la red de la figura siguiente:

Figura 3.14



Solución:

Para verificar si este arco pertenece a un circuito o corte elementales compatibles con la coloración se utilizará el algoritmo de Minty. Es decir, se resuelve el problema de la trayectoria coloreada de $N^+ = \{i_4\}$ a $N^- = \{i_5\}$

Utilizando el algoritmo de enrutamiento se obtiene la trayectoria compatible de i_4 a i_5

$$P: 4 \rightarrow 2 \leftarrow 1 \rightarrow 5$$

esta trayectoria P junto con el arco j_8 forman un circuito elemental.

EJEMPLO 3.13

Determine si el arco j_8 pertenece a un circuito elemental compatible o a un corte elemental compatible con la coloración dada en la red de la figura siguiente:

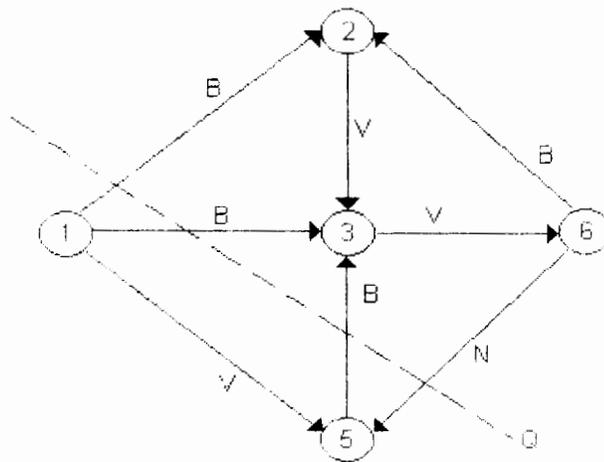


Figura 3.15

Solución

Aplicando el algoritmo de enrutamiento se determina $Q=[S,N\setminus S]$ donde $S=\{i_4, i_2, i_3\}$ que contiene a j_6 y es compatible con la coloración.

3.4 APLICACIONES A CONEXIDAD

El algoritmo de la red pintada proporciona un medio eficiente para probar cuándo una red es conexa y si no lo es nos sirve para determinar sus componentes.

Seleccione un nodo arbitrario s y aplique el algoritmo con $N^+ = \{s\}$ y $N^- = \emptyset$ y todos los arcos pintados de verde. En la terminación habrá un enrutamiento máximo con base s (es decir base $\{s\}$) correspondiente a cierto conjunto S que contiene a s . De la naturaleza del pintado, es claro que S consiste de todos los nodos alcanzables por trayectorias que empiezan en s y describe un sistema particular de trayectorias que completan la tarea. Si $S = N$, G está conectada. Si no, entonces los nodos en S , junto con los arcos incidentes a ellos forman la componente de G conteniendo a s .

Para determinar otra componente seleccione cualquier nodo que no está en S y repita el procedimiento. Después de una sucesión finita de tales cálculos todas las componentes se identificarán.

La prueba para una red fuertemente conexa, necesita el doble del esfuerzo. Aplicando el algoritmo como se hizo antes, y comenzando desde un nodo arbitrariamente seleccionado s , pero con los arcos pintados de blanco. Esto nos da un enrutamiento máximo compatible con la coloración θ_w asociado con un conjunto

de nodos S_w . Repetimos la aplicación con los arcos todos blancos, obteniendo θ_b y S_b .

Obviamente, los nodos en S_w (fuera de s) son aquellos que pueden alcanzarse por una trayectoria positiva desde s , mientras aquellos en S_b son los que pueden llegar a s a través de una trayectoria positiva. Entonces el conjunto $S = S_w \cap S_b$ está compuesto por todos los nodos fuertemente conectados a s . Si $S = N$, G es fuertemente conectada. Si no, S suministra una componente fuerte conteniendo a s (como podemos ver en la figura siguiente).

En el caso de una red en una ciudad se pueden pintar las calles de doble sentido de verde, todas las de un sentido de blanco, y todas las que estén cerradas por reparaciones de rojo. ¿Será posible ir de un nodo a cualquier otro nodo?

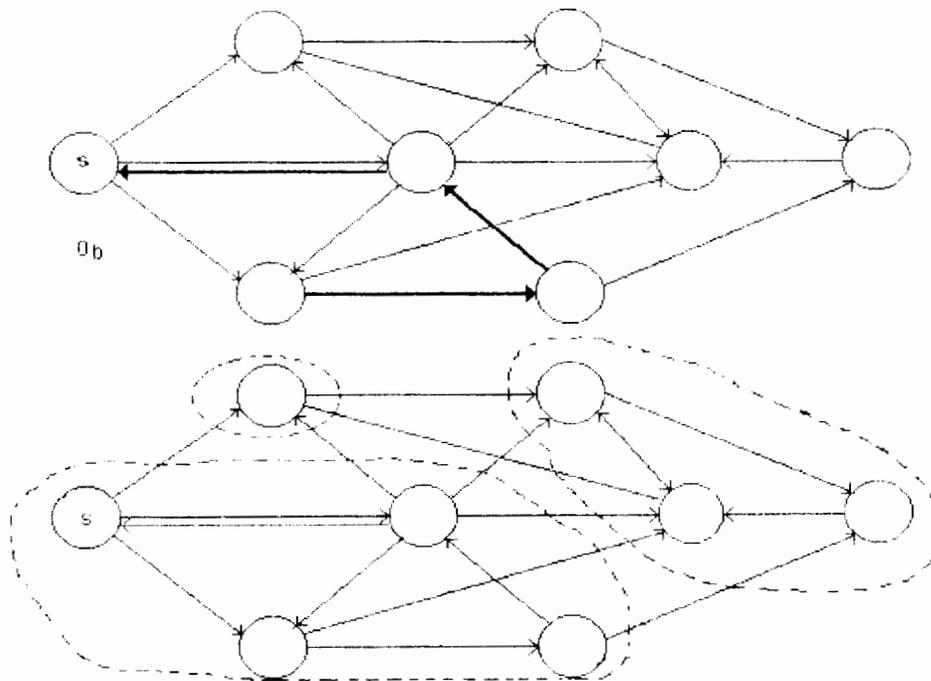


Figura 3.16

3.5 NOTAS HISTÓRICAS

La teoría de gráficas y redes ha sufrido de una falta de estandarización en cuanto a su terminología, diferentes autores usan diferentes palabras para el mismo concepto, o una misma palabra en diferente sentido. En parte, ésto se debe a que hay muchas áreas de aplicación.

En el caso de las "trayectorias" como las que se exponen aquí es difícil encontrar dos textos que estén de acuerdo, lo mismo sucede con cortes. La terminología usada aquí es acorde con el teorema de la red coloreada y el Lema de Minty (1960), aunque Minty sólo usó tres colores, él no admitió la categoría de usar negro, sólo se refiere al sentido inverso de los arcos blancos. Sin embargo, ésto no provoca ningún problema al usar arcos negros. Enrutamiento se usa por otros autores como arcos con raíz o arborescencias, que veremos más adelante.

CAPÍTULO 4 FLUJOS Y CAPACIDADES

4.1 INTRODUCCIÓN

En muchos problemas de redes se involucran flujos que tienen ciertas restricciones en al menos uno de los arcos, también se pueden tener restricciones en las divergencias permitidas en los nodos. El propósito de este capítulo es introducir los conceptos necesarios para resolver estos problemas así como analizar el problema clásico de flujo máximo, y sus correspondientes métodos de solución.

4.2 PRINCIPIO FUNDAMENTAL DE DIVERGENCIA

El problema de flujo máximo fue resuelto en 1986 por R. L. Ford y R.D. Fulkerson que de hecho fueron los iniciadores de la era de flujo en redes independientes de redes eléctricas.

INTERVALOS DE CAPACIDAD

El flujo en un arco j de una red varía en un intervalo cerrado $c(j)$, denominado intervalo de capacidad de j . Dicho intervalo se denota por:

$$c(j) = [c^-(j), c^+(j)]$$

donde $c^-(j)$ es la mínima capacidad y $c^+(j)$ es la máxima.

La única restricción es que $c(j)$ sea un intervalo no-vacío. En particular $c^+(j)$ puede ser $+\infty$ y $c^-(j)$ puede ser $-\infty$.

Un flujo se dice **factible** si $x(j) \in c(j)$ para toda $j \in A$.

Algunos ejemplos de intervalos de capacidad son:

1. $c(j) = [-c, c]$ con $0 \leq c < +\infty$; el flujo x puede usar el arco j en ambas direcciones pero el flujo debe satisfacer $|x(j)| \leq c$.
2. $c(j) = [0, c]$ con $0 \leq c < +\infty$ como en el ejemplo anterior, pero el arco sólo puede usarse en dirección positiva.
3. $c(j) = [0, \infty)$ el arco sólo puede usarse en dirección positiva, pero no hay una cota superior en el flujo.
4. $c(j) = (-\infty, +\infty)$ aquí no hay ninguna restricción de flujo en el arco j .
5. $c(j) = [c, c]$ con $-\infty < c < +\infty$; aquí hay un requerimiento exacto, $x(j) = c$

Una consecuencia directa de que los flujos en los arcos estén acotados es que se restringe el flujo que puede pasar a través de un corte. Específicamente, el flujo x a través de un corte Q se define como:

$$e_Q x = \sum_{j \in Q} x(j) - \sum_{j \in Q^c} x(j) \quad (4.1)$$

donde e_Q es la función de incidencia para Q . Este se puede interpretar como la cantidad neta de material que fluye a través de Q en la dirección de la orientación de Q .

Para reforzar esta interpretación escriba $Q = [S, N \setminus S]$ para el conjunto de nodos S , se define la divergencia de x desde S como:

$$y(S) = \sum_{i \in S} y(i) \quad (4.2)$$

Esta cantidad representa la cantidad neta de material originada en S (esto es, la cantidad total de la fuente menos la cantidad total del sumidero). Estas cantidades (4.1) y (4.2) se relacionan a través del **principio de divergencia**.

PRINCIPIO FUNDAMENTAL DE DIVERGENCIA

Sea x un flujo en la red G y sea el corte $Q = [S, N \setminus S]$. Entonces $y(S) = e_Q x$

[divergencia de x desde S] = [flujo de x a través del corte Q]

Demostración

Usando las definiciones de $y = \text{div } x$ y de e_Q se tiene:

$$\begin{aligned} y(S) &= \sum_{i \in S} y(i) = \sum_{i \in S} \sum_{j \in N} e(i, j) x(j) = \sum_{j \in N} \sum_{i \in S} e(i, j) x(j) = \\ &= \sum_{j \in Q} \sum_{i \in S} e(i, j) x(j) + \sum_{j \in Q^c} \sum_{i \in S} e(i, j) x(j) = \\ &= \sum_{j \in Q} x(j) - \sum_{j \in Q^c} x(j) = e_Q x \end{aligned}$$

Nota: La regla de divergencia total donde $y(N)=0$ es un caso particular del principio de divergencia ya que si $S = N$ entonces $Q = \emptyset$ y de aquí se concluye el resultado.

EJEMPLO 4.1

Dada la red

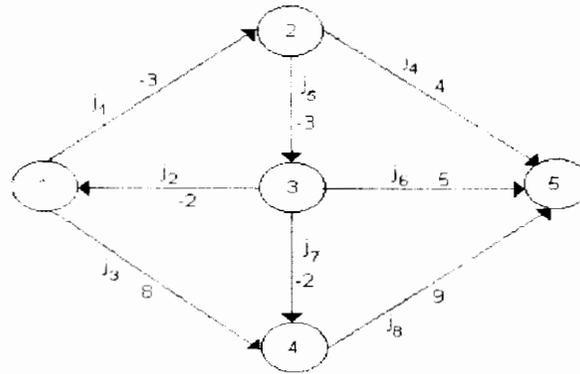


Figura 4.1

Con $y(1) = 7, y(2) = 4, y(3) = 4, y(4) = 3, y(5) = -18$

Sea $S = \{1,2,4\}$, entonces $Q = \{j_4, j_5, j_2, j_3, j_7\}$

Se puede observar que la divergencia de x desde $S : y(S) = 7 + 4 + 3 = 14$ y el flujo x a través de Q

$$Q^+ = \{j_4, j_5, j_8\} = 4 - 3 + 9 = 10$$

$$Q^- = \{j_2, j_7\} = -2 - 2 = -4$$

$$e_Q x = 10 - (-4) = 14, \text{ son iguales}$$

EJEMPLO 4.2

Dada la siguiente red

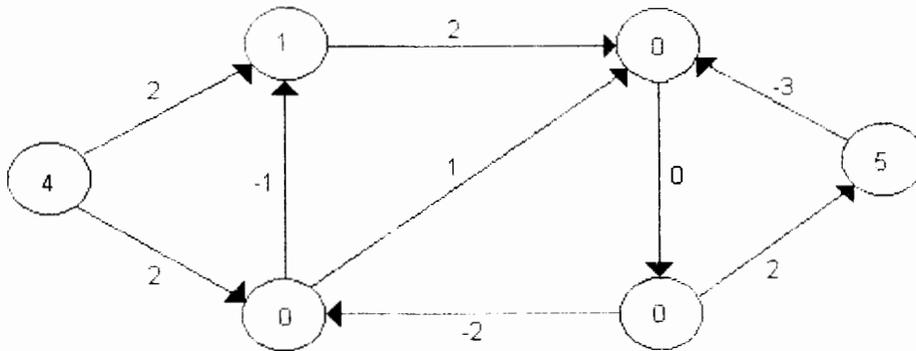


Figura 4.2

l	1	2	3	4	5	6	7
y_l	0	5	0	-3	0	-1	0

$$S = \{1, 2, 4\}$$

$$Q = \{J_4, J_5, J_2, J_6, J_7\}$$

$$y(S) = 2 = e_Q \cdot x$$

Ejercicio

Verifique el principio fundamental de divergencia en la siguiente red:

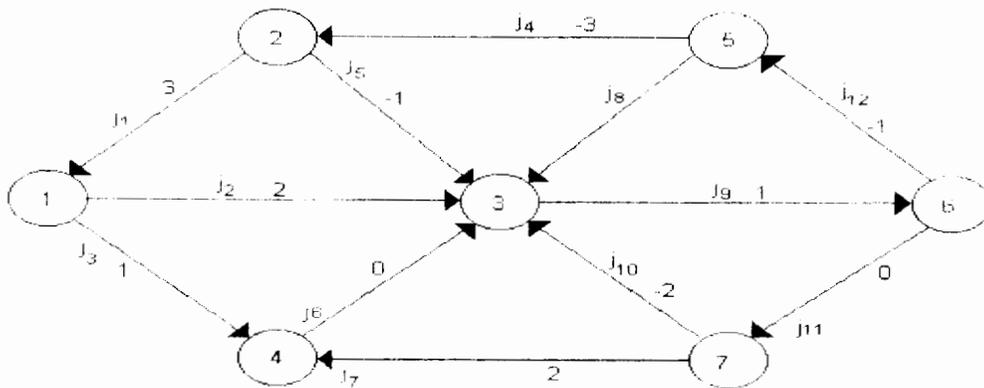


Figura 4.3

En un corte Q se observa:

$$c^-(j) \leq x(j) \leq c^+(j) \quad j \in Q^+$$

$$-c^+(j) \leq -x(j) \leq -c^-(j) \quad j \in Q^-$$

y sumando se tiene:

$$\sum_{j \in Q^+} c^-(j) \leq \sum_{j \in Q^+} x(j) \leq \sum_{j \in Q^+} c^+(j)$$

$$-\sum_{j \in Q^-} c^+(j) \leq -\sum_{j \in Q^-} x(j) \leq -\sum_{j \in Q^-} c^-(j)$$

$$\sum_{j \in Q^+} c^-(j) - \sum_{j \in Q^-} c^+(j) \leq \sum_{j \in Q^+} x(j) - \sum_{j \in Q^-} x(j) \leq \sum_{j \in Q^+} c^+(j) - \sum_{j \in Q^-} c^-(j)$$

$$C^-(Q) \leq \text{flujo a través del corte } Q \leq C^+(Q)$$

éstas son las capacidades inferior y superior del corte, el intervalo de capacidad asociado con Q es $c(Q) = [c^-(Q), c^+(Q)]$

4.3 PROBLEMA DE FLUJO MÁXIMO EN UNA RED

Sea G una red con intervalos de capacidad, y sean N^+ y N^- conjuntos de nodos disjuntos de G . Sea también cualquier flujo x que se conserva en todos los nodos que no pertenecen a N^+ ó N^- , en otras palabras, tales que tienen $y(i)=0$ para toda $i \notin (N^+ \cup N^-)$ y donde $y = \text{div } x$. Por la regla de divergencia total se tiene que:

$$0 = \sum_{i \in N^+} y(i) = \sum_{i \in N^+} y(i) + \sum_{i \in N^-} y(i)$$

lo que implica

$$y(N^+) = -y(N^-)$$

Esta última cantidad recibe el nombre de **flujo de x de N^+ a N^-**

PROBLEMA DE FLUJO MÁXIMO

El problema de flujo máximo queda entonces definido de la siguiente manera:

El problema consiste en maximizar el flujo de N^+ a N^- sobre todos los flujos x tales que $y(i) = 0$ para todas las $i \in N^+ \cup N^-$ factibles con respecto a las capacidades.

Se supondrá la existencia de al menos un flujo que satisfaga todas las restricciones de capacidad y conservación.

La mínima cota superior del conjunto de flujos factibles de N^+ a N^- se denomina el **supremo** en el problema de flujo máximo. El supremo puede ser infinito, un flujo cuyo valor es igual al supremo se dice **solución** al problema de flujo máximo.

Una formulación del problema en términos de programación lineal equivale a

$$\max z = \sum_{i \in N^+} \sum_{j \in A} e(i, j)x(j)$$

sujeto a

$$\sum_{j \in A} e(i, j)x(j) = 0 \quad i \notin N^+ \cup N^- \quad (\text{restricciones de conservación})$$

$$c^-(j) \leq x(j) \leq c^+(j) \quad j \in A \quad (\text{restricciones de capacidad})$$

Para resolver este problema se introduce el concepto dual de corte mínimo que será utilizado como herramienta de optimización. Como se había mencionado anteriormente, todas las trayectorias de N^+ a N^- utilizan algún arco de cualquier corte $Q: N^+ \downarrow N^-$, de este modo los cortes constituyen "cuellos de botella" para el valor del flujo de N^+ a N^- . Es decir:

PROPOSICIÓN 4.1

Sea x un flujo que satisface las restricciones del problema y sea $Q: N^+ \downarrow N^-$ un corte que separa N^+ de N^- entonces

$$\text{flujo de } x \text{ de } N^+ \text{ a } N^- \leq C^+(Q)$$

Demostración:

Puesto que Q es un corte que separa N^+ de N^- entonces es de la forma $[S, N \setminus S]$ con $N^+ \subseteq S$ y $S \cap N^- = \emptyset$. Además como x se conserva en todos los nodos excepto los de $N^+ \cup N^-$, se tiene que $y(i)=0$ para $i \in S \setminus N^+$, por tanto $y(N^+) = y(S)$, pues $y(i) = 0$ para toda $i \in S \setminus N^+$. Asimismo por el principio de divergencia $y(S) = e_Q x \leq C^+(Q)$ lo que demuestran la proposición.

PROBLEMA DE CORTE MÍNIMO

Minimizar $C^+(Q)$ sobre todos los cortes $Q: N^+ \downarrow N^-$

Una consecuencia inmediata, de la proposición anterior esta dada por:

Supremo en problemas de flujo máximo \leq mínimo en problemas de corte mínimo

La relación más importante entre los dos problemas es el hecho de que la igualdad se cumple en los óptimos. Para probar este resultado es necesario definir los conceptos de trayectoria aumentante de flujo y trayectoria de capacidad ilimitada.

Definición 4.1

Una trayectoria $P: N^+ \rightarrow N^-$ es **aumentante** para el flujo x

Si

$$x(j) < c^+(j) \text{ para toda } j \in P^+$$

y

$$x(j) > c^-(j) \text{ para toda } j \in P^-$$

Claramente, el flujo x puede mejorarse a través de una trayectoria aumentante. En efecto, puede garantizarse la existencia de un número $\alpha > 0$ tal que:

$$x'(j) = x(j) + \alpha e_p(j) = \begin{cases} x(j) + \alpha & j \in P^+ \\ x(j) - \alpha & j \in P^- \\ x(j) & j \notin P^+ \end{cases}$$

Es importante notar que el flujo x' construido de esta manera cumple con todas las restricciones del problema para todos los valores de α que cumplan

$$\begin{aligned} \alpha &\leq c^+(j) - x(j) && \text{para toda } j \in P^+ \\ \alpha &\leq x(j) - c^-(j) && \text{para toda } j \in P^- \end{aligned}$$

ya que con esto $c^-(j) \leq x(j) \leq c^+(j)$, para toda $j \in A$ además

$$\text{div } x' = \text{div } x + \alpha \text{ div } e_p$$

pero

$$\begin{aligned} \text{div } e_p(i) &= 1 && \text{si } i \in N^+ \\ \text{div } e_p(i) &= -1 && \text{si } i \in N^- \\ \text{div } e_p(i) &= 0 && \text{si } i \notin N^+ \cup N^- \\ \alpha &> 0 \end{aligned}$$

Esto implica que x' se conserva en todos los nodos fuera de $N^+ \cup N^-$ y también que:

$$[\text{flujo } x' \text{ de } N^+ \text{ a } N^-] = [\text{Flujo } x \text{ de } N^+ \text{ a } N^-] + \alpha$$

de aquí que x' sea mejor que x .

Al valor máximo del número α se le llama capacidad incremental de la trayectoria P . Se dice que una trayectoria $P: N^+ \rightarrow N^-$ tiene capacidad ilimitada si su capacidad incremental no es finita; es decir, si:

$$\begin{aligned} c^+(j) &= +\infty && \text{para toda } j \in P^+ \text{ y} \\ c^-(j) &= -\infty && \text{para toda } j \in P^- \end{aligned}$$

EJEMPLO 4.3

En la figura 4.4 se muestran trayectorias aumentantes, con capacidades incrementales 3, 5 y 1 respectivamente:

a) flujo inicial

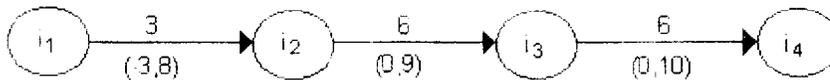
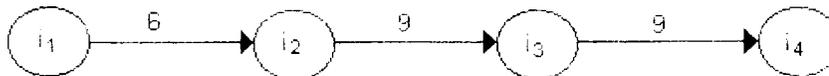


Figura 4.4 a

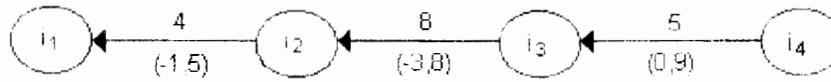
flujo actualizado: $\alpha=3$

Figura 4.4b



b) flujo inicial

Figura 4.4c



flujo actualizado: $\alpha=5$

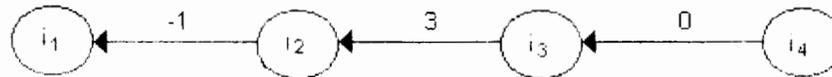


Figura 4.4d

c) flujo inicial

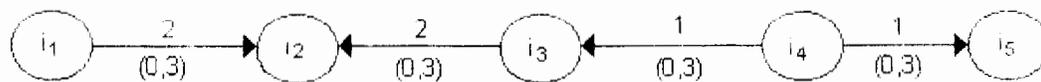


Figura 4.4e

flujo actualizado: $\alpha=1$

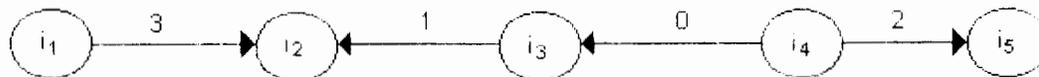


Figura 4.4f

La siguiente es una trayectoria no aumentante

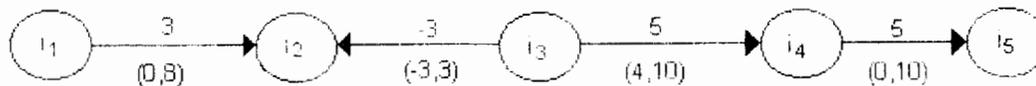


Figura 4.4g

Ya que se tiene $\alpha = 0$ en el segundo arco.

TEOREMA DE FLUJO MÁXIMO - CORTE MÍNIMO (FORD Y FULKERSON)

Suponga que existe al menos un flujo x que satisface todas las restricciones del problema, entonces:

$$\begin{array}{l} \text{Supremo en problemas de} \\ \text{flujo máximo} \end{array} = \begin{array}{l} \text{mínimo en problemas de} \\ \text{corte mínimo} \end{array}$$

Demostración:

La demostración de este teorema es constructiva por lo cual constituye un algoritmo para resolver ambos problemas. En vista del resultado que afirma que el supremo en el problema de flujo máximo es menor o igual que el ínfimo en el problema del corte mínimo, basta exhibir un flujo x y un corte Q tales que:

$$[\text{flujo de } x \text{ de } N^+ \text{ a } N^-] = C^+(Q)$$

Para ello, considere cualquier flujo x en la red G tal que se conserva en todos los nodos fuera de $N^+ \cup N^-$ y construya un corte aplicando

- i) $S = N^+$
- ii) Si $i \in N^+$
 - $j \sim (i, i') \in A$ y $x(j) < c^+(j)$
 - $j \sim (i', i) \in A$ y $x(j) > c^-(j)$

actualice $S = N^+ \cup \{i\}$

Repita (ii) hasta que no sea posible agregar nodos a S .

Es importante señalar que los nodos de S son aquellos a los cuales aún puede enviarse flujo. Pueden presentarse entonces dos casos: $S \cap N^- \neq \emptyset$, o bien $S \cap N^- = \emptyset$.

Caso 1 Si $S \cap N^- \neq \emptyset$ entonces aún puede enviarse flujo de N^+ a N^- , es decir, dada la construcción de S , existe una trayectoria $P: N^+ \rightarrow N^-$ tal que $x(j) < c^+(j)$ para toda $j \in P^+$ y $x(j) > c^-(j)$ para toda $j \in P^-$. De aquí se concluye que P es una trayectoria aumentante para el flujo X . Sea α la capacidad incremental de P y se define x como sigue:

$$x(j) = \begin{cases} x(j) + \alpha & \text{si } j \in P^+ \\ x(j) - \alpha & \text{si } j \in P^- \\ x(j) & \text{en otro caso} \end{cases}$$

Construya de nuevo el conjunto S con el procedimiento (i), (ii) utilizando este flujo mejorado

Caso 2 Si $S \cap N^- = \emptyset$, entonces $Q = [S, N \setminus S]$ es un corte de Q por construcción se tiene que $x(j) = c^+(j)$ para toda $j \in Q^+$ y $x(j) = c^-(j)$ para toda $j \in Q^-$; entonces puesto que $S \cap N^- = \emptyset$

$$\begin{aligned}
[\text{flujo de } x \text{ de } N^+ \text{ a } N^-] &= y(N^+) = y(S) = e_Q x = \\
&= \sum_{j \in Q^+} x(j) - \sum_{j \in Q^-} x(j) = \sum_{j \in Q^+} c^+(j) - \sum_{j \in Q^-} c^-(j) = \\
&= C^-(Q)
\end{aligned}$$

En este caso x es flujo máximo y Q es corte mínimo.

ALGORITMO DE FORD Y FULKERSON

Propósito: Determinar el flujo máximo de N^+ a N^- en una red G en la cual no existen trayectorias de capacidad ilimitada.

Descripción

PASO 1 Determinar x , un flujo que satisfaga todas las restricciones del problema.

PASO 2 Colorear los arcos de G de acuerdo a:

Verde	si	$c^-(j) < x(j) < c^+(j)$
Blanco	si	$c^-(j) = x(j) < c^+(j)$
Negro	si	$c^-(j) < x(j) = c^+(j)$
Rojo	si	$c^-(j) = x(j) = c^+(j)$

PASO 3 Utilizar el algoritmo de enrutamiento para determinar una trayectoria compatible con la coloración (es decir, una trayectoria aumentante para x).

3.1 Si se determina la trayectoria $P: N^+ \rightarrow N^-$ compatible con la coloración, calcular:

$$\alpha = \min \left\{ \begin{array}{l} c^-(j) - x(j) \quad j \in P^- \\ x(j) - c^+(j) \quad j \in P^+ \end{array} \right\}$$

hacer $x = x + \alpha e_p$ y regresar a 2

3.2 Si se determina un corte $Q: N^+ \downarrow N^-$ compatible con la coloración terminar con el flujo máximo x y el corte mínimo Q ya que este último satisface

$$\begin{aligned}
c^+(j) &= x(j) & j \in Q^+ \\
c^-(j) &= x(j) & j \in Q^-
\end{aligned}$$

y por lo tanto el flujo x a través de

$$Q = \sum_{j \in Q^+} x(j) - \sum_{j \in Q^-} x(j) = C^-(Q)$$

Nota: El valor en la igualdad del teorema puede ser $+\infty$ cuando el corte tiene capacidad ilimitada.

EJEMPLO 4.3

En la siguiente red se busca el flujo máximo que se puede enviar del nodo s al nodo s' , se aplica el algoritmo de flujo máximo, comenzando con un flujo cero $x(j) = 0$ para toda $j \in A$. Esto es posible porque todos los intervalos de capacidad contienen al cero.

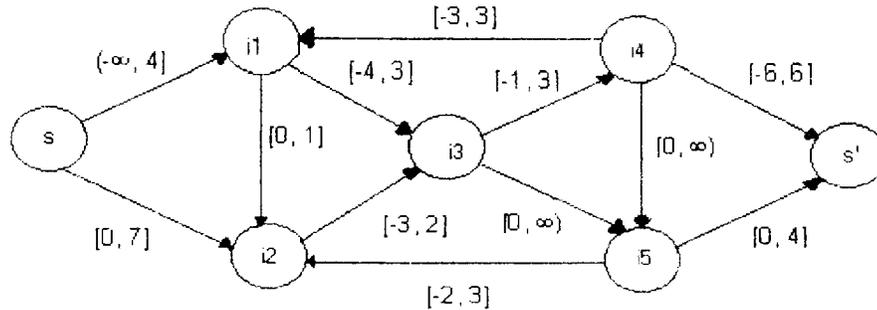


Figura 4.5

En siguiente figura se muestran los números en cada arco que corresponden a las iteraciones 1 a 5.

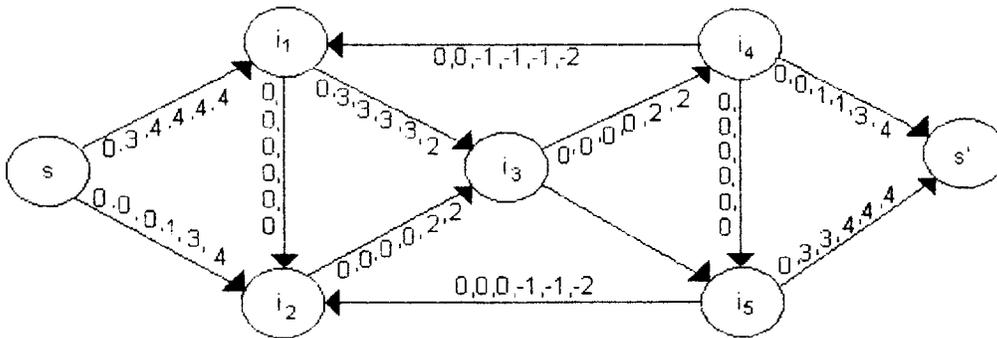


Figura 4.6

Las correspondientes trayectorias aumentantes son:

- | | |
|---|----------------|
| $P_1: s \rightarrow i_1 \rightarrow i_3 \rightarrow i_5 \rightarrow s'$ | $\alpha_1 = 3$ |
| $P_2: s \rightarrow i_1 \leftarrow i_4 \rightarrow s'$ | $\alpha_2 = 1$ |
| $P_3: s \rightarrow i_2 \leftarrow i_5 \rightarrow s'$ | $\alpha_3 = 1$ |
| $P_4: s \rightarrow i_2 \rightarrow i_3 \rightarrow i_4 \rightarrow s'$ | $\alpha_4 = 2$ |
| $P_5: s \rightarrow i_2 \leftarrow i_5 \leftarrow i_3 \leftarrow i_1 \leftarrow i_4 \rightarrow s'$ | $\alpha_5 = 1$ |

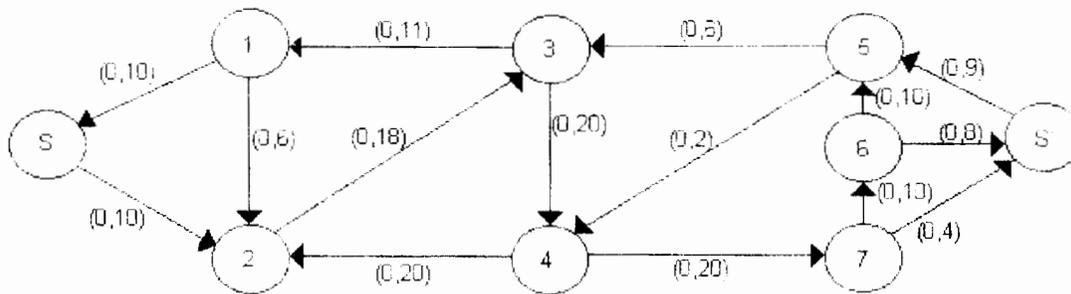
La iteración seis no nos proporciona otra trayectoria de flujo aumentante, sólo el corte $Q = [S, N \setminus S]$ correspondiente a $S = \{s, i_2\}$. Así, el flujo x en esta etapa resuelve el problema de flujo máximo mientras Q resuelve el problema de corte mínimo. El flujo de s a s' es 8, y éste es el mismo valor de $c^+(Q)$.

Note que la cancelación del flujo toma lugar en la iteración 5 en los arcos (i_1, i_3) e (i_3, i_5) donde las trayectorias P_1 y P_5 corren contrarias una con otra. Tal cancelación corresponde al enrutamiento de un flujo anterior, y el algoritmo no funcionaría si esto no fuera permitido.

Ejercicio

Considere la siguiente red y determine el flujo máximo de S a S' .

Figura 4.7



Justificación de convergencia del algoritmo.

Suponga que las capacidades de los arcos $c^+(j)$, $c^-(j)$ y los valores de flujo inicial $x(j)$ son conmensurables, es decir, son múltiplos de cierta cantidad q . En particular capacidades y flujos enteros son conmensurables con $q=1$. De esta condición de conmensurabilidad se concluye que los números α y $x^1(j)$, calculados durante el algoritmo, son también múltiplos de la cantidad q . De aquí que, en cada iteración, el flujo de N^+ a N^- se incrementa al menos en la cantidad positiva q y por lo tanto se realiza un número finito de iteraciones si no existen trayectorias de capacidad ilimitada. Sin esta condición puede no converger el algoritmo o que converja dando una solución errada.

Discriminación de arcos

La condición de conmensurabilidad puede eliminarse si el algoritmo de Ford y Fulkerson se utiliza con una pequeña modificación llamada criterio de discriminación de arcos. Este cambio consiste en utilizar, siempre que sea posible, arcos verdes durante la rutina de enrutamiento; cuando se utilizan trayectorias de arco verdes, el

flujo "mejorado" alcanzará la cota superior o la inferior para al menos un arco, el correspondiente al valor de α . De este modo, para la siguiente iteración, al menos un arco se vuelve blanco o negro mientras que los que tenían estos colores no se alteran. Después de un número finito de iteraciones no existen trayectorias aumentantes con todos los arcos verdes, por lo que para continuar con el proceso de aumento de flujo (si esto es posible) deberá recurrirse a los arcos blancos o negros. En tal momento hay un conjunto S correspondiente a un corte Q que no contiene arcos verdes; es decir, todo arco j de Q satisface $x(j) = c^+(j)$ o $x(j) = c^-(j)$ o ambas. De aquí que

$$\sum_{j \in Q^+} x(j) - \sum_{j \in Q^-} x(j) = \text{flujo a través del corte } Q = \text{flujo } x \text{ de } N^+ \text{ a } N^-$$

G- 612685

4.4 EL PROBLEMA DE FACTIBILIDAD DE FLUJO

El algoritmo de flujo máximo requiere inicialmente un flujo factible en la red, es decir, un flujo que satisfaga las restricciones de capacidad y conservación en arcos y nodos respectivamente. Se considera un caso especial de un modelo más general llamado el Problema de Distribución Factible, donde además se tiene que las divergencias en los nodos deben coincidir con cantidades preestablecidas de oferta y demanda en los nodos.

PROBLEMA DE DISTRIBUCIÓN FACTIBLE

Sean los intervalos de capacidad $C(j) = [c^-(j), c^+(j)]$ para todo arco j y sean los valores de oferta $b(i)$ para todo nodo i . Se desea determinar un flujo x tal que:

$$\begin{aligned} c^-(j) \leq x(j) \leq c^+(j) & \quad \text{para toda } j \in A \\ y(i) = b(i) & \quad \text{para toda } i \in N, \quad (y = \text{div } x) \end{aligned}$$

La función b recibe el nombre de función de oferta, entendiéndose como demanda una oferta negativa. En el caso particular en que $b=0$ el problema recibe el nombre de problema de circulación factible.

Debido al principio de divergencia total, una condición necesaria para que exista solución al problema de distribución factible es $b(N) = 0$; es decir, que la oferta total sea igual a la demanda total.

El problema puede ser más general si en vez de considerar un cierto valor para la divergencia, se permite que esta última pueda tomar valores en un determinado intervalo llamado de oferta y se formula como:



PROBLEMA DE FLUJO FACTIBLE GENERAL

Determinar un flujo x tal que

$$\begin{array}{ll} x(j) \in C(j) & \text{para toda } j \in A \\ y(i) \in C(i) & \text{para toda } i \in N \text{ (} y = \text{div } x \text{).} \end{array}$$

Donde $C(j)$ es el intervalo de capacidad para el arco j y $C(i)$ es el intervalo de oferta para el nodo i .

Puede verificarse fácilmente que el problema de distribución factible, circulación factible y flujo factible son equivalentes.

TEOREMA DE DISTRIBUCIÓN FACTIBLE (GALE Y HOFFMAN)

El problema de distribución factible tiene solución si y sólo si $b(N)=0$ y $b(S) \leq C^+(Q) \forall Q=[S, N \setminus S]$ con $S \subset N$.

Demostración:

Condición necesaria

La condición $b(N)=0$ debe cumplirse como se estableció en la formulación del problema. Por otro lado, según el principio de divergencia, si $S \subset N$ y $Q=[S, N \setminus S]$

$$b(S) = [\text{divergencia de } x \text{ desde } S] = \text{flujo de } x \text{ a través de } Q.$$

Si $\text{div } x = b$ como x debe ser un flujo factible con respecto a las capacidades de los arcos.

$$b(S) \in C(Q) = [c^-(Q), c^+(Q)]$$

En particular $b(S) \leq c^+(Q)$.

Condición suficiente

Se prueba constructivamente con el algoritmo de distribución factible.

ALGORITMO DE DISTRIBUCIÓN FACTIBLE

Propósito: Resolver el problema de distribución factible.

Descripción

PASO 1 Determinar un flujo factible x con respecto a las capacidades de los arcos.

PASO 2 Sean $N^+ = \{i \in N \mid b(i) > y(i)\}$, $N^- = \{i \in N \mid b(i) < y(i)\}$

Si $N^+ = N^- = \emptyset$, entonces el flujo x es la solución deseada, terminar.

Si $N^+ \neq \emptyset$ y $N^- \neq \emptyset$, colorear los arcos de la red de la siguiente manera:

Verde si $c^-(j) < x(j) < c^+(j)$

Blanco si $c^-(j) = x(j) < c^+(j)$

Negro si $c^-(j) < x(j) = c^+(j)$

Rojo si $c^-(j) = x(j) = c^+(j)$

ir al Paso 3.

PASO 3 Aplicar el algoritmo de enrutamiento a la red.

- a) Si se determina una trayectoria $P: N^+ \rightarrow N^-$ compatible con la coloración, calcular:

$$\alpha = \min \begin{cases} c^-(j) - x(j) & j \in P^+ \\ x(j) - c^+(j) & j \in P^- \\ b(i) - y(i) & i \text{ nodo inicial de } P \text{ (en } N^+) \\ y(i) - b(i) & i \text{ nodo terminal de } P \text{ (en } N^-) \end{cases}$$

Hacer $x' = x + \alpha e_p$ y regresar al Paso 2.

- b) Si se determina un corte $Q = [S, N \setminus S]$ compatible con la coloración, terminar, no existe solución al problema.

EJEMPLO 4.4

Determine un flujo factible en la red siguiente:

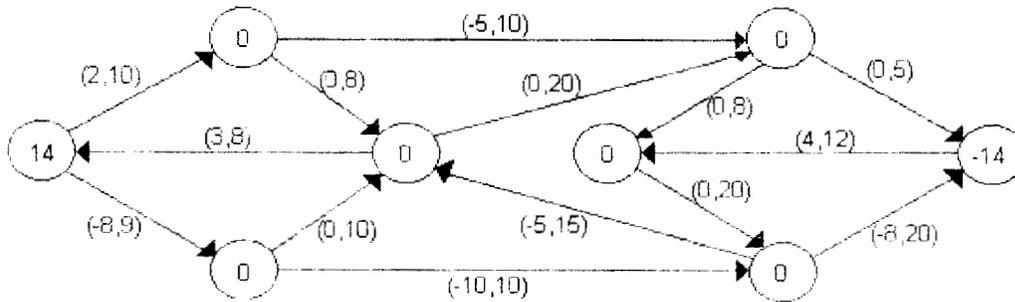


Figura 4.8

PASO 1

Se inicia con el flujo x factible con respecto a los arcos. En la figura se muestra en los arcos el flujo y el color del arco, así como los conjuntos N^+ y N^- de nodos cuya divergencia es mayor o menor que la demanda.

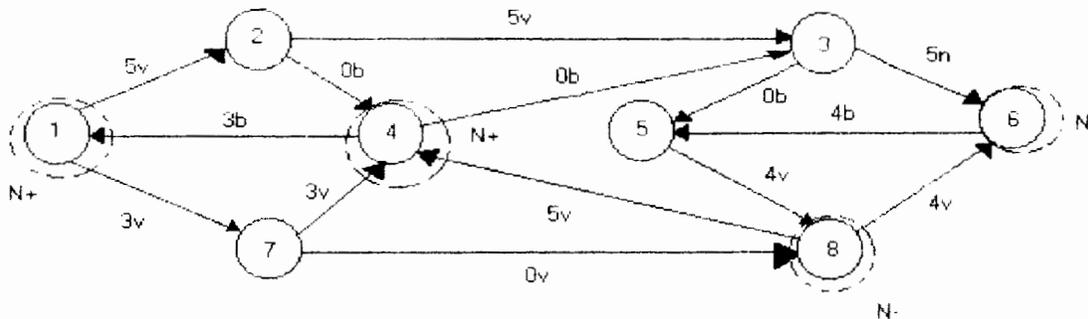


Figura 4.9

PASO 2

i	1	2	3	4	5	6	7	8
$y(i)$	5	0	0	-5	0	-5	0	+5
$b(i)$	14	0	0	0	0	-14	0	0

entonces $N^+ = \{1, 4\}$ y $N^- = \{6, 8\}$

PASO 3

Se obtiene la trayectoria compatible con la coloración

$$P: 1 \rightarrow 7 \rightarrow 8$$

En la figura 4.10 se muestra la trayectoria con arcos más gruesos y el flujo actualizado en los arcos así como la coloración.

$$\alpha = \min \{9-3, 10, 14-5, 5\} = 5$$

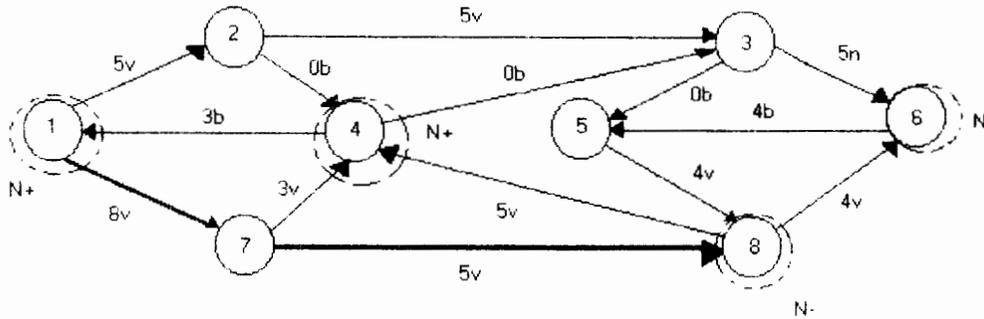


Figura 4.10

Iteración 1

i	1	2	3	4	5	6	7	8
y(i)	10	0	0	-5	0	-5	0	0
b(i)	14	0	0	0	0	-14	0	0

de donde

$$N^+ = \{1, 4\} \text{ y } N^- = \{6\}$$

con la coloración correspondiente y $P: 4 \leftarrow 8 \rightarrow 6$ y $\alpha = \min \{10, 16, 5, 9\} = 5$ con el nuevo flujo que se muestra en la figura 4.11

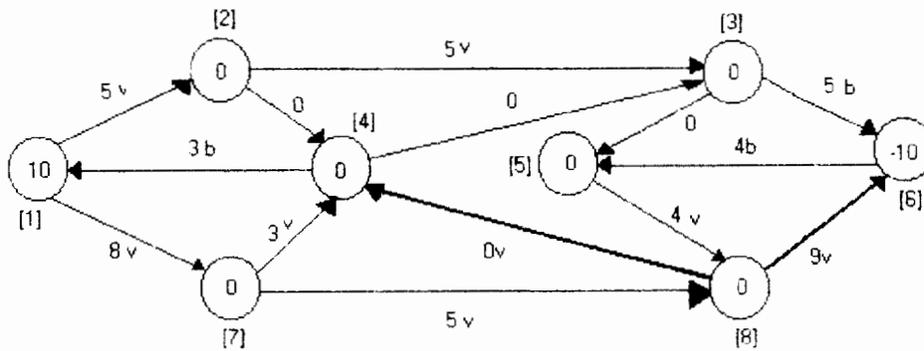


Figura 4.11

Iteración 2

La divergencia y las ofertas son

i	1	2	3	4	5	6	7	8
y(i)	10	0	0	0	0	-10	0	0
b(i)	14	0	0	0	0	-14	0	0

de donde $N^+ = \{1\}$ y $N^- = \{6\}$

y la trayectoria compatible con la coloración

$$P: 1 \rightarrow 2 \rightarrow 4 \leftarrow 8 \rightarrow 6$$

con $\alpha = \min \{10-5, 8-0, 0-(-5), 20-9, 14-10, -10-(-14)\} = \min \{5, 8, 5, 11, 4, 4\} = 4$

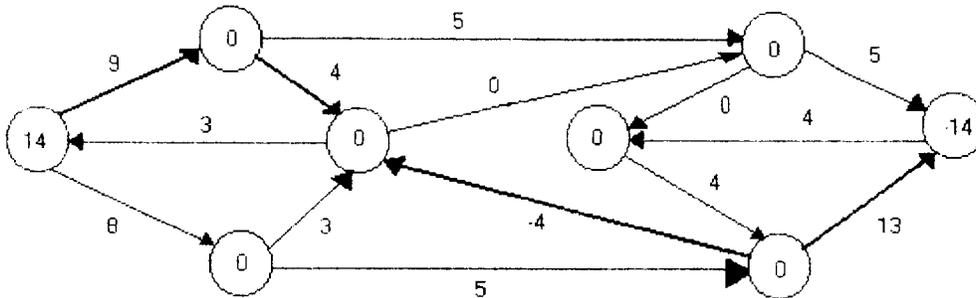


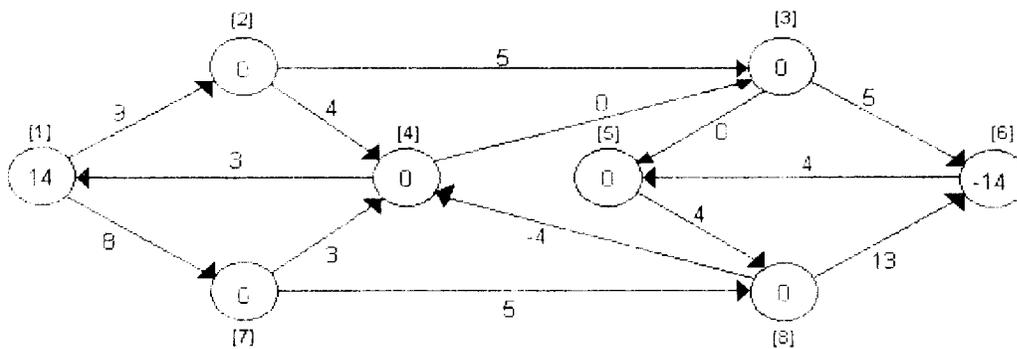
Figura 4.12

Iteración 3

i	1	2	3	4	5	6	7	8
y(i)	14	0	0	0	0	-14	0	0

$N^+ = N^- = \emptyset$ $y(i) = b(i)$ para toda i

Figura 4.13



ALGORITMO DE RECTIFICACIÓN DE FLUJO

Una alternativa natural para resolver el problema de distribución factible consiste en determinar inicialmente un flujo factible con respecto a las restricciones de oferta en los nodos y posiblemente no factible con respecto a las restricciones de capacidad sobre los arcos. Al igual que en el algoritmo de distribución factible se construye en cada iteración, un flujo que "rompa menos" las restricciones de capacidad en el sentido de que el flujo definido sobre el arco se acerque más a las cotas de capacidad en cada iteración.

Descripción

PASO 1 Determine un flujo x que es factible con respecto a las restricciones de divergencias, es decir $\text{div } x = b$

PASO 2 Defina los conjuntos de arcos:

$$A^+ = \{j \in A \mid x(j) > c^+(j)\} \quad A^- = \{j \in A \mid x(j) < c^-(j)\}$$

- a) Si $A^+ = A^- = \emptyset$, entonces x es la solución deseada, terminar.
 b) Si existe $j \in A^+$ o $j \in A^-$, colorear los arcos de la red, de la siguiente manera:

verde	si $c^-(j) < x(j) < c^+(j)$
blanco	si $x(j) \leq c^-(j)$; $x(j) < c^+(j)$
negro	si $x(j) \geq c^+(j)$; $x(j) > c^-(j)$
rojo	si $c^-(j) = x(j) = c^+(j)$

ir al paso 3

PASO 3 Aplique el lema de Minty:

- a) Si se determina un circuito elemental P compatible con la coloración, que contenga a j calcular:

$$\alpha = \min \begin{cases} c^+(j) - x(j) & j \in P^- \\ x(j) - c^-(j) & j \in P^+ \\ \alpha & \end{cases}$$

donde

$$\alpha = \min \begin{cases} c^-(j) - x(j) & j \in A^- \\ x(j) - c^+(j) & j \in A^+ \end{cases}$$

Hacer $x' = x + \alpha e_p$ e ir a 2

- b) Si se determina un corte $Q = [S, N \setminus S]$ compatible con la coloración que contenga a j terminar, ya que en este caso no existe solución al problema.

EJEMPLO 4.5

Determine un flujo factible en la red de la figura siguiente:

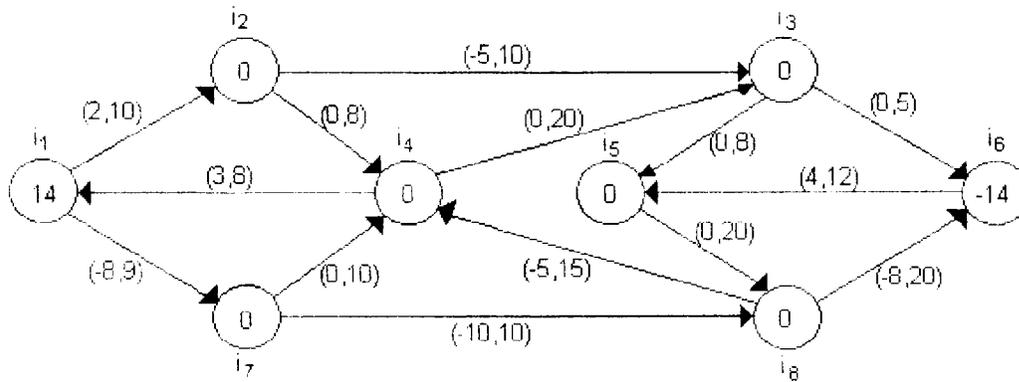


Figura 4.14

Iteración 1

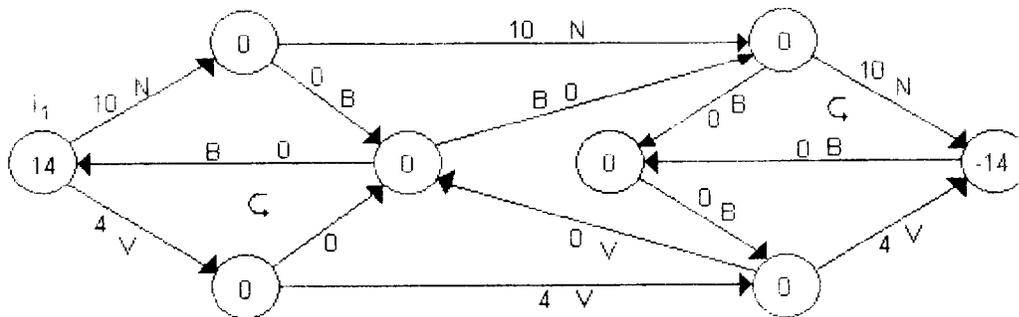


Figura 4.15

$$A^+ = \{(i_3, i_6)\} \quad A^- = \{(i_4, i_1), (i_6, i_5)\}$$

$$P_1 : i_4 \rightarrow i_1 \rightarrow i_7 \rightarrow i_4$$

$$P_2 : i_6 \leftarrow i_3 \rightarrow i_5 \rightarrow i_8 \rightarrow i_6$$

que contienen a los arcos (i_4, i_1) y (i_3, i_6) respectivamente

Para el primer circuito se obtiene:

$$\alpha_1 = \min\{9 - 4, 10 - 0, 8 - 0\} = \min\{5, 10, 8\} = 5$$

$$\alpha_2 = \min\{10 - 0, 8, 20 - 0, 20 - 4\} = \min\{10, 8, 20, 16\} = 8$$

resultando el flujo

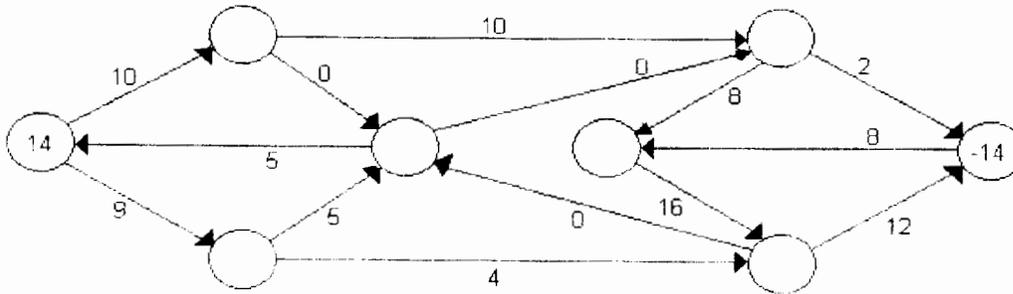


Figura 4.16

Iteración 2

$$A^+ = \phi \quad A^-(\{(i_6, i_5)\})$$

Sea $P_3: i_6 \rightarrow i_5 \rightarrow i_8 \rightarrow i_6$ que contiene a (i_6, i_5)

$$\alpha = \min \{12, 12, 8\} = 8$$

y resulta para este flujo $A^+ = A^- = \phi$

DESCRIPCIÓN DEL ALGORITMO DE ETIQUETADO DE FORD FULKERSON

El método que se describirá a continuación, resuelve el problema de flujo máximo a través de etiquetar nodos. A grandes rasgos consiste en lo siguiente: Primero, se etiqueta el nodo origen s con dos etiquetas $[s, \infty]$ indicando que en este nodo se dispone de cualquier cantidad de flujo. Después, si j es un nodo etiquetado y puede enviarse flujo del nodo j al nodo i , recibe dos etiquetas $[+, x(ji)]$. La primera etiqueta será de la forma $+j$ si $i \in P^+(j)$, es decir, si puede aumentarse flujo a través del arco j . Será de la forma $-j$ si $i \in P^-(j)$, es decir, si puede disminuirse el flujo a través del arco (j,i) . La segunda etiqueta es la cantidad de flujo que puede enviarse de j a i y se calculará por lo tanto como el mínimo entre $x(ji)$ y h donde

$$h = \begin{cases} c_{ji}^+ - x_{ji} & \text{si } i \in P^+(j) \\ x_{ji} & \text{si } i \in P^-(j) \end{cases}$$

Debe observarse que si $h = 0$ no puede enviarse flujo de j a i , por lo que no se etiquetará con $[j, x(ji)]$. Aquí c_{ji}^+ es la capacidad máxima del arco

Este proceso de asignación de etiquetas a nodos se repetirá mientras sea posible. Si el nodo destino t recibe etiquetas entonces, dado el modo de etiquetar, existe una trayectoria aumentante de s a t con una capacidad incremental igual a $x(t)$ y por lo

tanto se procederá a determinar ésta con la ayuda de la primer etiqueta, y se actualizará el flujo a través de ella. Si, por el contrario, t no recibe etiqueta alguna, entonces se habrá determinado el flujo máximo. Para justificar esto último considere el conjunto de arcos que tienen extremo inicial etiquetado y extremo final no etiquetado; este conjunto forma un corte de capacidad igual al valor del último flujo definido y por lo tanto éste es máximo.

Note la similitud de este algoritmo con la demostración del teorema de flujo máximo - corte mínimo.

Es importante observar que el algoritmo converge sólo si las capacidades para el flujo en los arcos son enteras; sin embargo, en casos donde las capacidades sean racionales, pueden transformarse éstas en enteras multiplicándolas por la potencia de 10 adecuada. De este modo, el algoritmo puede usarse también en estos casos.

ALGORITMO DE FORD Y FULKERSON

Propósito: Determinar el flujo máximo entre el origen y destino en una red G .

Descripción

- PASO 1 Iniciar con cualquier flujo factible x
- PASO 2 Etiquetar el origen con $[s, \infty]$
- PASO 3 Elegir un nodo etiquetado y no examinado; sea j dicho nodo y sean $[\pm j, x(j)]$ sus etiquetas.
- 3.1 Para todo $i \in P^+(j)$ que no esté etiquetado y tal que $x_{ji} < c_{ji}^+$ asignar la etiqueta $[+j, x(i)]$, donde $x(i) = \min \{x(j), c_{ji}^+ - x_{ji}\}$
- 3.2 Para todo $i \in P^-(j)$ que no este etiquetado y tal que $c_{ji}^+ > 0$ asignar la etiqueta $[-j, x(i)]$, donde $x(i) = \min \{x(j), x_{ij}\}$
- Se puede decir ahora que el nodo j ha sido examinado
- PASO 4 Repetir el paso 3 hasta que suceda:
- 4.1 El nodo destino t no tiene etiqueta y todos los nodos etiquetados han sido examinados. Terminar, ya que el flujo factible x es máximo o
- 4.2 El nodo t recibe etiqueta. Ir al paso 5
- PASO 5 Sea $y = t$
- 5.1 Si la etiqueta de y es de la forma $[+z, x(y)]$, hacer
$$x_{zy} = x_{zy} + x(y)$$
- 5.2 Si la etiqueta de y es de la forma $[-z, x(y)]$ hacer
$$x_{yz} = x_{yz} - x(y)$$
- PASO 6 Si $z = s$, borrar todas las etiquetas y regresar al paso 2 actualizando el flujo en la trayectoria.
- Si $z \neq s$, hacer $y = z$ y regresar al paso 5

EJEMPLO 4.6

Determine el flujo máximo de s a t en la siguiente red usando el algoritmo de Ford y Fulkerson.

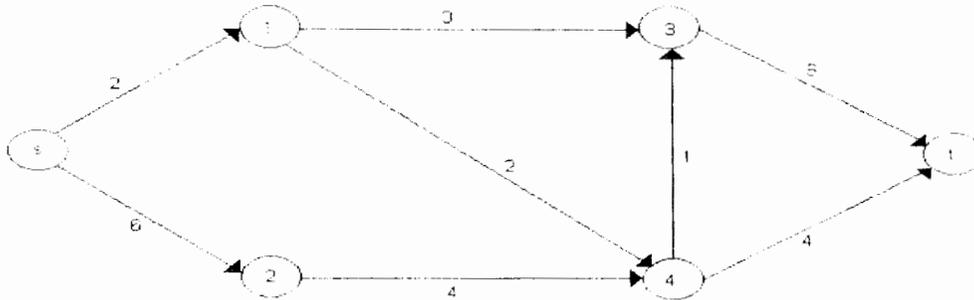


Figura 4.17

PASO 1

Como se puede observar de la figura 4.17 todos los arcos tienen al cero como cota inferior, es por esto que los números sobre los arcos lo que representan son las capacidades máximas de cada uno. Así que se propone el flujo factible inicial $x = 0$.

PASO 2

Primero se etiqueta el nodo s con $[s, \infty]$, después se calculan las otras etiquetas conforme a los pasos 3 y 4 del algoritmo.

PASO 3

Se elige el nodo s puesto que está etiquetado pero no examinado. Los sucesores i de s no etiquetados y tales que $x_{si} < q_{si}$ son 1 y 2 por lo cual

1 recibe etiqueta $[+s, \min\{\infty, 2\}]$

2 recibe etiqueta $[+s, \min\{\infty, 6\}]$

El nodo s ha sido examinado.

Se repite el paso 3.

PASO 4

Se elige el nodo 1 puesto que está etiquetado pero no examinado, los sucesores i de 1 no etiquetados y tales que $x_{1i} < c_{1i}$ son 3 y 4 por lo que

PASO 5

3 recibe etiqueta $[+1, \min\{2, 3\}]$

4 recibe etiqueta $[+1, \min\{2, 2\}]$

El nodo 1 ha sido examinado.

Iteración 1

Se elige el nodo 4

El sucesor i de 4 no etiquetado y tal que $x_{4i} < c^+_{4i}$ es t , de donde:

t recibe etiqueta $[+4,2]$

El nodo 4 ha sido examinado.

Puesto que el nodo t recibió etiqueta existe una trayectoria aumentante de s a t . Se determina ésta y se actualiza el flujo a través de ella en los pasos 5 y 6, ver figura 4.18

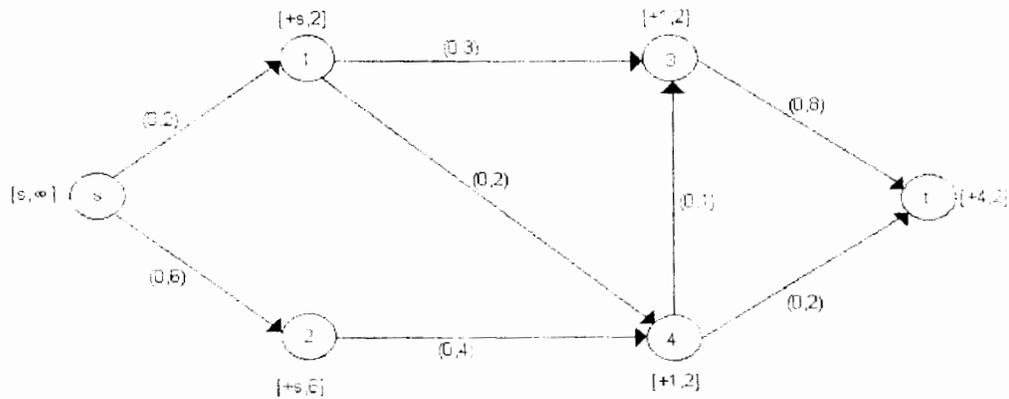


Figura 4.18

$y = t$, su etiqueta es $[+4, 2]$ entonces $x_{4t} = 2$
 $y = 4$, su etiqueta es $[+1, 2]$ entonces $x_{14} = 2$
 $y = 1$, su etiqueta es $[+s, 2]$ entonces $x_{s1} = 2$

Con este nuevo flujo de valor 2 se realiza otra iteración.

Iteración 2

Las etiquetas resultantes de esta iteración se muestran en la siguiente red:

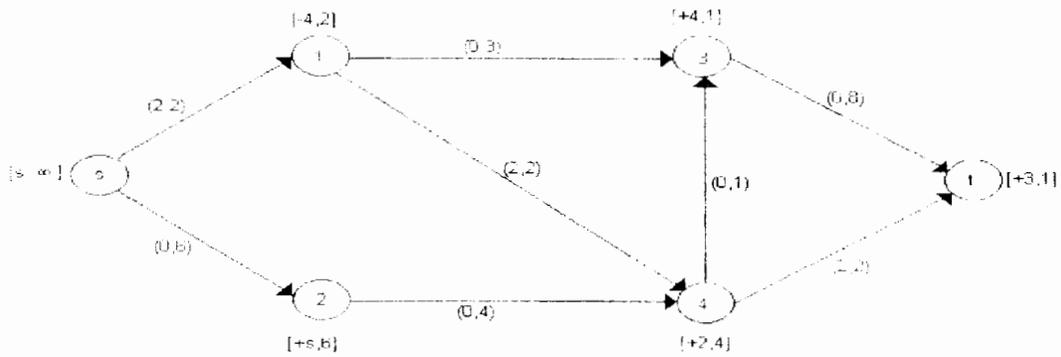


Figura 4.19

De nuevo, se etiquetó s con $[s, \infty]$, después:

Se elige el nodo s

El vecino i de s no etiquetado y tal que $x_{si} < c_{si}$ es 2 por lo cual

$$2 \text{ recibe etiqueta } [s, \min\{\infty, 6\}]$$

El nodo s ha sido examinado.

Se elige el nodo 2

El único sucesor no etiquetado de 2 es 4 y $x_{24} < c_{24}$ por lo que

$$4 \text{ recibe etiqueta } [+2, \min\{6, 4\}]$$

El nodo 2 ha sido examinado.

Se elige el nodo 4

El sucesor i de 4 no etiquetado y tal que $x_{4i} < c_{4i}$ es 3, de donde:

$$3 \text{ recibe etiqueta } [+4, \min\{4, 1\}]$$

El predecesor de 4 no etiquetado es 1 y $x_{14} > 0$ de donde

$$1 \text{ recibe etiqueta } [-4, \min\{4, 2\}]$$

El nodo 4 ha sido examinado.

Se elige el nodo 3

El sucesor no etiquetado de 3 es t y $x_{3t} < q_{3t}$, de donde

t recibe etiqueta $[+3, \min\{1,8\}]$

El nodo 3 ha sido examinado.

Puesto que t recibió una etiqueta, existe una trayectoria aumentante de s a t con capacidad incremental igual a 1. Se determina ésta y se actualiza el flujo a través de ella:

$y = t$, su etiqueta es $[+3, 1]$ entonces $x_{3t} = 1$
 $y = 3$, su etiqueta es $[+4, 1]$ entonces $x_{43} = 1$
 $y = 2$, su etiqueta es $[+s, 6]$ entonces $x_{s2} = 1$

Con este nuevo flujo de valor 3 se realiza otra iteración.

Iteración 3

Nuevamente las etiquetas asignadas se muestran en la red.

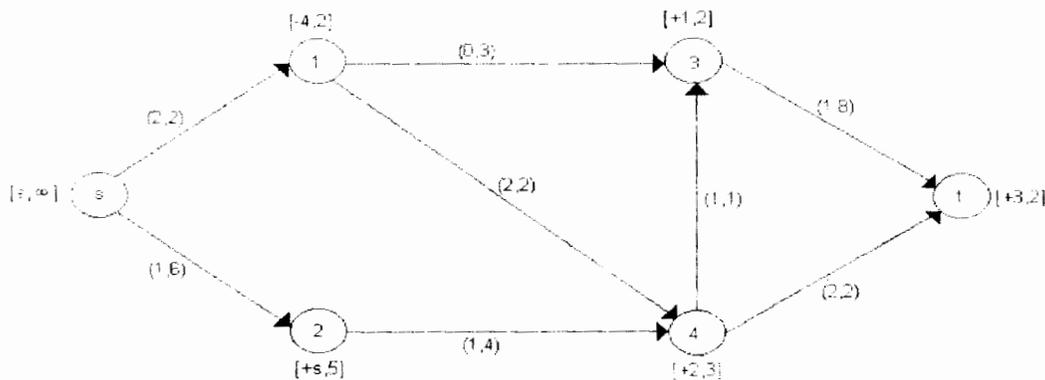


Figura 4.20

De nuevo, se etiquetó s con $[s, \infty]$, después:

Se elige el nodo s

El sucesor de s no etiquetado y tal que $x_{s2} < q_{s2}$ es 2 por lo cual

2 recibe etiqueta $[+s, \min\{\infty, 6-1\}]$

El nodo s ha sido examinado.

Se elige el nodo 2

El único sucesor no etiquetado de 2 es 4 y $x_{24} < q_{24}$ por lo que

4 recibe etiqueta $[+2, \min\{5, 4-1\}]$

El nodo 2 ha sido examinado.

Se elige el nodo 4

El predecesor de 4 no etiquetado es 1 y $x_{14} > 0$ de donde

1 recibe etiqueta $[-4, \min\{3, 2\}]$

El nodo 4 ha sido examinado.

Se elige el nodo 1

El sucesor no etiquetado de 1 es 3 y $x_{13} < q_{13}$, de donde

3 recibe etiqueta $[+1, \min\{2, 3\}]$

El nodo 1 ha sido examinado.

Se elige el nodo 3

El sucesor no etiquetado de 3 es t y $x_{3t} < q_{3t}$, de donde

t recibe etiqueta $[+3, \min\{2, 8-1\}]$

El nodo 3 ha sido examinado.

Puesto que t recibió una etiqueta, existe una trayectoria aumentante de s a t con capacidad incremental igual a 2. Se determina ésta y se actualiza el flujo a través de ella:

$y = t$, su etiqueta es $[+3, 2]$ entonces $x_{3t} = 1 + 2 = 3$

$y = 3$, su etiqueta es $[+1, 2]$ entonces $x_{13} = 2$

$y = 1$, su etiqueta es $[-4, 2]$ entonces $x_{14} = 2 - 2 = 0$

$y = 4$, su etiqueta es $[+2, 3]$ entonces $x_{24} = 1 + 2 = 3$

$y = 2$, su etiqueta es $[+s, 5]$ entonces $x_{s2} = 1 + 2 = 3$

Con este nuevo flujo de valor 5 se realiza otra iteración.

Iteración 4

Nuevamente las etiquetas asignadas se muestran en la red.

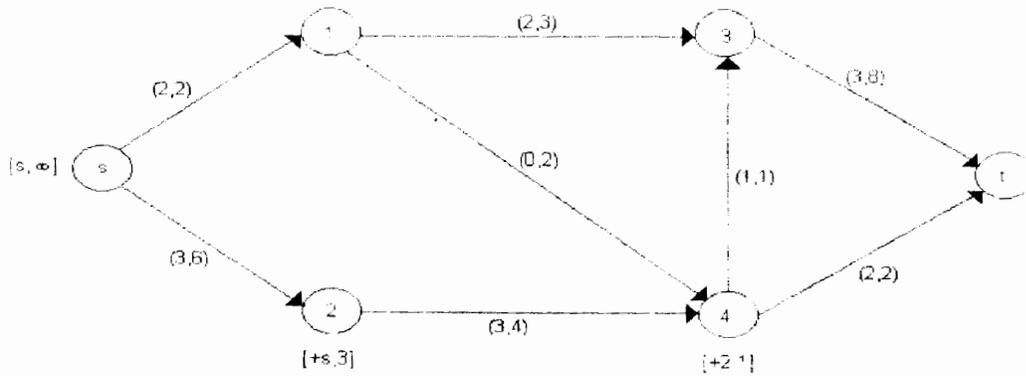


Figura 4.21

De nuevo, la etiqueta de s es $[s, \infty]$, después:

Se elige el nodo s

El sucesor de s no etiquetado y tal que $x_{s2} < c_{s2}$ es 2 por lo cual

$$2 \text{ recibe etiqueta } [+s, \min\{\infty, 6-3\}]$$

El nodo s ha sido examinado.

Se elige el nodo 2

El único sucesor no etiquetado de 2 es 4 y $x_{24} < c_{24}$ por lo que

$$4 \text{ recibe etiqueta } [+2, \min\{3, 4-3\}]$$

El nodo 2 ha sido examinado.

Se elige el nodo 4

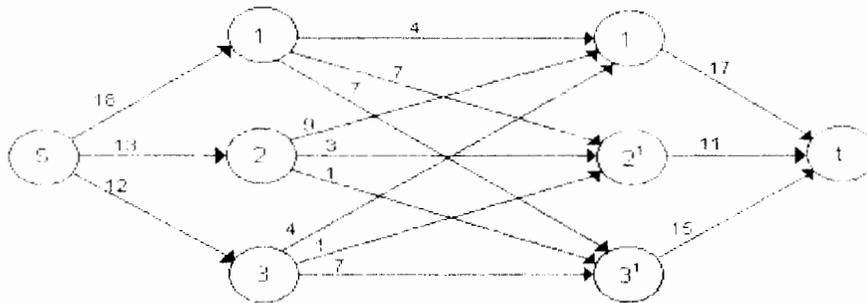
Ningún vecino de 4 puede recibir etiqueta.

Todos los nodos etiquetados han sido examinados y t no recibió etiqueta. Por lo tanto, el último flujo es máximo. Por otro lado, el conjunto de nodos etiquetados es $N = \{s, 2, 4\}$; por lo tanto el corte mínimo es $(N, N') = \{(s, 1), (4, 3), (4, t)\}$. Observe que su capacidad es $Q(N, N') = 2 + 1 + 2 = 5$ que es igual al valor del flujo máximo.

EJEMPLO 4.8

Un gran número de personas viajan en automóvil de la ciudad A a la ciudad B. Las rutas posibles se muestran en la red siguiente. El departamento de policía de caminos desea construir suficientes casetas de inspección de tal manera que todo automóvil pase por al menos una de ellas en su trayectoria de A a B. El costo de construcción de las casetas varía según su localización; el costo asociado con cada tramo se proporciona en cada arco de la red (en millones de pesos). Determine dónde deberán colocarse las casetas si se desea incurrir en el mínimo costo. La red que representa la problemática se muestra en la figura 4.21

Figura 4.22



Observe que, en términos de redes, se desea obtener un conjunto de arcos (tramos donde deberán construirse las casetas) de manera tal que si eliminamos esos arcos de la red ya no existan caminos de A a B (es decir, todo automóvil debe utilizar alguno o algunos de estos arcos en su trayectoria); este conjunto de arcos forma entonces una cortadura de la red. Por otro lado, se desea incurrir en el mínimo costo; debe determinarse entonces la cortadura de mínimo costo. Si se consideran los costos de los arcos como capacidades de un cierto flujo a través de ellos y se determina la cortadura de capacidad mínima, es claro que ésta corresponde a la de mínimo costo. Para ello se determinará el flujo máximo de A a B mediante el algoritmo de Ford y Fulkerson.

Iteración 1

Se aplicará el algoritmo con un flujo factible inicial de valor 11 definido en la siguiente red. Los números asociados a cada arco son el flujo a través de él y su capacidad. Las etiquetas asignadas a cada nodo durante esta iteración se muestran en la siguiente red:

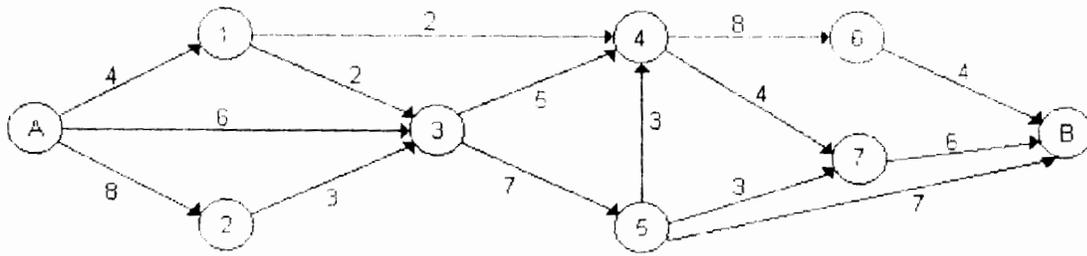


Figura 4.23

Puesto que $f(B)=2$ existe una cadena aumentante de capacidad incremental igual a 2; ésta es A, 3, 5, B. Actualizando el flujo a través de esta cadena se obtiene el definido en la siguiente red.

Iteración 2

De nuevo se asocia a cada nodo su correspondiente etiqueta.

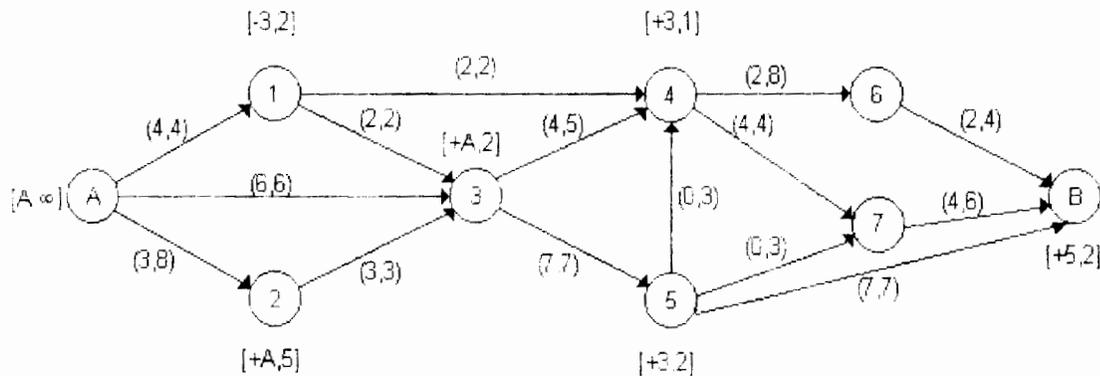


Figura 4.24

En esta iteración el nodo B no recibió etiqueta; por lo tanto este flujo de valor 13 es el máximo. El conjunto de nodos no etiquetados es $N=\{A, 2\}$ por lo que la cortadura de capacidad mínima es $(N, \tilde{N})=\{(A, 1), (A, 3), (2, 3)\}$ de capacidad $q(N, \tilde{N})=4+6+3=13$. Por tanto, la solución al problema del departamento de policía de caminos es construir las casetas de inspección en los tramos (A,1), (A, 3) y (2, 3) con un costo de construcción de 13 millones de pesos.

EJEMPLO 4.9

Determine el flujo máximo en la siguiente red a partir del flujo factible definido en ella.

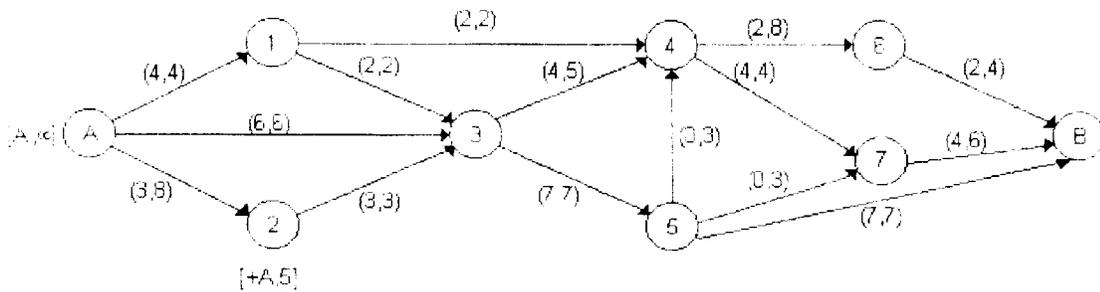


Figura 4.25

Los números asociados a cada arco son el flujo a través de él y su capacidad. Obsérvese primeramente que el valor de este flujo es $v=4$.

Iteración 1

Las etiquetas asociadas a los nodos durante esta iteración se muestran en la siguiente red.

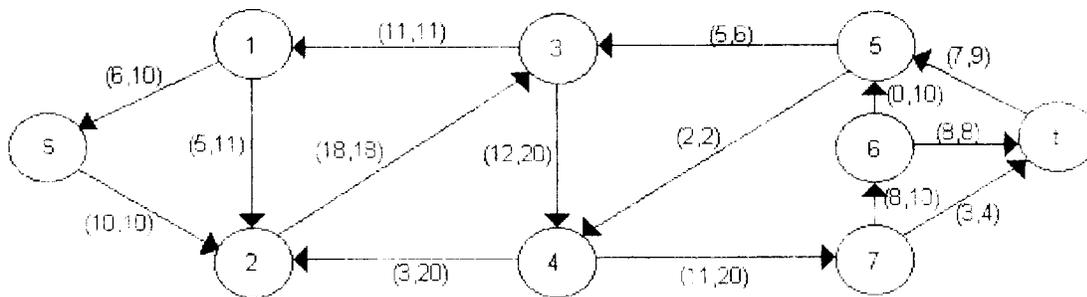


Figura 4.26

Puesto que $f(t)=5$ se ha determinado una cadena aumentante con capacidad incremental igual a 5 unidades de flujo, ésta es: $s, 1, 3, 5, t$. Actualizando el flujo se obtiene el definido en la siguiente red.

Iteración 2

De nuevo se asocia a cada nodo su correspondiente etiqueta.

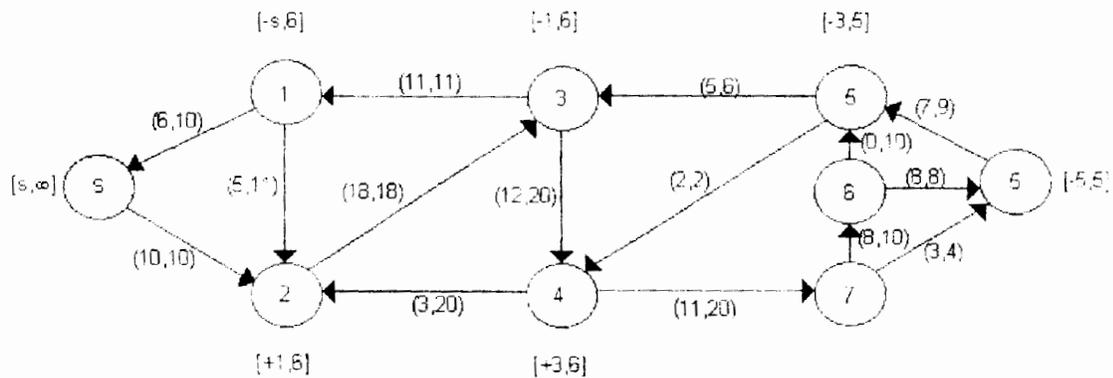


Figura 4.27

Puesto que $f(t)=1$ se ha determinado otra cadena aumentante con capacidad incremental igual a 1; ésta es: $s, 1, 2, 4, 7, t$. Actualizando el flujo se obtiene el definido en la siguiente red.

Iteración 3

De nuevo se asocia a cada nodo su etiqueta.

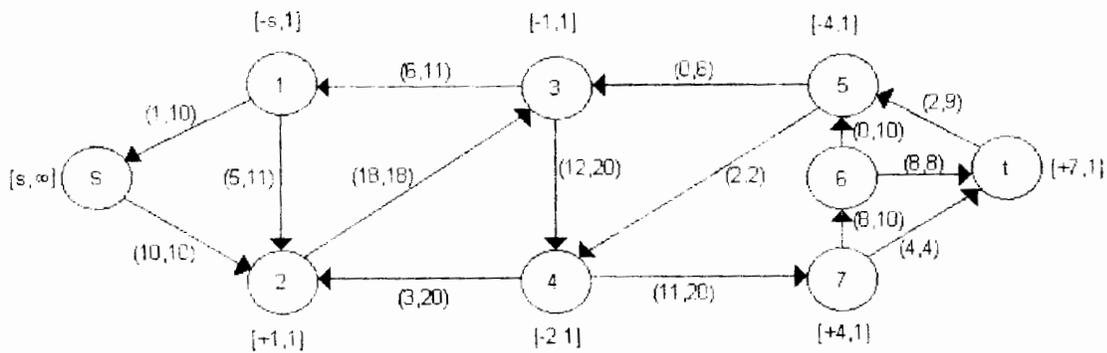


Figura 4.28

En esta iteración t no pudo ser etiquetado. Luego, este flujo de valor 10 es máximo. Puesto que el conjunto de nodos etiquetados es $N=\{s\}$, la cortadura mínima está dada por: $(N, \bar{N})=\{(s, 2)\}$ de capacidad igual a 10.

4.5 NOTAS HISTÓRICAS

El artículo de Ford y Fulkerson (1956) en el problema de flujo máximo estableció el celebrado teorema de flujo máximo - corte mínimo. Fulkerson y Dantzig (1955) y Elías, Feinstein y Shannon (1956) establecieron independientemente este resultado. Ford y Fulkerson y Elías, et. al. resolvieron el problema de flujo máximo a través de algoritmos de trayectorias aumentantes, mientras Fulkerson y Dantzig (1955) lo resolvieron usando el método simplex especializado para redes.

Desafortunadamente, el algoritmo de Ford y Fulkerson corre en tiempo pseudopolinomial; más aún, para redes con capacidades irracionales el algoritmo desarrolla una sucesión infinita de trayectorias aumentantes que pueden converger a un valor diferente del valor de máximo flujo. Existen varias versiones mejoradas que superan esta limitación.

En la primer versión del teorema, todos los intervalos de capacidad se toman de la forma $[0, c]$, y aparecen nodos sencillos en lugar de conjuntos de nodos N^+ y N^- . Tales restricciones no representan una pérdida significativa de generalidad, pues se pueden usar trucos para reformular el problema; sin embargo, el uso de intervalos arbitrarios y nodos múltiples fuente y sumidero hacen más fácil su aplicación.

En años recientes J. Edmonds, y R.M. Karp (1972) demostraron que si se usa una búsqueda de primer amplitud en la subrutina de pintado de la red, el algoritmo terminará (aún sin conmensurabilidad de los datos), y de hecho los pasos requeridos son del orden $O(|N| |A|^2)$ (por lo que es para el caso digráfico no peor que $O(|N|^5)$). El efecto de la búsqueda de primer amplitud consiste en seleccionar en cada iteración una trayectoria aumentante con la menor cantidad de arcos posible. Lo mismo se puede cumplir usando una versión de trayectorias múltiples de la subrutina de pintado de la red.

El teorema de distribución factible fue publicado por primera vez por Gale (1957) en el caso en que $c^-(j) \leq 0 \leq c^+(j)$ y por Hoffman (1960) en el caso de intervalos de capacidad cerrados arbitrarios. Ambos resultados en realidad datan de un artículo que no se publicó elaborado por ambos autores en 1956.

El algoritmo de rectificación de flujo se puede ver como una simplificación de un método de Minty (1962).

CAPÍTULO 5 ÁRBOL DE EXPANSIÓN MÍNIMA

5.1 INTRODUCCIÓN

Como ya se ha mencionado en el Capítulo 1, el problema de árbol de expansión mínima juega un papel importante en los problemas de redes de flujo. Como veremos en el siguiente capítulo, para resolver el problema de ruta más corta se necesita construir un árbol con raíz; asimismo, para resolver redes de flujo a costo mínimo se usan árboles para encontrar la solución. En este capítulo veremos el modelo de árbol de expansión mínima y definiremos un árbol de expansión T de una red G como una subgráfica acíclica conectada que conecta a todos los nodos.

ÁRBOLES DE EXPANSIÓN

Con mayor exactitud podemos definir un árbol de la siguiente manera:

DEFINICIÓN:

Una red se dice **árbol** si es conectada y sin circuitos. Por ejemplo, en la red de la figura 5.1 se muestra un árbol.

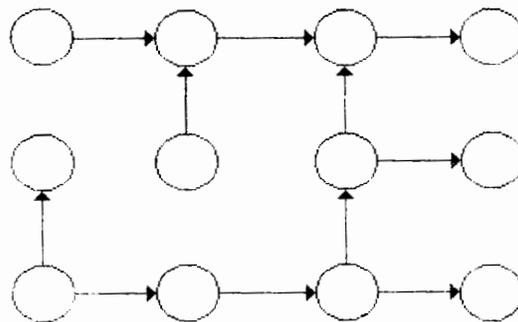


Figura 5.1

La definición anterior excluye aquellas redes que constan de un solo nodo.

Hay una serie de características de un árbol que podemos enunciar en el siguiente teorema:

TEOREMA 5.1

Un árbol es una red (o gráfica) caracterizada por cualquiera de los siguientes postulados:

- a) Conectada y sin circuitos.
- b) Existe una trayectoria única entre cada par de nodos.
- c) No tiene circuitos, pero exactamente uno se forma al añadir un arco.
- d) Es conectada, pero deja de serlo si algún arco se elimina.

Demostración:

a \rightarrow b) Es inmediato, pues si existe un par de nodos con dos trayectorias distintas que los una, se tendrá un circuito.

b \rightarrow c) Se tiene que la red no tiene circuitos. Si $j \leq (i, i')$ y hay una trayectoria única al añadir un arco, tendremos un circuito único.

c \rightarrow d) Suponemos que no es conectada, entonces se añade un arco que une dos componentes no conectados y no se forma un circuito (lo que es imposible ya que por hipótesis al añadir un arco se forma un circuito).

d \rightarrow a) Es inmediato que es conectada; supongamos que tiene circuitos, entonces podemos eliminar un arco y la red no se desconecta, pero esto contradice a d) por lo cual no debe tener circuitos.

Nota: las equivalencias de árbol no requieren que el número de nodos sea finito.

COROLARIO 5.1

Sea G una red con m nodos. Entonces cualquiera de los postulados siguientes caracteriza a un árbol:

- a) G es un árbol.
- b) G tiene $m-1$ arcos y no tiene circuitos.
- c) G tiene $m-1$ arcos y es conectada.

Un nodo se dice terminal si su grado (número de arcos que inciden) es 1.

LEMA

En una red la suma de los grados de los nodos es par.

Proposición 5.1

Sea G un árbol. Entonces G tiene al menos dos nodos terminales. Ver figura 5.2

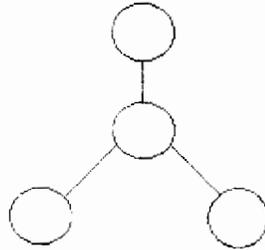


Figura 5.2

Suponga que G tiene m nodos y $m-1$ arcos. Entonces la suma de los grados de los nodos es $2(m-1)$.

Si G no tiene al menos dos nodos terminales, es sencillo concluir que la suma de los grados de los nodos será mayor que $2(m-1)$.

Proposición 5.2

Sea G un árbol. Cualquiera de las siguientes operaciones produce otro árbol.

- a) Se elimina un nodo terminal y el arco que incide.
- b) Se añade un nodo y se conecta a G con un arco.

PRUEBA PARA SABER CUANDO UNA RED ES UN ÁRBOL

Existe una forma de determinar algorítmicamente cuándo una red dada con $(|N| \leq 2)$ es un árbol. Seleccione cualquier nodo s y aplique el algoritmo de pintado de la red, con todos los arcos pintados de verde para obtener un enrutamiento máximo Θ con base en $N^+ = \{s\}$. Si éste es un enrutamiento para todo N (es decir, $\Theta(i)$ se define para cualquier $i \neq s$) y usa cada arco (esto es, cada arco j se designa como $\Theta(i)$ para algún nodo i , y es necesariamente único de acuerdo con la definición de enrutamiento) entonces la red es un árbol.

Si el enrutamiento tiene estas propiedades, entonces la red es conectada, ya que existe una trayectoria de s a cualquier nodo, y no puede tener circuitos elementales, pero si P fuera un circuito elemental, entonces cada arco j de P estaría en $\Theta(i)$ para uno de los dos nodos de P adyacente a él; ya que un circuito elemental tiene igual

número de nodos que de arcos, esto implicaría para cada nodo i de P que $\Theta(i)$ está definido y es un arco de P , entonces el enrutamiento no proporciona una trayectoria que une cualquier nodo de P con s . Por lo que la red debe ser un árbol.

Por otro lado, si la red es un árbol y se aplica este procedimiento, cada nodo se alcanzará debido a la conexidad. Además los nodos y los arcos usados por el enrutamiento constituirán un árbol por el argumento ya dado, y por lo tanto cualquier par de nodos i, i' se pueden unir por una trayectoria elemental limitada a los arcos que usan el enrutamiento.

Lo anterior muestra que la noción de un enrutamiento con base en s es equivalente a la de árbol con raíz, donde el nodo s es la raíz del árbol. Un árbol con raíz también recibe el nombre de arborescencia.

BOSQUES Y ÁRBOLES DE EXPANSIÓN

Un conjunto de arcos $F \subset A$ se dice que forma un bosque en una red G si cada componente de la subred compuesta por arcos de F y los nodos incidentes a ella es un árbol. En otras palabras, F es un bosque si y sólo si no se incluyen circuitos elementales en F . Por ejemplo, los arcos más gruesos en la figura 5.3 forman un bosque, si incluimos los nodos incidentes se tiene en cada caso un árbol.

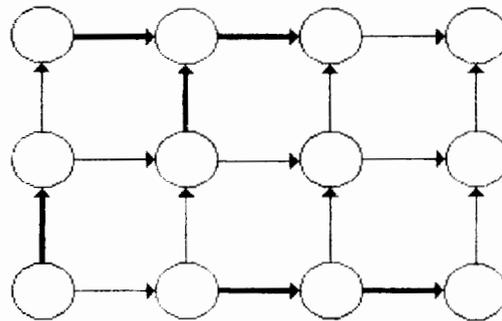


Figura 5.3

F forma un bosque en G si y sólo si es el conjunto de arcos usado por algún enrutamiento Θ (no necesariamente todos los de N con base $N^+ \subset N$).

La conexión entre bosques y flujos es la siguiente: Un conjunto de arcos F forma un bosque si y sólo si no existe una circulación distinta de cero cuyo soporte esté incluido en F , esto es, ninguna circulación x tal que $x(j) = 0$ para toda j que no esté en F pero $x(j) \neq 0$ para al menos una $j \in F$.

Las circulaciones x se caracterizan como soluciones del sistema.

$$\sum_{j \in A} e(i, j)x(j) = 0 \quad \text{para toda } i \in N$$

Sea $e^j(i) = e(i, j)$, así e^j denota a la "columna" de E correspondiente al arco j . El sistema entonces se convierte en

$$\sum_{j \in A} x(j)e^j = 0$$

En otras palabras, una circulación distinta de cero puede interpretarse como una elección de coeficientes $x(j)$ que producen una relación de dependencia lineal entre las columnas correspondientes de E . De aquí se sigue que un conjunto de arcos F forma un bosque en G si y sólo si las columnas correspondientes de la matriz de incidencia de G son linealmente independientes.

Es natural de este hecho pensar en el estudio de conjuntos de columnas que forma una base para el espacio de las columnas de E (es decir, el subespacio lineal de R^N generado por todas las e^j , $j \in A$), ya que entre otras cosas éste da una descripción del rango de E . Un *bosque máximo en G* se define como un bosque que no está estrictamente contenido en otro bosque de G .

Un árbol de expansión de G es un árbol que pasa por cada nodo de G , como en la figura 5.4

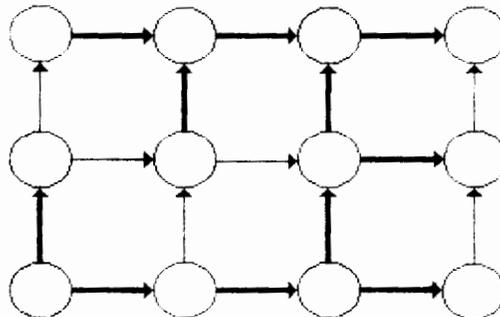


Figura 5.4

TEOREMA 5.2 : TEOREMA DE EXPANSIÓN

Para cualquier red G , las siguientes propiedades de un conjunto de arcos $F \subset A$ son equivalentes:

1. F forma un bosque máximo para G .
2. F forma un árbol de expansión para cada componente de G que no sea un nodo único.
3. F forma un bosque tal que $|F| = |N| - p$ donde p es el número de componentes de G .
4. Las columnas e^j de la matriz de incidencia E correspondiente a los arcos $j \in F$ forman una base para el espacio de las columnas de E .

Demostración:

La equivalencia de las propiedades 1 y 4 es clara de las observaciones precedentes acerca de la independencia lineal.

1 \rightarrow 2)

Note de la maximalidad de F que cada componente de G que no sea un nodo aislado, debe incluir un arco de F y por lo tanto uno de los árboles componentes de F .

Sea T tal árbol y sea S su conjunto de nodos, la componente de G que incluya a T no puede tener nodos fuera de S . Si así fuera entonces el corte $Q = [S, N \setminus S]$ sería no vacío y cualquier arco en Q se podría agregar a F sin romper la propiedad de bosque. Así, cada árbol componente del bosque corresponde a una componente de G que tiene el mismo conjunto de nodos que el árbol, y cada componente que no es un nodo aislado corresponde de esta forma a un árbol.

2 \rightarrow 3)

Es inmediata del hecho de que el número de arcos en un árbol siempre es igual al número de nodos menos uno, de aquí se generaliza que para cualquier componente G' de G :

$$\begin{array}{rcccc} \text{Número de arcos} & & \text{Número de nodos} & & \text{Número de árboles} & & \text{(Número de} \\ \text{en un bosque que} & = & \text{incidentes al} & - & \text{del bosque que} & \leq & \text{nodos en} \\ \text{está en } G' & & \text{bosque que está} & & \text{está en } G' & & \text{ } G') - 1 \end{array}$$

Entonces cualquier conjunto de arcos F que forman un bosque en G satisface:

$$|F| \leq |N| - [\text{número de componentes de } G]$$

y se cumple que F es máximo, así la propiedad 3 implica la propiedad 1.

COROLARIO 5.2

En una red conectada (con al menos un arco) un bosque máximo es lo mismo que un árbol de expansión.

En programación lineal la propiedad 4 tiene un significado especial, ya que en programación los problemas involucran la restricción $Ex = b$, que se puede escribir:

$$\sum_{j \in A} x(j)e^j = b$$

Es importante saber cuáles conjuntos $F \subset A$ son tales que, los elementos e^j para $j \in F$ forman una base del espacio de las columnas (suponiendo que contiene a b). Cada uno de éstos determina una única x que satisface la ecuación y teniendo $x(j) = 0$ para toda j que no está en F . El método simplex depende de generar una secuencia de tales conjuntos F . Geométricamente, estos conjuntos pueden identificarse con los árboles de expansión para la red, suponiendo que la red es conectada.

Asociado con el concepto de árbol con raíz o arborescencia se encuentra el problema de ruta más corta que se aborda en el anexo A y se presentan algoritmos de solución para las distintas variantes que hay, también se desarrollan ejemplos para cada algoritmo.

5.2 ÁRBOL DE EXPANSIÓN MÍNIMA.

Una característica especial de los problemas de árbol de expansión mínima es que los arcos no son dirigidos ((i, j) ó (j, i) se refieren al mismo arco).

Los tres algoritmos que veremos en este capítulo son el Algoritmo de Kruskal, el Algoritmo de Prim y el Algoritmo de Sollin todos con la característica de ser algoritmos de tipo "glotón", en el sentido de que en cada paso aumentan un arco de costo mínimo como candidato de la lista, siempre y cuando este arco no forme ningún circuito con los arcos ya seleccionados. Los tres algoritmos mantienen un bosque que contiene arcos que ya han sido seleccionados y se aumenta uno o más arcos para aumentar el tamaño del bosque.

Para el algoritmo de Kruskal, la lista de candidatos es la red completa; para el algoritmo de Prim, el bosque es un único árbol más un conjunto de nodos aislados y la lista de candidatos contiene todos los arcos entre el árbol único y los nodos que no están en él. El algoritmo de Sollin es un híbrido que mantiene varias componentes en el bosque, como el algoritmo de Kruskal, pero se aumentan varios arcos en cada iteración, seleccionando (como en el algoritmo de Prim) el arco de costo mínimo que conecta cada componente del bosque a los nodos que no están en esa componente.

Ya que los algoritmos tipo glotón se usan en muchos otros contextos de optimización discreta, se puede ver que generalizaciones del algoritmo de Kruskal resuelven una amplia clase de problemas combinatorios abstractos como los de optimización en matroides.

La programación matemática tiene otra forma útil de ver el problema de árbol de expansión mínima: se puede formular un problema de árbol de expansión mínima como un modelo de programación entera y usar argumentos de programación lineal para establecer otra prueba de la validación del algoritmo de Kruskal. Esta discusión sirve para varios propósitos:

1. Da otra visión útil de los árboles de expansión mínima.
2. Ilustra una técnica de prueba, vía programación lineal que ha demostrado ser poderosa en el campo de la optimización combinatoria.
3. Proporciona un puente entre el problema de árbol de expansión mínima y poliedros combinatorios (o poliedros enteros).

5.3 ALGORITMO DE KRUSKAL

Propósito: Determinar el árbol de expansión cuyo "costo o peso" es mínimo en una red conectada G con N nodos y función de costo $C: A \rightarrow R$ conocida.

Descripción

- PASO 1 (Inicio) Ordene el conjunto de arcos en forma creciente respecto a la función de costos. Sean j_1, j_2, \dots, j_m los arcos ordenados, hacer $k = 0$, $l = 1$ y $\underline{A} = \phi$
- PASO 2 (Añadir arco) Si el arco j_i no forma circuito con el conjunto de arcos de \underline{A} entonces $\underline{A} = \underline{A} \cup \{j_i\}$ y hacer $k = k + 1$ y vaya a 3. De otra manera siga a 3 y no aumente k .
- PASO 3 (Criterio de Terminación) Si $k < n-1$ hacer $l = l+1$ y regresar a 2. En caso contrario $T = [N, \underline{A}]$ es el árbol de expansión mínima.

EJEMPLO 5.1

Considere la red de la figura 5.5 y determine el árbol de peso mínimo mediante el algoritmo de Kruskal.

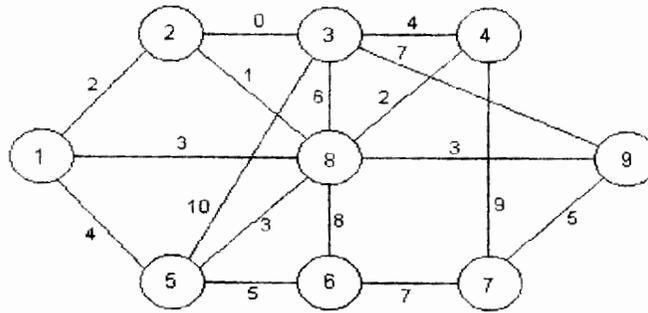


Figura 5.5

Iteración 1

Paso 1: Se ordenan los arcos:

- | | | | |
|-------------------|-------------------|-------------------|-------------------|
| $j_1 = (2, 3)$ | $j_2 = (2, 8)$ | $j_3 = (1, 2)$ | $j_4 = (8, 4)$ |
| $j_5 = (1, 8)$ | $j_6 = (5, 8)$ | $j_7 = (8, 9)$ | $j_8 = (1, 5)$ |
| $j_9 = (3, 4)$ | $j_{10} = (5, 6)$ | $j_{11} = (7, 9)$ | $j_{12} = (3, 8)$ |
| $j_{13} = (6, 7)$ | $j_{14} = (3, 9)$ | $j_{15} = (8, 6)$ | $j_{16} = (4, 7)$ |
| $j_{17} = (3, 5)$ | | | |

$\underline{A} = \{j_1\}$, $k = 1$. Como $k < n-1 = 8$, aún no se tiene el árbol.

En el siguiente cuadro se presenta un resumen de las operaciones realizadas en cada iteración. En la primera columna aparece el número de iteración, en la segunda, el arco que se agrega al conjunto \underline{A} para ir formando el árbol; por último, en la tercera columna se tiene el número de arcos que ya se han incluido en el árbol hasta esa iteración

Número de iteración (1)	Arco agregado a \underline{A}	Valor de K
1	$j_1=(2,3)$	1
2	$j_2=(2,8)$	2
3	$j_3=(1,2)$	3
4	$j_4=(8,4)$	4
5	ninguno, j_5 forma un circuito	4
6	$j_6=(5,8)$	5
7	$j_7=(8,9)$	6
8	ninguno, j_8 forma un circuito	6
9	ninguno, j_9 forma un circuito	6
10	$j_{10}=(5,6)$	7
11	$j_{11}=(7,9)$	8

Puesto que en la iteración 11 se tiene $k = 8 = n - 1$, se ha obtenido la solución óptima; misma que está dada por la red de la figura 5.6 donde $\underline{A} = \{j_1, j_2, j_3, j_4, j_6, j_7, j_{10}, j_{11}\}$

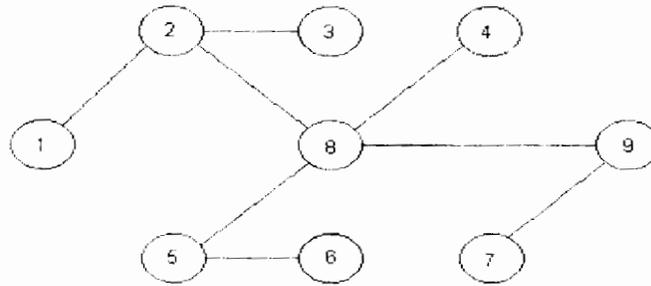


Figura 5.6

Este árbol de expansión tiene un peso de:

$$0 + 1 + 2 + 2 + 3 + 3 + 5 + 5 = 21$$

EJEMPLO 5.2

Considere la red de la figura 5.7 y determine el árbol de peso mínimo mediante el algoritmo de Kruskal.

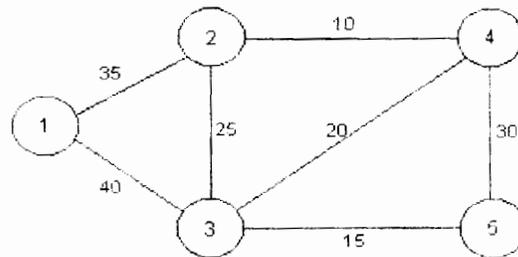


Figura 5.7

Paso 1

Se ordena la lista de arcos

j	Arco	Valor
1	(2, 4)	10
2	(3, 5)	15
3	(3, 4)	20
4	(2, 3)	25
5	(4, 5)	30
6	(1, 2)	35
7	(1, 3)	40

Paso 2		Paso 3	
No. de iteración	Arista a agregar	¿Hace ciclo?	Valor de K
1	(2,4)	No	1
2	(3,5)	No	2
3	(3,4)	No	3
4	(2,3)	Sí	3
5	(4,5)	Sí	3
6	(1,2)	No	4

El árbol de expansión mínima se presenta en la figura 5.8

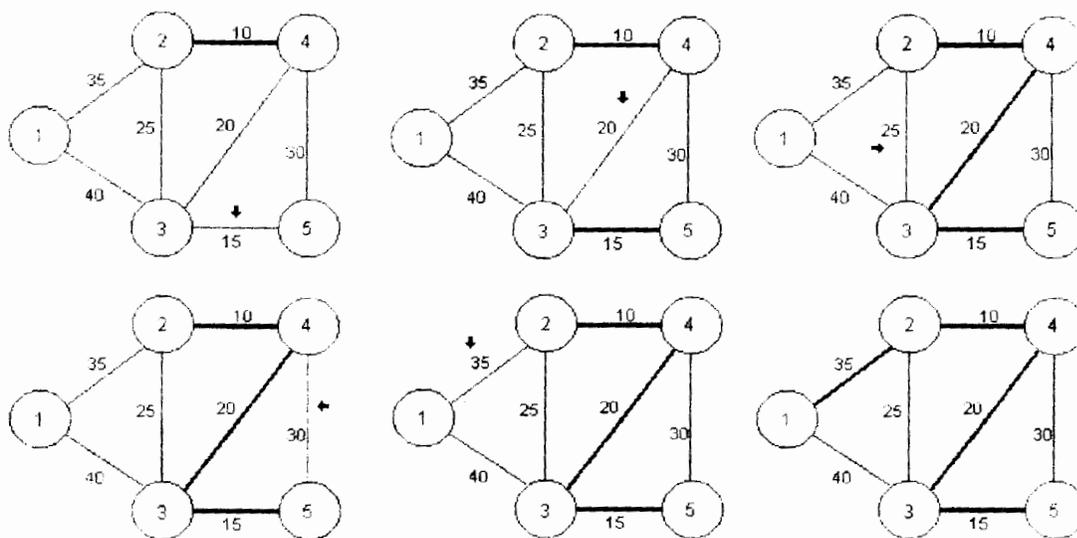


Figura 5.8

COMPLEJIDAD COMPUTACIONAL DEL ALGORITMO

El tiempo de corrida del algoritmo de Kruskal se compone del tiempo para ordenar los arcos y el tiempo para detectar ciclos. Para una red arbitrariamente grande, el ordenamiento requiere $O(m \log m) = O(m \log n^2) = O(m \log n)$ de tiempo. El tiempo para detectar un ciclo depende del método que se use en este paso. Por ejemplo, la lista que se tiene en el ejemplo 2 y el segundo paso es (2, 4) y (3, 5). Se denotan estos conjuntos de nodos por los árboles N_1 y N_2 se guardan estos conjuntos en dos listas ligadas. Mientras se examina un arco (k, l) se busca en estas listas ligadas y se verifica si k y l están en la misma lista si ambos nodos están en listas distintas se agrega el arco si no, se elimina. Si se aumenta el arco se unen las listas que contienen a k y l en una sola lista; ésta estructura de datos requiere un tiempo $O(n)$. Usando esta estructura de datos el algoritmo de Kruskal corre en $O(nm)$

5.4 ALGORITMO DE PRIM

Un método alternativo de solución para el problema del árbol de peso mínimo de una red conexa con n nodos, es el algoritmo de Prim. Este algoritmo consiste en considerar, inicialmente, una red formada por cualquier nodo de la red original, después se agregará el arco de menor peso adyacente a él y su otro extremo. Luego se aumenta el arco más pequeño, que tenga exactamente un extremo en la red formada, junto con su otro extremo. Se procede de esta manera, sucesivamente, hasta tener $n-1$ arcos en la red generada.

A diferencia del algoritmo de Kruskal, la red construida en cada iteración es conexa y además tiene $n-1$ arcos, luego, esta red es un árbol expandido de la red original. Debe notarse también, que el algoritmo termina en $n-1$ iteraciones exactamente; esto constituye otra diferencia con el algoritmo de Kruskal.

ALGORITMO DE PRIM

Propósito: Determinar el árbol de expansión cuyo costo o peso es mínimo en una red conectada G con N nodos y función de peso o costo $C: A \rightarrow R$ conocida.

Descripción

PASO 1 (Inicio) Sea x_0 (arbitrario) elemento de N y $k=0$. Sea
 $N_0 = \{x_0\}$ $A_0 = \phi$

PASO 2 (Añadir un arco). Sea F_k el conjunto de arcos de A que tienen exactamente un extremo en N_k . Sea j_k el arco de costo mínimo en F_k y denote por x_k el extremo de j_k que no pertenece a F_k . Hacer
 $N_{k+1} = N_k \cup \{x_k\}$ $A_{k+1} = A_k \cup \{j_k\}$

PASO 3 Hacer $k = k+1$. Si $k < n-1$ regrese a 2. En caso contrario, termine. La red $T_{n-1} = [N_{n-1}, A_{n-1}]$ representa el árbol de expansión mínima de G .

EJEMPLO 5.3

Aplique el algoritmo de Prim y determine el árbol de expansión mínima en la red conectada de la figura 5.9

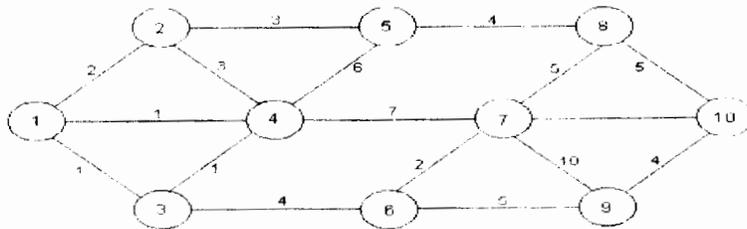


Figura 5.9

Iteración 1:

Comenzaremos con $x_0 = \{7\}$. En el siguiente cuadro se presentan resumidas, las iteraciones del algoritmo.

Número de iteración (K)	F_k	j_k	N_k	A_k
1	{{(4,7), (6,7), (7,8), (7,9)}}	(6,7)	{7,6}	{{(6,7)}}
2	{{(3,6), (6,9), (4,7), (7,8), (7,9)}}	(3,6)	{7,6,3}	{{(6,7), (3,6)}}
3	{{(1,3), (3,4), (6,9), (4,7), (7,8), (7,9)}}	(1,3)	{7,6,3,1}	{{(6,7), (3,6), (1,3)}}
4	{{(1,2), (1,4), (3,4), (6,9), (4,7), (7,8), (7,9)}}	(1,4)	{7,6,3,1,4}	{{(6,7), (3,6), (1,3), (1,4)}}
5	{{(1,2), (2,4), (4,5), (6,9), (7,8), (7,9)}}	(1,2)	{7,6,3,1,4, 2}	{{(6,7), (3,6), (1,3), (1,4), (1,2)}}
6	{{(2,5), (4,5), (6,9), (7,8), (7,9)}}	(2,5)	{7,6,3,1,4, 2,5}	{{(6,7), (3,6), (1,3), (1,4), (1,2), (2,5)}}
7	{{(5,8), (6,9), (7,8), (7,9)}}	(5,8)	{7,6,3,1,4, 2,5,8}	{{(6,7), (3,6), (1,3), (1,4), (1,2), (2,5), (5,8)}}
8	{{(6,9), (7,9), (8,10)}}	(6,9)	{7,6,3,1,4, 2,5,8,9}	{{(6,7), (3,6), (1,3), (1,4), (1,2), (2,5), (5,8), (6,9)}}
9	{{(8,10), (9,10), (7,8), (7,9)}}	(9,10)	{7,6,3,1,4, 2,5,8,9,10}	{{(6,7), (3,6), (1,3), (1,4), (1,2), (2,5), (5,8), (6,9), (9,10)}}

En la primer columna aparece el número de iteración k , en la segunda el conjunto c_k resultante de la k -ésima iteración, en la tercera el arco de peso mínimo de c_k (j_k) y por último en las columnas cuarta y quinta respectivamente, los nodos y arcos que irán formando el árbol de expansión mínima de la red.

En la iteración $k = n-1 = 9$ se observa que el árbol de peso mínimo generado por el algoritmo es como el que se muestra en la figura 5.10

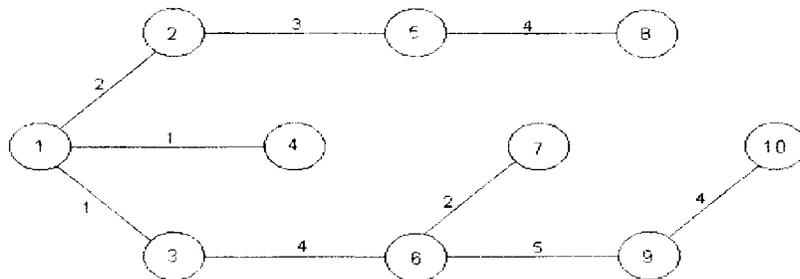


Figura 5.10

EJEMPLO 5.4

La siguiente serie de redes muestran paso a paso el uso del algoritmo de Prim para resolver el problema de encontrar un árbol de expansión mínima.

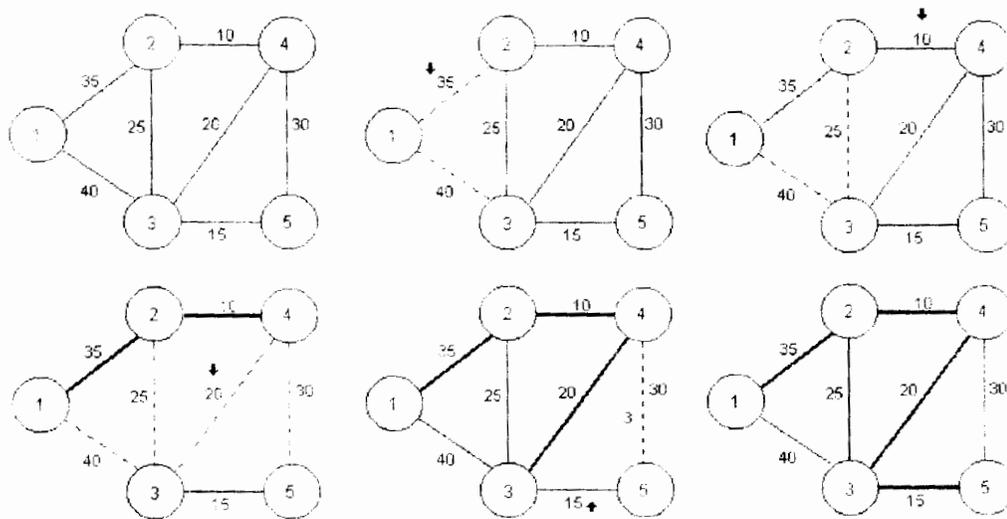


Figura 5.11

COMPLEJIDAD COMPUTACIONAL DEL ALGORITMO

Para analizar el tiempo de ejecución del algoritmo de Prim, consideramos cada una de las $n-1$ iteraciones que el algoritmo desarrolla como si aumentara un arco a la vez al árbol, hasta que se tiene un árbol de expansión con $n-1$ arcos. En cada iteración el algoritmo selecciona el arco de costo mínimo en el corte $[S, \bar{S}]$. Si podemos revisar la lista completa de arcos para identificar el arco de costo mínimo, esta operación requiere un tiempo de ejecución de $O(m)$, lo que nos da una cota de tiempo de $O(nm)$ para el algoritmo. Por lo cual podemos afirmar que el algoritmo se ejecuta en un tiempo $O(nm)$.

El cuello de botella en el algoritmo de Prim es la identificación del arco de costo mínimo en el corte $[S, \bar{S}]$. Podemos mejorar la eficiencia en este paso manteniendo dos índices para cada nodo $j \in \bar{S}$:

1. Una etiqueta para la distancia $d(j)$ que representa el costo mínimo de los arcos en el corte incidente al nodo j que no está en S , es decir

$$d(j) = \min \{c_{ij} : (i,j) \in [S, \bar{S}]\}$$

2. Una etiqueta predecesora $pred(j)$ que representa el otro punto final del arco de costo mínimo en el corte incidente al nodo j . Por ejemplo en la tercer iteración del ejemplo 4.4 los tres arcos $(1, 3)$, $(2, 3)$ y $(4, 3)$ en el corte son incidentes al nodo 3. Entre esos arcos el arco $(4, 3)$ tiene el costo mínimo de 20.

Entonces $d(3) = 20$ y $pred(3) = 4$. Para el mismo caso $d(5) = 30$ y $pred(5) = 4$. Si mantenemos esos índices podemos encontrar fácilmente el costo mínimo de un arco en el corte; simplemente calculamos

$$\min \{d(j) \mid j \in S\}$$

si el nodo i alcanza este mínimo, $(pred(i), i)$ es el arco de costo mínimo en el corte. Observe que si movemos el nodo i desde S hasta \bar{S} , necesitamos actualizar las etiquetas distancia y predecesor sólo para los nodos adyacentes al nodo i .

5.5 ALGORITMO DE SOLLIN

Podemos usar las condiciones de optimalidad para derivar otro algoritmo para el problema de árbol de expansión mínima. Este algoritmo lo podemos ver como una versión híbrida de los algoritmos de Kruskal y de Prim. Como en el algoritmo de Kruskal, el algoritmo de Sollin mantiene una colección de árboles de expansión de los nodos N_1, N_2, N_3, \dots y aumenta arcos a esa colección. Sin embargo en cada iteración aumenta arcos de costo mínimo que emanan de esos árboles, una idea tomada del algoritmo de Prim. Como resultado se obtiene un algoritmo que usa estructuras de datos sencillas y corre en un tiempo $O(m \log n)$. El algoritmo de Sollin desarrolla repetidamente las dos siguientes operaciones básicas.

Vecino más cercano: (N_k, i_k, j_k) . Esta operación toma como inicio un árbol que une a los nodos N_k y determina un arco (i_k, j_k) de costo mínimo entre todos los arcos que emanan de N_k , es decir

$$C_{i_k j_k} = \min \{C_{ij} \mid (i,j) \in A, i \in N_k, \text{ y } j \notin N_k\}$$

Para desarrollar esta operación necesitamos revisar todos los arcos en las listas de adyacencia de los nodos en N_k , y encontrar el arco de costo mínimo entre esos arcos que tenga un punto final que no este en N_k .

Acoplar (i_k, j_k) : Esta operación toma como inicio dos nodos i_k y j_k y si los dos nodos están en distintos árboles entonces une esos dos árboles en un solo árbol.

ALGORITMO DE SOLLIN

Propósito: Resolver el problema de árbol de expansión mínima.

Descripción

PASO 1 Para cada $i \in N$ haga $N_i = \{i\}$, $T^* = \emptyset$

PASO 2 Mientras $|T^*| < (n-1)$ entonces para cada árbol N_k haga (N_k, i_k, j_k) : donde el arco (i_k, j_k) es el arco de costo mínimo entre todos los que emanan de N_k es decir:

$$\min c_{ij} = \{(i,j) \mid (i,j) \in A, i \in N_k, j \notin N_k\}$$

y vaya al paso 3

PASO 3 Si los nodos i_k y j_k están en distintos árboles una (i_k, j_k) y actualice $T^* = T^* \cup \{(i_k, j_k)\}$ y regrese al paso 2.

EJEMPLO 5.5

En la siguiente red encuentre el árbol de expansión mínima usando el algoritmo de Sollin.

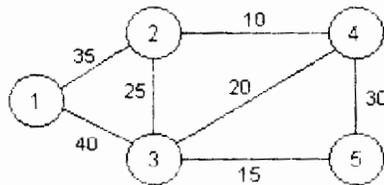


Figura 5.12

Se comienza con cinco árboles que conforman un bosque. Cada árbol consta de un solo nodo y el arco es el de menor costo que conecta cada nodo.

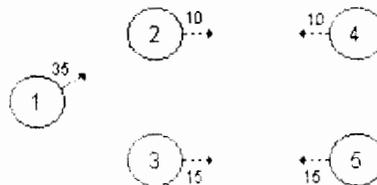


Figura 5.13

Se unen nodos reduciendo el número de árboles a solo dos. El arco de menor costo que emana de esos árboles es el (3,4).

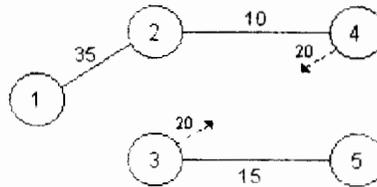


Figura 5.14

Se une ese arco y obtenemos el árbol.

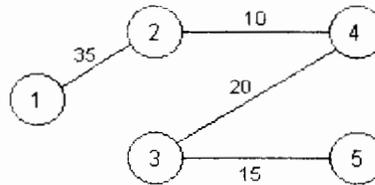


Figura 5.15

COMPLEJIDAD COMPUTACIONAL DEL ALGORITMO

Para poder analizar el tiempo de ejecución del algoritmo necesitamos discutir la estructura de datos necesaria para implantarlo. Mostraremos que el algoritmo desarrolla $O(\log n)$ ejecuciones del ciclo del paso 2, y que se pueden desarrollar las operaciones vecino mas cercano y acoplamiento en un tiempo $O(m)$. Estos resultados establecen una cota de tiempo de $O(m \log n)$ para el algoritmo.

Almacenamos los nodos del árbol en una lista circular doblemente ligada, lo que nos permite visitar cada nodo del árbol comenzando en cualquier nodo. Asignamos una etiqueta numérica con cada nodo en la red; la etiqueta satisface las siguientes dos propiedades:

- 1.- Nodos del mismo árbol que tienen la misma etiqueta.
- 2.- Nodos de diferentes árboles con diferentes etiquetas. Al inicio del algoritmo, asignamos la etiqueta i a cada nodo $i \in N$.

Usando esta estructura de datos podemos verificar fácilmente cuándo un arco (i, j) tiene sus puntos finales en el mismo árbol. Contestamos esta pregunta verificando las etiquetas de los nodos i y j .

Esta observación implica que podemos desarrollar la operación de vecino más cercano para cada árbol en el bosque en un tiempo total de $O(\sum_{i \in N} |A(i)|) = O(m)$

Desarrollamos las operaciones de acoplamiento en el ciclo del paso 2-3 usando el siguiente esquema iterativo: En cada iteración seleccionamos un árbol no examinado, por ejemplo N_1 , y consideramos el arco de costo mínimo (i_1, j_1) que emana de N_1 (el nodo i_1 está en N_1 y el j_1 puede o no estar en N_1). Suponga que los nodos en N_1 tienen la etiqueta α . Si el nodo j_1 también tiene la etiqueta α , la iteración termina.

En caso contrario buscamos los nodos del árbol, por ejemplo N_2 que contenga al nodo j_1 y les asignamos la etiqueta α . Después, consideramos los arcos de costo mínimo (i_2, j_2) que emanan de N_2 . Si el nodo j_2 tiene etiqueta α , la iteración termina; en caso contrario, buscamos en los nodos del árbol N_3 que contenga al nodo j_2 y les asignamos la etiqueta α . Repetimos este proceso hasta que la iteración termina. Note que en cada iteración podemos asignar a los nodos de varios árboles la etiqueta del primer árbol. Cuando una iteración termina iniciamos una nueva iteración seleccionando otro árbol no examinado. Terminamos este proceso iterativo cuando hayamos examinado todos los árboles. De esta descripción se puede ver que este método asigna una etiqueta a cada nodo y se ejecuta en un tiempo $O(n)$.

Cada ejecución en el ciclo del algoritmo requiere un tiempo $O(m)$, obtenemos entonces una cota del número de ejecuciones del ciclo. Cada ejecución del ciclo reduce el número de árboles en el bosque por un factor de al menos dos porque acoplamos cada árbol en un árbol mayor. Esta observación tiene como consecuencia que desarrollaremos $O(\log n)$ ejecuciones del ciclo. Se tiene entonces el siguiente teorema:

TEOREMA:

El algoritmo de Sollin tiene un tiempo de ejecución de $O(m \log n)$.

En este capítulo hemos descrito tres algoritmos para resolver el problema del árbol de expansión mínima. Todos ellos son fáciles de implantar y tienen excelentes tiempos de ejecución, además son muy eficientes en la práctica.

La tabla siguiente resume estos tres algoritmos:

ALGORITMO	TIEMPO DE EJECUCIÓN	OBSERVACIONES
Kruskal	$O(m+n \log n)$	<ol style="list-style-type: none"> 1.- Examina arcos en orden no decreciente de sus costos más el tiempo necesario para ordenar m arcos y los incluye en un árbol de expansión si el arco que se aumenta no forma un ciclo con los arcos seleccionados. 2.-La prueba del algoritmo usa condiciones de optimalidad para una trayectoria. 3.-Es un algoritmo atractivo si los arcos ya se encuentran ordenados en forma creciente.
Prim	$O(m+n \log n)$	<ol style="list-style-type: none"> 1.-Mantiene un árbol de expansión en un subconjunto de nodos S y aumenta un arco de costos mínimos en el corte $[S, \bar{S}]$. 2.-La prueba del algoritmo usa condiciones de optimalidad para cortes. 3.-Se puede implantar usando una variedad de estructuras de apilamiento. Esta cota de ejecución se refiere al tiempo de la estructura de datos de Fibonacci.
Sollin	$O(m \log n)$	<ol style="list-style-type: none"> 1.-Mantiene una colección de árboles, en cada iteración aumenta un árbol de costo mínimo que emana de cada árbol. 2.-La prueba del algoritmo usa condiciones de optimalidad en cortes.

5.6 ANÁLISIS DE SENSIBILIDAD

Una vez presentados los métodos de solución para el problema del árbol de expansión mínima, es interesante observar el efecto de eliminar un arco de la gráfica original. Es claro que, si este arco no pertenece a la solución óptima, la solución óptima del problema de encontrar el árbol de peso mínimo en la red resultante de remover dicho arco, es la misma que para la red original. Sin embargo, si el arco forma parte de la solución óptima es importante contar con una herramienta que permita resolver el nuevo problema aprovechando los resultados que ya se han obtenido. Enseguida se analiza este problema.

Suponga que se ha obtenido el árbol expandido de peso mínimo $T = [X, A^*]$ de una red conexa $G = [X, A]$, también suponga que por alguna razón, es necesario eliminar un arco de la red G que, además, pertenece al árbol T . Considere, ahora, el problema de encontrar el árbol expandido de peso mínimo de la red resultante de eliminar dicho arco.

Sea $j' = (i, i')$ el arco que debe eliminarse y sea G' la red resultante de tal eliminación, es decir, $G' = [X, A - \{j'\}]$. Entonces, el problema consiste en encontrar el árbol requerido. A continuación desarrollamos un procedimiento para obtenerlo, aprovechando que ya se cuenta con el árbol T .

Al eliminar (i, i') de T , éste se descompone en dos componentes conexas. Sean éstas $T_1 = [X_1, A_1]$ y $T_2 = [X_2, A_2]$. Observe que un extremo del arco j' está en X_1 y el otro en X_2 . Suponga, sin pérdida de generalidad, que i está en X_1 y que i' está en X_2 . Sea U el conjunto de arcos de G distintos de j' que tienen un extremo en X_1 y otro en X_2 . Observe que si U es vacío el problema no tiene solución puesto que G' no sería conexa y por lo tanto no existe árbol expandido de G . Suponga entonces, que U no es vacío, sea u^* el arco de U tal que $p(u^*) \leq p(u)$ para todo arco $u \in U$. La solución del problema se establece en la siguiente proposición:

Proposición 5.4

La red $T^* = [X, A_1 \cup A_2 \cup \{u^*\}]$ es un árbol de expansión mínima de G' .

Demostración:

Sean las redes conexas $G_1 = [X_1, A_1]$ y $G_2 = [X_2, A_2]$ donde A_i es el conjunto de arcos de A que tienen ambos extremos en X_i ($i = 1, 2$). Se demostrará primero que T_1 y T_2 son los árboles de expansión mínima de G_1 y G_2 respectivamente.

Suponga que T_1 no es un árbol de expansión mínima de G_1 . Sea $N = [X_1, U_1]$ el árbol de expansión mínima de dicha gráfica, se tiene entonces que $p(N) < p(T_1)$.

Sea $T' = [X, U_1 \cup U_2 \cup \{j'\}]$, note que T' es un árbol de expansión de G , ya que T' es acíclica debido a que el arco j' tiene un extremo en X_1 y otro en X_2 y las redes N y T_2 son acíclicas.

Ahora bien,

$$p(T') = p(U_1) + p(A_2) + p(j') = p(N) + p(A_2) + p(j')$$

por otro lado

$$p(T) = p(A_1) + p(A_2) + p(j') = p(T_1) + p(A_2) + p(j')$$

por lo tanto

$$P(T') < P(T)$$

Lo que contradice que T sea un árbol de expansión mínima de G . Por lo tanto T_1 es un árbol de expansión de peso mínimo de G_1 . Análogamente, se prueba que T_2 es un árbol de expansión mínima de G_2 .

Por otro lado, G' puede expresarse como la red $[X, A_1 \cup A_2 \cup U]$, y puesto que $p(u^*) \leq p(u)$ para todo arco $u \in U$, se tiene que T^* es un árbol de expansión mínima de G' y con ésto se concluye la prueba.

Este procedimiento lo podemos aplicar en el siguiente problema:

Suponga que se ha encontrado el árbol de peso mínimo $T = [X, A^*]$ de la red $G = [X, A]$ y suponga también que se modifica el peso de un arco $j \in A^*$. Se desea encontrar ahora el árbol de peso mínimo en la gráfica G con la función de peso, asociada a los arcos de G , modificada para el arco j .

Si el nuevo peso de j es menor que el que tenía anteriormente, es claro, que la solución óptima seguirá siendo la misma; el problema surge cuando el nuevo peso de j es mayor que su peso anterior. Es en este último caso cuando puede aplicarse el método expuesto anteriormente. Para encontrar la nueva solución, la única modificación que deberá hacerse al procedimiento será no excluir al arco j del conjunto U .

EJEMPLO 5.6

Considere la red del ejemplo 5.4 y su árbol de expansión mínima. Suponga que se elimina el arco $(2,8)$. La gráfica resultante G' se muestra en la figura 5.16

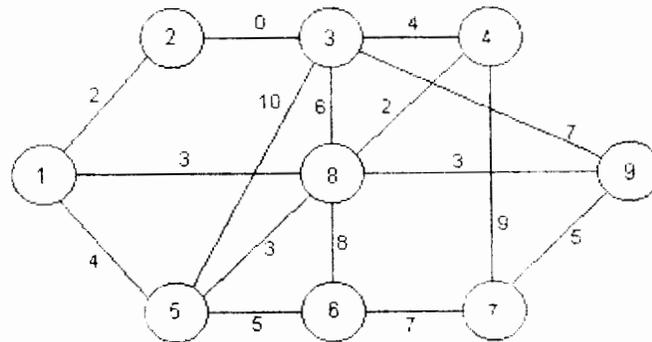


Figura 5.16

Determine el árbol de expansión mínima en esta nueva red. Las componentes conexas en que se divide T al eliminar $(2, 8)$ se muestran en la figura 5.17

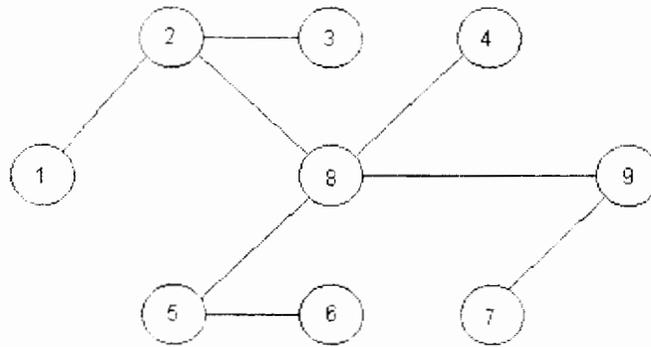


Figura 5.17

El conjunto U estará formado entonces, por los arcos siguientes, con sus respectivos pesos:

Arco	Peso	Arco	Peso
(1, 5)	4	(3, 5)	10
(1, 8)	3	(3, 8)	6
(3, 4)	4	(3, 9)	7

De aquí se tiene que $u^* = (1, 8)$. Por lo tanto el árbol de expansión mínima de G' se muestra en la figura 5.18

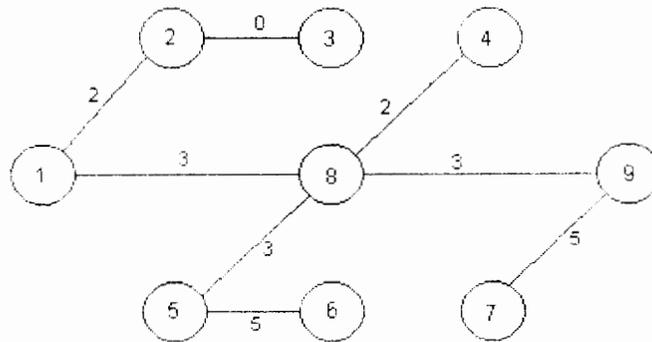


Figura 5.18

cuyo peso está dado por:

$$0+2+2+3+3+3+5+5 = 23 = p(T) - p(2, 8) + p(1, 8)$$

5.7 NOTAS HISTÓRICAS

Los primeros algoritmos de árbol de expansión mínima se desarrollaron en 1926. El artículo de Graham and Hell (1985) presenta una reseña histórica de los algoritmos de árbol de expansión mínima.

Boruvka (1926) y Jamick (1930) independientemente desarrollaron y resolvieron el problema. Más tarde otros investigadores redescubrieron esos algoritmos.

Kruskal (1956) y Loberman y Weinberger (1957) independientemente desarrollaron el algoritmo que lleva el nombre de Kruskal.

Prim (1957) desarrolló el algoritmo que lleva su nombre.

Sollin presentó su algoritmo, discutido en un seminario en 1961; nunca se publicó. Claude Berge, que estaba presente en el seminario, reportó este algoritmo en su libro Berge and Ghouila-Houri (1962).

Más tarde, diversos investigadores descubrieron que el algoritmo de Sollin es similar al de Boruvka y que el algoritmo de Prim es similar al de Jamick.

Una versión mejorada del algoritmo de Kruskal fue desarrollada por Tarjan en 1984 y se ejecuta en $O(m \alpha(n, m))$ más el tiempo requerido para ordenar los arcos. $\alpha(n, m)$ es la función de Ackermann que para efectos prácticos es menor que 6.

Gabow, Galil, Spencer y Tarjan en 1986 presentaron una variedad del algoritmo de Prim que corre en un tiempo $O(m \log \beta(m, n))$ con la función $\beta(m, n)$ definida como $\beta(m, n) = \min \{i: \log^{(i)}(m/n) \leq 1\}$. En esta expresión $\log^{(i)}x = \log \log \log \dots \log x$ en \log iterado i veces. Así, $\beta(m, n)$ es una función que crece muy lentamente. Por ejemplo, si $m/n = 2^{264,000}$ entonces $\beta(m, n) = 6$.

Actualmente, el algoritmo más rápido para resolver el problema del árbol de expansión mínima es el de Kruskal modificado por Tarjan (1984) si los arcos ya están ordenados.

CAPÍTULO 6

PROBLEMA DE RUTA MÁS CORTA

6.1 INTRODUCCIÓN

En este capítulo se presentan métodos de solución para los siguientes problemas de rutas más cortas de una red:

1. Ruta más corta entre dos nodos específicos s y t .
2. Rutas más cortas entre un nodo específico s y todo nodo i de la red.
3. Ruta más corta entre todo par de nodos.

Para ejemplificar considere lo siguiente:

EJEMPLO 6.1

En una terminal de camiones para pasajeros se desea establecer la ruta que deberá seguir el autobús que presta servicio de la ciudad s a la ciudad t de tal manera que la distancia recorrida sea lo más corta posible. A este problema se le puede representar en una red $G = [N, A, d]$ donde:

$N = \{\text{Ciudades a las cuales se ofrece el servicio}\}$

$A = \{\text{Tramos de carretera entre las ciudades}\}$

$d: A \rightarrow R$, donde para toda a elemento de A , $d(a)$ es la longitud o distancia del tramo de carretera a .

En general en una red $G = [N, A, d]$, al número $d(a)$ asociado a cada arco se le llama longitud o costo de a . Por otro lado se define la longitud de una ruta o camino como la suma de longitudes de los arcos que la forman; aquella ruta tal que su longitud sea mínima se llama la ruta más corta o camino más corto.

El problema entonces de la terminal de autobuses es encontrar la ruta más corta entre dos nodos específicos: los que representan a las ciudades s y t . Observe que en este caso las longitudes definidas son no negativas; sin embargo, el problema de encontrar la ruta más corta entre dos nodos específicos puede generalizarse a cualquier red puesto que la función de longitud d , puede representar, además de distancia o tiempo, costos o alguna otra cantidad.

Si la red contiene arcos con longitudes negativas pueden presentarse circuitos negativos (circuitos de longitud negativa). En este caso el problema puede no ser acotado puesto que para cualquier ruta entre s y t que contenga al circuito negativo existe otra mejor, a saber aquella que contiene una vez más al circuito. Por ejemplo, en la siguiente red:

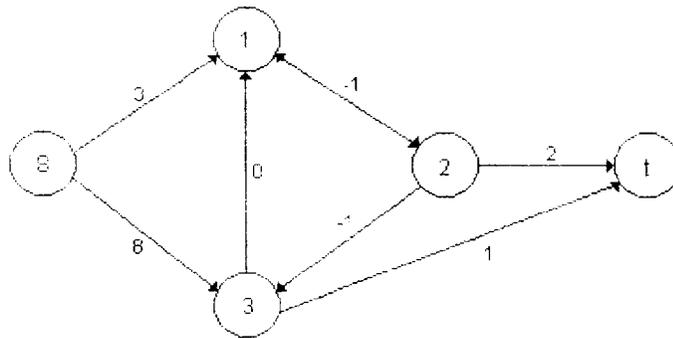


Figura 6.1

Una ruta de s a t es: $s, 3, 1, 2, 3, t$ de longitud 7. Otra ruta de s a t mejor que la anterior es: $s, 3, 1, 2, 3, 1, 2, 3, t$, de longitud 5. De hecho, si se considera una ruta que contenga el arco $(s, 3)$ M veces el circuito $3, 1, 2, 3$ ($M > 0$) y el arco $(3, t)$ la longitud será $9 - 2M$, de tal modo que si M tiende a infinito, la longitud de la ruta tiende a menos infinito. Luego, la ruta más corta entre s y t no existe.

Se concluye entonces, que para que el problema de la ruta más corta entre dos nodos específicos tenga solución, deberá cumplirse:

- i) Existe alguna trayectoria entre s y t ;
- ii) No existen circuitos negativos tales que haya un camino de s a algún nodo del circuito y otro de algún nodo del circuito a t .

Suponga ahora que en la terminal de autobuses se desea mejorar el servicio que se proporciona a la ciudad s ; con este objeto se requiere encontrar las rutas más cortas entre la ciudad s y todas las demás ciudades a las cuales se les da el servicio. A este problema puede asociarse, de nuevo, la red definida anteriormente.

Antes de analizar cuándo tiene solución este problema se definirán algunos conceptos de gran utilidad.

Sea $G = [N, A]$ una red dirigida y sea s un nodo en N ; entonces a s se le llama *raíz* de G , si existe una trayectoria de s a i para toda i en N .

Una *arborescencia* es un árbol que tiene una raíz. La siguiente red es una arborescencia de raíz 1:

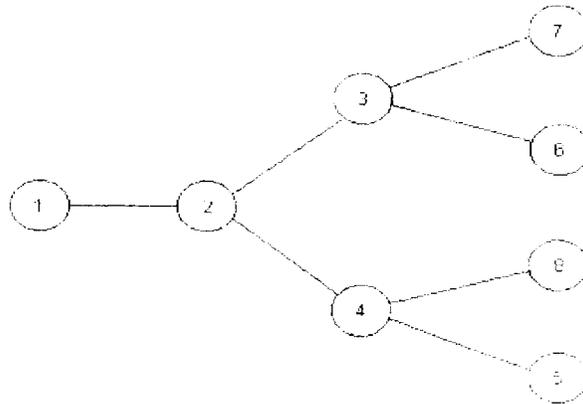


Figura 6.2

Sea $G = [N, A]$ una gráfica dirigida. Una *arborescencia de G* es un árbol expandido de G que contiene un nodo que es raíz.

En una arborescencia de raíz s el camino o trayectoria de s a i , para toda i elemento de N , es único.

Considere ahora una red G . Una *arborescencia de rutas más cortas* de G es aquella arborescencia tal que la única ruta de s a i , para toda i elemento de N , es una ruta más corta de s a i .

Una vez definidos estos conceptos, puede decirse que el problema de la terminal de autobuses es encontrar la arborescencia de rutas más cortas de raíz s de la red $G=[N,A,d]$.

Haciendo una analogía con el problema de la ruta más corta entre dos nodos específicos puede concluirse que para que exista la arborescencia de rutas más cortas de raíz s en una red cualquiera G , ésta deberá cumplir que:

- i) Existen caminos de s a i , para toda i en N . Es decir, que s sea la raíz de la red.
- ii) No existen circuitos negativos en la red G , ya que de presentarse éstos el problema sería no acotado (por la misma razón expuesta para el problema de ruta más corta entre dos nodos específicos).

Finalmente, suponga que en la terminal de autobuses se tiene interés en encontrar las rutas más cortas para todos los camiones que prestan servicio entre cada par de ciudades. Nuevamente, se asocia a este problema la red G definida anteriormente, se deben encontrar entonces, las rutas más cortas entre todo par de nodos en la red G .

Este último problema es una generalización inmediata de los anteriores. Por esto se deduce que, para que exista solución en cualquier red G , deberá cumplirse lo siguiente:

- i) Existe, al menos, una trayectoria entre todo par de nodos.
- ii) No existen circuitos negativos en la red G .

Algunas propiedades de las arborescencias se dan a continuación.

6.2 CARACTERIZACIÓN DE UNA ARBORESCENCIA

Existen ciertas propiedades de las arborescencias que serán utilizadas en la búsqueda de la solución del problema de la arborescencia de rutas más cortas de raíz x_0 . Estas propiedades se demuestran en el siguiente teorema en el cual se postulan las distintas caracterizaciones de este tipo de redes.

Es importante hacer notar que, en general, la arborescencia de rutas más cortas es distinta del árbol de expansión mínima en una red, puesto que, mientras el segundo concepto es no orientado el primero sólo es aplicable a redes dirigidas.

Aún en el caso en que la red sea dirigida, un árbol no necesariamente es arborescencia ya que ésta puede no tener raíz. Por ello, para determinar la arborescencia de rutas más cortas no es posible aplicar ninguno de los algoritmos presentados en el capítulo anterior. Podría suceder incluso que un árbol de peso mínimo resulte ser una arborescencia; sin embargo, de ningún modo puede garantizarse que ésta sea de rutas más cortas.

Por ejemplo en la siguiente red:

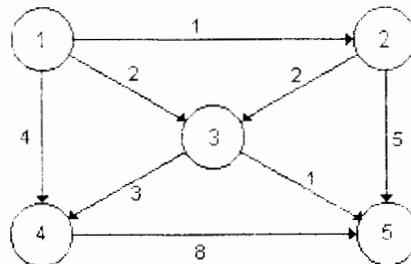


Figura 6.3

Un árbol de peso mínimo es:

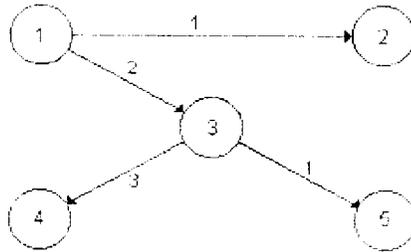


Figura 6.4

el cual resulta ser una arborescencia de raíz 1. Sin embargo, la única ruta en la arborescencia entre los nodos 1 y 4 tiene una longitud de 5 unidades y existe otra ruta de longitud menor entre esos dos nodos en la red: la ruta 1, 4 tiene una longitud de 4 unidades. Por lo tanto, puede concluirse que la arborescencia no es de rutas más cortas.

TEOREMA 6.1

Sea G una red con n nodos. Suponga que $n \geq 2$. Los postulados siguientes son equivalentes y caracterizan una arborescencia:

- G es un árbol y tiene un nodo x_0 que es raíz.
- Para todo nodo i existe un camino único de x_0 a i .
- G tiene al nodo x_0 que es raíz y si se elimina un arco entonces x_0 ya no es raíz.
- G es conexa.
- G es acíclica.
- G tiene como raíz a x_0 y es acíclica.
- G tiene como raíz a x_0 y posee $n-1$ arcos.

Demostración

(a implica b) Sea i un nodo de G . Puesto que x_0 es raíz de G entonces existe una trayectoria de x_0 a i . Suponga que existen, al menos, dos trayectorias de x_0 a i . Entonces G tendría un ciclo y esto contradice que G es un árbol. Luego, el camino de x_0 a i es único.

(b implica c). Puesto que existe una trayectoria entre x_0 e i , para todo nodo i , entonces x_0 es raíz. Sea $a = (i_k, j_k)$ un arco de G y suponga que x_0 es raíz de la red

$F' = [N, A - \{a\}]$ entonces existe una trayectoria de x_0 a j_k en G' (note que esta trayectoria no contiene a a). Puesto que x_0 es raíz de G , existe una trayectoria de x_0 a i_k ; esta trayectoria y el arco a , forman una trayectoria de x_0 a j_k ; entonces existen en G dos trayectorias distintas de x_0 a j_k , lo cual es una contradicción. Por lo tanto, G no tiene a x_0 como raíz.

(c implica d). Puesto que x_0 es raíz de G , entonces G es conexo. Suponga que $g(x_0) \neq 0$, es decir que existe un arco (x, x_0) . Como x_0 es raíz, entonces existe al menos un camino de x_0 a x ; luego, al eliminar (x, x_0) de G , x_0 sigue siendo raíz, lo cual es una contradicción por lo tanto $g(x_0)$ es igual a 0. Por otro lado, por ser x_0 raíz, se tiene que $g(x) \geq 1$ para todo nodo $x \neq x_0$. Suponga que $g(x) > 1$, es decir, que existen al menos dos arcos (i_k, x) y (j_k, x) entonces al suprimir cualquiera de estos dos arcos x_0 sigue siendo raíz de G , por lo tanto, $g(x) = 1$, para toda $x \neq x_0$.

(d implica e). La red G es conexa y tiene $n-1$ arcos puesto que $g(x_0) = 0$ y $g(x) = 1$ para toda $x \neq x_0$. Entonces G es un árbol y como consecuencia es acíclica.

(e implica f). Si x_0 no fuera raíz, entonces G tendría un ciclo puesto que $g(x) = 1$ para toda $x \neq x_0$.

(f implica g). G es acíclica y además conexa puesto que x_0 es raíz. Entonces G es un árbol y por lo tanto tiene $n-1$ arcos.

(g implica a). G tiene $n-1$ arcos y es conexa puesto que x_0 es raíz, luego G es un árbol.

6.3 MÉTODOS DE SOLUCIÓN Y JUSTIFICACIÓN

En esta parte del capítulo se presentan los métodos de solución para el problema de la arborescencia de rutas más cortas de raíz s en una red $R = [N, A, d]$. El primero de ellos sólo es aplicable a redes que tienen arcos con costos no negativos y el segundo puede utilizarse para cualquier red.

En la práctica existe gran cantidad de problemas que involucran costos no negativos (tiempo, distancia, etc.); es por esta razón que se justifica el desarrollo de algoritmos que se aplican sólo a estos casos. Por otro lado, el método presentado para redes con esta característica, constituye el primer caso del algoritmo general posteriormente expuesto. Es importante mencionar que estos algoritmos tienen la ventaja de que, además de proporcionar la solución óptima cuando existe, la detectan cuando ésta no existe; ya sea que dicha solución no exista porque s no es raíz de la red o por la presencia de circuitos negativos. Cabe señalar que estos métodos sirven también para encontrar la solución óptima del problema de la ruta más corta entre todo par de nodos.

Para el problema de la ruta más corta entre todo par de nodos se presenta un algoritmo con la misma ventaja que los anteriores. Más aun, si la ruta más corta existe para algunos pares de nodos y para otros no, el algoritmo proporciona las longitudes de las rutas existentes y detecta para qué pares de nodos no existe ninguna ruta.

Otra observación importante que debe hacerse es que, como se verá posteriormente, el problema de la ruta más corta resulta ser un subproblema del problema de flujo a costo mínimo. Otro subproblema que surge en la búsqueda del flujo a costo mínimo, es el de la detección de circuitos negativos por lo cual los métodos en este capítulo serán de gran utilidad.

ARBORESCENCIA DE RUTAS MÁS CORTAS

Caso de redes con costos no negativos

El método de solución presentado para el problema de la arborescencia de rutas más cortas en redes que tienen arcos con costos no negativos fue desarrollado por Dijkstra (1959) y está considerado como uno de los métodos más eficientes para resolver este problema.

Este método se basa en la asignación de etiquetas "permanentes" a los nodos para los cuales ya se conocen las longitudes de las rutas más cortas de la raíz a ellos. Sea S este conjunto de nodos. Las etiquetas de los nodos de S representan precisamente las longitudes de las rutas más cortas buscadas. Los nodos restantes se etiquetan "temporalmente" en una cota superior de la longitud más corta de la raíz al nodo etiquetado.

En la primera iteración el conjunto T contendrá únicamente al nodo raíz; es decir, sólo la raíz estará etiquetada permanentemente. Las etiquetas temporales se mejoran continuamente y en cada iteración se agrega exactamente un nodo x a S ; este nodo es aquel tal que la longitud desde la raíz es la más corta posible.

Puesto que todos los arcos tienen costos no-negativos, siempre puede encontrarse una ruta más corta de la raíz a x que pase sólo por nodos de S ; en este caso la etiqueta de x representa la longitud de la ruta más corta correspondiente. Una vez que todos los nodos estén en S , las etiquetas de todos los nodos serán las correspondientes a las longitudes más cortas desde la raíz y por lo tanto se habrá encontrado la solución deseada. En el caso en que se desee sólo la ruta más corta entre dos nodos específicos, se obtendrá la solución cuando se etiquete "permanentemente" el nodo final del camino buscado.

ALGORITMO DE DIJKSTRA

Propósito: Obtener la arborescencia de las rutas más cortas de raíz s en una red $G = [N, A, d]$ con costos no negativos en los arcos.

Descripción

PASO 1 (Iniciación de etiquetas). Sea $d(s)=0$ y márquese esta etiqueta como permanente. Sea $d(x)=\infty$, para todo $x \neq s$ y considérense estas etiquetas como temporales. Sean $a(x)=x$ (estas etiquetas indicarán el predecesor de x en la arborescencia). Sea $p=s$.

PASO 2 (Actualización de etiquetas). Para todo $x \in I^+(p)$ que tenga etiqueta temporal, actualizar etiquetas de acuerdo a:

$$d(x) = \min \{d(x), d(p) + d(p, x)\}$$

si $d(x)$ se modificó, hacer $a(x)=p$. Sea x^* tal que $d(x^*) = \min \{d(x) \mid d(x) \text{ es temporal}\}$. Si $d(x^*) = \infty$, terminar. En este caso no existe arborescencia alguna de raíz s . En otro caso, marcar la etiqueta $d(x^*)$ como permanente. sea $p=x^*$.

PASO 3 (i) (Si sólo se desea la ruta de s a t). Si $p=t$, terminar: $d(p)$ es la longitud del camino más corto. Si $p \neq t$, ir al paso 2.

(ii) (Si se desea la arborescencia). Si todos los nodos tienen etiquetas permanentes, terminar; ésta es la longitud deseada del camino y el conjunto de arcos $\{a(x), x\}$ forman la arborescencia de caminos más cortos. En otro caso, ir al paso 2.

Justificación del algoritmo

Primero observe que el algoritmo termina en un número finito de iteraciones, ya sea en el paso 2 ó en el paso 3 puesto que el número de nodos es finito. Se justificará la optimalidad del algoritmo para el caso del problema de arborescencia de rutas más cortas ya que el otro caso está comprendido en éste.

Note que si el algoritmo termina en el paso 3, la red generada tendrá $n-1$ arcos y a s como raíz; por esta razón dicha red es una arborescencia. Por otro lado se tiene que, por construcción, $d(x)$ es la longitud del único camino de s a x en esta arborescencia. Ahora se probará que la arborescencia generada es de rutas más cortas. Para ello se demostrará, por inducción sobre el número de iteraciones, que las etiquetas permanentes de los nodos son las longitudes de las rutas más cortas de s a x , para

todo $x \in X$. Esto es claro en la primera iteración. Suponga también que es válido en la k -ésima iteración.

Sea S el conjunto de nodos con etiquetas permanentes y S' el conjunto de nodos con etiquetas temporales en la iteración k . Al final del paso 2 de la iteración $k+1$ la etiqueta temporal $d(x)$, para $x \in S'$ es la longitud de una ruta más corta de s a x que contiene solamente nodos de S . En efecto, en cada iteración sólo se etiqueta permanentemente un nodo; por lo tanto, sólo es necesaria la comparación efectuada en el paso 2. En particular esto sucede para x^* (nodo con la mínima etiqueta temporal).

Suponga ahora que la ruta más corta de la raíz a x^* no contiene sólo nodos de S . Sea y el primer nodo, en el camino más corto de s a x^* , que no está en S . Puesto que los costos de los arcos son no negativos, entonces la porción del camino de s a x^* , que une a s con x^* , es no negativa. Sea D esta longitud. Note que la porción del camino de s a x^* , que une a s con y , es un camino que contiene solamente nodos de S . Pero $d(y)$ es la longitud de una ruta más corta que contiene todos sus nodos en S , luego:

$$d(y) + D < d(x^*)$$

lo que implica

$$d(y) < d(x^*) - D < d(x^*)$$

lo cual constituye una contradicción puesto que $d(x^*)$ es el mínimo de las etiquetas temporales. Luego, se concluye que la ruta más corta de s a x^* contiene sólo nodos de S y, por lo tanto, $d(x^*)$ es su longitud. Por lo tanto, la etiqueta permanente de x es igual a la longitud de la ruta más corta de s a x en la iteración $k+1$.

Finalmente, observe que si $d(x^*)$ es igual a infinito (paso 2) en alguna iteración, entonces existe algún nodo (x^* y todos los que tengan etiqueta temporal) para el cual no existe ruta alguna desde s ; puede entonces concluirse que el problema no tiene solución puesto que en este caso s no es raíz de la red.

EJEMPLO 6.1

Suponga que en un aeropuerto se considera la posible adquisición de un equipo que, a causa de ciertas modificaciones, será inútil dentro de 3 años. Los costos de utilización y de reventa del equipo usado son conocidos. El equipo puede ser utilizado durante uno, dos o tres años, revenderse al final del periodo de utilización y comprarse uno nuevo para usarse durante el tiempo restante. Los costos totales en los que se incurre (costo de compra más costo de utilización menos precio de venta) están dados en la matriz de costos:

$$C = \begin{matrix} & \begin{matrix} 1 & 2 & 3 \end{matrix} \\ \begin{matrix} 0 \\ 1 \\ 2 \end{matrix} & \begin{bmatrix} 4 & 8 & 16 \\ - & 5 & 11 \\ - & - & 6 \end{bmatrix} \end{matrix}$$

El elemento (i, j) de la matriz es igual al costo total en millones de pesos en el que se incurre si se compra un equipo al final del año i , se utiliza hasta el final del año j y se revende. Se desea encontrar la estrategia más económica de compra y reventa de equipo.

Para resolver este problema considere la red $G = [N, A, C]$ donde $N = \{0, 1, 2, 3\}$ y cada elemento de C indica el final de un año; $(i, j) \in A$ si y sólo si es posible comprar el equipo al final del año i y revenderlo al final del año j ; finalmente, $C(i, j)$ es el costo total en el que se incurre por comprar el equipo al final del año i , usarlo y revenderlo al final del año j .

Una forma esquemática de la red es:

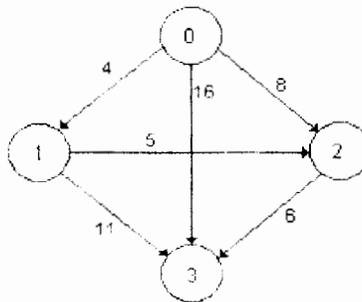


Figura 6.5

Observe que una ruta de 0 a 3 en la red corresponde a una estrategia posible y viceversa; por otro lado, el costo de una estrategia posible es el mismo que el de la ruta correspondiente. Por ejemplo, la ruta 0,1,3 (de longitud 15) corresponde a comprar al final del año 0, utilizar el equipo durante un año, revenderlo y comprar uno nuevo al final del año 1, utilizarlo durante dos años y revenderlo al final del año 3 (con un costo total de $4+11=15$ millones). Por otra parte, comprar el equipo al final el año 0, utilizarlo durante tres años y revenderlo al final del año 3 (con un costo total de 16 millones) corresponde a la ruta 0,3 (de longitud 16).

Para encontrar la estrategia óptima se deberá entonces encontrar la ruta más corta entre los nodos 0 y 3 de la red. Para esto se aplicará el algoritmo de Dijkstra considerando $s=0$ y $t=3$.

Iteración 1

PASO 1

$d(0)=0$ (etiqueta permanente); $d(x)=\infty$, para $x=1,2,3$ (etiquetas temporales), $p=0$.

PASO 2

Actualización de etiquetas: $\Gamma^+(p)=\{1,2,3\}$

$$\begin{array}{ll} d(1) = \min \{ \infty, 4 \} = 4 & a(1)=0 \\ d(2) = \min \{ \infty, 8 \} = 8 & a(2)=0 \\ d(3) = \min \{ \infty, 16 \} = 16 & a(3)=0 \end{array}$$

Sea x^* tal que $d(x) = \min \{ d(1), d(2), d(3) \} = d(1)$

PASO 3

De donde $x^*=1$ (nodo con mínima etiqueta temporal). Se marca $d(1)$ como permanente; $p=1$. ($p \neq t = 3$).

Iteración 2

Actualización de etiquetas: $\Gamma^+(p)=\{2,3\}$

$$\begin{array}{ll} d(2) = \min \{ 8, 4 + 5 \} = 8 & a(2) \text{ no cambia} \\ d(3) = \min \{ 16, 4 + 11 \} = 15 & a(3)=1 \end{array}$$

De donde $x^*=2$ (nodo con mínima etiqueta temporal). Se marca $d(2)$ como permanente; $p=2$. ($p \neq t$).

Iteración 3

Actualización de etiquetas: $\Gamma^+(p)=\{3\}$

$$d(3) = \min \{ 15, 8 + 6 \} = 14 \quad a(3)=2$$

De donde $x^*=3$ (nodo con mínima etiqueta temporal). Se marca $d(3)$ como permanente; $p = 3 = t$. Se termina.

Para recuperar la ruta más corta de 0 a 3, de longitud $d(3)=14$, se utilizan las etiquetas $a(x)$. La ruta deseada, en sentido inverso, es: 3, $a(3) = 2$, $a(2) = 0$. Luego, la estrategia óptima para el problema del aeropuerto es comprar el equipo al final del año 0, utilizarlo durante dos años, revenderlo y comprar un nuevo al final del año 2, utilizar este último durante un año y venderlo al final del año 3 con un costo total de 14 millones de pesos.

EJEMPLO 6.2

Considere el siguiente mapa de los Ferrocarriles Nacionales de México, donde las vías están representadas por arcos y el número asociado al arco (i,j) representa el tiempo, en horas que tarda un ferrocarril en recorrer el tramo de vía (i,j) :

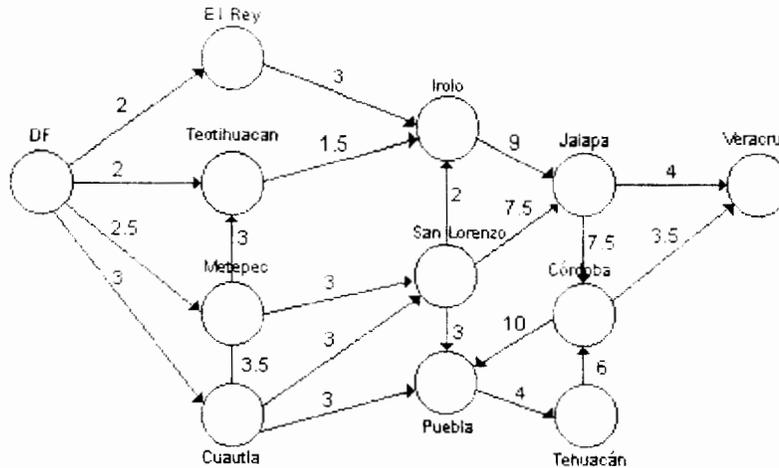


Figura 6.6

Suponga que se desea encontrar las rutas más rápidas para los ferrocarriles que presten servicio desde el D.F. hasta cada una de las ciudades del mapa.

En términos de redes se desea determinar la arborescencia de rutas más cortas de raíz D.F. Para resolver este problema usaremos algoritmo de Dijkstra puesto que los costos en los arcos son no negativos.

Iteración 1

PASO 1

$d(\text{D.F.}) = 0$ (etiqueta permanente)

$d(x) = \infty$ para toda x en N tal que $x \neq \text{D.F.}$ (etiquetas temporales)

$p = \text{D.F.}$

PASO 2

Actualización de etiquetas: $\Gamma^+(p) = (\text{El Rey}, \text{Teotihuacan}, \text{Metepec}, \text{Cuautla})$:

$d(\text{El Rey})$	$= \min \{ \infty, 2 \}$	$= 2;$	$a(\text{El Rey})$	$= \text{D.F.}$
$d(\text{Teotihuacan})$	$= \min \{ \infty, 2 \}$	$= 2;$	$a(\text{Teotihuacan})$	$= \text{D.F.}$
$d(\text{Metepec})$	$= \min \{ \infty, 2.5 \}$	$= 2.5;$	$a(\text{Metepec})$	$= \text{D.F.}$
$d(\text{Cuautla})$	$= \min \{ \infty, 3 \}$	$= 3;$	$a(\text{Cuautla})$	$= \text{D.F.}$

de donde $x^* = \text{El Rey}$, puesto que éste es el nodo con mínima etiqueta temporal (los empates se rompen arbitrariamente). Se marca $d(\text{El Rey})$ como permanente, $p = \text{El Rey}$ (aún existen etiquetas temporales).

Iteración 2

Actualización de etiquetas: $\Gamma^+(p) = (\text{Irolo})$
 $d(\text{Irolo}) = \min \{ \infty, 2+3 \} = 5$; $a(\text{Irolo}) = \text{El Rey}$

de donde $x^* = \text{Teotihuacan}$, puesto que éste es el nodo con mínima etiqueta temporal. Se marca $d(\text{Teotihuacan})$ como permanente, $p = \text{Teotihuacan}$ (aún existen etiquetas temporales).

Iteración 3

Actualización de etiquetas: $\Gamma^+(p) = (\text{Irolo})$
 $d(\text{Irolo}) = \min \{ 5, 2+1.5 \} = 3.5$; $a(\text{Irolo}) = \text{Teotihuacan}$

de donde $x^* = \text{Metepec}$, puesto que éste es el nodo con mínima etiqueta temporal. Se marca $d(\text{Metepec})$ como permanente, $p = \text{Metepec}$ (aún existen etiquetas temporales).

Iteración 4

Actualización de etiquetas: $\Gamma^+(p) = (\text{Teotihuacan}, \text{San Lorenzo})$ (Teotihuacan ya tiene etiqueta permanente).
 $d(\text{San Lorenzo}) = \min \{ \infty, 2.5+3 \} = 5.5$; $a(\text{San Lorenzo}) = \text{Metepec}$

de donde $x^* = \text{Cuautla}$, puesto que éste es el nodo con mínima etiqueta temporal. Se marca $d(\text{Cuautla})$ como permanente, $p = \text{Cuautla}$ (aún existen etiquetas temporales).

Iteración 5

Actualización de etiquetas: $\Gamma^+(p) = (\text{Metepec}, \text{San Lorenzo}, \text{Puebla})$ (Metepec ya tiene etiqueta permanente).
 $d(\text{San Lorenzo}) = \min \{ 5.5, 3+3 \} = 5.5$;
 $a(\text{San Lorenzo}) = \text{Metepec}$
 $d(\text{Puebla}) = \min \{ \infty, 3+3 \} = 6$; $a(\text{Puebla}) = \text{Cuautla}$

de donde $x^* = \text{Irolo}$, puesto que éste es el nodo con mínima etiqueta temporal. Se marca $d(\text{Irolo})$ como permanente.

$p = \text{Irolo}$ (aún existen etiquetas temporales).

Iteración 6

Actualización de etiquetas: $\Gamma^+(p) = (\text{Jalapa})$
 $d(\text{Jalapa}) = \min \{\infty, 3.5 + 9\} = 12.5;$
 $a(\text{Jalapa}) = \text{Irolo}$

x^* = San Lorenzo, puesto que éste es el nodo con mínima etiqueta temporal. Se marca $d(\text{San Lorenzo})$ como permanente.
 p = San Lorenzo (aún existen etiquetas temporales).

Iteración 7

Actualización de etiquetas: $\Gamma^+(p) = (\text{Jalapa, Puebla, Irolo})$
 $d(\text{Jalapa}) = \min \{12.5, 5.5 + 7.5\} = 12.5;$
 $a(\text{Jalapa}) = \text{Irolo}$
 $d(\text{Puebla}) = \min \{6, 5.5 + 3\} = 6;$ $a(\text{Puebla})$ no se modifica

x^* = Puebla, puesto que éste es el nodo con mínima etiqueta temporal. Se marca $d(\text{Puebla})$ como permanente.
 p = Puebla (aún existen etiquetas temporales).

Iteración 8

Actualización de etiquetas: $\Gamma^+(p) = (\text{Tehuacán})$
 $d(\text{Tehuacán}) = \min \{\infty, 6 + 4\} = 10;$
 $a(\text{Tehuacán}) = \text{Puebla}$

x^* = Tehuacán, puesto que éste es el nodo con mínima etiqueta temporal. Se marca $d(\text{Tehuacán})$ como permanente.
 p = Tehuacán (aún existen etiquetas temporales).

Iteración 9

Actualización de etiquetas: $\Gamma^+(p) = (\text{Córdoba})$
 $d(\text{Córdoba}) = \min \{\infty, 10 + 6\} = 16;$
 $a(\text{Córdoba}) = \text{Tehuacán}$

x^* = Jalapa, puesto que éste es el nodo con mínima etiqueta temporal. Se marca $d(\text{Jalapa})$ como permanente.
 p = Jalapa (aún existen etiquetas temporales).

Iteración 10

Actualización de etiquetas: $\Gamma^+(p) = (\text{Veracruz}, \text{Córdoba})$

$d(\text{Veracruz}) = \min \{x, 12.5+4\} = 16.5$; $a(\text{Veracruz}) = \text{Jalapa}$

$d(\text{Córdoba}) = \min \{16, 12.5+7.5\} = 16$;

$a(\text{Córdoba}) = \text{Tehuacán}$

$x^* = \text{Córdoba}$, puesto que éste es el nodo con mínima etiqueta temporal. Se marca $d(\text{Córdoba})$ como permanente.

$p = \text{Córdoba}$ (aún existe una etiqueta temporal).

Iteración 11

Actualización de etiquetas: $\Gamma^+(p) = (\text{Puebla}, \text{Veracruz})$

$d(\text{Veracruz}) = \min \{16.5, 16+3.5\} = 16.5$; $a(\text{Veracruz}) = \text{Jalapa}$

$x^* = \text{Veracruz}$, puesto que éste es el nodo con mínima etiqueta temporal. Se marca $d(\text{Veracruz})$ como permanente.

Todos los nodos tienen etiqueta permanente, detenerse.

La arborescencia de rutas más cortas de raíz D.F. está formada por el conjunto de arcos en la siguiente red:

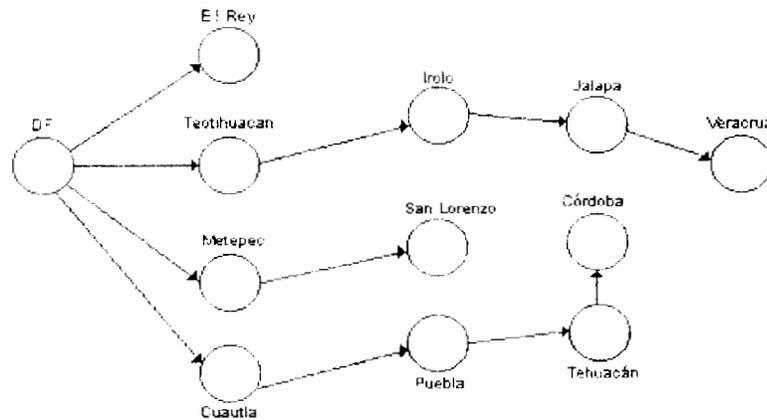


Figura 6.7

El único camino entre el D.F. y x para todo nodo x en N , en esta arborescencia es la ruta más rápida a seguir por ferrocarril que proporciona servicio entre las ciudades D.F. y x y su longitud es el número asociado al nodo x (etiqueta permanente).

RUTA MÁS CORTA ENTRE TODO PAR DE NODOS

Una manera de resolver el problema de rutas más cortas entre todo par de nodos en una red G consiste en encontrar la arborescencia de rutas más cortas de raíz x para toda x elemento de N . Sin embargo, existen procedimientos más eficientes como el que se expone en esta sección. Este procedimiento fue desarrollado por R.W. Floyd (1962) y es aplicable a redes que admiten cualquier costo en sus arcos. En dicho algoritmo se supondrá una numeración de los nodos de la red $1, 2, 3, \dots, n$ y se utilizará una matriz C de dimensión n para calcular las longitudes de las rutas más cortas entre cada par de nodos; al terminar de aplicar el algoritmo, la longitud de la ruta más corta entre los nodos i y j estará dada por el elemento (i, j) de C .

En el algoritmo de Floyd, en la k -ésima iteración se calcula la longitud de la ruta más corta entre i y j que pueda admitir a los primeros k nodos, o a alguno de ellos, como nodos intermedios, este número se almacena en la entrada (i, j) de la matriz C . Al inicio se asigna el costo del arco (i, j) al elemento (i, j) de la matriz C , si $i \neq j$; si dicho arco no existe, entonces se asigna ∞ , los valores en la diagonal serán igual a cero. Con ésto quedan calculadas las longitudes de las rutas más cortas entre todo par de nodos i y j , que no contengan ningún nodo como nodo intermedio.

Al inicio de la k -ésima iteración, la entrada (i, j) de C es igual a la longitud de la ruta más corta entre i y j que contiene a los primeros $k-1$ nodos, o a alguno de ellos, como nodos intermedios. Durante esta iteración se compara la longitud de esta ruta con la de aquella formada por la unión de las rutas más cortas que contienen a los primeros $k-1$ nodos como nodos intermedios entre i y k y k y j ; de esta manera se obtiene la ruta más corta entre i y j que contiene a los primeros k nodos, o a algunos de ellos, como nodos intermedios. Procediendo de este modo se tendrá que, al final de la n -ésima iteración, la entrada (i, j) de C es la longitud de la ruta más corta entre i y j , que contiene a los primeros n nodos como nodos intermedios o a alguno de ellos; es decir, se habrá calculado la longitud de la ruta más corta entre i y j .

Debe observarse que si, al finalizar el algoritmo, alguna entrada de C es igual a ∞ , ésto querrá decir que no existe ruta alguna entre los nodos correspondientes. Por otro lado, si algún elemento de la diagonal de C , por ejemplo (i, i) , es menor que cero en alguna iteración, se habrá encontrado una ruta de i a i de longitud negativa (es decir, un circuito negativo), luego en este caso el problema no tiene solución. Este algoritmo por lo anterior es importante porque detecta circuitos negativos.

ALGORITMO DE FLOYD

Propósito: Obtener las rutas más cortas entre todo par de nodos en una red G con n nodos.

Descripción

PASO 1 Construya la matriz C de dimensión n cuyos elementos c_{ij} son como sigue:

PASO 2 Haga $k = k+1$ para toda $i \neq k$ tal que $c_{ij} \neq \infty$ y para todo $j \neq k$ tal que $c_{kj} \neq \infty$, haga:

$$C_{ij} = \begin{cases} 0 & \text{si } i = j \\ \infty & \text{si } (i, j) \notin A \\ d(i, j) & \text{si } (i, j) \in A \end{cases}$$

$k = 0$

$$c_{ij} = \min\{c_{ij}, c_{ik} + c_{kj}\}$$

PASO 3 i) Si $c_{ii} < 0$ para alguna i, termine. En este caso existe un circuito negativo que contiene al nodo i y por lo tanto no hay solución.

ii) Si $c_{ij} \geq 0$, para toda i y $k=n$ termine. c_{ij} es la longitud del camino mas corto de i a j.

iii) Si $c_{ij} \geq 0$ para toda i, y $k < n$ ir al paso 2.

RECUPERACIÓN DE LAS RUTAS

Para recuperar las rutas más cortas puede construirse una matriz A de dimensión n, donde el elemento a_{ij} de esta matriz será el predecesor del nodo j en la ruta de i a j encontrada en cada iteración. Dada la definición de A, sus entradas se inicializarán $a_{ij} = i$, para todo par de nodos $i, j \in N$.

La matriz A se modificará en el paso 2 de la k-ésima iteración de acuerdo a la siguiente asignación:

$$a_{ij} = \begin{cases} a_{ki} & \text{si } c_{ik} + c_{kj} < c_{ij} \\ \text{no cambia} & \text{si } c_{ij} \leq c_{ik} + c_{kj} \end{cases}$$

Justificación del algoritmo

El algoritmo de Floyd termina en exactamente n iteraciones, donde n es el número de nodos de la red, a menos que termine en el inciso i) del paso 3, en cuyo caso terminará en menos de n iteraciones. Se mostrará la optimalidad por inducción sobre el número de iteraciones.

Al principio de la iteración 1 (Paso 1) la entrada (i,j) de la matriz representa la longitud de la ruta más corta, que no contiene ningún nodo intermedio entre los nodos i y j . Durante el paso 2 de esta iteración se realiza una comparación entre dicha longitud (c_{ij}) y la de aquella ruta formada por la unión de la ruta entre i y 1 y la ruta entre 1 y j ($c_{i1} + c_{1j}$); de este modo se obtiene la longitud de la ruta más corta entre los nodos i y j , que no contiene ningún nodo intermedio o que contiene al nodo 1 como intermedio.

Suponga que al final de la iteración $k-1$, c_{ij} representa la longitud de la ruta más corta entre i y j , que contiene los primeros $k-1$ nodos como nodos intermedios o a algunos de ellos. Durante el paso 2 de la iteración k , se realiza la comparación entre esta última longitud (c_{ij}) y la de aquella ruta formada por la unión de las rutas más cortas, que admiten a los primeros $k-1$ nodos como nodos intermedios entre i y k y entre k y j ($c_{ik} + c_{kj}$); entonces, al final de la k -ésima iteración, c_{ij} es igual a la longitud de la ruta más corta entre i y j , que contiene a los primeros k nodos como intermedios o a algunos de ellos.

De lo anterior se concluye que, al final de la iteración n , c_{ij} es la longitud de la ruta más corta entre los nodos i y j .

EJEMPLO 6.3

Considere la siguiente red y determine las rutas más cortas entre todo par de nodos; se usará el algoritmo de Floyd para la determinación de las rutas:

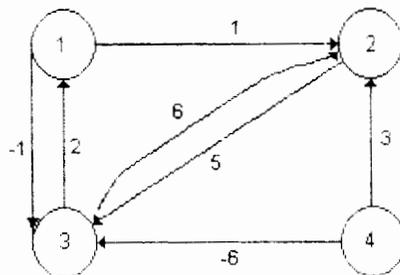


Figura 6.8

Iteración 1

Las matrices C y A iniciales son las siguientes:

PASO 1

$$C = \begin{bmatrix} 0 & 1 & -1 & \infty \\ \infty & 0 & 5 & \infty \\ 2 & 6 & 0 & \infty \\ \infty & 3 & -6 & 0 \end{bmatrix} \quad A = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 2 & 2 & 2 \\ 3 & 3 & 3 & 3 \\ 4 & 4 & 4 & 4 \end{bmatrix}$$

PASO 2

Sea $k=1$. Se actualizan los valores de los elementos:

$$C_{32} = \min \{6, 2+1\} = 3 \quad a_{32} = a_{12} = 1$$

$$C_{33} = \min \{0, 2, -1\} = 0 \quad \text{no se modifica } a_{33}$$

Entonces las matrices resultantes de esta iteración son las siguientes:

$$C = \begin{bmatrix} 0 & 1 & -1 & \infty \\ \infty & 0 & 5 & \infty \\ 2 & 3 & 0 & \infty \\ \infty & 3 & -6 & 0 \end{bmatrix} \quad A = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 2 & 2 & 2 \\ 3 & 1 & 3 & 3 \\ 4 & 4 & 4 & 4 \end{bmatrix}$$

PASO 3

iii) $C_{ii} \geq 0 \quad \forall i, k < n$, ir al paso 2

Iteración 2

Se asigna $k=2$ y se actualizan los elementos:

$$C_{13} = \min \{-1, 1+5\} = -1 \quad \text{no se modifica } a_{13}$$

$$C_{33} = \min \{0, 3+5\} = 0 \quad \text{no se modifica } a_{33}$$

$$C_{43} = \min \{-6, 3+5\} = -6 \quad \text{no se modifica } a_{43}$$

Luego, las matrices C y A no se modificaron durante la segunda iteración.

Iteración 3

Se asigna $k=3$ y se actualizan los elementos:

$$C_{12} = \min \{1, -1+3\} = 1 \quad \text{no se modifica } a_{12}$$

$$C_{21} = \min \{\infty, 5+2\} = 7 \quad a_{21} = a_{31} = 3$$

$$C_{22} = \min \{0, 5+3\} = 0 \quad \text{no se modifica } a_{22}$$

$$C_{42} = \min \{3, -6+3\} = -3 \quad a_{42} = a_{32} = 1$$

Entonces las matrices resultantes de esta iteración son las siguientes:

$$C = \begin{bmatrix} 0 & 1 & -1 & \infty \\ 7 & 0 & 5 & \infty \\ 2 & 3 & 0 & \infty \\ -4 & -3 & -6 & 0 \end{bmatrix} \quad A = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 3 & 2 & 3 & 3 \\ 3 & 1 & 3 & 3 \\ 3 & 1 & 4 & 4 \end{bmatrix}$$

Iteración 4

Se asigna $k=4$. En esta iteración no se realiza ningún cambio puesto que $C_{i4}=\infty$, para $i=1,2,3$. Entonces la última matriz C es la matriz de longitudes más cortas. El elemento a_{ij} de la última matriz A es el predecesor del nodo j , en la ruta más corta de i a j , siempre y cuando $C_{ij} \neq \infty$.

Observe que, puesto que $C_{i4} = \infty$ (para $i=1,2,3$), entonces no existe ninguna ruta entre los nodos i y 4 (para $i=1,2,3$); esto puede verificarse fácilmente en la red siguiente.

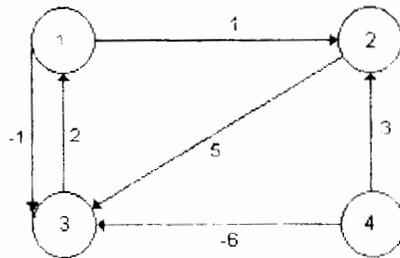


Figura 6.9

El Algoritmo de Floyd se puede reformular de la siguiente manera.

ALGORITMO DE FLOYD

PASO GENERAL K. Defina el renglón k y la columna k como renglón pivote y columna pivote. Aplique la triple operación a cada elemento d_{ij} en D_{k-1} para todo i y j . Si la condición $d_{ik} + d_{kj} < d_{ij}$ ($i \neq k, j \neq k, i \neq j$) se satisface, haga los cambios siguientes:

- a) Crear D_k reemplazando d_{ij} en D_{k-1} con $d_{ik} + d_{kj}$.
- b) Crear S_k reemplazando s_{ij} en S_{k-1} con k .

Esto se puede ver fácilmente como sigue: El renglón k y la columna k son la columna y el pivote actuales.

El renglón i representa cualquiera de los renglones $1, 2, \dots, y k-1$

El renglón p representa cualquiera de los renglones $k+1, k+2, \dots, n$

La columna j representa a las columnas $1, 2, \dots, k-1$.

La columna q representa a las columnas $k+1, k+2, \dots, n$.

- 1.- Si $d_{ik} + d_{kj} < d_{ij}$, reemplace d_{ij} con $d_{ik} + d_{kj}$ y cambie s_{ij} por k .
- 2.- Si $d_{ik} + d_{kq} < d_{iq}$, reemplace d_{iq} con $d_{ik} + d_{kq}$ y cambie s_{iq} por k .
- 3.- Si $d_{pk} + d_{kj} < d_{pj}$, reemplace d_{pj} con $d_{pk} + d_{kj}$ y cambie s_{pj} por k .
- 4.- Si $d_{pk} + d_{kq} < d_{pq}$, reemplace d_{pq} con $d_{pk} + d_{kq}$ y cambie s_{pq} por k .

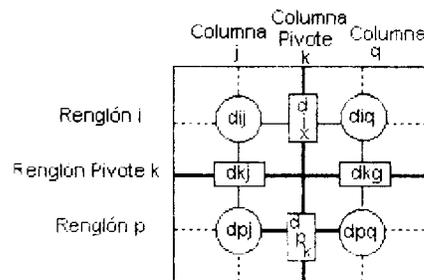


Figura 6.10

Después de n pasos, podemos determinar la ruta más corta entre los nodos i y j de las matrices D_n y S_n usando las siguientes reglas:

1. De D_n , d_{ij} da la distancia más corta entre los nodos i y j .
2. De S_n , determinar el nodo intermedio $k = s_{ij}$, el cual proporciona la ruta $i \rightarrow k \rightarrow j$. Si $s_{ik} = k$, detener; todos los nodos intermedios de la ruta han sido hallados. De otra forma, repetir el procedimiento entre los nodos i y k .

EJEMPLO 6.4

Para la siguiente red, encuentre las rutas más cortas entre cada par de nodos. Las distancias están dadas en los arcos. El arco (5, 6) es direccional, por lo que no se permite flujo del nodo 6 al nodo 5. Todos los otros arcos permiten flujos en ambas direcciones.

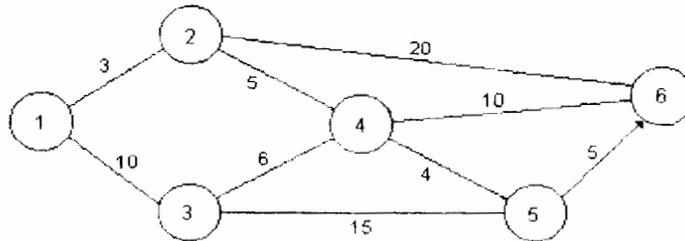


Figura 6.11

Iteración 1.

Las matrices D_0 y S_0 proporcionan la representación inicial de la red. Note que $d_{65} = \infty$ debido a que no se permite el tráfico del nodo 6 al nodo 5. El renglón y la columna pivotes son dados por el primer renglón y la primer columna (marcados). $k=1$.

	D_0						S_0					
	1	2	3	4	5	6	1	2	3	4	5	6
1	0	3	10	∞	∞	∞	1	2	3	4	5	6
2	3	0	∞	5	∞	20	1	2	3	4	5	6
3	10	∞	0	6	15	∞	1	2	3	4	5	6
4	∞	5	6	0	4	10	1	2	3	4	5	6
5	∞	∞	15	4	0	5	1	2	3	4	5	6
6	∞	20	∞	10	∞	0	1	2	3	4	5	6

En la matriz D_0 , los elementos d_{23} y d_{32} son los únicos que pueden ser mejorados. Por lo tanto, para obtener D_1 y S_1 de D_0 y S_0 :

1. Reemplace d_{23} con $d_{21} + d_{13} = 3 + 10 = 13$ y haga $s_{23} = 1$.
2. Reemplace d_{32} con $d_{31} + d_{12} = 10 + 3 = 13$ y haga $s_{32} = 1$.

Estos cambios se muestran en D_1 y S_1 .

Iteración 2

Haga $k=2$. Los cambios en los elementos de D_1 se muestran en D_2 y S_2 .

		D_1						S_1						
		1	2	3	4	5	6	1	2	3	4	5	6	
1	[0	3	10	∞	∞	∞	1	1	2	3	4	5	6]
2	[3	0	13	5	∞	20	2	1	2	1	4	5	6]
3	[10	13	0	6	15	∞	3	1	1	3	4	5	6]
4	[∞	5	6	0	4	10	4	1	2	3	4	5	6]
5	[∞	∞	15	4	0	5	5	1	2	3	4	5	6]
6	[∞	20	∞	10	∞	0	6	1	2	3	4	5	6]

Iteración 3

Fije el renglón y la columna pivotes en $k=3$.

		D_2						S_2						
		1	2	3	4	5	6	1	2	3	4	5	6	
1	[0	3	10	8	∞	23	1	1	2	3	2	5	2
2	[3	0	13	5	∞	20	2	1	2	1	4	5	6]
3	[10	13	0	6	15	33	3	1	1	3	4	5	2
4	[8	5	6	0	4	10	4	2	2	3	4	5	6]
5	[∞	∞	15	4	0	5	5	1	2	3	4	5	6]
6	[23	20	33	10	∞	0	6	2	2	2	4	5	6]

Iteración 4.

Fije el renglón y la columna pivotes en $k=4$.

		D_3						S_3						
		1	2	3	4	5	6	1	2	3	4	5	6	
1	[0	3	10	8	25	23	1	1	2	3	2	3	2]
2	[3	0	13	5	28	20	2	1	2	1	4	3	6]
3	[10	13	0	6	15	33	3	1	1	3	4	5	2]
4	[8	5	6	0	4	10	4	2	2	3	4	5	6]
5	[25	28	15	4	0	5	5	3	3	3	4	5	6]
6	[23	20	33	10	48	0	6	2	2	2	4	3	6]

Iteración 5.

Fije el renglón y la columna pivotes en $k=5$.

	D_4					
	1	2	3	4	5	6
1	0	3	10	8	12	18
2	3	0	11	5	9	15
3	10	11	0	6	10	16
4	8	5	6	0	4	10
5	12	9	10	4	0	5
6	18	15	16	10	14	0

	S_4					
	1	2	3	4	5	6
1	1	2	3	2	4	4
2	1	2	4	4	4	4
3	1	4	3	4	4	4
4	2	2	3	4	5	6
5	4	4	4	4	5	6
6	4	4	4	4	4	6

Iteración 6.

Fije el renglón y la columna pivotes en $k=5$. Como muestra la matriz D_5 , ya no es posible mejorar ninguna de las rutas. Por lo tanto, D_5 y S_5 son las matrices óptimas para la red.

	D_5					
	1	2	3	4	5	6
1	0	3	10	8	12	17
2	3	0	11	5	9	14
3	10	11	0	6	10	15
4	8	5	6	0	4	9
5	12	9	10	4	0	5
6	18	15	16	10	14	0

	S_5					
	1	2	3	4	5	6
1	1	2	3	2	4	5
2	1	2	4	4	4	5
3	1	4	3	4	4	5
4	2	2	3	4	5	5
5	4	4	4	4	5	6
6	4	4	4	4	4	6

Estas matrices contienen toda la información necesaria para determinar la ruta más corta entre todo par de nodos en la red. Por ejemplo, la distancia más corta entre el nodo 1 y el nodo 6 es $d_{16}=17$. Para determinar la ruta, sea cada segmento (i, j) una ruta directa si $s_{ij}=j$, de otra manera i y j están ligadas por al menos otro nodo intermedio.

Como $s_{16}=5$ y $s_{56}=6$, la ruta está dada inicialmente por $1 \rightarrow 5 \rightarrow 6$. Como $s_{15} \neq 5$, el segmento $(1, 5)$ no es una ruta directa y necesitamos determinar sus nodos intermedios; dados $s_{15}=4$ y $s_{45}=5$, la ruta $1 \rightarrow 5$ es reemplazada por $1 \rightarrow 4 \rightarrow 5$ y como $s_{14}=2$ y $s_{24}=4$, $1 \rightarrow 4$ es reemplazada por $1 \rightarrow 2 \rightarrow 4$. Finalmente, tenemos $s_{12}=2$ y no existen más nodos intermedios, lo que implica que $1 \rightarrow 2 \rightarrow 4 \rightarrow 5 \rightarrow 6$ es la ruta óptima y la distancia asociada a la ruta es 17.

Otra manera de resolver el problema de ruta más corta es usando programación dinámica como se muestra en la siguiente sección.

6.4 PROBLEMA DE RUTA MÁS CORTA CON PROGRAMACIÓN DINÁMICA

La programación dinámica es un procedimiento de optimización que es aplicable particularmente a problemas que requieren una sucesión de decisiones interrelacionadas. Cada decisión transforma la situación actual en una situación nueva. Una sucesión de decisiones, que a su vez producen una sucesión de situaciones, que buscan maximizar (o minimizar) alguna medida de valor. El valor de la sucesión de decisiones generalmente es igual a la suma de los valores de decisiones individuales y situaciones en la sucesión. De esta forma se puede plantear al problema de encontrar la mejor trayectoria de un lugar a otro.

PROBLEMA DE TRAYECTORIA SIMPLE

Suponga que vive en una ciudad cuyas calles se encuentran como en la figura siguiente:

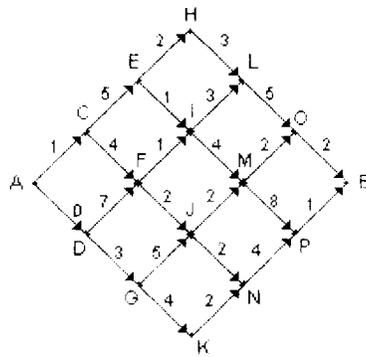


Figura 6.12

Todas las calles son de un solo sentido y los números representan el esfuerzo (que usualmente es tiempo, pero algunas veces costo o distancia) requerido para atravesar cada bloque o cuadra individual. Si usted vive en A y quiere llegar a B con el mínimo esfuerzo total. Se puede resolver este problema enumerando todas las trayectorias posibles de A a B. Sumando los esfuerzos bloque por bloque para cada trayectoria y positivamente escogiendo la menor suma.

Sin embargo hay 20 trayectorias posibles de A a B (¿Por qué?).

Si enumeramos las calles de la siguiente manera:

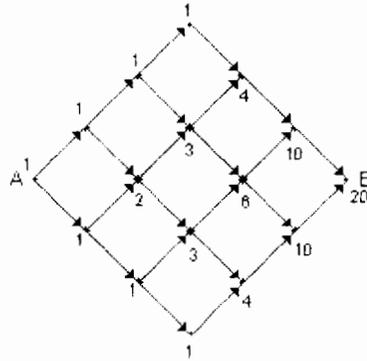


Figura 6.13

Se pone un 1 en A pues para llegar ahí sólo hay una posibilidad (de hecho se va a salir de ahí). Después se pone un 1 en cada esquinas inmediata porque sólo hay una forma de ir a ellas desde A, dos en la siguiente esquina pues hay 2 formas de llegar a ella y así en cada esquina se pone la suma de los dos números más cercanos situados sobre el o los caminos que conduzcan hasta ella.

Si cortamos a la mitad esta figura y la rotamos:

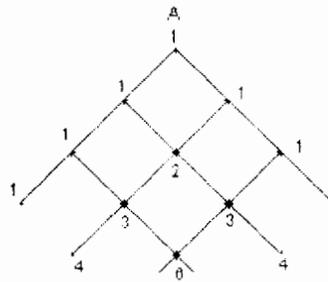


Figura 6.14

Los números de este triángulo son los números del triángulo de Pascal; entonces, el algoritmo para contar el número de trayectorias mínimas que bajan desde la cúspide, no es otro que el método usual para construir el triángulo de Pascal.

Entonces para una red de $n \times m$ el número de trayectorias posibles de A a B es igual al número de combinaciones de $n+m$ elementos tomados de n en n .

$$\frac{n+m}{n}$$

Si $n=m$ se tiene:

$$\frac{(2n)!}{n!n!} = \frac{(2(3))!}{3!3!} = \frac{6!}{3!3!} = \frac{6 \times 5 \times 4}{3 \times 2} = \frac{120}{6} = 20$$

Si se toma una red de 30×30 son 2000 años.

Se tiene entonces 20 caminos de A a B y 5 sumas que producen la suma de 6 números a lo largo de la trayectoria, así 100 sumas podrían producir las 20 sumas de las trayectorias a ser comparadas. Ya que una comparación produce el menor de 2 números, una comparación adicional (de ese número con un tercero) produce el menor de 3, y así sucesivamente. Las enumeraciones completan esta solución del problema.

Sin embargo se puede resolver este problema usando programación dinámica.

LA SOLUCIÓN DE PROGRAMACIÓN DINÁMICA

Para desarrollar el problema en términos de programación dinámica, razonamos de la siguiente forma: "No sé si voy a avanzar diagonalmente hacia arriba o diagonalmente hacia abajo desde A, pero si sólo conozco 2 números adicionales, -es decir el esfuerzo total requerido para ir de C a B por la mejor trayectoria (el mínimo esfuerzo) y el esfuerzo requerido de ir de D a B por la mejor trayectoria- puedo entonces hacer la mejor selección desde A".

Denotamos el mínimo esfuerzo desde C a B por S_C y el mínimo esfuerzo desde D a B por S_D , se puede entonces aumentar el esfuerzo de ir de A a C a S_C , obteniendo así el esfuerzo requerido de la mejor trayectoria de ir de A a B usando C. Lo mismo se hace con S_D y finalmente se comparan estas 2 sumas para encontrar el mínimo esfuerzo y la mejor decisión.

Por supuesto, todo esto presupone el conocimiento de los números S_C y S_D . Sin embargo, una de las ideas centrales de la programación dinámica se ha establecido ya que sólo el esfuerzo a través de las mejores trayectorias desde C y desde D hacia B son relevantes para los cálculos y el esfuerzo a lo largo de las otras 9 trayectorias respectivas desde C y D a B nunca necesitan calcularse. Esta observación a menudo se conoce como el principio de optimalidad y se establece como sigue:

Principio de Optimalidad

La mejor trayectoria de A a B tiene la propiedad de que cualquiera que sea la decisión inicial en A, la trayectoria remanente a B, comenzando desde el siguiente punto después de A, debe ser la mejor trayectoria desde ese punto a B.

Sea S_C y S_D como ya se había mencionado, podemos citar el principio de optimalidad para justificar la fórmula:

$$S_A = \min \begin{cases} 1 + S_C \\ 0 + S_D \end{cases}$$

donde S_A es el mínimo esfuerzo de ir de A a B y el símbolo "min []" significa "el menor entre x e y".

La segunda idea importante es: Mientras los dos números S_C y S_D son desconocidos para nosotros, inicialmente podemos calcular S_C si conocemos S_E y S_F (los esfuerzos mínimos desde E y F para llegar a B respectivamente) y remitiéndonos al principio de optimalidad escribimos:

$$S_C = \min \begin{cases} 5 + S_E \\ 4 + S_F \end{cases}$$

así mismo:

$$S_D = \min \begin{cases} 7 + S_E \\ 3 + S_G \end{cases}$$

Así continuamos con S_E , S_F y S_G si no los conocemos, los podemos calcular si S_H , S_I , S_J y S_K se conocen y continuamos con este razonamiento hasta llegar a O y P, trabajando hacia atrás desde O y P hasta A tenemos:

$$S_E = \min \begin{cases} 2 + S_H = 2 + 10 \\ 1 + S_I = 1 + 8 \end{cases} = 9$$

$$S_F = \min \begin{cases} 1 + S_I = 1 + 8 \\ 2 + S_J = 2 + 6 \end{cases} = 8$$

$$S_G = \min \begin{cases} 5 + S_J = 5 + 6 \\ 4 + S_K = 4 + 7 \end{cases} = 11$$

$$S_C = \min \begin{cases} 5 + S_E = 5 + 9 \\ 4 + S_F = 4 + 8 \end{cases} = 12$$

$$S_D = \min \begin{cases} 7 + S_E = 7 + 8 \\ 3 + S_G = 3 + 11 \end{cases} = 14$$

$$S_A = \min \begin{cases} 1 + S_C = 1 + 12 \\ 0 + S_D = 0 + 14 \end{cases} = 13$$

$$S_L = 5 + S_O = 7$$

$$S_H = 3 + S_L = 10$$

$$S_M = \min \begin{cases} 2 + S_O = 2 + 2 \\ 8 + S_P = 8 + 1 \end{cases} = 4$$

$$S_N = 4 + S_P = 5$$

$$S_I = \min \begin{cases} 3 + S_L = 3 + 7 \\ 4 + S_M = 4 + 4 \end{cases} = 8$$

$$S_J = \min \begin{cases} 2 + S_M = 2 + 4 \\ 2 + S_N = 2 + 5 \end{cases} = 6$$

$$S_K = 2 + S_N = 2 + 5 = 7$$

Como la mejor trayectoria de $S_A=13$ desarrollamos una suma en cada punto H, L, O, K, N y P (que son 6 puntos) donde sólo fue posible una decisión y desarrollamos 2 sumas y una comparación por cada uno de los 9 puntos remanentes donde las 2 decisiones iniciales fueron posibles. Esto da 24 sumas y 9 comparaciones, lo cual contrasta con 100 sumas y 19 comparaciones usando enumeración exhaustiva (o fuerza bruta).

Una vez que conocemos el esfuerzo total queremos saber cuál es la trayectoria, ésta se obtiene fácilmente fijándonos en las 2 primeras decisiones y buscando el mínimo en nuestro cálculo previo.

Si denotamos x un nodo inicial cualquiera y P_x el nodo posterior al nodo x en la trayectoria óptima desde x a B entonces se tiene:

$$P_M=O \text{ ya que } 2+S_O \text{ fue menor que } 8+S_P$$

$$P_I=M \text{ ya que } 4+S_M < 3+S_L \text{ etc.}$$

Así, se tiene que:

$$P_O = B \quad P_P = B$$

$$P_L = O \quad P_M = O \quad P_N = P$$

$$P_H = L \quad P_I = M \quad P_J = M \quad P_K = N$$

$$P_R = I \quad P_F = J \quad P_G = J \text{ o } K$$

$$P_C = F \quad P_D = G$$

$$P_A = C$$

La mejor trayectoria es:

$$A \rightarrow C \rightarrow F \rightarrow J \rightarrow M \rightarrow O \rightarrow B$$

Sumando las distancias a lo largo de la trayectoria se tiene

$$1 + 4 + 2 + 2 + 2 + 2 = 13$$

que es igual a S_A .

TERMINOLOGIA.

La regla que asigna los valores a varios subproblemas es la función de valor óptimo (en este caso S).

El subíndice de S -por ejemplo, S_A - es el argumento de la función S y cada argumento se refiere a un subproblema en particular (por la definición de S , en este caso A significa la mejor trayectoria de A a B que es deseada).

La regla que asocia la primer mejor decisión con cada subproblema -la función P- es la función de política óptima.

El principio de optimalidad produce una fórmula o un conjunto de fórmulas relativas a varios valores de S. Esta fórmula es la relación de recurrencia. Finalmente, el valor de la función de valor óptimo S para ciertos argumentos se supone obvia del tipo de problema y de la definición de S que no requieren cálculos. Estos valores obvios se conocen como condiciones de frontera a S.

En este caso: Sea

S_x = trayectoria más corta de x a B

X = A, C, D, ...

$S_B = 0$ (Condición de frontera)

Ecuaciones Recursivas.

donde

C_{xy} = longitud del nodo x al y

C_{xz} = longitud del nodo x al z

además se tiene la longitud mínima de todo nodo al nodo B.

6.5 NOTAS HISTORICAS

El problema de la ruta más corta y sus generalizaciones tienen una voluminosa literatura. Como una guía al respecto hasta antes de 1984, remitimos al lector a la extensa bibliografía compilada por Deo y Pang (1984). En este escrito presentamos algunas referencias seleccionadas; referencias adicionales pueden encontrarse en los artículos de Ahuja, Magnanti y Orlin (1989, 1991).

El primer algoritmo de etiquetado fue sugerido por Dijkstra (1959) e, independientemente, por Dantzing (1960) y Whiting y Hillier (1960). La instrumentación original del algoritmo de Dijkstra corre en un tiempo de $O(n^2)$, el cual es el tiempo óptimo de corrida para redes completamente densas [aquellas con $m = \Omega(n^2)$] debido a que cualquier algoritmo debe examinar todos los arcos.

Además, el uso de apilamientos nos permite obtener mejores tiempos de corrida para redes esparcidas. El uso del apilamiento-d del algoritmo de Dijkstra con $d = \max\{2, \lceil m/n \rceil\}$ corre en un tiempo de $O(m \log_d n)$ y fue hecho por Johnson (1977a). El uso del apilamiento de Fibonacci hecho por Fredman y Tarjan (1984) corre en un tiempo de $O(m + n \log n)$. Johnson (1982) sugirió el uso del algoritmo de Dijkstra $O(m \log \log C)$ basado en un trabajo anterior de Boas, Kaas y Sijlstra (1977). El algoritmo escalable de Gabow (1985) es otro eficiente algoritmo de ruta más corta.

El algoritmo de Floyd-Warshall, publicado por Floyd (1962), se basó en el algoritmo de Warshall (1962).

CAPÍTULO 7

PROBLEMA DE FLUJO A COSTO MÍNIMO

7.1 INTRODUCCIÓN

En este capítulo se analiza el problema de una red de flujo donde se tienen capacidades en los arcos y se desea enviar flujo a costo mínimo. Se verán tres algoritmos, el primero de ellos se refiere al uso de la programación lineal y el método simplex, que se adaptará al problema de redes. El segundo método se refiere a la eliminación de circuitos negativos y, finalmente, el tercero se basa en rutas más cortas. El problema de flujo a costo mínimo generaliza el modelo de flujo máximo en 4 aspectos:

1. Todos los arcos son dirigidos.
2. Un flujo de costo unitario está asociado con cada arco.
3. Los arcos pueden tener capacidades mínimas y máximas.
4. Cualquier nodo en la red puede actuar como nodo fuente (con oferta) o como nodo destino (con demanda).

El objetivo de este modelo es determinar los flujos en los diferentes arcos que minimizan el costo total mientras satisfacen las restricciones de flujo en los arcos y las cantidades de oferta y demanda en los nodos.

Sea la red con restricciones de capacidad $G = (N,A)$ donde N es el conjunto de nodos y A el conjunto de arcos.

x_{ij} = La cantidad de flujo del nodo i al nodo j
 $u_{ij}(l_{ij})$ = Capacidad superior (inferior) del arco (i, j)
 c_{ij} = Unidad de costo del flujo del nodo i al nodo j
 f_i = Flujo neto en el nodo i

Esto se puede ver en la red de la figura siguiente:

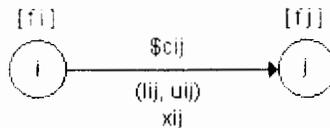


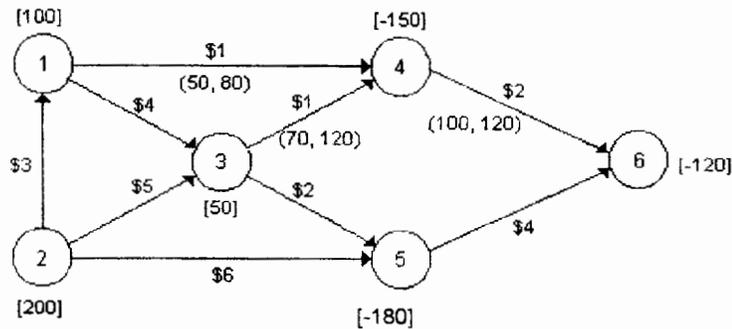
Figura 7.1

La etiqueta $[f_i]$ supone un valor positivo (negativo) cuando se tiene una oferta (demanda) asociada al nodo i .

EJEMPLO 7.1

MASECA debe distribuir maíz de tres silos a tres granjas. Las cantidades de oferta en los tres silos son 100, 200 y 50 toneladas y la demanda de las tres granjas es de 150, 80 y 120 toneladas. MASECA usa principalmente ferrocarril para transportar el maíz a las granjas con excepción de tres rutas que usan trailers. La siguiente red muestra la problemática:

Figura 7.2



Los nodos 1, 2 y 3 representan los silos con sus respectivas ofertas. Las granjas están representadas por los nodos 4, 5 y 6 cuyas demandas están representadas por [-150], [-180] y [-120]. Las rutas permiten transporte entre los silos. Las rutas de los trailers están representadas por (1, 4), (3, 4) y (4, 6). Las capacidades en las rutas están dadas en términos de las toneladas a transportar (asociadas a la capacidad de los trailers); por ejemplo, en la ruta (1, 4) se tiene entre 50 y 80 toneladas. Las otras rutas que usan tren tienen capacidad ilimitada. Los costos de transportación por tonelada están indicados en los arcos.

La solución óptima será el flujo a costo mínimo en los diferentes arcos que satisfagan las restricciones de flujo en el modelo.

7.2 FORMULACIÓN DE PROGRAMACION LINEAL

El modelo de la red con capacidad limitada se puede formular como un modelo de programación lineal.

$$\min z = \sum_{(i,j) \in A} \sum c_{ij} x_{ij}$$

Sujeto a

$$\sum_{(i,j) \in A} x_{jk} - \sum_{(i,j) \in A} x_{ij} = f_j \quad j \in N$$

$$l_{ij} \leq x_{ij} \leq u_{ij}$$

Si lo escribimos en formato de tabla

	X_{12}	X_{13}	X_{14}	X_{23}	X_{25}	X_{34}	X_{35}	X_{46}	X_{56}	
min	3	4	1	5	6	1	2	2	4	
nodo 1	1	1	1							= 100
nodo 2	-1			1	1					= 200
nodo 3		-1		-1		1	1			= 50
nodo 4			-1			-1		1		= -150
nodo 5					-1		-1		1	= -80
nodo 6								-1	-1	= -120
cotas inf.	0	0	50	0	0	70	0	100	0	
cotas sup.	∞	∞	80	∞	∞	120	∞	120	∞	

Sustituyendo las variables:

$$X_{14} = X_{14}' + 50$$

$$X_{34} = X_{34}' + 70$$

$$X_{46} = X_{46}' + 100$$

	X_{12}	X_{13}	X_{14}	X_{23}	X_{25}	X_{34}	X_{35}	X_{46}	X_{56}	
min	3	4	1	5	6	1	2	2	4	
nodo 1	1	1	1							= 100
nodo 2	-1			1	1					= 200
nodo 3		-1		-1		1	1			= 50
nodo 4			-1			-1		1		= -150
nodo 5					-1		-1		1	= -80
nodo 6								-1	-1	= -120
cotas sup.	∞	∞	30	∞	∞	50	∞	20	∞	

La red correspondiente después de que se han sustituido las cotas inferiores se muestra en la figura 7.4

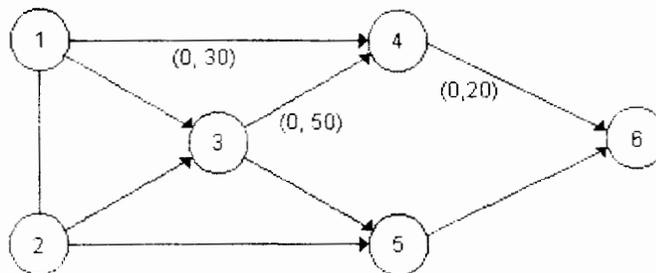


Figura 7.4

Note que la sustitución de cotas inferiores se puede efectuar directamente de la figura 7.3 a la figura 7.4

El siguiente ejemplo proporciona una ilustración de un modelo de redes que inicialmente no satisface el requerimiento de flujo en nodos, esto es, que el flujo que sale menos el que entra no es igual al nodo neto de la red, pero se puede convertir a esta estructura a través de manipulaciones especiales de las restricciones en el programa lineal.

EJEMPLO 7.2

La agencia de empleos "El Empleo Seguro", tiene un contrato para proporcionar trabajadores para los próximos cuatro meses (de enero a abril) de acuerdo con el siguiente calendario:

Mes	Enero	Febrero	Marzo	Abril
Número de Trab.	100	120	80	170

En general, debido a la demanda errática, podría resultar económico tener más trabajadores que los necesarios en un mes determinado. La agencia estima que el costo por reclutar y mantener trabajadores es una función del periodo de empleo como se muestra en la tabla siguiente:

Período de empleo (meses)	1	2	3	4
costo por trabajador (\$)	100	130	180	220

Sea x_{ij} = número de trabajadores contratados al inicio del periodo i y terminado al inicio del mes j .

Por ejemplo, x_{12} nos da el número de trabajadores contratados en Enero por sólo un mes.

Para poder plantear el problema de programación lineal para un periodo de cuatro meses, necesitamos definir la variable que representa contratar en abril para abril. Esto se logra aumentando el mes de mayo (mes 5), tal que x_{45} definirá la variable deseada. Naturalmente, mayo no tiene demanda.

Las restricciones reconocen que la demanda para algún periodo k puede satisfacerse por todas las x_{ij} tales que $i \leq k < j$. Sean $S_i \geq 0$ las variables de exceso del número de trabajadores en el mes i , el programa lineal está dado por:

	X ₁₂	X ₁₃	X ₁₄	X ₁₅	X ₂₃	X ₂₄	X ₂₅	X ₃₄	X ₃₅	X ₄₅	S ₁	S ₂	S ₃	S ₄	
	100	130	180	220	100	130	180	100	130	100					Min
Enero	1	1	1	1							-1				100
Feb.		1	1	1	1	1	1					-1			120
Mar.			1	1		1	1	1	1				-1		30
Abril				1			1		1	1				-1	170

El programa lineal anterior no tiene la estructura especial (-1, +1) de la red de flujo. Sin embargo, notamos que dado un programa lineal satisface dos condiciones especiales:

1. Los coeficientes de las restricciones de cada variable son cero o uno.
2. Todos los números unos son consecutivos (no intervienen ceros).

El programa lineal con n ecuaciones que satisfaga esas propiedades se puede convertir en un modelo de flujo equivalente a través de efectuar las siguientes manipulaciones aritméticas:

1. Crear una nueva ecuación n+1, multiplicando la ecuación n por -1
2. Dejar la ecuación 1 sin cambio.
3. Para $i=2,3,\dots,n$ reemplace cada ecuación i con (ecuación i)-(ecuación i-1)

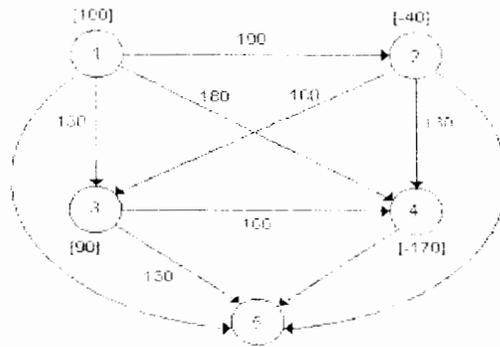
La aplicación de estas manipulaciones algebraicas se pueden ver en la siguiente tabla:

	X ₁₂	X ₁₃	X ₁₄	X ₁₅	X ₂₃	X ₂₄	X ₂₅	X ₃₄	X ₃₅	X ₄₅	S ₁	S ₂	S ₃	S ₄	
	100	130	180	220	100	130	180	100	130	100					Min
Enero	1	1	1	1							-1				100
Feb	-1				1	1	1				1	-1			-20
Mar		-1			-1			1	1			1	-1		-40
Abril			-1			-1		-1		1			1	-1	90
Mayo				-1			-1		-1	-1				1	-170

Nota: en esta tabla los espacios en blanco representan ceros en las columnas.

Usando la formulación anterior, el modelo se puede representar equivalentemente por la red de costo de flujo mínimo en la figura 7.5

Figura 7.5



7.3 ALGORITMO SIMPEX PARA REDES CON CAPACIDADES EN LOS ARCOS

El algoritmo simplex para redes con capacidades en los arcos está basado en los pasos del método simplex regular. El "nuevo" método, sin embargo, explota la estructura especial del modelo de costo mínimo de la red.

Dada f_i , el flujo neto en el nodo i se define en el programa lineal como en la sección 7.2, el algoritmo simplex para la red capacitada estipula que la red debe satisfacer la condición

$$\sum_{i=1}^n f_i = 0$$

Esta condición dice que la demanda total en la red debe ser igual a la demanda total. Podemos satisfacer siempre esta condición aumentando un nodo destino, conectado a todos los otros nodos en la red y que tenga en los arcos un costo cero, y capacidades ilimitadas. Note que el balanceo de la red no garantiza que el modelo tendrá una solución factible.

ALGORITMO SIMPLEX PARA REDES

Propósito: Resolver el problema de redes a costo mínimo

Descripción

PASO 0 Determine una solución básica factible inicial en los arcos, para la red. Vaya al paso 1.

PASO 1 Determine un arco (variable) entrante usando la condición de optimalidad del método simplex, si la solución es óptima deténgase, de otra forma vaya al paso 2.

PASO 2 Determine el arco (variable) saliente usando la condición de factibilidad del método simplex. Cambie la base y vaya al paso 1.

Un nodo n con flujo neto igual a cero, $f_1 + f_2 + \dots + f_n = 0$ consiste de $n-1$ restricciones independientes. Así, una solución básica asociada debe incluir $n-1$ arcos. Recuerde que una solución básica corresponde siempre a un árbol de expansión en la red. Esto significa que el conjunto de arcos que definen una solución básica no puede incluir circuitos.

El arco que entra (Paso 1) está determinado al calcular sus coeficientes objetivo $z_{ij} - c_{ij}$ para todos los arcos no básicos actuales (i,j) , si se cumple que para toda $z_{ij} - c_{ij} \leq 0$, entonces la base actual es óptima. En otro caso, seleccionamos el arco no básico con el más positivo de los $z_{ij} - c_{ij}$ para entrar a la base.

Los cálculos de $z_{ij} - c_{ij}$ están basados en el uso de la dualidad exactamente como se hace en el modelo de transporte. Usando el modelo de programación lineal de la sección 7.2 sea w_i la variable dual asociada con la restricción del nodo i , entonces el problema dual asociado con las restricciones principales (excluyendo las cotas superiores) está dado por:

$$\max z = \sum_{i=1}^n f_i w_i$$

Sujeto a

$$\begin{array}{ll} w_i - w_j \leq c_{ij} & (i,j) \in A \\ w_i \text{ irrestricta} & i = 1, 2, \dots, n \end{array}$$

de la teoría de la programación lineal tenemos

$$w_i - w_j = c_{ij} \text{ para el arco básico } (i,j)$$

ya que el programa lineal original tiene una restricción redundante por definición, podemos asignar un valor arbitrario a una de las variables duales. Por conveniencia, asumiremos que $w_1 = 0$. Podemos entonces resolver las ecuaciones básicas $w_i - w_j = c_{ij}$ para determinar los valores duales remanentes. Subsecuentemente, podemos calcular $z_{ij} - c_{ij}$ para las variables no básicas como

$$z_{ij} - c_{ij} = w_i - w_j - c_{ij}$$

El único detalle es como se determina la variable que sale.

EJEMPLO 7.3

Una red de agua conecta dos plantas desalinadoras a dos ciudades. La oferta total diaria de las dos plantas es de 40 y 50 millones de galones y las demandas diarias de las ciudades 1 y 2 son 30 y 60 millones de galones. Ambas plantas 1 y 2 tienen rutas directas a cada ciudad. El agua desalinada de las plantas se puede transportar de la planta 2 a una estación especial de bombeo. Adicionalmente, la planta 1 está unida a la planta 2 y la ciudad 1 a la ciudad 2. El modelo está balanceado porque la oferta en los nodos 1 y 2 es igual a la demanda en los nodos 4 y 5.

La figura siguiente muestra la red

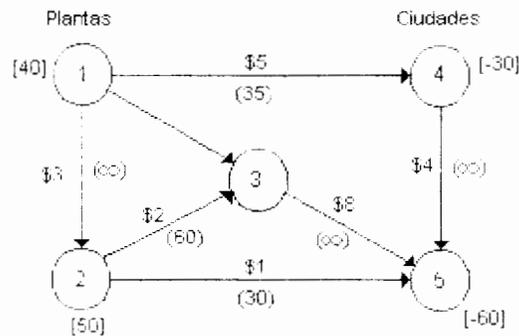


Figura 7.6

Iteración 1

PASO 0: Determinación de una solución básica factible inicial

Nuestra primera tarea es encontrar un árbol de expansión factible. El árbol en la figura siguiente, figura 7.7 se obtuvo por inspección.

Otra forma de obtenerlo es a través de encontrar las columnas linealmente independientes en la matriz de incidencia.

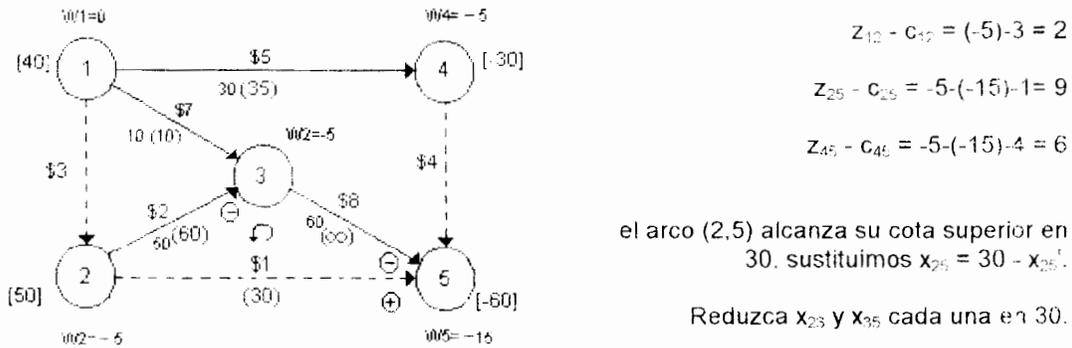


Figura 7.7

Normalmente podríamos usar la técnica de la variable artificial para encontrar tal solución. En la figura 7.7 la solución básica factible consiste de los arcos (1,3), (1,4), (2,3) y (3,5) con los flujos factibles de 30, 10, 50 y 60 unidades. Entónces los arcos punteados (1,2), (2,5) y (4,5) representan las variables no básicas. Usamos la notación $x(c)$ para indicar un flujo de x unidades que se asigna a un arco cuya capacidad es c . Los valores por default para x y c son 0 e ∞ respectivamente.

Iteración 2

PASO 1: Determinación del arco entrante

Obtenemos los valores duales resolviendo las ecuaciones básicas actuales.

$w_1 = 0$

$w_i - w_j = c_{ij}$ para (i,j) básica

entonces

arco (1,3) : $w_1 - w_3 = 7$ implica $w_3 = -7$

arco (1,4) : $w_1 - w_4 = 5$ implica $w_4 = -5$

arco (2,3) : $w_2 - w_3 = 2$ implica $w_2 = -5$

arco (3,5) : $w_3 - w_5 = 8$ implica $w_5 = -15$

Ahora $z_{ij} - c_{ij}$ para las variables no básicas es

arco (1,2) : $w_1 - w_2 - c_{12} = 0 - (-5) - 3 = 2$

arco (2,5) : $w_2 - w_5 - c_{25} = (-5) - (-15) - 1 = 9$

arco (4,5) : $w_4 - w_5 - c_{45} = (-5) - (-15) - 4 = 6$

Así, el arco (2,5) entra a la solución básica.

PASO 2. Determinación del arco que sale

De la figura 7.7, observamos que el arco (2, 5) forma un ciclo con los arcos básicos (2, 3) y (3, 5). *De acuerdo con la definición del árbol de expansión no puede formarse ningún otro ciclo.*

Como el flujo en el nuevo arco (2, 5) debe ser incrementado, necesitamos ajustar los arcos de los ciclos mediante una cantidad igual para mantener la factibilidad de la nueva solución. A fin de lograrlo, identificamos el flujo positivo (+) en el ciclo mediante la dirección del flujo del arco que entra (es decir, de 2 a 5). Después asignamos (+) ó (-) a los arcos restantes del ciclo dependiendo de si el flujo de cada arco está en o contra la dirección del flujo del arco que entra. Lo anterior se muestra en la figura 7.7

A continuación, para determinar el nivel máximo del flujo en el arco que entra (2, 5), debemos garantizar que:

1. El nuevo flujo en los arcos básicos actuales del ciclo no puede ser negativo.
2. El nuevo flujo en el arco que entra no puede exceder su capacidad.

La aplicación de la regla 1 muestra que los flujos en los arcos (2, 3) y (3, 5) no pueden disminuir en más que $\min\{50, 60\} = 50$ unidades. La regla 2 estipula que el flujo en el arco (2, 5) puede incrementarse en, a lo más, la capacidad del arco (= 30 unidades). Así, el cambio máximo del flujo en el ciclo es $\min\{30, 50\} = 30$ unidades. Los flujos en el ciclo son, de este modo, 30 unidades en el arco (2,5), $50-30=20$ unidades en el arco (2, 3) y $60-30=30$ unidades en el arco (3, 5).

Debido a que ninguno de los arcos básicos actuales deja la base al nivel cero, el nuevo arco (2, 5) debe permanecer no básico en una cota superior. Para evitar el tratar con arcos no básicos que no están en un nivel de flujo cero, llevamos a cabo la sustitución:

$$x_{25} = 30 - x_{52}, 0 \leq x_{52} \leq 30$$

Esta sustitución se efectúa en las ecuaciones de flujo asociadas con los nodos 2 y 5. Considere la ecuación de flujo actual

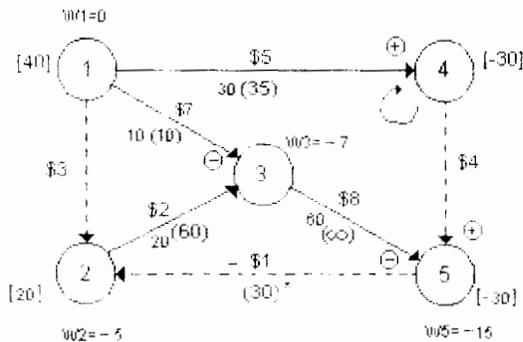
$$\text{en el nodo 2: } 50 + x_{12} = x_{23} + x_{25}$$

$$\text{en el nodo 5: } x_{25} + x_{35} + x_{45} = 60$$

Entonces, la sustitución $x_{25} = 30 - x_{52}$ nos da

nueva ecuación del flujo en el nodo 2= $20 + x_{12} + x_{52} = x_{23}$
nueva ecuación del flujo en el nodo 5= $x_{35} + x_{45} = x_{52} + 30$

Los resultados de estos cambios se muestran en la figura 7.8. Note que la dirección del flujo en el arco (2, 5) es ahora revertida a $5 \rightarrow 2$ con $x_{52} = 0$, como se deseaba. La sustitución también requiere cambiar el costo unitario del arco (5, 2) a $-\$1$. Indicaremos esta dirección inversa en la red etiquetando el arco con un asterisco.



$$z_{12} - c_{12} = 0 - (-5) - 3 = 2$$

$$z_{52} - c_{52} = -15 - (-5) - (-1) = -9$$

$$z_{45} - c_{45} = -5 - (-15) - 4 = 6$$

el arco (4,5) entra en el nivel 5, el arco (1,4) sale con su cota superior y sustituimos $x_{14} = 35 - x_{14}$.

Reduzca x_{13} y x_{35} cada una en 5.

Figura 7.8

Iteración 3

La figura 7.8 resume los nuevos valores de $z_{ij} - c_{ij}$ y muestra que el arco (4, 5) entra en la solución básica. Esto define tanto el ciclo asociado con el nuevo arco que entra como los signos de sus arcos.

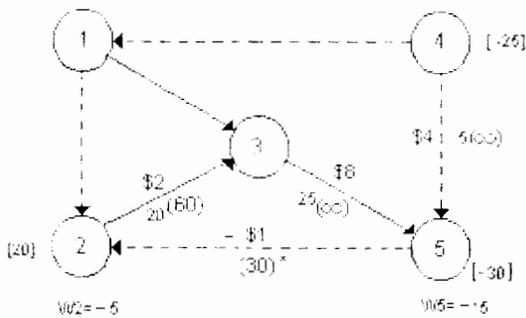
El flujo en el arco (4, 5) puede incrementarse mediante el menor de los siguientes:

1. Incremento máximo permitido en el arco que entra (4, 5)= x
2. Incremento máximo permitido en el arco (1, 4)= $35 - 30 = 5$ unidades.
3. Decremento máximo permitido en el arco (1, 3)= 10 unidades.
2. Decremento máximo permitido en el arco (3, 5)= 30 unidades.

De este modo, el flujo en el arco (4, 5) puede incrementarse a 5 unidades. Este incremento hará a (4, 5) básico y forzará al arco básico (1, 4) a ser no básico en su cota superior (= 35).

Usando la sustitución $x_{14} = 35 - x_{41}$, la red cambia como se muestra en la figura 7.9, con los arcos (1, 3), (2, 3), (3, 5), y (4, 5) formando la solución básica del árbol de expansión. Note el flujo inverso en el arco (1, 4) y el cambio de su costo

unitario a -\$5. Además, convéznase que la sustitución en las ecuaciones de flujo de los nodos 1 y 4 darán 5 unidades netas que entran a cada nodo.



$$z_{12} - c_{12} = 0 - (-5) - 3 = 2$$

$$z_{41} - c_{41} = -11 - 0 - (-5) = -6$$

$$z_{52} - c_{52} = -15 - (-5) - (-1) = -9$$

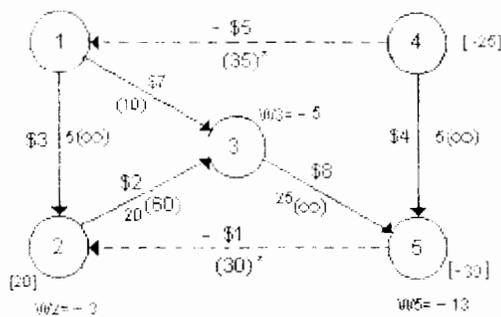
el arco (1,2) entra al nivel 5, el arco (1,3) sale al nivel 0.

Incremente x_{13} por 5

Figura 7.9

Iteración 4

El cómputo del nuevo $z_{ij} - c_{ij}$ para los arcos no básicos (1, 2), (4, 5) y (5, 2) se observa en la figura 7.9, la cual muestra que el arco (1, 2) entra a nivel 5 y el arco (1, 3) se convierte en no básico al nivel 0. La nueva solución se describe en la figura 7.10



$$z_{13} - c_{13} = 0 - (-5) - 7 = -2$$

$$z_{41} - c_{41} = -9 - 0 - (-5) = -4$$

$$z_{52} - c_{52} = -13 - (-3) - (-1) = -9$$

Solución óptima

$$x_{12} = 5, x_{13} = 0$$

$$x_{23} = 35 - 0 = 35$$

$$x_{34} = 25$$

$$x_{25} = 30 - 0 = 30$$

$$x_{35} = 25, x_{45} = 5$$

Costo total \$495

Figura 7.10

Iteración 5

El nuevo $z_{ij} - c_{ij}$ en la figura 7.10 muestra que la solución es óptima. Los valores de las variables originales son obtenidos mediante sustitución hacia atrás como se muestra en la figura 7.10

7.4 MÉTODO BASADO EN LA ELIMINACIÓN DE CIRCUITOS NEGATIVOS

El primer paso de este método consiste en determinar un flujo factible. Para ello se puede usar el algoritmo de Ford y Fulkerson con una modificación del criterio de detención del algoritmo. Esta modificación consiste en lo siguiente: si en el paso 5, la etiqueta $f(t)$ de t más el valor v' del flujo actual es mayor o igual que v , entonces se modifica el flujo en la cantidad $v-v'$ y se termina con el flujo del valor v ; si éste no es el caso se realiza otra iteración.

Observe que si se determina el flujo máximo, usando esta modificación del algoritmo de Ford y Fulkerson, y el valor de este es menor que v , entonces el problema no tiene solución. Una vez determinado el flujo de valor v se procede a "probar" la optimalidad. Probar la optimalidad consiste en verificar si en la red marginal existen circuitos negativos o no, si no existen, el flujo es de costo mínimo; si existe alguno, éste se eliminará.

Dicho en otras palabras, si existe un circuito negativo en la red marginal, se determina su capacidad incremental y se modifica el flujo a través del ciclo correspondiente, en la red original, en esta cantidad para obtener un flujo de menor costo. Puesto que el nuevo flujo a través de alguno de los arcos del ciclo es igual a su capacidad o a cero, el circuito negativo no estará en la red marginal con respecto a este nuevo flujo; es decir, el circuito fue eliminado. Este procedimiento se repite hasta que se hayan eliminado todos los circuitos negativos. En este caso el flujo de valor v actual es de costo mínimo.

Note que durante este procedimiento, es necesario determinar circuitos negativos, para determinarlos puede usarse el algoritmo de Floyd o el algoritmo general para determinar arborescencias de rutas más cortas si es que existe una raíz.

Recuerde que durante el algoritmo de Floyd se determinan las rutas más cortas entre todo par de nodos, si existen, o se determina algún circuito negativo, concluyendo en este caso, que no hay solución al problema. Es posible entonces, emplear este método para detectar los circuitos negativos en la red marginal o para garantizar la no-existencia de éstos en el caso de que el algoritmo termine con las rutas más cortas.

ALGORITMO DE FLOYD

Propósito: Determinar el flujo a costo mínimo de valor v en la red $r=[X, A, q, c]$.

Descripción

PASO 1 Determine un flujo factible f de valor v mediante el algoritmo de Ford y Fulkerson.

PASO 2 Construya la red marginal, con respecto a f , $R'(f)$.

PASO 3 Mediante algún algoritmo de rutas más cortas, identifiqúese algún circuito negativo en $R'(f)$. Si no existen circuitos negativos, terminar. El flujo actual f es el requerido; en otro caso, sea C el circuito negativo. Ir al paso 4.

PASO 4 Sea $d = \text{Min } \{q_{ij}\} \quad (i, j) \in C$

(i) Para todo arco $(i, j) \in A$ tal que $(i, j) \in A_1 \cap C$, actualizar.

$$f_{ij} = f_{ij} + d$$

(ii) Para todo arco $(i, j) \in A$ tal que $(i, j) \in A_2 \cap C$, actualizar

$$f_{ij} = f_{ij} - d$$

Con este nuevo flujo ir al paso 2.

EJEMPLO 7.4

Determine el flujo a costo mínimo de valor 5 en la red de la figura 7.11.

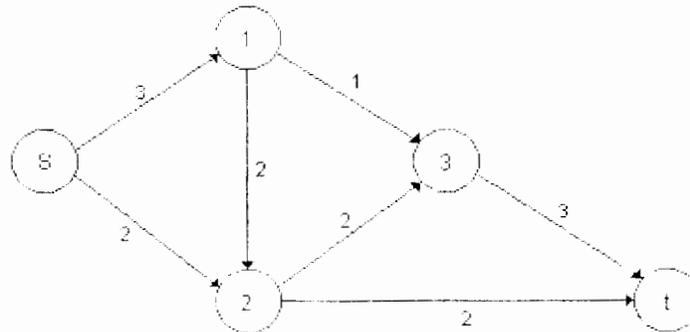


Figura 7.11

Iteración 1

PASO 1

Se aplicará el algoritmo a partir del flujo, de costo 44. Este flujo se muestra nuevamente en la figura 7.11

PASO 2

La red marginal con respecto a este flujo se muestra en la figura 7.12

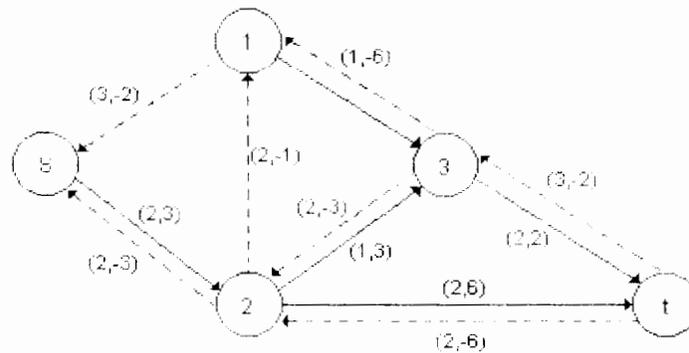


Figura 7.12

PASO 3

Utilizando el algoritmo de Floyd, se determina el circuito negativo 1, s, 2, 3, 1 de costo -2

PASO 4

De aquí se tiene $d = \min \{q'_{1s}, q'_{s2}, q'_{23}, q'_{31}\} = 1$. Actualizando el flujo se obtiene el definido en la figura 7.13

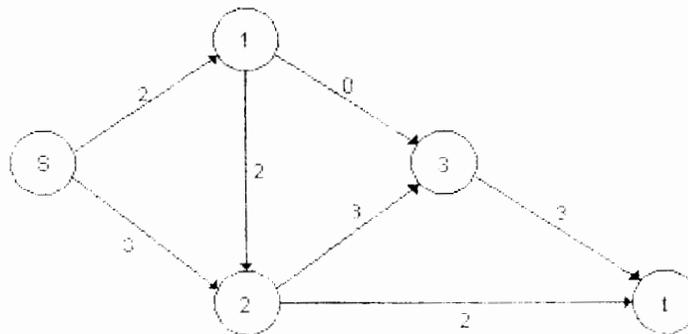
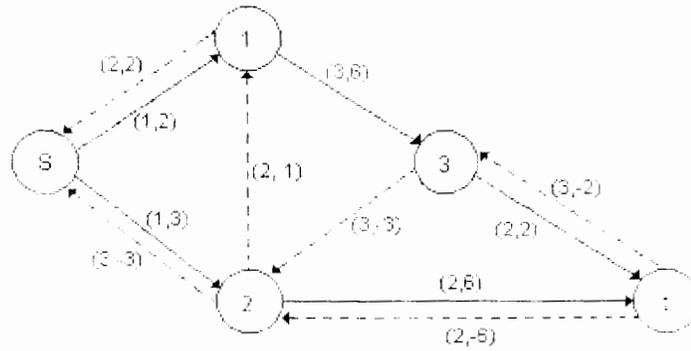


Figura 7.13

Iteración 2

La red marginal con respecto a este nuevo flujo se muestra en la figura 7.14. Al utilizar el algoritmo de Floyd se observa que la red marginal no contiene circuitos negativos, por lo que el flujo de costo 42 definido en la figura 7.14 es el flujo a costo mínimo de valor 5.

Figura 7.14



EJEMPLO 7.5

Determine el flujo a costo mínimo de valor 20 en la siguiente red mediante el algoritmo basado en eliminación de circuitos negativos.

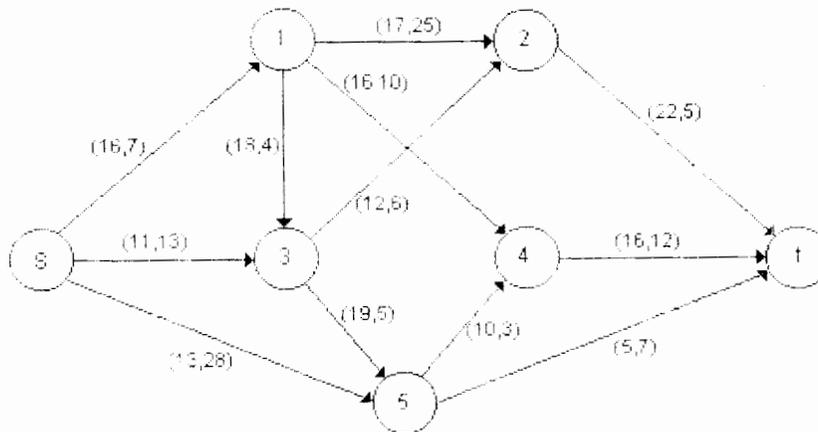


Figura 7.15

Iteración 1

Utilizando el algoritmo de Ford y Fulkerson se determina primero la cadena aumentante $s, 1, 2, t$ de capacidad incremental igual a 16 unidades y después la cadena $s, 5, t$ de capacidad incremental igual a 5 unidades por lo que a través de esta última cadena se envían sólo 4 unidades.

PASO 1

El flujo inicial resultante se muestra en la figura 7.16 (a). El costo de este flujo es 732.

PASO 2

La red marginal con respecto a este flujo se muestra en la figura 7.16 (b) en la cual se ha asociado a cada arco su capacidad y el costo unitario del flujo a través de él, respectivamente.

PASO 3

Mediante el algoritmo de Floyd se detecta el circuito negativo $3, 2, 1, 3$ de costo -15, de donde

PASO 4

$d = \text{Min} \{q'_{32}, q'_{21}, q'_{13}\} = 12$. Actualizando el flujo se obtiene el definido en la figura 7.16 © de costo $732 - 15(12) = 552$.

Iteración 2

La red marginal con respecto a este nuevo flujo se muestra en la figura 7.16 (d) En ella se ha detectado el circuito negativo $5, s, 3, 5$ de costo -10. En tonces $d = \text{min} \{q'_{5s}, q'_{s3}, q'_{35}\} = 4$. Actualizando se obtiene el flujo de la figura 7.16 (e) de costo $552 - 10(4) = 512$.

Iteración 3

La red marginal se muestra en la figura 7.16 (f) en donde se identifica el circuito negativo $t, 2, 1, 4, t$ de costo -8. Luego, $d = \text{min} \{16, 4, 16, 16\} = 4$. Actualizando se obtiene el flujo de la figura 7.16 (g) de costo $512 - 8(4) = 480$.

Iteración 4

La red marginal resultante se considera en la figura 7.16 (h). En ésta se identifica el circuito negativo $s, 3, 5, t, 4, 1, s$ de costo -4. De donde, $d = \text{min} \{7, 15, 1, 4, 4, 16\} = 1$. El flujo que se obtiene es el correspondiente a la figura 7.16 (i) de costo $480 - 4 = 476$.

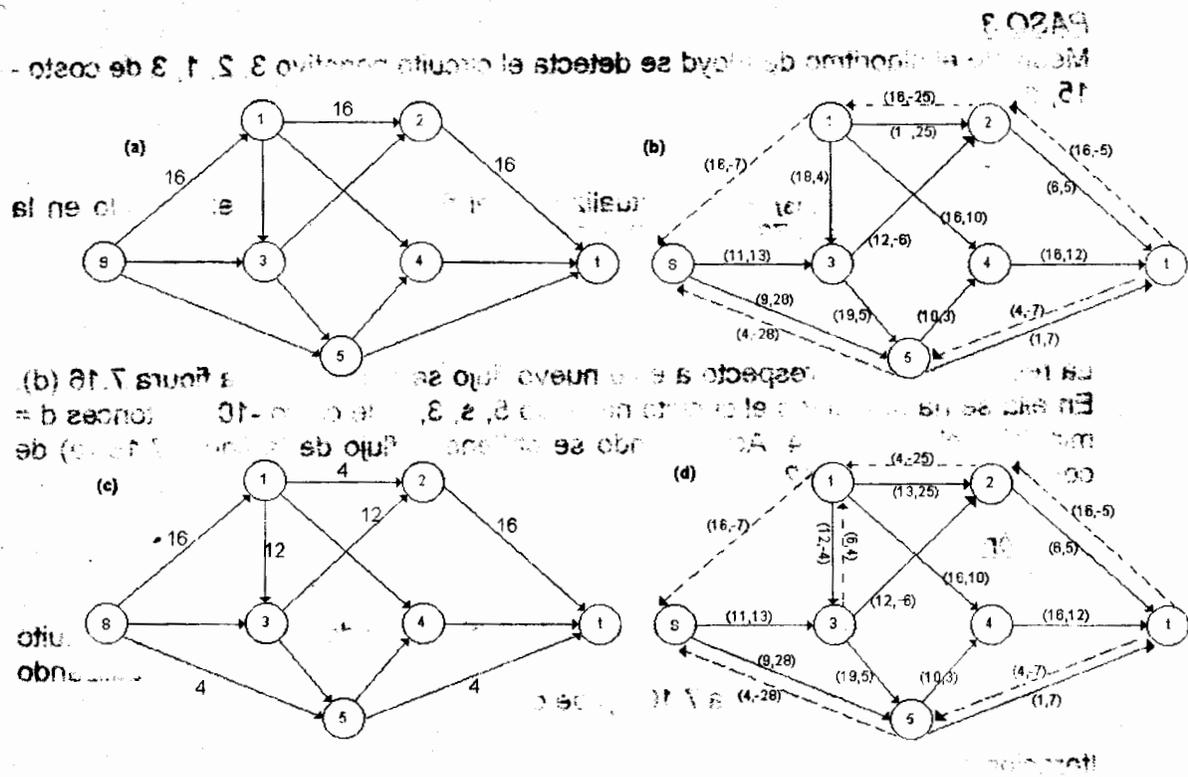
Iteración 5

En la figura 7.16 (j) se muestra la nueva red marginal. En ella se detecta un circuito negativo s, 1, 3, s de costo -2. Por tanto $d = \min \{1, 6, 5\} = 1$. El nuevo flujo, de costo $476 - 2 = 474$, se muestra en la figura 7.16 (k).

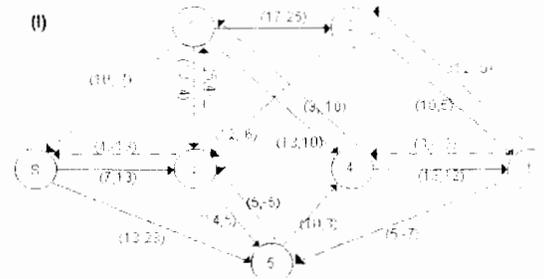
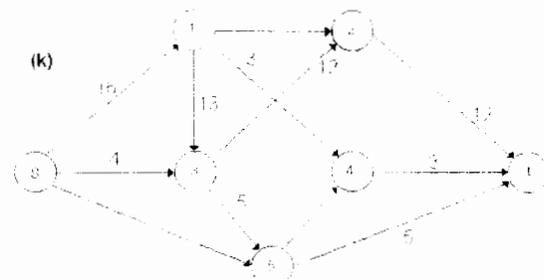
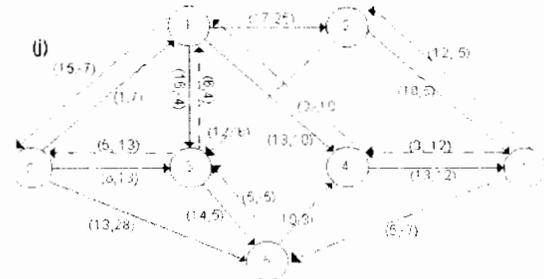
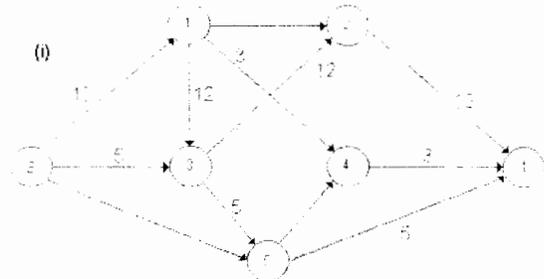
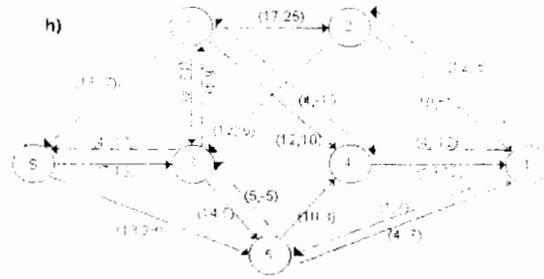
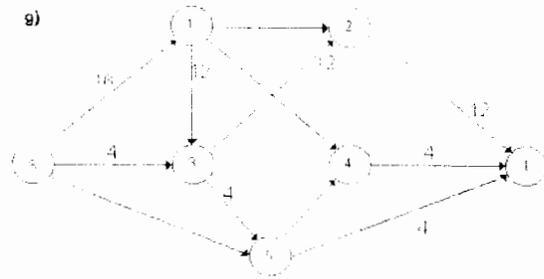
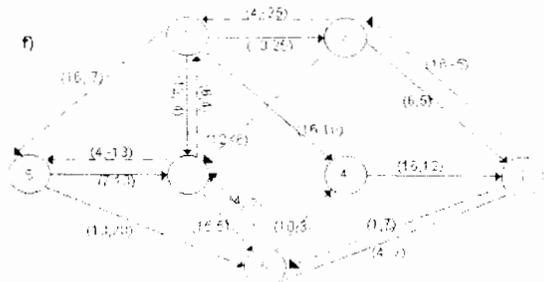
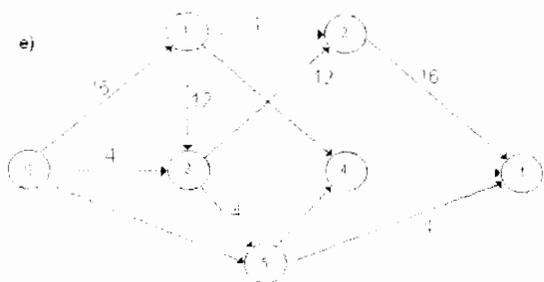
Iteración 6

La nueva red marginal, figura 7.16 (m), no contiene circuitos negativos por lo cual el flujo definido en la figura 7.16 (k) es el de costo mínimo de valor 20.

Figura 7.16



La red marginal resultante de esta iteración se muestra en la figura 7.16 (l). En ella se detecta un circuito negativo s, 1, 3, s de costo -2. De donde $d = \min \{1, 6, 5\} = 1$. El nuevo flujo que se obtiene es el correspondiente a la figura 7.16 (m) de costo $474 - 2 = 472$.



7.5 MÉTODO BASADO EN RUTAS MÁS CORTAS

Primeramente se determinará un flujo factible de valor v' , de costo mínimo, donde v' es menor que el valor v requerido. En general, se utiliza $v'=0$. Para determinar este flujo deben eliminarse todos los circuitos negativos que existan en la red marginal. El siguiente paso consiste en determinar la ruta más corta P^* , de s a t , en la red marginal; de nuevo es posible utilizar, con este propósito, el algoritmo general para determinar arborescencias de rutas más cortas presentado en el capítulo 6.

A través de la cadena correspondiente a P^* se enviará la máxima cantidad posible de s a t ; el nuevo flujo así definido será de costo mínimo por el segundo teorema presentado en el capítulo 6. Se procederá de este modo en cada iteración hasta alcanzar el valor v requerido.

Aquí hay que hacer algunas observaciones: Si en alguna iteración no existe ruta alguna de s a t en la red marginal y el valor v' del flujo actual f es menor que v entonces no existe ningún flujo del valor requerido ya que, al no existir ruta de s a t en la red marginal, no existen cadenas aumentantes de s a t en la red original y por tanto f es el flujo máximo. Por otro lado, si v' más la capacidad incremental de la cadena encontrada es mayor que v entonces sólo será necesario enviar la cantidad $v-v'$ a través de esta cadena para determinar el flujo deseado. En el algoritmo se denotará como $f+k$ o (P^*) al flujo definido en R de la siguiente manera, donde P es la cadena correspondiente a P^* :

ALGORITMO BASADO EN RUTAS MÁS CORTAS

Propósito. Determinar el flujo a costo mínimo de valor v en la red $R = [X, A, q, c]$.

Descripción

- PASO 1 Determine un flujo factible f de costo mínimo de valor 0 en R .
- PASO 2 Construya la red marginal, con respecto a f , $R'(f)$.
- PASO 3 Determine la ruta más corta P^* de s a t en $R'(f)$.
- PASO 4 Sea $d = \text{Min} \{q_{ij}\} \quad (i, j) \in P^*$
- PASO 5
- (i) Si el valor del flujo $f+d$ ó (P^*) es menor que v , actualizar $f = f+d$ ó (P^*) e ir a 2.
 - (ii) Si el valor del flujo $f+d$ ó (P^*) es igual a v , actualizar $f = f+d$ ó (P^*) y terminar. En este caso f es el flujo a costo mínimo de valor v .
 - (iii) Si el valor del flujo $f+d$ ó (P^*) es mayor que v y v' es el valor de f , actualizar $f = f + (v-v')$ ó (P^*) y terminar, ya que f es el flujo requerido.

EJEMPLO 7.6

Determine el flujo a costo mínimo de valor 20 en la red del ejemplo 2 utilizando el método basado en rutas más cortas.

Iteración 1

PASOS 1 Y 2

Si se define un flujo igual a 0 a través de todos los arcos de la red se obtiene un flujo de valor 0 de costo mínimo puesto que la red marginal, con respecto a este flujo, coincide con la original y no existen en ella circuitos negativos.

PASO 3

Utilizando el algoritmo de Dijkstra se determina la ruta más corta P^* : s, 1, 3, 2, t de costo 22.

PASO 4

En este caso $d = \min \{16, 16, 12, 22\} = 12$.

PASO 5

Entonces se actualiza el flujo $f = f+12$ ó (P^*); es decir, se aumentan 12 unidades al flujo a través de los arcos (s, 1), (1, 3), (3, 2) y (2,t).

Iteración 2

La red marginal se muestra en la figura 7.17 (a). Mediante el algoritmo general para determinar arborescencias de rutas más cortas, se obtiene la ruta P^* : s, 1, 3, 5, t, de costo 23. Luego, $d = \min \{4, 6, 19, 5\} = 4$. Se actualiza $f = f+4$ ó (P^*) definiéndose, de este modo, el flujo mostrado en la figura 7.17 (b).

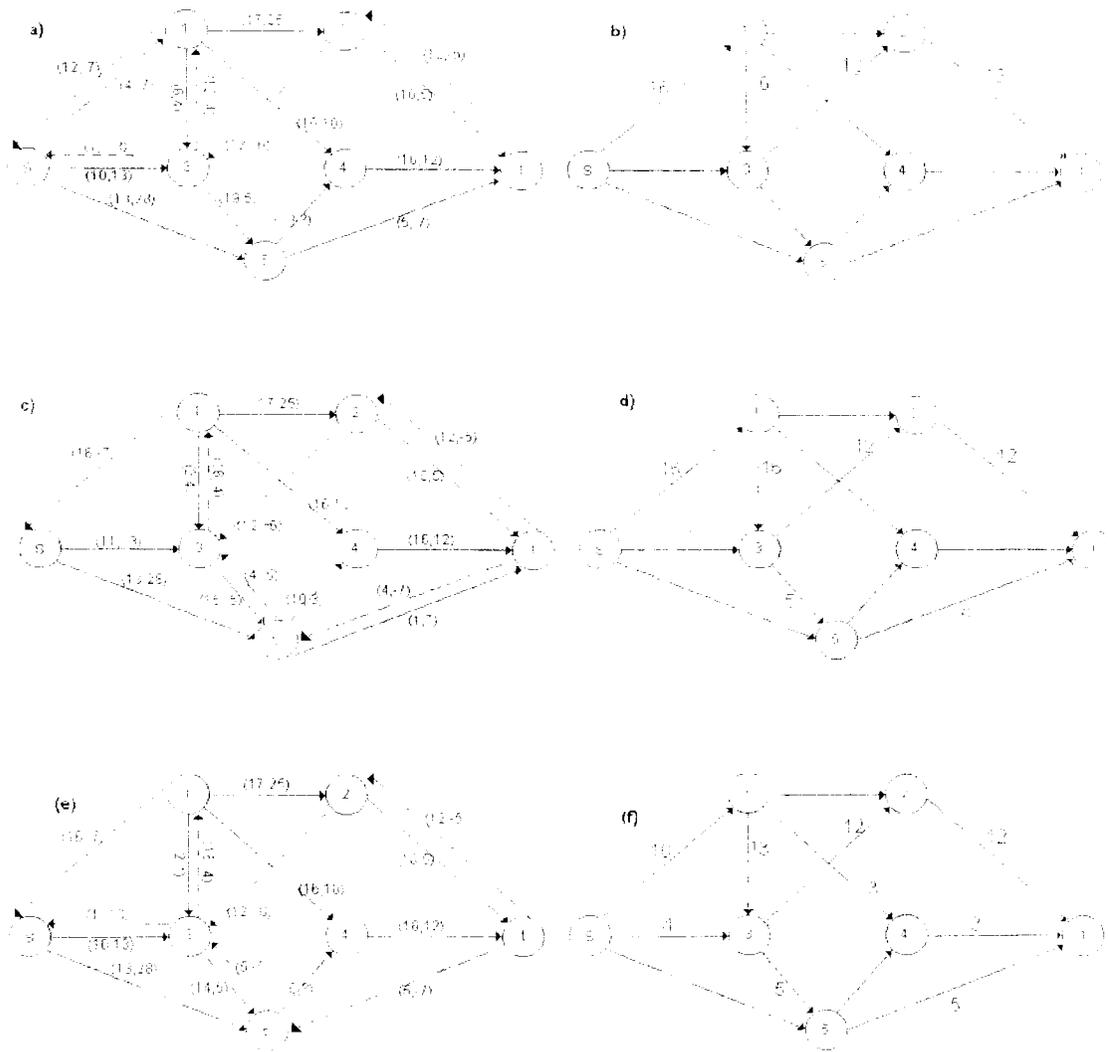
Iteración 3

La figura 7.17 (c) corresponde a la nueva red marginal. P^* resulta ser: s, 3, 5, t de costo 25, de donde $d = \min \{11, 15, 1\} = 1$. El flujo $f+1$ ó (P^*) se muestra en la figura 7.17 (d).

Iteración 4

La red marginal es la mostrada en la figura 7.17 (e). P^* es: s, 3, 1, 4, t, de costo 31. De aquí $d = \min \{10, 16, 16, 16\} = 10$. Puesto que el valor del flujo actual es $12+4+1=17$, sólo se envía la cantidad $20-17=3$ a través de esta última cadena; es decir, se actualiza $f = f+3$ ó (P^*). De este modo se obtiene el flujo a costo mínimo deseado que se muestra en la figura 7.17 (f).

Figura 7.17



EJEMPLO 7.7

Considere la siguiente red de flujo:

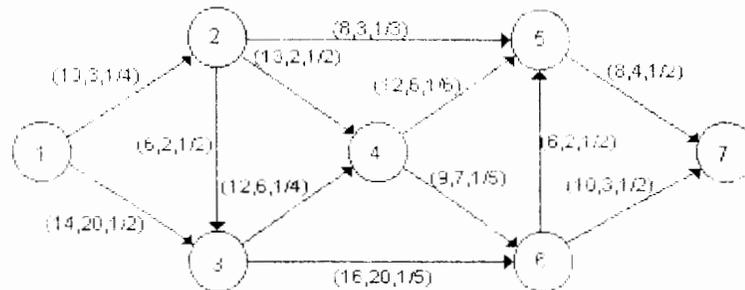


Figura 7.18

Determine el flujo máximo a costo mínimo.

Iteración 1

PASOS 1 Y 2

El flujo cero en cada arco es óptimo cuando se demanda un flujo igual a cero en el nodo sumidero denotado por el nodo 7. La red marginal es en este caso la red original y el árbol de rutas más cortas es:

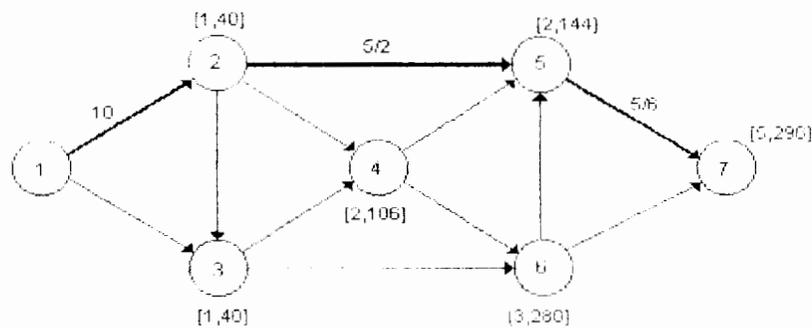


Figura 7.19

donde $[., .]$ = [nodo predecesor, valor de la ruta].

PASO 3

De dicho árbol se observa que la ruta más corta de 1 a 7 es dada por los arcos (1, 2), (2, 5) y (5, 7); y que el flujo máximo que llega al nodo sumidero es:

PASO 4

$$\begin{aligned} AF_T &= \min \{10a_{1,2}, a_{2,5} \cdot 10, 8a_{5,7} \cdot 10, 8a_{1,3} \cdot 10\} \\ &= \min \{10(1/4)(1/3)(1/2), 8(1/3)(1/2), 8(1/2)\} = 10/24 \end{aligned}$$

PASO 5

Asimismo, el flujo en cada arco es:

$$\begin{aligned} f_1 &= AF_T / a_{11} a_{51} a_{10} = 10 \\ f_5 &= AF_T / a_{55} a_{10} = 5/2 \\ f_{10} &= AF_T / a_{10} = 5/6 \end{aligned}$$

y cero en el resto de los arcos.

Iteración 2

La red marginal asociada al flujo óptimo es:

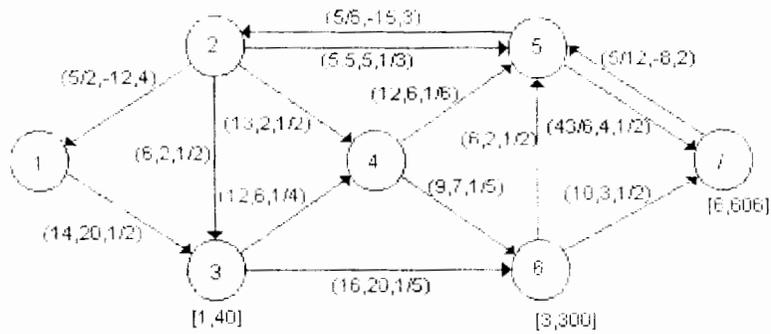


Figura 7.20

El árbol de rutas más cortas es dado por:

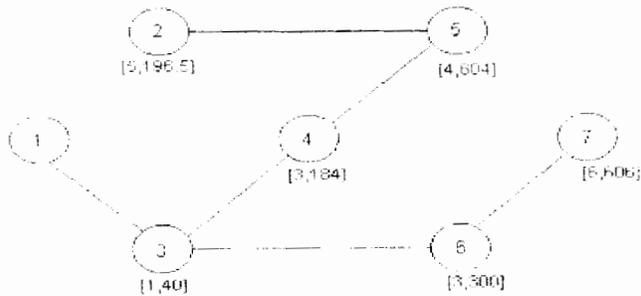


Figura 7.21

y el flujo máximo aumentante es:

$$AF_T = \min \{14(1/2)(1/5)(1/2), 16(1/5)(1/2), 10(1/2)\} = 7/10$$

El nuevo flujo en la red es:

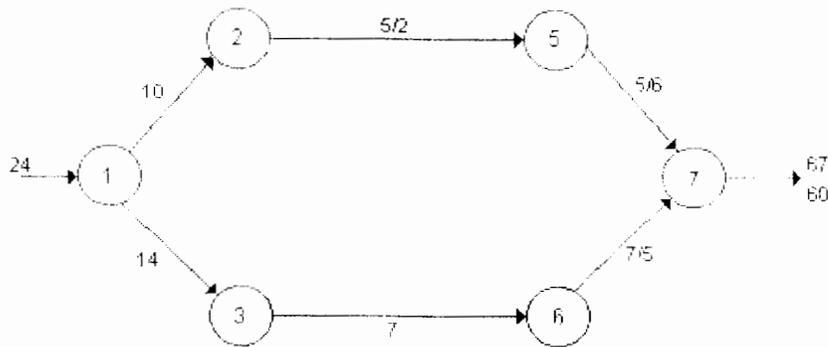


Figura 7.22

Usando la red marginal es inmediato que dicho flujo es el máximo flujo a costo mínimo (pues los arcos (1, 2) y (1, 3) están en su cota máxima).

Este algoritmo tiene una complejidad computacional de $O(nU)$ y tiene la característica de que mantiene un pseudo flujo x que satisface las condiciones de optimalidad y aumenta el flujo a través de trayectorias de ruta más corta desde los nodos fuente hasta los nodos sumidero. En cada iteración resuelve un problema de ruta más corta con costos no negativos y es muy flexible ya que, seleccionando cuidadosamente los aumentos en cada arco, se obtienen algoritmos de tiempo polinomial.

El algoritmo de cancelación de circuitos negativos tiene una complejidad computacional de $O(mCU)$, mantiene un flujo factible en cada iteración y aumenta flujo a lo largo de circuitos negativos. En cada iteración resuelve un problema de ruta más corta con costos no negativos en los arcos. Es muy flexible ya que algunas reglas para seleccionar circuitos negativos dan como resultado algoritmos de tiempo polinomial.

7.6 NOTAS HISTORICAS

Dantzig (1951) desarrolló el algoritmo simplex para redes para el problema de transporte sin restricciones de capacidad, a través de especializar su método simplex de programación lineal. Probó la propiedad de árbol de expansión de la base y la propiedad de variable entera de la solución óptima. Más tarde desarrolló una técnica de cota superior para programación lineal, lo que llevó a una especialización eficiente del método simplex para el problema de flujo a costo mínimo.

El algoritmo simplex para redes ganó su actual popularidad en los inicios de los 70's, cuando la comunidad de Investigación de Operaciones comenzó a desarrollar algoritmos de pruebas usando índices de árbol eficientes. Johnson (1966) sugirió los primeros índices de árbol, Srinivasan y Thompson (1973), Glover, Karney, Klingman y Napier (1974) implantaron estas ideas encontrando que el algoritmo simplex para redes era sustancialmente más rápido que los existentes.

Ford y Fulkerson (1957) desarrollaron los algoritmos primal-dual para el problema de transporte con restricciones de capacidad, más tarde generalizaron este método para resolver el problema de flujo a costo mínimo desarrollando el algoritmo de rutas más cortas.

El algoritmo de eliminación de circuitos negativos se atribuye a Klein (1967). Tres aplicaciones de este algoritmo corren en tiempo polinomial: la primera debida a Barahona y Tardos (1989), la segunda debida a Goldberg y Tarjan (1988), y la tercera, a Wallacher y Zimmerman (1991).

BIBLIOGRAFIA

1. Ahuja, Ravindra; Magnanti, Thomas & Orlin, James. *Network Flows: Theory, Algorithms and Applications*. Edit. Prentice Hall, 1993.
2. Fuentes Maya, Sergio. *Notas de Teoría de Redes y Programación Monotrópica*.
3. Hu, T.C. *Integer Programming and Network Flows*. Edit. Addison Wesley, 1970.
4. Jensen, Paul & Barnes, Wesley. *Network Flow Programming*. Edit. John Wiley & Sons, 1980.
5. Kennington, Jeff & Helgasn, Richard. *Algorithms for Network Programming*. Edit. John Wiley & Sons, 1980.
6. Rockafellar, R.T. *Network Flows and Monotropic Optimization* Edit. John Wiley & Sons, 1984.
7. Taha, Hamdy. *Investigación de Operaciones*. Edit. Alfa-Omega, 1995
8. Taha, Hamdy. *Apuntes de Teoría de Redes*. Universidad de Arkansas, 1996.

Esta obra se terminó de imprimir
en abril de 2001
en el taller de imprenta del
Departamento de Publicaciones
de la Facultad de Ingeniería,
Ciudad Universitaria, México, D.F.
C.P. 04510

Secretaría de Servicios Académicos

El tiraje consta de 300 ejemplares
más sobrantes de reposición.

APUNTE
198-B

FACULTAD DE INGENIERIA UNAM.



612685

1999
G.- 612685

4 - 612685
198-B