Universidad de Sonora
Departamento de Matemáticas

Tesis

El Problema de Flujo Constante a Costo Mínimo en Redes

Para obtener el título de

Licenciado en Matemáticas

Presenta

Myriam Cisneros Molina

Hermosillo, Sonora, 1 de Octubre de 1993

Universidad de Sonora

Repositorio Institucional UNISON

A mi Padre que amo, admiro
y le debo todo lo que soy y más.

A mis hermanos José Luis, Karina y Marcos
que siempre han estado y estarán
en mi corazón.

A la memoria de una magnífica persona
a quien le aprendí disfrutar la vida
Daniel.

A Antonio y Marcela a quienes quiero demasiado
por toda la ayuda y cariño que me brindan.

A toda la Familia de Astronomía quienes
enriquecieron y cambiaron mi vida durante
los años juntos.

Agradezco infinitamente a Pedro Flores P.
por ser una gran persona e impulsarme a
realizar éste trabajo, sabiéndo siempre
que haría tesis con él.

Agradezco a todos los profesores que intervinieron
en mi formación como Matemática.

Agradezco a la vida por permitirme tener
una satisfacción mas.

Indice

Dentro de los factores que influyeron para la realización del proyecto PAREIMM están la poca difusión de los temas de Redes en la Universidad de Sonora, y a la carencia de Software de este tipo. Lo que imposibilita que se puedan resolver numerosos problemas reales, que se plantean en el área.

En la primera parte del trabajo se analizan y plantean problemas hasta llegar a obtener la red que los modela; el resolver dichos problemas es equivalente a encontrar un flujo de valor constante con costo mínimo. En el capítulo siguiente se establece toda la teoría completa necesaria para la demostración del teorema que garantiza la convergencia de cada uno de los dos algoritmos que resuelven el problema y que se implementaron. Los capítulos tres y cuatro, como ya se comentó establecen cada uno el teorema central para cada algoritmo y se presenta la demostración detallada de él. Además se dá un panorama del tipo de estructuras y el manejo de la información que se utilizó en la implementación. También se presenta una corrida de escritorio con gráficas y con las estructuras de datos correspondientes para efectos de comparación.

Al final del trabajo se presentan tres anexos importantes; el primero contiene los conceptos básicos de teoría de Gráficas que se requieren para entender el presente trabajo y el segundo contiene un pequeño manual del usuario, donde se explica de una manera sencilla como se opera cada uno de los programas y como se debe leer la solución al problema resuelto, también contiene éste anexo el código de los programas. El último de los anexos contiene los códigos de los programas implementados.

Una importante observación es que en la realidad una solución de estos algoritmos puede no resolver completamente un problema real, debido a que las condiciones reales son más complejas que las implementadas en la teoría y generalmente son afectados por otros factores difíciles de modelar completamente, pero las soluciónes teóricas pueden dar un acercamiento a la solución real del problema.

Capítulo 1

Planteamiento del Problema y Ejemplos

En este Capítulo veremos en que consiste el problema de flujo constante a costo mínimo, para esto partiremos del problema ya conocido de flujo máximo, recordémoslo planteando el siguiente ejemplo: Supongamos que acabamos de comprar un rancho y que lo utilizaremos para el cultivo de maíz, el abastecimiento de agua para nuestro rancho es por medio de un río pero se encuentra un poco alejado de nosotros, hay diversas tuberias desde el río que llegan a nuestro rancho que pasan por rancherías vecinas. En cada ranchería hay una compuerta para el paso de agua de ahí hacia nuestro rancho y es manipulada por cada dueño. Las tuberías son de diferentes tamaños por lo que tenemos una cantidad máxima de agua que puede pasar por cada una de ellas; nosotros queremos saber cuales tuberías debemos usar de tal manera que nos suministren la mayor cantidad de agua posible para establecer las hectáreas de maíz que sembraremos.

Como sabemos, el algoritmo de Ford y Fulkerson (de flujo máximo) resuelve este problema y despeja nuestra incógnita.

¿Que pasaría si por la grave situación económica que se vive en el país los dueños de las rancherías deciden cobrar cierta cantidad de dinero por mantener el grifo abierto?. Los costos que éstos pongan dependerá del criterio y necesidades de cada dueño,ahora nuestro problema se complica un poco más, ya que no solo hay que encontrar la cantidad máxima de agua que recibiremos, sino también nos interesa que sea lo más barato posible. Esta nueva situación que acaba de surgir se denomina problema de flujo máximo-costo mínimo. El problema en sí es mas general, ya que no solo se puede plantear para el valor v máximo de flujo, sino para cualquier otro valor $v\prime$ de flujo, con la condición que $v\prime < v$, para que haya factibilidad.

Aquí también se expondrán algunos ejemplos y sus debidos planteamientos hasta llegar a diseñar una red que lo represente; dichos ejemplos nos ayudarán a comprendel mejor el problema.

Un comentario importante es que el Método Símplex para redes es un caso particular de este problema, ya que al adecuar la red para aplicar el Símplex se pone un nodo auxiliar con conexiones hacia todos los nodos, arcos de salida si los nodos son de demanda, de llegada si son de productores

y en los nodos intermedios puede ser de llegada o salida ya que el costo que se les asocia es cero; con este arreglo nos queda una red con flujo constante de valor la demanda total, o bien la producción total que son iguales.

Mas adelante veremos cómo la información en la red nos es muy útil para diseñar dos algoritmos que resuelven el problema, el primero basado en localizar circuitos de costo negativo, y el segundo encontando rutas mas cortas, por lo tanto los algoritmos de Ruta más corta de Floyd y el de Dijkstra Generalizado serán de mucha ayuda para la implementación.

1.1 Planteamiento del Problema de Flujo Constante a Costo Mínimo

Establezcamos formalmente el modelo del problema de flujo Constante a Costo Mínimo, para esto veamos algunas definiciones que se utilizarán muy a menudo.

Siempre que haya referencia a una red $R = [X, A, k, q, c]$ tendremos que:

- X es el conjunto de nodos que componen la red.
- A es el conjunto de arcos de la red.
- k es la función de capacidad mínima de los arcos.
- q es la función de capacidad máxima de los arcos.
- c es la función de costos unitarios del flujo a través de los arcos.

Al considerar un flujo sobre una red, supondremos que hablamos de un flujo factible, esto es, que cumple con las restricciones de capacidad mínima, capacidad máxima y cumple con las leyes de Kirchhoff.

Sabemos que dada una red R se puede encontrar el flujo máximo que puede circular por esta, mediante el algoritmo Ford-Fulkerson. Mas adelante veremos que es fácil modificar el algoritmo para obtener flujos con cualquier valor menor que el máximo. Ahora si asociamos costo a cada unidad de flujo en los arcos, el problema de flujo constante a costo mínimo consiste en, dado un valor v de flujo en una red $R = [X, A, k, q, c]$, encontrar la distribución del flujo que sea de costo menor.

Definiremos el costo de un flujo factible f a la cantidad $\sum_{(i,j)\in A} c_{ij} f_{ij}$, donde c_{ij} es el costo unitario del flujo a través del arco (i, j), y f_{ij} es la cantidad de flujo a través de (i, j). Esto nos va a servir para calcular los costos de los flujos en cada iteración de los algoritmos.

1.2 Ejemplos de Problemas y su Planteamiento

A continuación se presentan algunos ejemplos [1] estudiados hasta plantear una red que los represente, para poder aplicar el algoritmo y encontrar el costo mínimo de un flujo de valor establecido; el algoritmo se presentará en el siguiente capítulo.

1.2.1 Distribución de Producción

Una planta manufacturadora posee tres máquinas especificadas con 1, 2 y 3 para elaborar su producto y tres almacenes marcados con los números 4, 5 y 6, donde los guarda. Para colocar los m productos en los almacenes tiene un sistema de bandas de transporte distribuidos como se indican en la figura. Cada banda puede transportar a lo mas 50 unidades por hora y la descarga en los almacenes es 60 unidades por hora. Cada máquina tiene una producción de P_1, P_2 y P_3, respectivamente y se pretende hacer la distribución de los productos mediante las bandas de tal manera que se minimice la distancia que recorrerán dichos productos. Las distancias se indican en la figura.

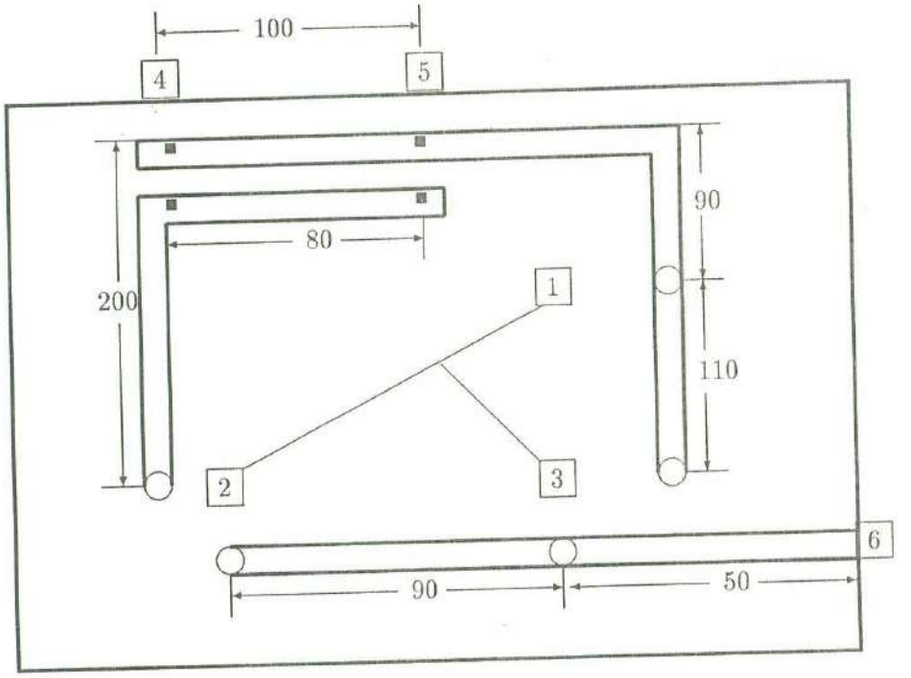


FIG. 1.1 Distribución de las máquinas y almacenes de descarga

El problema se puede plantear mediante flujo en redes. A cada punto donde están situadas las máquinas que elaboran los productos asignémosle un nodo; así también a los almacenes de descarga. Cada arco que pongamos en la red representará el recorrido que puede hacer un producto, ya sea

[1] Todos los ejemplos son modificaciones de los encontrados en [4]

de una máquina a otra o a un centro de descarga. A cada arco le pondremos la información de la capacidad mínima y máxima de productos que pueden pasar por las bandas, en el caso de los almacenes, las capacidades de almacenamiento. Debido a que la producción de cada máquina es una cantidad fija, agregaremos un nodo mas por cada uno de los ya establecidos para poder representar ese flujo fijo de productos. Entonces el ejemplo lo podemos plantear como un problema de flujo constante (que en este caso es la suma de las producciones de las máquinas) a costo mínimo, ya que el costo lo podemos asociar a la distancia que deberán recorrer los productos, y queremos que sea lo menos posible.

Los nodos $1, 2$ y 3 representan a las máquinas; $1', 2'$ y $3'$ son nodos auxiliares para representar las restricciones en los nodos $1, 2$ y 3 ya que se tiene un flujo fijo que es la producción que tiene cada máquina.

Los nodos $4, 5$ y 6 representan los almacenes y los nodos auxiliares $4', 5'$ y $6'$ al igual que en las máquinas sirven para la restricción de desecho fijo de productos en el almacén.

Los datos de los arcos están compuestos por:

(Cota Inferior del flujo, Cota Superior del flujo, Costo)

donde:

- El flujo es las unidades por hora que se transportan
- El Costo es la distancia que recorren los productos

La red que modela este problema queda de la siguiente manera:

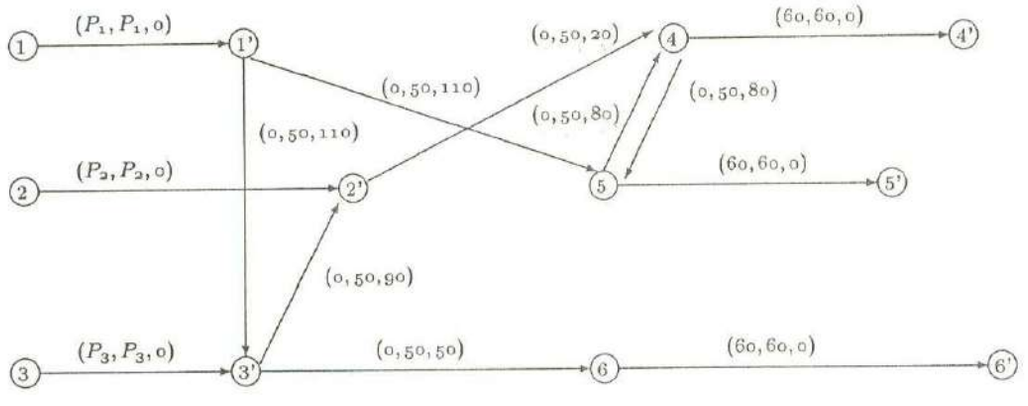


FIG. 1.2 Red que modela el problema de Distribución de la Producción

1.2.2 Fábrica de Herramientas

Una tienda recibe un pedido para elaborar 6 tipos de herramientas. La compañía posee solo dos máquinas para la elaboración de las herramientas con la restricción de que la herramienta 4 no puede ser hecha por la máquina 2. El problema a resolver es minimizar el tiempo para hacer dichos trabajos pero a fin de balancear la producción se requiere que cada máquina haga tres herramientas.

Los tiempos requeridos se presentan en la siguiente tabla.

	Herramientas					
Máquina	1	2	3	4	5	6
1	3	7	9	2	9	4
2	7	12	10	M	6	6

El problema se enfoca en minimizar el tiempo total requerido por las 2 máquinas para hacer las herramientas.

La letra M asignada como tiempo a la heramienta 4 en la máquina 2 representa una cantidad muy grande, que significa asociarle un costo alto para garantizar, que al momento de encontrar la solución no se asigne esa herramienta a esa máquina, ya que la condición del problema así lo establecía. Tomando en cuenta que, si una herramienta no se realiza en primer lugar, el tiempo requerido para ella es el tiempo de espera mas lo propio en hacerse; es decir si C_1, C_2 y C_3 son los costos de las herramientas hechas en los lugares 1, 2 y 3 respectivamente por la máquina 1 el tiempo total será:

$$C_1 + (C_1 + C_2) + (C_1 + C_2 + C_3) = 3C_1 + 2C_2 + C_3$$

Si los nodos 1 al 6 representan los seis tipos de herramientas que se pueden hacer; $1'$ al $6'$ son nodos auxiliares para garantizar flujo constante y los nodos ij representan la posición i de la máquina j y los costos de los arcos es el costo de hacer la herramienta k en el lugar i por la máquina j, tenemos la siguiente relación para los costos de los arcos.

$$C_{kij} = (3 - i + 1)C_{kj}$$

Donde C_{kij} es el costo de hacer la herramienta k en el lugar i en la máquina j y C_{kj} es el costo dado de hacer la herramienta k en la máquina j.

Los costos se pueden ver en la tabla sig.

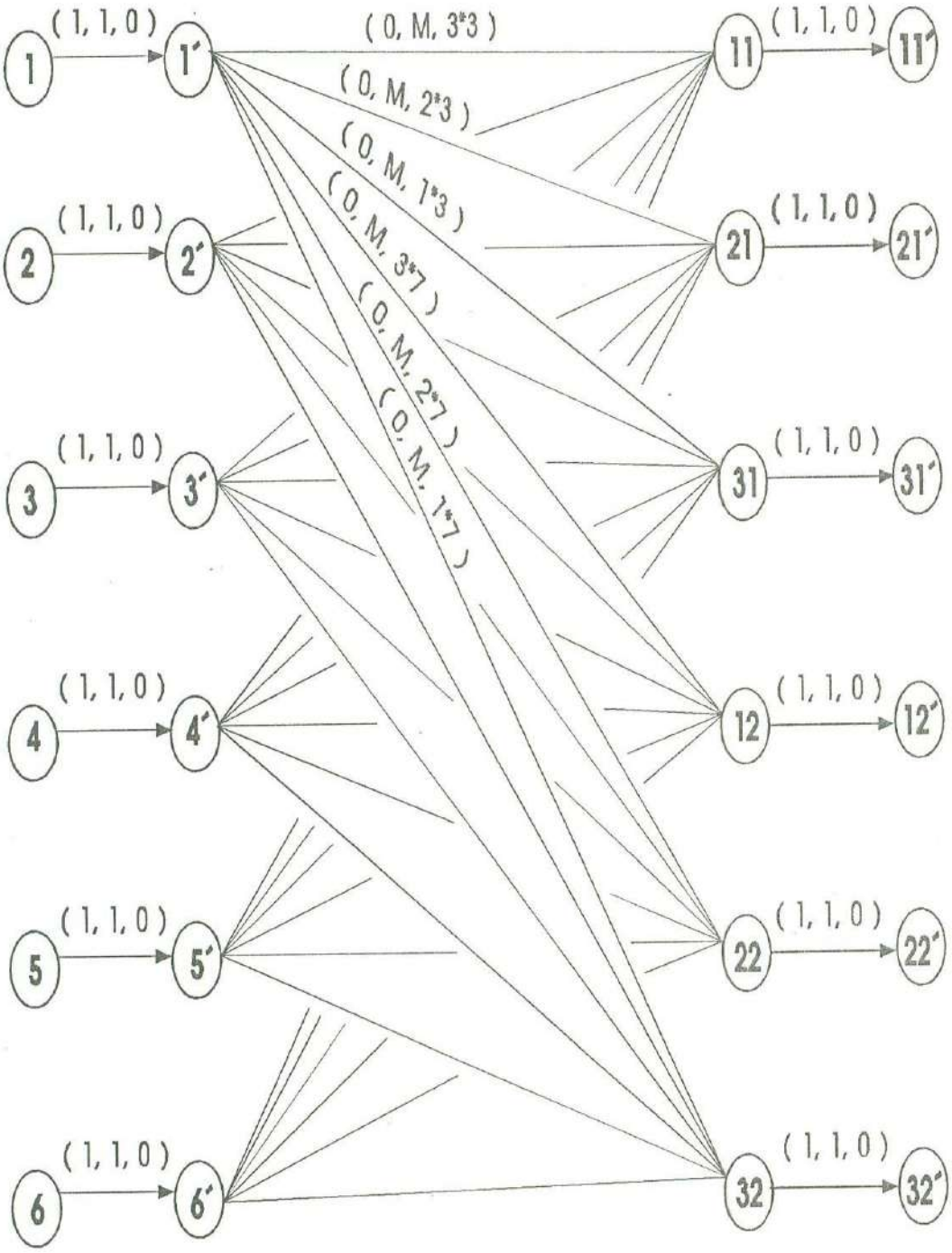


FIG. 1.3. Modelo de red para el problema de la Fábrica de Herramientas

1.2.3 La Cantina de Emilio

G. Emilio Quintana tiene y trabaja una cantina que se encuentra a un lado de la Universidad, por esta razón, la mayoría de su clientela son estudiantes universitarios. Debido a los problemas que se empezaron a suscitar a raiz de que los estudiantes salían de la cantina muy bebidos, y para evitar que le clausuraran su Cantina, Emilio tuvo que llegar a un convenio con el Departamento de Seguridad de la Universidad, el acuerdo fue que no se lo clausurarían mientras no permitiera que salieran muy borrachos.

Emilio está en un dilema, ya que necesita tener su cantina en pie para poder subsistir económicamente, pero también requiere que estén contentos los encargados de seguridad.

Una situación típica de cualquier noche en el mesón es:

Cuatro clientes sedientos están sentando en la barra con 750ml de Tequila para dividirlo entre ellos. (ninguno abandonará la Cantina mientras haya licor). Como Emilio está sirviendo, él controla la cantidad de Tequila que cada cliente recibirá. Si son buenos clientes atiguos, sabe que cada uno deberá recibir al menos 70ml del licor, pero como el cliente 1 pagó la botella se puede quejar si recibe menos de 300ml y el cliente 3 tiene problemas amorosos por lo que no debe beber más de 100ml.

La probabilidad de que el cliente i no se emborrache después de haber bebido xi mililitros de Tequila es:

$$P_i = e^{-\gamma_i x_i}$$

donde

$$\gamma_1 = .01 \quad \gamma_2 = .05 \quad \gamma_3 = .10 \quad y \quad \gamma_4 = .01$$

¿Cuántos mililitros de Tequila debería Emilio dar a cada cliente para maximizar la probabilidad de que los cuatro clientes permanezcan sobrios?

Claramente la probabilidad de que estén sobrios los cuatro clientes es $P = P_1 * P_2 * P_3 * P_4$ y queremos maximizar el valor de P.

Esta función objetivo no lineal no es apropiadas para técnicas de programación de flujo en redes. Sin embargo, recordando que un producto de variables pueden estar expresado como la suma de los logaritmos de las variables, eso hará a nuestra función objetivo apropiada para solución por técnicas de redes.

$$\ln P = \ln P_1 + \ln P_2 + \ln P_3 + \ln P_4$$

Entonces

$$\ln P = -.01x_1 - .05x_2 - .10x_3 - .01x_4$$

Maximizando P es equivalente a maximizar $\ln P$, lo cual es igual a minimizar $-\ln P$.

Por lo tanto si Emilio proporciona a sus cuatro clientes la cantidad de licor tal que la suma

$$0.01x_1 + 0.05x_2 + 0.10x_3 + 0.01x_4$$

se minimice, se maximizará la probabilidad de que estén sobrios.

Los nodos A, A', B y B' son nodos auxiliares para permitir flujo constante que representa la cantidad de licor (750 ml.) disponibles.

Los nodos 1, 2, 3 y 4 representan los cuatro clientes que Emilio tiene en una noche.

La información de los arcos es la siguiente:

(Cota Inferior del flujo, Cota Superior de flujo, Costo)

donde:

- El flujo representa los mililitros de licor que se van a proporcionar a cada cliente.
- El costo representa el coeficiente γ_i para el cliente i en la distribución de la probabilidad

La figura ilustra la red que Emilio utilizó para llegar a la distribución óptima de Tequila.

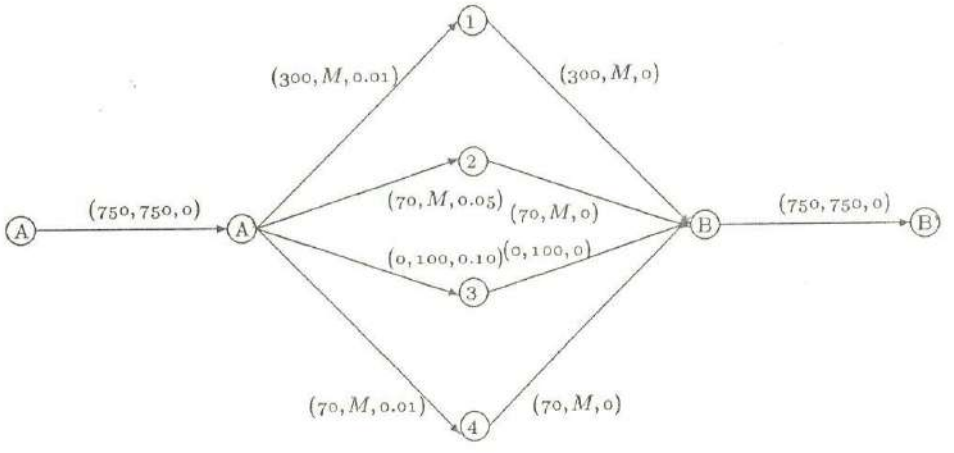


FIG. 1.4 Red modeladora del Problema de la Cantina de Emilio

1.2.4 Racismo en una Escuela del Distrito de Smallville

En el distrito de Smallville hay problemas por cuestiones raciales que han provocado cada vez mas disturbios. En un intento por mejorar la situación, los encargados de repartir al estudiantado del Distrito entre las escuelas, han decidido que no habrá privilegios de inscripción para nadie, y no solo eso, sino que han decidido que la repartición de estudiantes sea aproximadamente del 50% de blancos y 50% de latinos en cada escuela para que el reparto sea equitativo, ya que algunas escuelas tienen mala fama por su ubicación y son poco concurridas.

El Distrito se divide en cinco subdistritos, que deberán hacer uso de las tres únicas preparatorias. Las reglas adicionales son:

- Todos los estudiantes deberán ir a la escuela
- Ninguna escuela puede tener mas estudiantes que la capacidad indicada

Las tablas siguientes contienen información importante para este dramático y penoso asunto. La información de la distancia de cada subdistrito a cada escuela preparatoria está dada para que el estudiantado camine lo menos posible. Para facilitar un poco las cosas, la distribución de estudiantes podrá ir de %40–%60 a %60–%40 en cada escuela.

Subdistrito	Número de blancos	Número de Latinos	Total
1	35	15	50
2	25	10	35
3	90	5	95
4	10	115	125
5	15	30	45
Total	175	175	350

Escuela preparatoria	Capacidad
A	170
B	80
C	100

Distancia a las escuelas en Km			
Subdistrito	Prepa A	Prepa B	Prepa C
1	3	7	9
2	4	4	12
3	6	4	10
4	7	5	4
5	4	7	3

En suma, se trata de distrubuir mas o menos mitad de blancos y mitad de latinos en cada escuela, procurando que se alejen lo menos posible de sus casas.

Los nodos iL representan a la distribución de Latinos en el distrito i, de igual manera los nodos iB son la distribución de Blancos en el distrito i; los nodos iL' y iB' son nodos auxiliares para tener un flujo constante de personas sobre cada distrito.

Los nodos AL y AB representan la distribución de latinos y blancos para la escuela A, respectivamente; BL y BB, así , también la distribución de latinos y blancos pero ahora para la escuela B, análogamente los nodos CL y CB lo representan para la escuela C. Los Nodos A, B y C representan a las tres diferentes escuelas en el distrito.

La información de los arcos es la siguiente:

(Cota Inferior de flujo, Cota Superior de flujo, Costo)

donde

- El flujo representa la cantidad de personas que se distribuyen
- El costo es la distancia que los estudiantes tienen que recorrer para llegar a la escuela.

Una red que modela este problema es la siguiente:

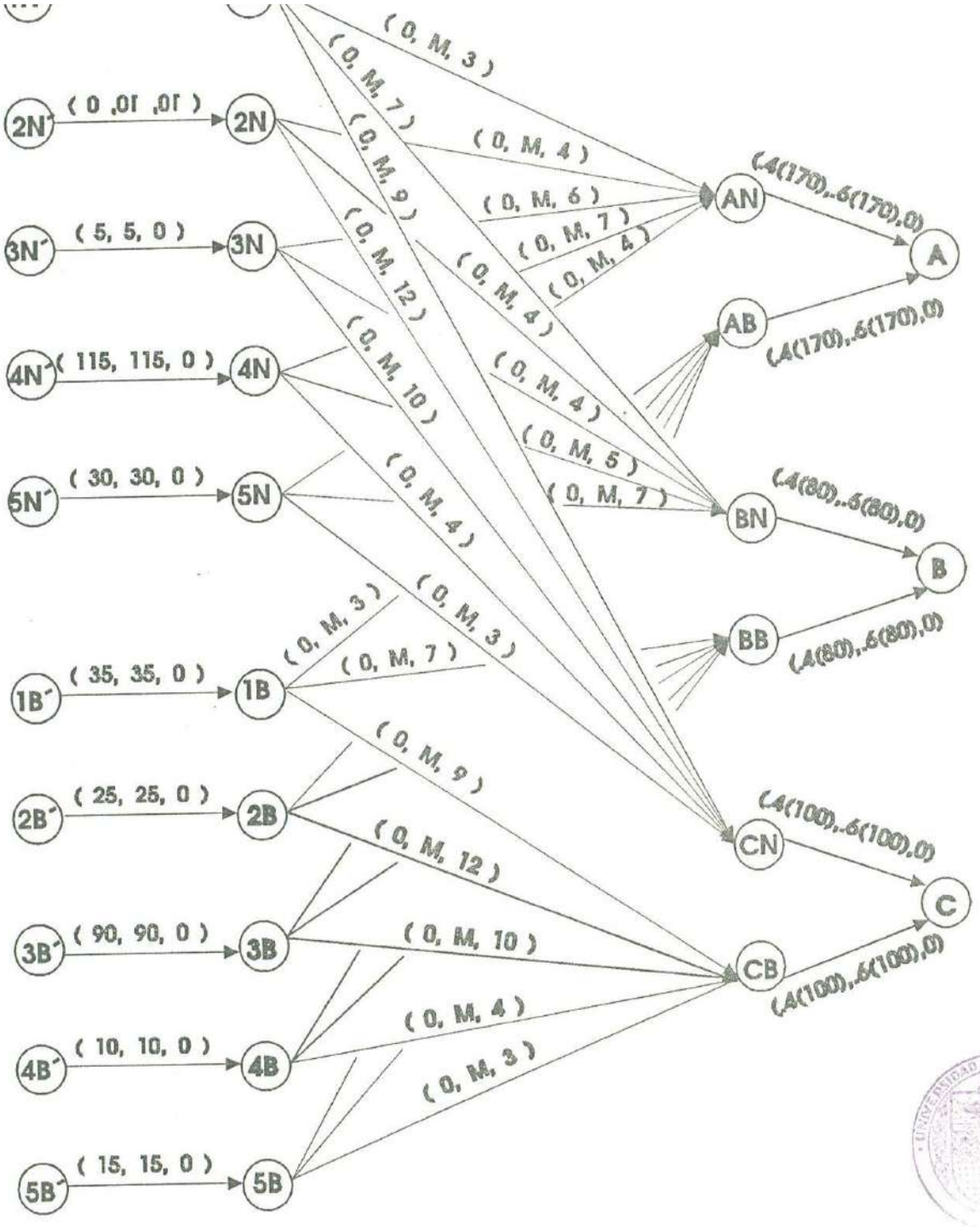


FIG. 1.5 Red que representa el Problema del Racismo en el Distrito de Smallville.

1.2.5 Un Gran Trabajo

A una fábrica de cierta gran compañía se le ha asignado una labor excepcional, de la más alta prioridad, dividida en cinco partes. Esta fábrica posee siete máquinas casi del mismo tipo, y solo se necesitan cinco para efectuar tal labor.

La siguiente tabla indica el tiempo (en minutos) que se tardaría cada máquina en cambiar de su labor actual a cada una de las 5 partes del trabajo.

El problema aquí es asignar una máquina para cada parte del trabajo, de tal manera que el tiempo de cambio sea mínimo.

	Máquinas						
Parte	1	2	3	4	5	6	7
1	22	28	34	41	45	22	52
2	19	27	45	52	44	28	21
3	52	45	37	28	21	17	22
4	37	55	40	45	55	39	31
5	56	28	33	54	25	31	56

Los nodos de 1 a 7 representan las siete máquinas con las que cuenta la compañía y los nodos $1'$ al $7'$ son auxiliares para limitar la capacidad de flujo de cada parte; los nodos 11 al 15 representan las cinco partes en que se divide el trabajo, así los nodos $11'$ a $15'$ son también nodos auxiliares para garantizar el flujo constante en la red.

La información de los arcos es la siguiente:

(Cota Inferior del flujo, Cota Superior del flujo, Costo)

donde

- El flujo es las actividades en que se descompone el trabajo
- El Costo es el tiempo en que tarda cada máquina en cambiar su labor.

Una red que modela este problema pudiera ser la siguiente:

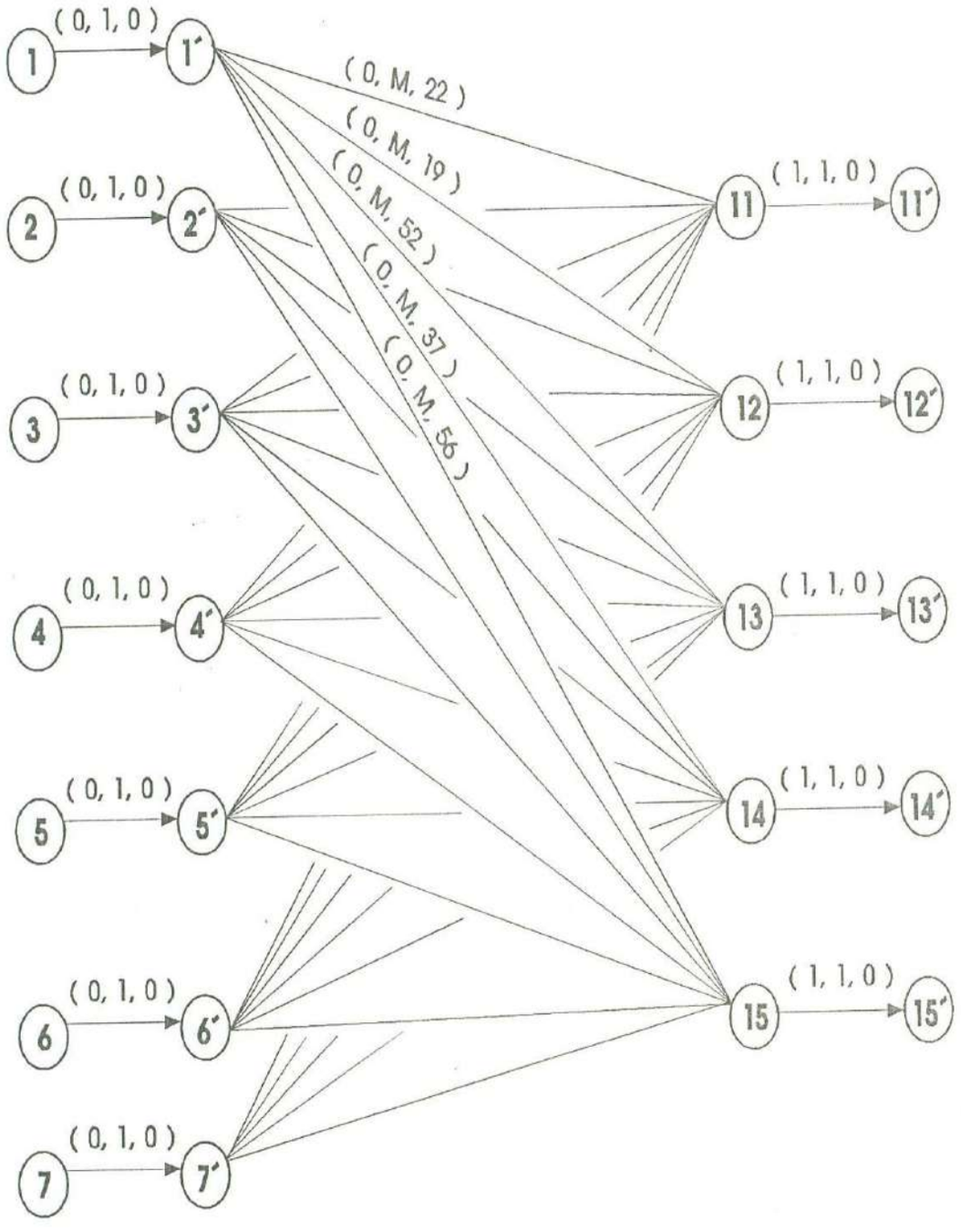


FIG. 1.6 Red que modela el problema de Un Gran Trabajo

1.2.6 Eliminación de Productos Químicos

Para eliminar 210 litros de líquido altamente corrosivo, la Compañía Química de Noroeste tiene disponible solamente cuatro contenedores de 50 litros cada una con una resistencia a la corrosión diferente.

La probabilidad de que se rompa el recipiente i al agregarle la sustancia es:

$$P_i = 1 - e^{-r_i v_i}$$

donde r_i es el coeficiente de resistencia y v_i es el volumen del líquido corrosivo en litros.

Formulando un modelo de flujo en redes que pueda ser utilizado para determinar la cantidad de líquido a poner en cada recipiente tal que la probabilidad de que ninguno de los contenedores fallen sea máxima, vemos que el problema es un flujo a costo mínimo, donde el flujo es el volúmen de líquido (en litros) de cada recipiente y el costo el coeficiente de corrosión.

Si P_i es la probabilidad de no efectividad de cada contenedor i, entonces el problema se traduce en

$$\text{minimizar } P = P_1 * P_2 * P_3 * P_4$$

que significa minimizar el mayor desastre, o bien para tener una función lineal tenemos que

$$\begin{aligned} \text{minimizar } \ln P &= \ln P_1 + \ln P_2 + lnP_3 + \ln P_4 \\ \text{minimizar } \ln P &= -r_1 v_1 - r_2 v_2 - r_3 v_3 - r_4 v_4 \\ \text{maximizar } -\ln P &= r_1 v_1 + r_2 v_2 + r_3 v_3 + r_4 v_4 \end{aligned}$$

Los nodos$A$ y A' son auxiliares para garantizar el flujo fijo de la cantidad de líquido por distribuir. Los nodos 1 al 4 representan a los tipos diferentes de contenedores.

La información en los arcos es la siguiente:

(Cota Inferior del flujo, Cota Superior del flujo, Costo)

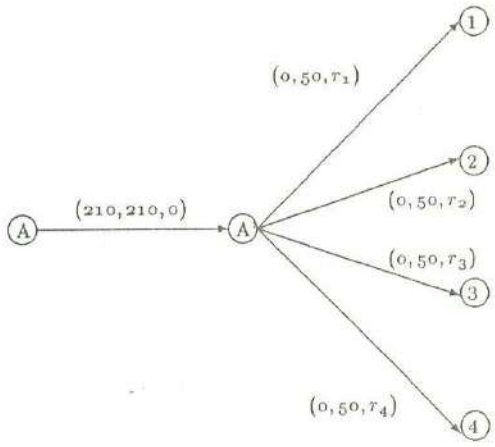


FIG. 1.7 Red que modela el problema de la Compañía de Productos Químicos

1.2.7 Enfriamiento de un Río en Sonora

En el estado de Sonora se encuentra un río que sirve como descarga de desechos de cuatro Industrias designadas por las letras A, B, C y D según su localización a partir de la fuente del río. La Comisión Ecológica del Estado ha encontrado que aunque los desechos no son toxicos para las fauna de ese río, si le afecta el aumento de 10 grados de temperatura que causan, es por eso que ha impuesto como requisito que cada Industria tenga un sistema interno para disminuir la temperatura de los desechos, y así no aumentar tanto la temperatura del río en esa parte. El costo por decrementar un grado la temperatura de los desechos es diferente para cada industria, ya que depende del tipo de enfriamiento que haya adoptado cada una, pero todas tienen la capacidad de disminuir a lo mas 10 grados la temperatura. La Comisión Ecológica determinó que para que las especies de peces que ahí habitan puedan sobrevivir, la temperatura del río debe estar entre 50 y 60 grados.

En la fuente del río situada antes de las cuatro industrias siempe hay una temperatura de 50 grados, y debido a la corriente del río éste realiza una baja de temperatura de 3 grados de A a B, 5 grados de B a C, 4 grados de C a D y 2 grados de D a la desembocadura del río.El problema entonces es encontrar cuantos grados necesita enfriar sus desechos cada industria, de tal manera que cumpla con los lineamientos de la Comisión y que les cueste lo mas barato posible. El costo por grado de enfriamiento para cada Industria se muestra en la tabla de abajo.

Industria	Costo de enfriamiento
A	10
B	15
C	20
D	15

Los nodos 1 y 2 son auxiliares para garantizar el flujo fijo de entrada y salida respectivamente; los nodos A, B, C, D y E representan los puntos donde están localizadas las industrias a través del Río, cada uno de estos nodos posee un nodo auxiliar primado para expresar las restricciones de los nodos. Los nodos biprimados se utilizan para representar el aumento de 10 grados de calor que cada industria.

Los cuatro nodos que poseen letras minúsculas se establecen para representar el decremento en la temperatura de ese punto del río.

La información de los arcos es

(Cota Inferior de flujo, Cota Superior de flujo, Costo)

donde

- El flujo representa los grados de temperatura en el río
- El costo es el de enfriemiento del río.

Formulando un modelo de red que las compañías puedan utilizar para minimizar el costo y cumplir con la restricción del gobierno.

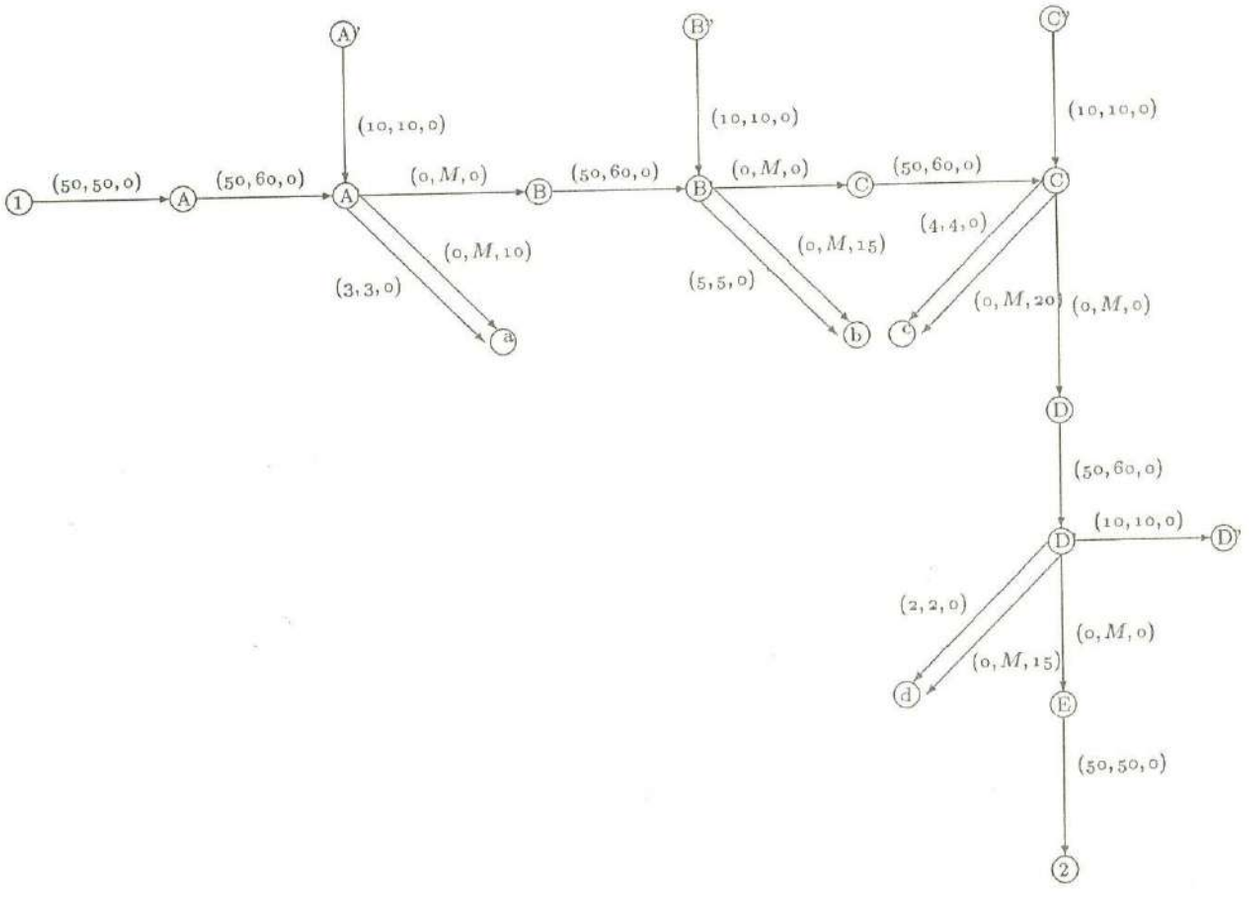


FIG. 1.8 Modelo de red del problema de Enfriamiento de un Río en Sonora

1.2.8 General Ramos 1

Una gran batalla se ha estado sosteniendo entre dos pueblos. Los enemigos están prontos en llegar para atacar al General Ramos y su tropa; él necesita hacer una operación rápida y eficaz, para lo cual están a su disposición 8 mensajeros que pueden ir a toda velocidad por refuerzos a 12 fuertes de apoyo. El número de tropas que se manden desde cada fuerte amigo dependerá del poder de convencimiento, sobre los Coroneles de cada fuerte, que cada mensajero tenga. Solo se puede enviar un mensajero a cada fuerte de ayuda.

Los ocho mensajeros se encuentran repartidos en 4 estaciones, tres en la primera, dos en la segunda, solo uno en la tercera y los restantes en la cuarta.

El general Ramos desea saber hacia donde mandar cada mensajero, de tal manera que puedan recibir la mayor cantidad de ayuda posible para no perder la batalla. El número de tropas asociado a cada mensajero por fuerte de ayuda se muestra en la tabla.

Mensajero	Fuerte											
	1	2	3	4	5	6	7	8	9	10	11	12
1	225	200	210	150	60	175	0	10	25	25	0	190
2	200	210	210	25	90	90	22	0	100	100	0	120
3	210	220	225	75	50	160	75	65	50	160	35	65
4	75	90	95	100	75	135	90	50	75	200	25	175

Considerando a los mensajeros como el flujo sobre una red, y el costo al número de tropas posibles de mandar, el problema se plantea como flujo constante a costo máximo. Para adecuarlo al tipo de problemas que aquí se plantean, el número de tropas, que son los costos se multiplican por menos uno, para poder minimizar costo. La figura 1.9 muestra la red que utilizará el General Ramos.

Los nodos del **13** al **16** sirven para representar las estaciones desde donde pueden partir los mensajeros, el nodo **17** es auxiliar para representar los 4 mensajeros que harían falta para cubrir los 12 fuertes por visitar, esto es, la asignación de alguno de estos cuatro mensajeros fantasmas querrá decir realmente que no se asignó ningún hombre hacia ese fuerte. Para cada uno de los nodos anteriores existe uno primado que es útil para representar la cantidad fija de personas que se asignan desde ahí.

Los 12 fuertes de ayuda también se representan mediante nodos que, al igual que en las estaciones, poseen nodos primados para que llegue solo un mensajero; el costo de los arcos que van desde los nodos hacia sus nodos auxiliares es cero.

El costo de los arcos que salen del nodo **17** es cero, ya que simboliza que no llegará mensajero y no se mandarán tropas.

La información de los arcos, al igual en los casos anteriores es:

(Cota inferior de flujo, Cota Superior de flujo, Costo)

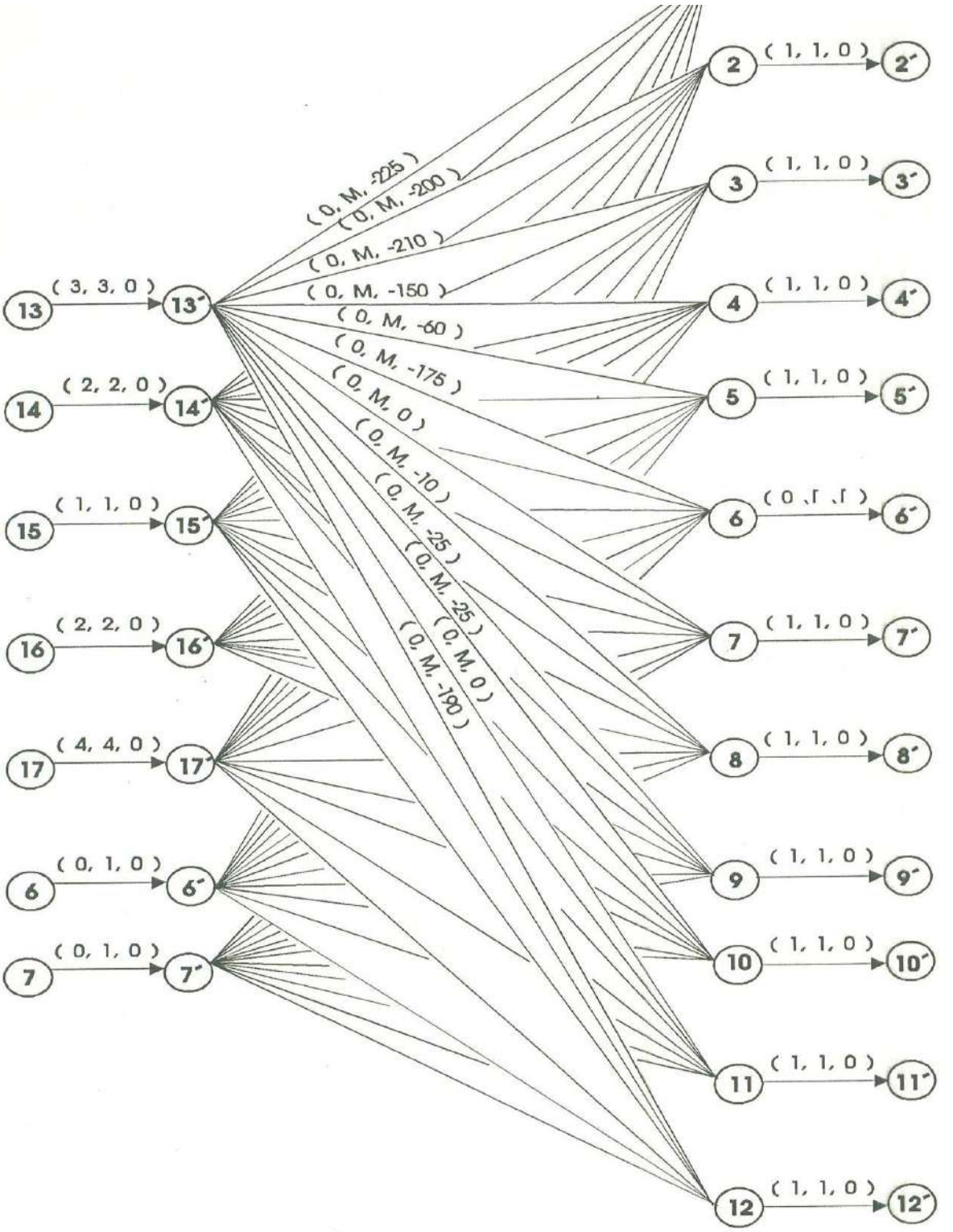


FIG. 1.9 Modelo de red del Problema del General Ramos 1

1.2.9 General Ramos 2

El General Ramos mandó a hacer un estudio más exaustivo de los mensajeros a su disposición, pidió que se les hicieran pruebas de velocidad, tenacidad y resistencia entre otras. Los resultados reflejaban que no todos los mensajeros tenian buenas aptitudes, y se llegó a la conclusión de que esto afectaría al número de tropas que los fuertes amigos pudieran mandarles para ganar la batalla.

La tabla que se muestra establece los cambios que se tubieron que hacer en el número esperado de tropas a recibir.

Mensajero	Campamento											
	1	2	3	4	5	6	7	8	9	10	11	12
1A	245	175	210	145	45	165	0	15	35	25	0	165
1B	220	225	235	155	30	225	0	10	10	25	0	190
1C	205	200	195	175	75	140	0	10	45	20	0	235
2A	235	235	245	110	75	200	85	95	45	60	20	35
2B	195	205	220	60	40	145	70	50	75	200	45	90
3A	210	220	225	75	50	160	75	65	50	160	35	65
4A	70	70	70	85	65	110	65	90	100	210	30	190
4B	90	110	120	120	95	150	110	65	50	150	10	140

El problema sigue teniendo el modelo de flujo constante a costo mínimo, solo que ahora se asigna un nodo para cada mensajero, ya que los costos, que son el número de tropas, está en función de las características de cada mensajero. Estos 12 nodos se representan con una combinación de números y letras, donde el número representa la estación de donde parte, y la letra se le asigna para diferenciar a los mensajeros.

Los doce nodos correspondientes a los fuertes de ayuda también están situados en la red que el General Ramos utilizará. Para cada nodo hay un nodo auxiliar primado, que, al igual que el ejemplo anterior sirven para representar una cantidad fija de flujo (mensajeros) que pasa a través de él.

La información que aparece en los arcos es

(Cota inferior de flujo, Cota Superior de flujo, Costo)

donde

- el flujo es el número de mensajeros a ese campo.
- el costo es el número esperado de tropas que refleja las habilidades de cada mensajero.

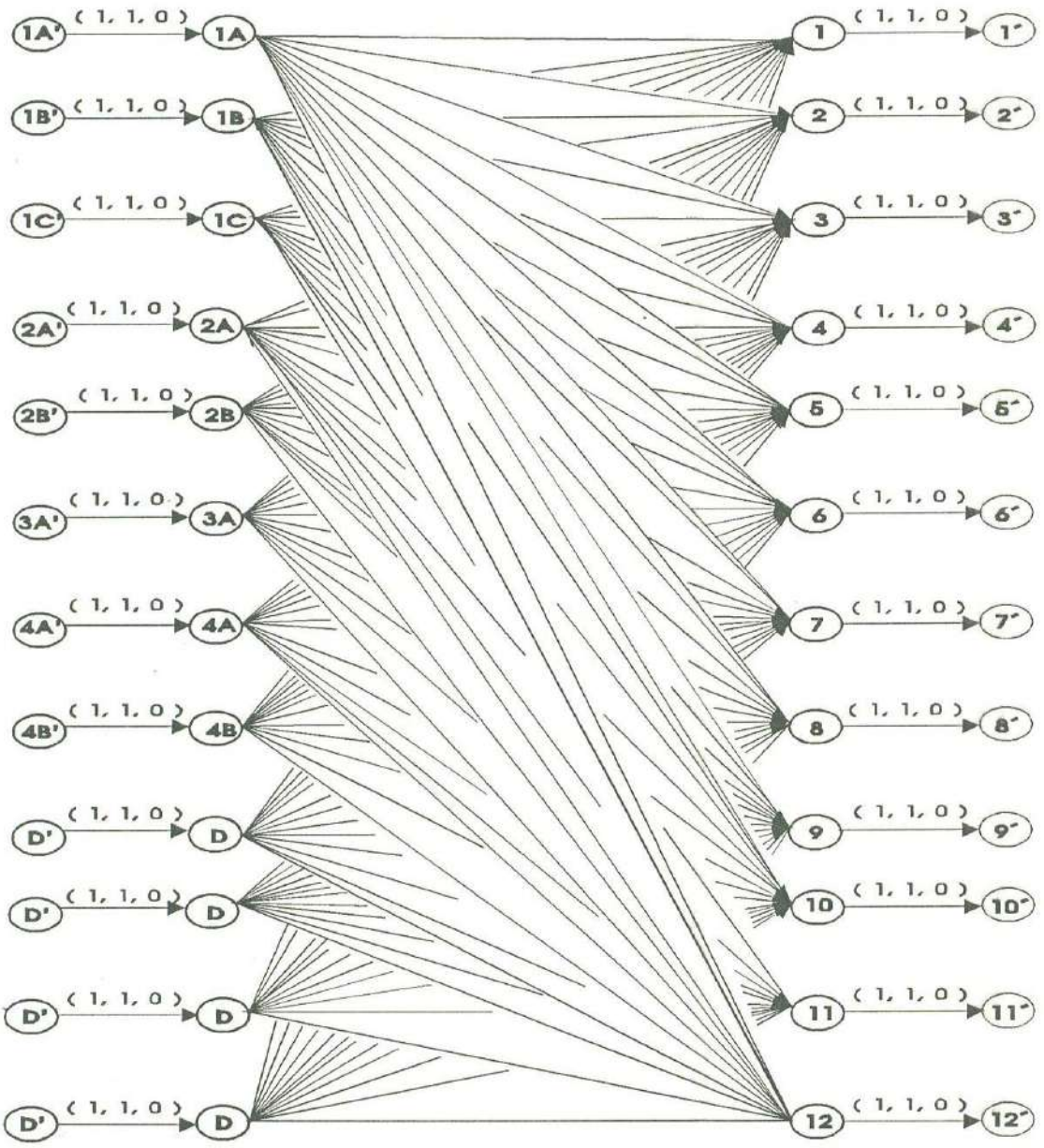


FIG. 1.10 Red que modela el Problema del General Ramos 2

Con el planteamiento de estos diversos problemas se puede deducir una manera de como asociar la red a la información que se tiene. Se establece primeramente que dato es apropiado para asignarlo al flujo, el costo se establece a la información que involucra la minimización o bien, cambiando de signo lo que se requiera maximizar, es muy natural poner los nodos, ya que siempre representan lugares, productores, etc. Los arcos simbolizan las restricciones.

Con el planteamiento de las redes no termina el problema, en el capítulo 3 se establecerá la teoría necesaria, así como uno de los algoritmos que resuelven completamente los problemas, este algoritmo se basa en la eliminación de circuitos negativos en una red llamada marginal o incremental, que se construye a partir de la red del problema, de esta manera se va modificando el flujo del valor deseado, hasta llegar a uno que sea de costo mínimo. El segundo algoritmo que se analizará en el cuarto Capítulo consiste en encontrar rutas mas cortas entre el nodo fuente y destino en la red marginal construida, las justificaciones teóricas de éste método también se presentan en este cuarto capítulo. La utilización de uno u otro algoritmo va a depender del tipo de problema, ya que para utilizar el primero de los algoritmos mencionados es necesario tener un flujo factible inicial del valor deseado, que puede ser encontrado por el algoritmo de Ford y Fulkerson para flujo máximo, haciéndole una ligera modificación, la cual encontraremos especificada en el capítulo siguiente, así como algunos resultados teóricos importantes para poder asegurar la convergencia de los dos algoritmos mencionados; el segundo de los algoritmos, en cambio no requiere tener una distribución inicial de flujo si todas sus cotas inferiores son cero, de no ser así , si se requiere un flujo inicial factible óptimo de menor valor al deseado, ya que el algoritmo se basa en ir aumentando unidades de flujo en la red cuidando que esta asignación de flujo sea siempre óptima.

Capítulo 2

Teoría Sobre Flujo Constante a Costo Mínimo

En este capítulo se establecerán algunas definiciones y resultados importantes sobre la teoría del Problema de Flujo Constante a Costo Mínimo. Se tomará como base que se conocen los conceptos básicos de Teoría de Gráficas [1] y el Algoritmo para encontrar un flujo máximo en una red (Algoritmo Ford-Fulkerson). La teoría planteada en este capítulo es necesaria para demostrar los teoremas que aseguran la convergencia de los dos algoritmos que se establecerán en los capítulos 3 y 4.

A continuación una definición de las más importantes para la descripción de cualesquiera de los dos algoritmos.

2.1 Red Marginal o Incremental

Aquí introduciremos un concepto sumamente importante para todo el desarrollo del trabajo; una nueva red que nos permitirá efectuar las iteraciones de los algoritmos. La nueva red que llamaremos marginal, comprenderá dos tipos de arcos: aquellos que nos permitan aumentar el flujo a través de ellos y que tendrán costos positivos, y los arcos que señalen que se puede disminuir flujo sobre ellos con costo negativo. Esta forma de considerar los costos permitirán identificar circuitos negativos que están asociados a ciclos en la red original, con los cuales, manteniendo el mismo valor de flujo se decrementará el costo. Por otra parte, la red marginal permitirá encontrar rutas mas cortas, que en la red original significa encontrar cadenas aumentantes para subir flujo conservando la optimalidad en costo mínimo.

Definición. Dada una red $R = [X, A, k, q, c]$, con X el conjunto de nodos, A el conjunto de arcos que la componen, k la función de capacidades mínimas asociadas a los arcos, q la función de capacidades máximas asociadas a los arcos y c los costos por unidad de flujo que pasa por los

[1] ver Anexo A

arcos. Definimos para una distribución f de flujo factible a través de R

$$R\prime\,(f) \;=\; [\,X, A_1 \cup A_2, q\prime, c\prime\,]$$

donde:

$$\begin{aligned} A_1 &= \{\,(i,j) \in A \mid f_{ij} < q_{ij}\,\} \\ A_2 &= \{\,(i,j) \mid (j,i) \in A \text{ y } f_{ij} > k_{ij}\,\} \end{aligned}$$

$q\prime_{ij}$ describe la capacidad de los arcos de $R\prime\,(f)$ como sigue

$$q\prime_{ij} \;=\; \begin{cases} q_{ij} - f_{ij} & \text{para todo } (i,j) \in A_1 \\ f_{ij} - k_{ij} & \text{para todo } (i,j) \in A_2 \end{cases}$$

$c\prime$ describe el costo unitario del flujo a través de los arcos de $R\prime\,(f)$ de la siguiente manera:

$$c\prime_{ij} \;=\; \begin{cases} c_{ij} & \text{para todo } (i,j) \in A_1 \\ -c_{ij} & \text{para todo } (j,i) \in A_2 \end{cases}$$

La nueva red marginal posee el mismo número de nodos. El conjunto A_1 de arcos son aquellos arcos de la red R tales que el flujo es menor que la capacidad, esto es, cada arco que se coloca en la red marginal representa lo que se puede aumentar de flujo en esa dirección, manteniendo el mismo costo; en cambio los arcos del conjunto A_2, que son aquellos tales que en sentido inverso están en R con flujo mayor que su capacidad mínima, es decir, arcos de regreso con capacidad lo que se puede decrementar sobre él, con costo negativo al original.

Una observación es que en la red marginal los arcos tienen como cota inferior cero, es por eso que no se hacen explícitas al crear dicha red.

Sintetizando lo que pasa al construir la red marginal tenemos que:

- Si $k_{ij} < f_{ij} < q_{ij}$ se agregan dos arcos en la red marginal, uno de ida y el otro de retorno.
- Si $f_{ij} = q_{ij}$ solo se agrega el arco (j,i) de regreso en la red marginal.
- Si $f_{ij} = k_{ij}$ solo se agrega el arco original (i,j) en la red marginal.

La red que se presenta a continuación será de mucha importancia, ya que en todo el desarrollo del presente trabajo se utilizará para aclarar conceptos, así como para realizar las iteraciones de los dos algoritmos.

Tomemos la red con el flujo dado y encontremos su red marginal asociada a ese flujo. Simbolizaremos los arcos del conjunto A_2 con líneas punteadas, y los arcos equivalentes a la red original con línea contínua.

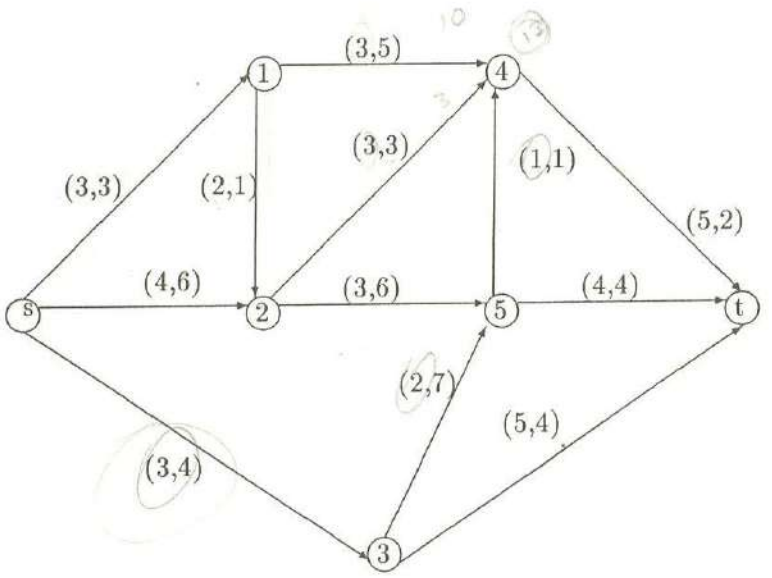
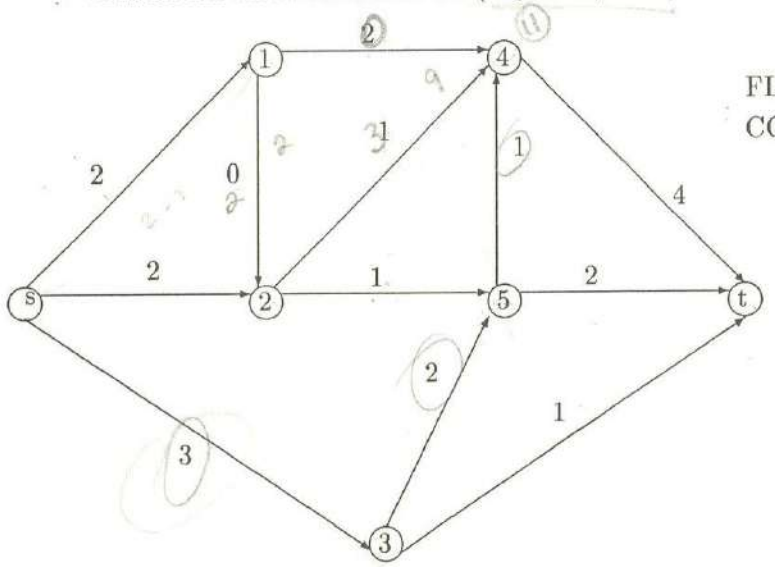


FIG. 2.1 Los valores en los arcos son (Capacidad, Costo)



FLUJO = 7
COSTO = 84

FIG. 2.2 El número asociado a cada arco es el flujo a través de él.

Entonces la red marginal respecto al flujo definido en la ***fig.2.2*** es:

$A_1 = \{ (s,1), (s,2), (1,4), (1,2), (2,4), (2,5), (3,t), (4,t), (5,t) \}$

$$
\begin{aligned}
qf_{s1} &= q_{s1} - f_{s1} = 3 - 2 = 1 & \qquad cf_{s1} &= c_{s1} = 3\\
qf_{s2} &= q_{s2} - f_{s2} = 4 - 2 = 2 & \qquad cf_{s2} &= c_{s2} = 6\\
qf_{14} &= q_{14} - f_{14} = 3 - 2 = 1 & \qquad cf_{14} &= c_{14} = 5\\
qf_{12} &= q_{12} - f_{12} = 2 - 0 = 2 & \qquad cf_{12} &= c_{12} = 1\\
qf_{24} &= q_{24} - f_{24} = 3 - 1 = 2 & \qquad cf_{24} &= c_{24} = 3\\
qf_{25} &= q_{25} - f_{25} = 3 - 1 = 2 & \qquad cf_{25} &= c_{25} = 6\\
qf_{3t} &= q_{3t} - f_{3t} = 5 - 1 = 4 & \qquad cf_{3t} &= c_{3t} = 4\\
qf_{4t} &= q_{4t} - f_{4t} = 5 - 4 = 1 & \qquad cf_{4t} &= c_{4t} = 2\\
qf_{5t} &= q_{5t} - f_{5t} = 4 - 2 = 2 & \qquad cf_{5t} &= c_{5t} = 4
\end{aligned}
$$

$$A_2 = \{\ (1,s),\ (2,s),\ (3,s),\ (4,1),\ (4,2),\ (5,2),\ (5,3),\ (t,3),\ (t,4),\ (4,5),\ (t,5)\ \}$$

$$
\begin{aligned}
qf_{1s} &= f_{s1} = 2 & \qquad cf_{1s} &= -c_{s1} = -3\\
qf_{2s} &= f_{s2} = 2 & \qquad cf_{2s} &= -c_{s2} = -6\\
qf_{3s} &= f_{s3} = 3 & \qquad cf_{3s} &= -c_{s3} = -4\\
qf_{41} &= f_{14} = 2 & \qquad cf_{41} &= -c_{14} = -5\\
qf_{42} &= f_{24} = 1 & \qquad cf_{42} &= -c_{24} = -3\\
qf_{52} &= f_{25} = 1 & \qquad cf_{52} &= -c_{25} = -6\\
qf_{53} &= f_{35} = 2 & \qquad cf_{53} &= -c_{35} = -7\\
qf_{t3} &= f_{3t} = 1 & \qquad cf_{t3} &= -c_{3t} = -4\\
qf_{t4} &= f_{4t} = 4 & \qquad cf_{t4} &= -c_{4t} = -2\\
qf_{45} &= f_{54} = 1 & \qquad cf_{45} &= -c_{54} = -1\\
qf_{t5} &= f_{5t} = 2 & \qquad cf_{t5} &= -c_{5t} = -4
\end{aligned}
$$

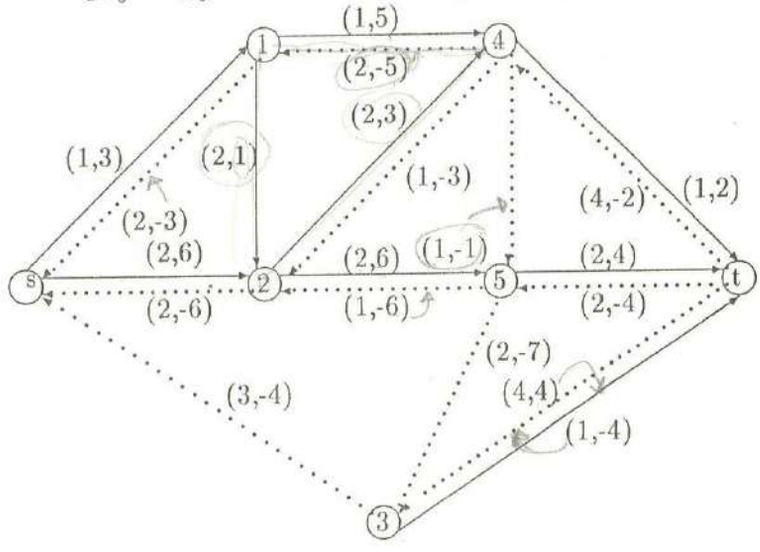


FIG. 2.3 Los números asociados son (capacidad, costo) de la nueva red marginal.

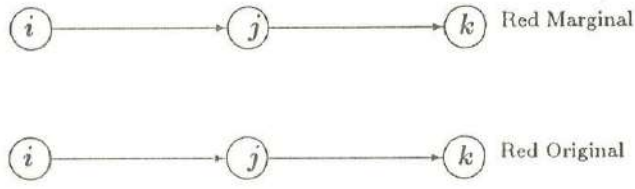
Acabamos de ver cómo se contruye la red marginal asociada a un flujo dado en una red, existe una estrecha relación entre estas dos digráficas, cuando se quiere resolver un problema sobre la red original no siempre resulta fácil, pero en ocasiones es más cómodo trabajar con la red marginal y después traducir los resultados hacia la red original.

2.1.1 Relaciones entre la Red Original y la Red Marginal

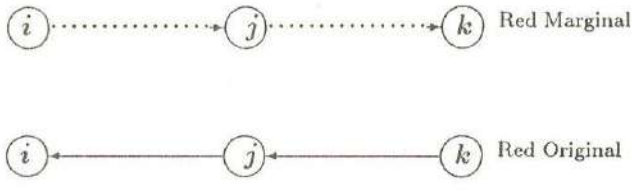
Cuando se construye la red marginal asociada a un flujo dado en la red, como ya se vió anteriormente, existen dos tipos de arcos, los que pertenecen a un conjunto denotado por A_1 que son los arcos por donde se puede incrementar el flujo y los del conjunto A_2 por los cuales se puede bajar el valor del flujo. Cuando en la red marginal se detecta un circuito, claramente corresponde a un ciclo en la red original, ya que los nodos que conforman ambas redes son los mismos, y para cada arco de la red original corresponde al menos uno en la red marginal, que puede ser en el mismo sentido o en el contrario.

Además, cuando encontramos trayectorias en la red marginal (caminos dirigidos elementales) éstas corresponden a cadenas aumentantes en la red original, para ver lo anterior basta plantearlo para un par de arcos consecutivos de la trayectoria encontrada en la red marginal, cuyos nodos son i, j y k. Representando a los arcos de A_1 con línea contínua, y los arcos del conjunto A_2 con línea punteada.

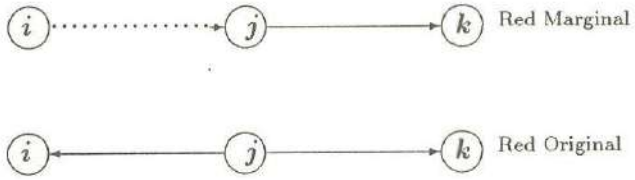
a) Cuando ambos arcos (i,j) y (j,k) pertenecen a A_1 implica que en la red original son arcos donde se puede aumentar el flujo, por lo tanto sí corresponden a una cadena aumentante.



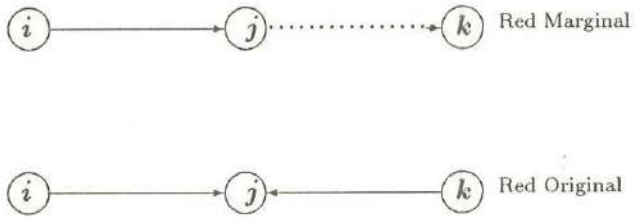
b) Cuando ambos arcos pertenecen al conjunto A_2, significando que en la red original se puede decrementar el flujo a través de ellos, en la red original aparecen los arcos en el sentido contrario del que aparecen en la red marginal, significando que se puede decrementar el flujo en una cierta cantidad a través de ellos conservando la factibilidad del flujo, lo cual significa que pertenecen a una cadena aumentante en la red original.



c) Si el arco $(i,j) \in A_2$ y $(j,k) \in A_1$, en la red original se encontrarán los arcos (j,i) (j,k), al modificar el flujo en una cantidad d apropiada (que no sobrepase los límites de capacidades de todos los arcos de la cadena) el cambio se realiza sumando d en los arcos de A_1 y restando esa misma cantidad en los arcos del otro conjunto, de esta manera se mantiene la factibilidad de la red original y corresponde a una cadena aumentante de ésta.



d) Cuando (i,j) pertenece A_1 y $(j,k) \in A_2$, donde en el primero se puede aumentar flujo y en la red original se encuentra en la misma posición, en cambio en el arco (j,k) se puede disminuir el flujo y en la red original se encuentra en la posición contraria; cuando se hace el reajuste de flujo, nuevamente define un flujo factible de mayor valor que el anterior, siendo entonces ésta una cadena aumentante.



2.2 Modificación del Flujo a Través de un Circuito

Como ya se mencionó antes, un ciclo en la red R corresponde a un circuito en la red marginal, y si existe una cadena aumentante "abierta" entre s y t en R entonces existe un camino "abierto" entre s y t en $R\prime\,(f)$ y viceversa. La importancia de lo anterior radica en que los algoritmos se basan en la construcción de la red marginal y la utilizan encontrando circuitos negativos o rutas mas cortas, para llegar a la optimalidad.

Sobre los circuitos en $R\prime\,(f)$ (o ciclos de R) se puede modificar el flujo ya que sobre los arcos $(i,j) \in A_1$ corresponden arcos de A con $k_{ij} < f_{ij} < q_{ij}$ donde se puede incrementar el flujo, y los arcos $(i,j) \in A_2$ corresponden a arcos $(j,i) \in A$ tales que $f_{ij} > k_{ij}$ por los cuales puede decrementarse el flujo a través de él.

De aquí en adelante se supondrá, sin pérdida de generalidad, que las redes utilizadas solo tienen un nodo fuente y un nodo destino, si no es así se agregan dos nodos auxiliares $s\prime$ y $t\prime$ haciendo las veces de únicos fuente y destino, $s\prime$ se conecta con arcos de salida con todos los nodos fuentes, y el nodo $t\prime$ con arcos de llegada con los nodos destinos originales, las capacidades inferiores de los nuevos arcos será cero, y la capacidad máxima de ellos un número M, que retomando lo establecido en el capítulo anterior simbolizaba una cantidad sumamente grande, a estos arcos se les asigna el costo cero, ya que son arcos auxiliares sin costo asignado. Otra suposición que es importante establecer es que los flujos a los que se refieran son de valores enteros, ya que si fueran racionales, basta multiplicar todos ellos por una constante lo suficientemente grande para lograr el objetivo; el trabajar con múltiplos en los flujos y/o capacidades no altera en absoluto el problema, ya que

se aplica el mismo esquema del algoritmo de Ford y Fulkerson que se puede aplicar solo si hay capacidades enteras.

Si el cambio en el flujo se hace como se indica en el siguiente teorema, se obtendrá un nuevo flujo factible del mismo valor v, pero de diferente costo. Así , cuando se utilicen circuitos negativos, se logrará bajar el costo del flujo conservando su valor.

Teorema. Sea una red $R = [X, A, k, q, c]$ con red marginal $R' = [X, A_1 \cup A_2, q', c']$ y flujo factible de valor v. Si existe circuito en R' y se modifica el flujo en cierta cantidad d, tal que se cumpla lo siguiente

$$k_{ij} < f_{ij} - d$$

$$f_{ij} + d < q_{ij}$$

modificando como sigue

$$f'_{ij} = \begin{cases} f_{ij} & \text{si } (i,j) \text{ no pertenece al ciclo} \\ f_{ij} + d & \text{si } (i,j) \in A_1 \text{ y pertenece al ciclo} \\ f_{ij} - d & \text{si } (j,i) \in A_2 \text{ y } (i,j) \text{ pertenece al ciclo} \end{cases}$$

se obtiene un nuevo flujo factible f' de valor v.

Demostración.

Para demostrar que f' es factible y de valor v también, basta ver que se cumpla

$$\sum_j f'_{ij} - \sum_k f'_{ki} = \begin{cases} v & \text{si } i = s \\ 0 & \text{si } i \neq s, t \\ -v & \text{si } i = t \end{cases}$$

que es la condición de flujo de las redes sin ganancia (Leyes de conservación del flujo o Leyes de Kirchoff).

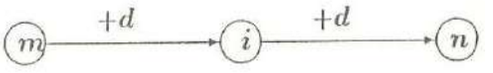
Si el vértice i no pertenece al ciclo es claro que cumple la condición de flujo en la red R

$$\sum_j f_{ij} - \sum_k f_{ki} = \begin{cases} v & \text{si } i = s \\ 0 & \text{si } i \neq s, t \\ -v & \text{si } i = t \end{cases}$$

Ahora si i es un vértice del ciclo tiene dos vecinos m y n en el ciclo.

Si en el circuito correspondiente en R' (f) están los arcos (m,i) e (i,n) tenemos varios casos:

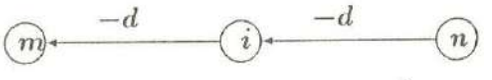
a) Si (m,i) e $(i,n) \in A_1$ tenemos en la red R



Donde m es predecesor y n es sucesor de i en el ciclo, entonces

$$\begin{aligned}
\sum_j fl_{ij} - \sum_k fl_{ki} &= \sum_{j,j\neq n} fl_{ij} + fl_{in} - \sum_{k,k\neq m} fl_{ki} - fl_{mi} \\
&= \sum_{j,j\neq n} f_{ij} + (f_{in} + d) - \sum_{k,k\neq m} -_{ki} - (f_{mi} + d) \\
&= \sum_j f_{ij} - \sum_k f_{ki} \\
&= \begin{cases} v & \text{si } i = s \\ 0 & \text{si } i \neq s,t \\ -v & \text{si } i = t \end{cases}
\end{aligned}$$

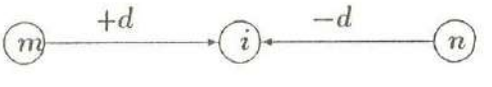
b) Si $(m,i),(i,n) \in A_2$ tenemos en la red R



Donde n es predecesor y m sucesor de i, entonces

$$\begin{aligned}
\sum_j fl_{ij} - \sum_k fl_{ki} &= \sum_{j,j\neq n} fl_{ij} + fl_{in} - \sum_{k,k\neq m} fl_{ki} - fl_{mi} \\
&= \sum_{j,j\neq n} f_{ij} + (f_{in} - d) - \sum_{k,k\neq m} -_{ki} - (f_{mi} - d) \\
&= \sum_j f_{ij} - \sum_k f_{ki} \\
&= \begin{cases} v & \text{si } i = s \\ 0 & \text{si } i \neq s,t \\ -v & \text{si } i = t \end{cases}
\end{aligned}$$

c) Si $(m,i) \in A_1$ y $(i,n) \in A_2$ tenemos en la red R



Donde m y n son predecesores de i, entonces

$$\begin{aligned}
\sum_j f'_{ij} - \sum_k f'_{ki} &= \sum_j f'_{ij} - \sum_{k,k\neq m,k\neq n} -f'_{mi} - f'_{ni} \\
&= \sum_j f_{ij} - \sum_{k,k\neq m,k\neq n} -(f_{mi}+d) - (f_{ni}-d) \\
&= \sum_j f_{ij} - \sum_k f_{ki} \\
&= \begin{cases} v & \text{si } i = s \\ 0 & \text{si } i \neq s,t \\ -v & \text{si } i = t \end{cases} .
\end{aligned}$$

y el último caso.

d) Si $(m,i) \in A_2$ y $(i,n) \in A_1$ tenemos en la red R

Donde m y n son sucesores de i, entonces

$$\begin{aligned}
\sum_j f'_{ij} - \sum_k f'_{ki} &= \sum_{j,j\neq n,j\neq m} f'_{ij} + f'_{im} + f'_{in} - \sum_k f'_{ki} \\
&= \sum_{j,j\neq n,j\neq m} f_{ij} + (f_{im}-d) + (f_{in}+d) - \sum_k f_{ki} \\
&= \sum_j f_{ij} - \sum_k f_{ki} \\
&= \begin{cases} v & \text{si } i = s \\ 0 & \text{si } i \neq s,t \\ -v & \text{si } i = t \end{cases}
\end{aligned}$$

■

La red marginal nos indica entonces la manera de obtener otros flujos factibles del mismo valor, y no solo eso sino también nuevo costo, ya que incrementar una unidad de flujo en los arcos $(i,j) \in A$ significa incrementar el costo en c_{ij}; y decrementar flujo en arcos (i,j) implica decrementar el costo en c_{ij} . El cambio de costo estará dado por el producto de las unidades de flujo con las suma de los costos de los arcos del circuito en R'; a este valor le llamaremos Costo del Circuito. El siguiente diagrama nos ejemplificará mejor lo anterior.

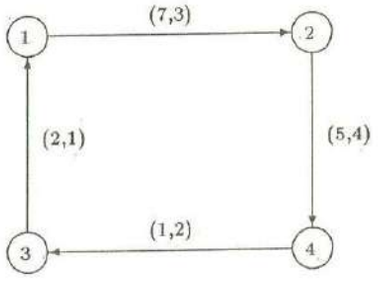


FIG. 2.4 Circuito 1, 2, 4, 3 donde (Capacidad, Costo), entonces el costo del circuito es $3+4+2+1 = 10$

De lo anterior se puede concluir que para bajar el costo de una red con un flujo constante basta encontrar circuitos en la red marginal asociada al flujo que tengan costos negativos y modificar el flujo a través de él.

Como el comportamiento al decrementar el costo en la red es lineal, una pregunta interesante sería ¿Cuál es la cantidad máxima de costo que se puede bajar en un circuito negativo?.

La única condición que se tiene que cumplir al cambiar el flujo es que no se rebasen las capacidades de los arcos que forman el circuito, entonces la máxima cantidad de flujo d que pueda pasar por el ciclo será lo más que se pueda bajar el costo de la red, donde

$$d = \min \{ q_{ij} \text{ con } (i,j) \text{ en el circuito de la red marginal} \}$$

Retomando nuevamente la red de la *fig.2.3* veamos cuanto podemos modificar el flujo para disminuir el costo en el circuito negativo $(1, 2, 4, 1)$ de costo -1, tenemos que

$$d = \min \{ 2, 2, 2 \} = 2$$

entonces podemos modificar a lo mas 2 unidades en flujo que produce una baja en el costo de -2 quedando finalmente la red siguiente con costo 82.

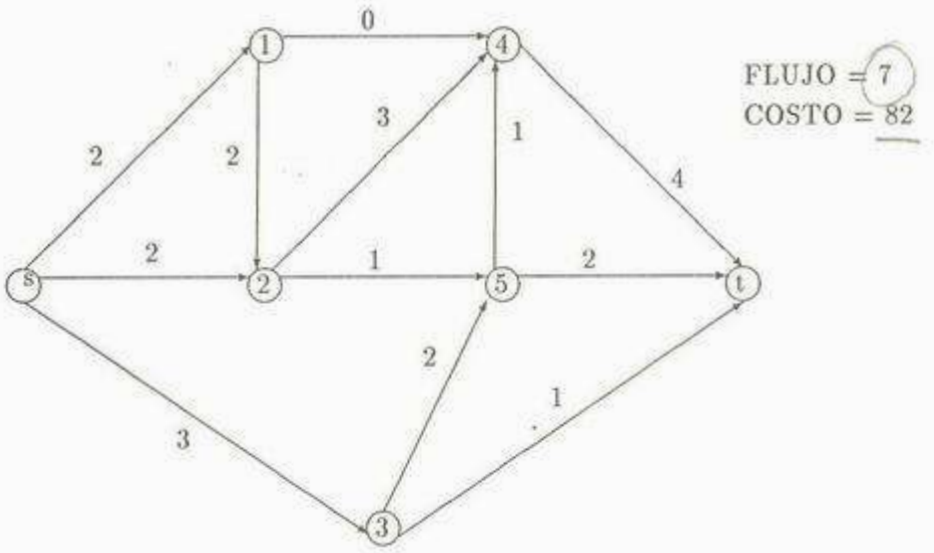


FIG. 2.5 Red con cambio en el flujo para disminuir el costo.
Los Números en los arcos representan el flujo a través de ellos.

Con el procedimiento anterior acabamos de bajar el costo del flujo en la red pero aún no es de costo mínimo ya que existen otros circuitos negativos como lo son (s, 1, 2, s), (2, 4, 5, 2), (5, 3, t, 5) donde también se puede decrementar el costo de la red.

Antes de establecer en un teorema las condiciones necesarias y suficientes para tener un flujo de costo mínimo necesitamos un resultado importante de descomposición de flujos.

2.3 Flujo Conforme

La descomposición de un flujo en unidades elementales es necesaria para la demostración de uno de los teoremas centrales, que garantiza la convergencia del Algoritmo de eliminación de circuitos negativos, cuando se efectúa tal descomposición, cada unidad posee un tipo especial de flujo llamado conforme, que es un concepto muy útil en la demostración, por lo cual es conveniente establecer primeramente lo que es un flujo conforme.

Antes de establecer formalmente la definición de lo que es un flujo conforme hay que clarificar el concepto de flujo total a través de un par de vértices.

Si dados dos nodos i e j con arcos en la dirección (i,j) y (j,i) (pueden ser mas de un arco en la misma dirección) y denotemos por w_{ij} y w_{ji} al flujo que circula a través de cada arco con las direcciones respectivas señaladas anteriormente, definimos el flujo neto que pasa a través de esos nodos en la dirección i a j como: $f_{ij} = \sum_{(i,j)\in R} w_{ij} - \sum_{(j,i)\in R} w_{ji}$.

Definición. Sea una red R sobre la cual hay definido un flujo f. Decimos que un flujo es ***Conforme*** si para cada par de vértices i e j con f_{ij} el flujo neto a través de ellos, hay un número total de unidades f_{ij} de flujo todas usadas en la dirección de i a j.

Los flujos pueden no ser conformes, por ejemplo, si hay un número total de $f_{ij} + p$ unidades de flujo a través del arco en la dirección i a j y p unidades de flujo a través del arco en la dirección opuesta, produce un flujo neto de f_{ij} de i a j. En la siguiente figura se puede apreciar mas este comentario.

$f_{12} = 3 + 5 - 2 = 6$
Flujo no Conforme

$f_{12} = 4 + 2 = 6$
Flujo Conforme

FIG. 2.6 Ejemplo de Flujo Conforme y no Conforme

2.4 Descomposición de un Flujo

En esta sección veremos como descomponer un flujo dado en una red como suma de flujos mas simples. En la práctica esto no es muy usual, pero en cambio en desarrollos teóricos es útil tener una simplificación tal de un flujo. La descomposición mencionada consiste en primeramente obtener una nueva red, con el mismo número de nodos y un arco por cada unidad de flujo; posteriormente se desglosa la red en circuitos y trayectorias elementales, por las cuales circule solamente una unidad de flujo.

Denotaremos por h o (S) al flujo de un conjunto S arbitrario de arcos (i,j) en una red R, donde $f_{ij} = h$ en arcos $(i,j) \in S$ y $f_{ij} = 0$ si $(i,j) \notin S$. El conjunto S puede ser un circuito o una trayectoria.

El siguiente ejemplo nos clarificará la notación establecida anteriormente.

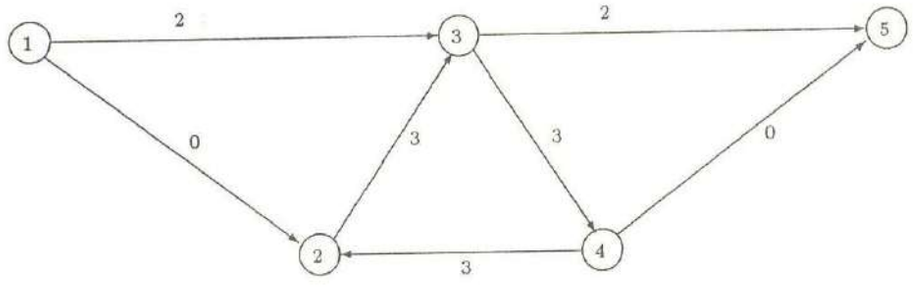


FIG. 2.7 Donde $S = \{(2,3), (3,4), (4,2)\}$ entonces 3 o (S) es el flujo de valor 3 que pasa a través de S.

La manera como se dá la descomposición de un flujo se establece en el siguiente.

Teorema. Si f es un flujo (de s a t) de valor v (entero) en una red R, entonces f se puede descomponer como:

$$f = 1\,o\,(P_1) + 1\,o\,(P_2) + \ldots + 1\,o\,(P_v) + 1\,o\,(\Phi_1) + 1\,o\,(\Phi_2) + \ldots + 1\,o\,(\Phi_{k_1})$$

donde $P_1, \ldots, P_v$ son trayectorias de s a t de R y $\Phi_1, \ldots, \Phi_k$ circuitos elementales también en R.(Los P_i y Φ_i no necesariamente distintos).

Demostración.

Dada la red $R = [X, A, k, q, c]$ con un flujo f, construimos la red unitaria $R^e = [X^e, A^e, k, q, c]$ como sigue:

- El conjunto X^e de vértices de R^e es el mismo que el conjunto X de vértices de R.
- Si f_{ij} es el flujo en el arco (i, j) de R entonces ponemos f_{ij} arcos paralelos entre los correspondientes nodos i^e y j^e de R^e. Si $f_{ij} = o$, entonces no se colocarán arcos entre i^e y j^e.

Cada arco en R^e corresponde a una unidad de flujo de los arcos de la red R, entonces R^e representa el flujo f en R.

En la red R^e el grado de los vértices satisface la siguiente condición dado que la condición de flujo para redes sin ganancia se cumple:

$$\begin{aligned} \Gamma^-(i^e) &= \Gamma^+(j^e) \quad \forall\, i^e \neq s^e\, o\, t^e \\ \Gamma^-(s^e) &= \Gamma^+(t^e) = v \end{aligned}$$

Como el total de unidades de flujo que energen del nodo fuente llegan al nodo destino, y dado que por cada arco solamente puede pasar una unidad de flujo a través de ella, tenemos v caminos dirigidos simples de s a t[2] que conducen dichas unidades de flujo. Sean $P\prime_1, P\prime_2, \ldots, P\prime_v$ esos caminos. Los caminos $P\prime_i$ no necesariamente es elemental , pero un camino no elemental se puede considerar como la suma de una trayectoria (de s^e *a* t^e) y un número de circuitos elementales cuyos arcos son disjuntos[3], entonces tenemos:

$$f = 1\,o\,(P_1) + 1\,o\,(P_2) + \ldots + 1\,o\,(P_v) + 1\,o\,(\Phi_1) + 1\,o\,(\Phi_2) + \ldots + 1\,o\,(\Phi_k)$$

donde los P_i son las trayectorias elementales (de s^e *a* t^e) y los Φ_i son los circuitos elementales

■

A continuación veremos un ejemplo de un flujo f que es descompuesto en trayectorias de s a t y circuitos.

[2]ver Anexo A
[3]ver Anexo A

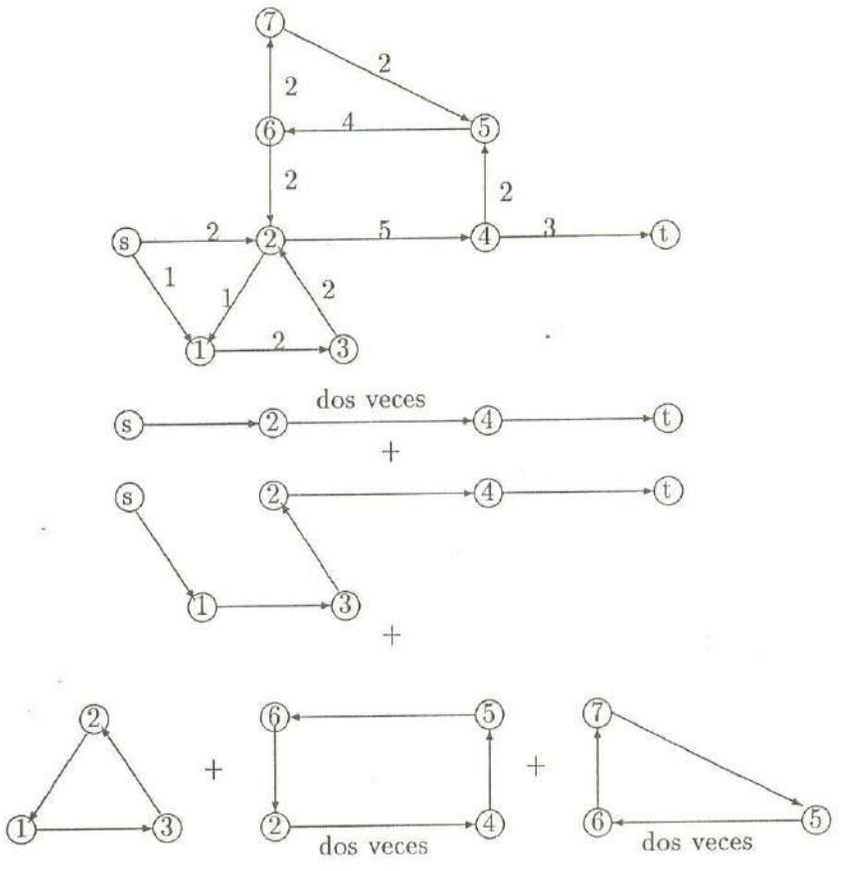


FIG. 2.8 Descomposición de un flujo f en trayectorias elementales (de s a t) y circuitos

En el ejemplo anterior se puede apreciar la descomposición de un flujo f en unidades de flujos unitarios (trayectorias y circuitos), como se puede ver las unidades de la descomposición no son todas diferentes.Una observación interesante es que la descomposición en esta forma de un flujo no es trivial como pudiera parecer.

En general no todos las trayectorias y circuitos son distintos. Si solamente la primer trayectoria $v\prime$ y el circuito $k\prime$ son distintos, con la trayectoria P_i apareciendo h_i veces en la lista $P_1, \ldots, P_v$ y el circuito Φ_i aparece l_i veces en la lista $\Phi_1, \ldots, \Phi_k$, entonces f se puede escribir:

$$f = \sum_{i=1}^{v\prime} h_i \circ (P_i) + \sum_{i=1}^{k\prime} l_i \circ (\Phi_i)$$

Un resultado importante obtenido en el desarrollo de la anterior demostración es que las unidades de flujo en el cual f ha sido descompuesto son ***conformes***, que como se vió anteriormente quiere decir que el flujo total que pasa por cualquier arco (i, j) se encuentra precisamente solo en

esa dirección, y es fácil de ver, ya que el flujo fue descompuesto en circuitos y/o trayectorias con flujo unitario.

A continuación se presentará una proposición que es muy importante en el desarrollo de la demostración del Teorema que justifica el método basado en la eliminación de circuitos negativos. Para establecer la proposición como tal hay que recordar, del teorema anterior, que si se tiene una descomposición de un flujo f, ésta es en unidades conformes de flujo.

No solo es importante esta proposición para el teorema, sino que además es un resultado teórico muy interesante. Lo que establece la proposición es que si se tiene una descomposición del un flujo factible, cualquier suma parcial de los elementos de la descomposición será un flujo factible también para la red.

Proposición. Sea f un flujo factible en una red R, cón descomposición

$$f = 1 \circ (P_1) + 1 \circ (P_2) + \ldots + 1 \circ (P_v) + 1 \circ (\Phi_1) + 1 \circ (\Phi_2) + \ldots + 1 \circ (\Phi_k)$$

entonces cualquier suma

$$f = 1 \circ (P_1) + 1 \circ (P_2) + \ldots + 1 \circ (P_m) + 1 \circ (\Phi_1) + 1 \circ (\Phi_2) + \ldots + 1 \circ (\Phi_l)$$

con $1 \leq m \leq v$ y $1 \leq l \leq k$ es factible, donde el flujo es cero en los elementos de la descomposición que no aparecen en la suma .

Demostración.

Para demostrar que la suma dada antes sea factible, basta ver que cada vértice cumple las ecuaciones de conservación de flujo.

La demostración se hará por inducción.

Tomemos la distribución de flujo que consiste de una sola unidad, (ya sea circuito o trayetoria) de la descomposición, y todos los demás arcos con flujo igual a cero, esta distribución es factible, ya que cada término de la descomposición cumple las leyes de Kirchhoff.

Veamos ahora que pasa si nuestra distribución de flujo consiste de dos elementos de la descomposición, ya sean circuitos o trayectorias.

Sea i un nodo de la red, claramente si este nodo pertenece únicamente a un elemento de la descomposición del flujo, se siguen cumpliendo las ecuaciones de conservación de flujo.

Sean f y g las distribuciónes de flujo respecto a dos elementos de la descomposición, entonces tenemos para el vértice i que $\sum_j f_{ij} - \sum_k f_{ki} = 0$ y $\sum_j g_{ij} - \sum_k g_{ki} = 0$. Si el vértice i se encuentra en dos unidades de la descomposición, las leyes de Kirchhoff aplicadas a él serán:

$$\sum_j (f_{ij} + g_{ij}) - \sum_k (f_{ki} + g_{ki}) =$$

$$= \sum_j f_{ij} + \sum_j g_{ij} - \sum_k f_{ki} - \sum_k g_{ki}$$

$$= \sum_j f_{ij} - \sum_k f_{ki} + \sum_j g_{ij} - \sum_k g_{ki}$$

$$= \mathbf{o}$$

Siguiendo el procedimiento anterior se demuestra para $k + 1$ unidades de flujo partiéndo de la hipótesis de inducción válida para k elementos, con esto el resultado queda demostrado.

■

Retomemos el ejemplo anterior de descomposición de un flujo para observar que cualquier suma parcial de elementos de la descomposición es un flujo factible para la red.

Tomando la primera trayectoria y los dos últimos circuitos de los elementos de la descomposición, y en aquellos arcos que no se encuentren en ninguno de las anteriores componentes definimos el flujo de cero, nos queda definido el siguiente flujo.

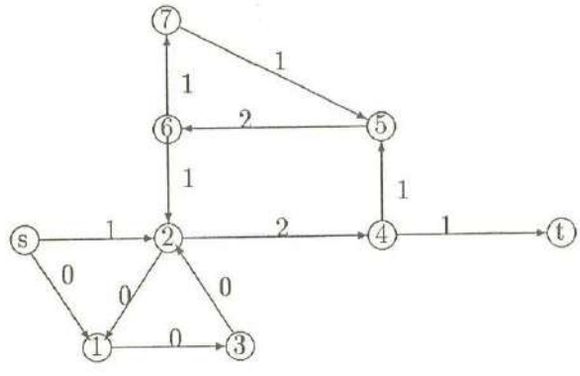


FIG. 2.9 Flujo factible definido mediante la suma parcial de elementos de la descomposición

Recordando que para aplicar el algoritmo de eliminación de circuitos negativos es necesario tener un flujo factible de cierto valor v menor o igual al máximo sobre la red que deseamos resolver, se comentó anteriormente que una modificación ligera al algoritmo de Ford y Fulkerson nos era útil. Esta modificación del Ford-Fulkerson también nos puede servir para el segundo de los algoritmos, ya que éste opera a partir de un flujo de valor menor al deseado que sea factible, aunque el flujo inicial que siempre se toma por facilidad es el cual todos los arcos tiene cero en flujo si las cotas inferiores para todos los arcos es cero.

En la siguiente sección se comentará sobre el cambio que hay que hacer el algoritmo de Flujo máximo para nuestros propósitos.

2.5 Modificación de Ford-Fulkerson para Obtener una Red con Flujo de Valor v

Como ya sabemos el algoritmo de Ford y Fulkerson está basado en encontrar cadenas aumentantes del nodo fuente s al nodo destino t, hasta que ya no se pueda aumentar el flujo; para obtener un flujo f de cierto valor v con éste método solo hay que hacer una modificación en el criterio de paro. Cada que se encuentre una nueva cadena aumentante se tendrá que revisar si el valor $v\prime$ de flujo obtenido con esa nueva cadena es menor que el valor v deseado de flujo, si es así , se continúa; si los valores de los flujos son iguales, el algoritmo se detiene y ya se posee el flujo deseado.Si el nuevo flujo sobrepasa al valor deseado, basta regresar y efectuar la iteración para un flujo $v - v\prime$

Recordemos que en el algoritmo de Ford y Fulkerson se manejan solo cantidades enteras en los flujos, y que en cada iteración del algoritmo hay aumento, es por eso que sirve sin problema a nuestros fines.

Aquí termina la parte de los conceptos y resultados importantes que son base para demostrar los dos teoremas centrales que justifican la convergencia del Algoritmode Eliminación de circuitos negativos, y el de rutas mas cortas, que como ya se comentó antes, el primero de ellos consiste en encontrar en la red marginal circuitos con costo negativo, y a través de ellos modificar el flujo haciendo con esto bajar el costo total del flujo; el segundo algoritmo consiste en, a partir de un flujo de valor menor al deseado encontrar rutas mas cortas en la red marginal, que corresponden a cadenas aumentantes en la red original por donde se mandará el máximo posible de unidades de flujo, manteniendo la optimalidad del flujo, hasta llegar al valor deseado de flujo.Estos algoritmos serán detallados en los dos capítulos siguientes,tres y cuatro, también se hará la corrida de las iteraciones necearias para resolver dos problemas.

Capítulo 3

Algoritmo Basado en la Eliminación de Circuitos Negativos

En este Capítulo se presentará uno de los teoremas centrales del presente trabajo, en donde se justifica la convergencia del método basado en eliminación de circuitos negativos y nos lleva a formular un primer algoritmo que nos resuelva el problema de flujo constante a costo mínimo. Los detalles de la implementación computacional de este algoritmo también será presentada aquí , además, se resolverán problemas utilizando la información en una red representada gráficamente y se realizan corridas de escritorio del algoritmo con las estructuras de datos utilizadas.

Este Capítulo y el siguiente serán la parte central del presente trabajo.

3.1 Condición de Optimalidad

El aspecto teórico que justifique la convergencia de un algoritmo es muy importante, ya que con esto se garantiza que se va a tener una solución confiable al aplicarlo. en esta sección se plantea el teorema que nos da la condición de optimalidad, así como su demostración.

Teorema. Un flujo f de valor v es de mínimo costo si y solo si no existe circuito Φ en $R_f(f)$ en el cual la suma de los costos de los arcos en Φ es negativo.

Demostración.

Dado que el teorema indica una caracterización de un flujo óptimo de valor mínimo mediante la ausencia de circuitos de costo negativo, la demostración se hará en dos partes, La primera parte se demostrará por contradicción partiendo del hecho de que el flujo definido en la red es de costo mínimo y suponiendo la existencia de un circuito de costo negativo en la red marginal, dado que sobre cualquier circuito en la red marginal se puede modificar el flujo por lo menos una unidad adicional sin que el flujo total en la red se modifique, se obtiene un nuevo flujo de menor costo, ya que el circuito es de costo negativo y de esta manera se contradice el hecho de que el flujo inicial era óptimo de costo mínimo.

Para la demostración de la segunda parte se inicia del hecho de que no existen circuitos de costo negativo en la red marginal asociada a un flujo f pero que no es de costo mínimo, se utilizan resultados derivados de la descomposición de un flujo en unidades elementales de flujo conforme hasta llegar a la contradicción de que el flujo f no es mínimo.

Sea $c[\, f\,]$ el costo del flujo f en la red R y $c[\, \Phi \mid R/(f)\,]$ la suma de los costos de los arcos en el circuito Φ con respecto a la red $R/$.

NECESIDAD. Sea $c[\, \Phi \mid R/(f)\,] < 0$ para algún circuito Φ de $R/(f)$. La circulación de una unidad adicional de flujo al rededor del circuito Φ produce el nuevo flujo $f + 1 \circ (\Phi)$ y el valor del flujo v de s a t no cambia. El costo del flujo $f + 1 \circ (\Phi)$ es $c[\, f\,] + c[\Phi \mid R/(f)\,] < c[\, f\,]$ lo cual contradice la aseveración que f es el flujo de costo mínimo de valor v.

SUFICIENCIA. Asumamos que $c[\, \Phi \mid R/(f)\,] \geq 0$ para cada circuito Φ en la red $R/(f)$ y que f^* $(\neq f)$ es el flujo de mínimo costo de valor v.

Ahora tomemos $f^* - f$, donde para el arco (i, j) es $f^*_{ij} - f_{ij}$.

Cada flujo f^* y f pued ser descompuesto en la suma de flujos a lo largo de trayectorias y circuitos elementales, como lo establece un teorema anterior, pero como ambos flujos son de valor v, solo difieren respecto a los arcos que se encuentran en circuitos, de aquí que la descomposición de la diferencia de esos flujos solo va a involucrar circuitos, es decir, si $\Phi_1, \Phi_2, \ldots, \Phi_k$ son los circuitos elementales dados podemos escribir:

$$f^* - f = 1 \circ (\Phi_1) + 1 \circ (\Phi_2) + \ldots + 1 \circ (\Phi_k)$$

Dado que cada flujo $1 \circ (\Phi_i)$ $i = 1, \ldots, k$ es conforme (por un teorema anterior), y sabemos que el flujo $f^* = f + 1 \circ (\Phi_1) + 1 \circ (\Phi_2) + \ldots + 1 \circ (\Phi_k)$ es factible, cualquier suma $f^* = f + 1 \circ (\Phi_1) + 1 \circ (\Phi_2) + \ldots + 1 \circ (\Phi_l)$ es factible para cualquier $1 \leq l \leq k$. Entonces considerando el flujo $f + 1 \circ (\Phi_1)$ tenemos:

$$\begin{aligned} c[\, f + 1 \circ (\Phi_1)\,] &= c[\, f\,] + c[\, \Phi_1 \mid R/(f)\,] \\ &\geq c[\, f\,] \end{aligned}$$

Consideremos ahora la red incremental $R/(\, f + 1 \circ (\Phi_1)\,)$. Los únicos arcos de la red los cuales tienen los costos reducidos comparados con con los correspondientes costos en la red $R/$ son

aquellos que van en "reversa" en el circuito Φ_1. Ahora, ya que los flujos $1 \circ (\Phi_1)$, $1 \circ (\Phi_1), \ldots$ son conformes y bajo la suposición de que no existen circuitos negativos en la red incremental, tenemos que:

$$c[\, \Phi_l \mid R_f(f + 1 \circ (\Phi_1))\,] \geq c[\, \Phi_l \mid R_f(f)\,]$$

para cualquier $l = 1, 2, \ldots, k$.

Realizando un procedimiento análogo al anterior vemos que el flujo $f + 1 \circ (\Phi_1) + 1 \circ (\Phi_2)$ es:

$$\begin{aligned} c[\, f + 1 \circ (\Phi_1) + 1 \circ (\Phi_2)\,] &= c[\, f + 1 \circ (\Phi_1)\,] + c[\, \Phi_2 \mid R_f(f) + 1 \circ (\Phi_1))\,] \\ &\geq c[\, f + 1 \circ (\Phi_1)\,] + c[\, \Phi_2 \mid R_f(f)\,] \\ &\geq c[\, f + 1 \circ (\Phi_1)\,] \\ &\geq c[\, f\,] \end{aligned}$$

Continuando de esta manera, finalmente tenemos que $c[\, f^*\,] \geq c[\, f\,]$ lo cual contradice la suposición de que f^* es el flujo de costo mínimo.

■

Entonces de acuerdo a este teorema para encontrar el flujo de costo mínimo es necesario tener un flujo inicial de valor deseado; lo anterior puede ser encontrado con el Algoritmo de Ford y Fulkerson para flujo máximo haciendole la modificación en el criterio de paro. Para verificar si existen circuitos de costo negativo en la red se pueden utilizar el algoritmo de Floyd, que nos da la ruta mas corta de un nodo dado a todos los demás y si existe un circuito negativo lo señala, o el algoritmo General de Dijkstra, que nos proporciona la ruta mas corta entre un par de vértices o bien el ciclo negativo que exista.
De esta manera teniendo un flujo factible, verificamos si existen circuitos negativos, si así es modificamos el flujo d unidades, donde d es la máxima cantidad de flujo que podemos hacer pasar en ese ciclo, con la certeza de que el nuevo flujo es de costo menor que el anterior, y se repite este procedimiento hasta no tener circuitos negativos en la red marginal asociada.

3.2 Descripción del Algoritmo

La descripción del algoritmo se establece en los pasos que a continuación se presentan, cada uno de ellos están justificados en el teorema anterior, o bien en resultados del capítulo dos.

PASO 1 [Obtención del flujo factible de valor v] Determínese un flujo factible de valor v mediante la modificación del algoritmo de Ford y Fulkerson.

PASO 2 [Red Marginal] Constrúyase la red marginal $R'(f)$ con respecto a f.

PASO 3 [Circuitos negativos] Mediante algún algoritmo de rutas mas cortas, identifíquese algún circuito negativo en $R'(f)$. Si no existen circuitos negativos, terminar. El flujo actual f es el requerido; en otro caso, sea Φ el circuito negativo. Ir al paso 4.

PASO 4 [Actualización del flujo] Sea $d = \min_{(i,j)\in\Phi}\{q'_{ij}\}$

(i) Para todo arco $(i,j) \in A$ tal que $(i,j) \in A_1 \cap \Phi$, actualizar $f_{ij} = f_{ij} + d$.

(ii) Para todo arco $(i,j) \in A$ tal que $(j,i) \in A_2 \cap \Phi$, actualizar $f_{ij} = f_{ij} - d$.

Con este nuevo flujo ir al paso 2.

3.3 Implementación Computacional del Algoritmo

Como observamos en la secciom anterior no basta tener un algoritmo eficiente que nos resuelva un problema, es también importante tener una implementación computacional de él, ya que permite resolver problemas de gran tamaño en un tiempo corto, y mientras mas eficiente sea ésta, mejores resultados se tendrán. En esta sección se presentará una implementación realizada del algoritmo presentado, realizado con Lenguaje C por ser un lenguaje muy versátil, con fácil manejo de estructuras de datos, que se utilizan en la implementación.

En las siguientes secciones veremos en general como opera el Algoritmo bajo las estructuras de información definidas, el código del programa fuente, así como de las funciones que se utilizan se presenta en el Anexo C.

Algo sumamente importante para la implementación de un algoritmo es tener la información agrupada en una manera clara y fácil de utilizar. Para el algoritmo de eliminación de circuitos negativos se utilizaron Estructuras de Datos [1] como listas, listas doblemente enlazadas. La red principal es una lista ordenada de nodos, cada nodo tiene ligadas una lista de arcos sucesores; la red marginal se construye de una manera similar agregandole una lista de arcos antecesores. Para guardar el circuito negativo que detecta el algoritmo de Floyd aplicado a la red marginal se utiliza una lista doblemente ligada para facilidad de recuperar la información.

En la implementación del Algoritmo basado en la eliminación de circuitos negativos se utilizó el algoritmo de Floyd. Una pregunta interesante sería ¿Por qué utilizar Floyd en lugar del Algoritmo de Dijkstra?. Una de los hechos por que se prefirió el algoritmo de Floyd es que el algoritmo de Dijkstra va formando en la solución una arborescencia de peso mínimo con los arcos, los arcos que no entran en la solución inicial se revisan para ver si ayudan a reducir el costo de la solución, pero mediante esta revisión de arcos no necesariamente se detectan todos los circuitos de costo negativo que pudieran haber en la red marginal, otra ventaja que ayudó a la decisión de implementar con Floyd fue que es un algoritmo sencillo y rápido, sin demasiada sofisticación, ya que solo se utiliza para el cálculo de circuitos negativos, en cambio para el algoritmo de determinación de rutas mas cortas solamente se puede utilizar el Dijkstra Generalizado, ya que se necesita determinar rutas mas cortas en redes con pesos negativos, y de los dos es el único que lo encuentra.

[1] ver referencia [7]

Aunque se aventaja en sencillez y rapidez con el algoritmo de Floyd, éste tiene la desventaja que descompone la red que se utilice, es por eso que fue necesario construir la red marginal aparte del la red original, y calcularla en cada iteración. Una implementación más eficiente sería aquella en donde se actualiza la red marginal, ya que de una iteración a otra solo cambia en los arcos que fue modificado el flujo.

3.3.1 Manejo de la Información

Como ya se ha comentado el manejar de una manera adecuada la información es muy importante para una buena y eficiente implementación de un algoritmo, a continuación se muestran las estructuras de datos que se utilizaron para el guardado de la información, cada estructura distribuye de diferente manera la información que se requiere de acuerdo al uso que se le vaya a dar dentro del algoritmo. Las estructuras de datos para los nodos o arcos de ambas redes (Marginal y original) no se encuentran aisladas, hay cierto tipo de ligamientos entre ellas,algunas están ligadas sencillamente pudiendo solamente accesar al elemento siguiente, otras tienen la característica de interaccionar con el elemento anterior y posterior debido a una doble ligadura, el tipo de uniones depende de la manera como son utilizadas a lo largo de la implementación del algoritmo. En la sección siguiente se detallará mas sobre la manera en que se va ligando la información para formar la red marginal, el circuito de costo negativo detectado, etc.

- Red Original

  En la red original tenemos definida las siguientes estructuras:

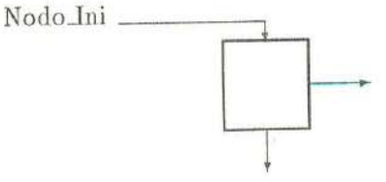
```
Estructura  NODO {
              NUMERO
              DIR_ARCO_SUC
              DIR_SIGUIENTE
            }
```

  En la estructura NODO se guarda la información de los nodos de la red, donde cada elemento representa lo siguiente:

  - NUMERO. Corresponde al número asignado al nodo en la red.
  - DIR_ARCO_SUC. Nos permite tener acceso a la lista de los arcos de salida de ese nodo.
  - DIR_SIGUIENTE. Da acceso al siguiente nodo de la lista ordenada.

  El nombre asignado al inicio de la lista de nodos de la red original es NODO_INI.

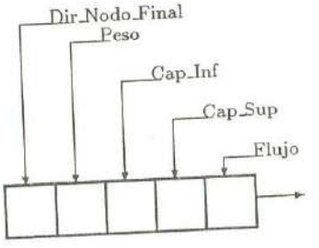
  La figura siguiente muestra un diagrama de la representación de un nodo mediante estructuras de datos.



```
Estructura  ARCO_SUC {
              DIR_NODO_FINAL
              PESO
              CAP_INF
              CAP_SUP
              FLUJO
              DIR_SIGUIENTE
            }
```

En la estructura ARCO_SUC se guarda la información de los arcos de salida de la red (Arcos Sucesores), donde cada elemento representa lo siguiente:

- DIR_NODO_FINAL. Nodo de llegada del arco.
- PESO. Corresponde al costo del arco por unidad de flujo.
- CAP_INF. Capacidad mínima de flujo que puede circular por el arco.
- CAP_SUP. Capacidad máxima de flujo que puede circular por el arco.
- FLUJO. Flujo que circula a través de ese arco.
- DIR_SIGUIENTE. Da acceso al siguiente arco de la lista ordenada.



- Red Marginal

Para la red marginal definimos otro tipo de estructuras, adecuadas para poder aplicar el algoritmo de Floyd y encontrar ciclos negativos.

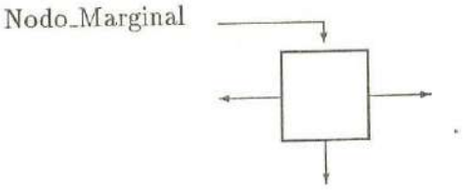
```
Estructura  NODOM {
              NUMERO
              DIR_ARCO_SUCM
              DIR_ARCO_ANTM
              DIR_SIGUIENTE
            }
```

En la estructura NODOM se guarda la información de los nodos de la red marginal, donde cada elemento representa lo siguiente:

- NUMERO. Corresponde al número asignado al nodo en la red marginal.

- DIR_ARCO_SUCM. Nos permite tener acceso a la lista de los arcos de salida de ese nodo.
- DIR_ARCO_ANTM. Nos permite tener acceso a la lista de los arcos de llegada de ese nodo.
- DIR_SIGUIENTE. Da acceso al siguiente nodo de la lista ordenada.

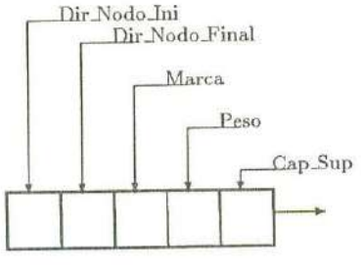
El nombre asignado al inicio de la lista de nodos de la red marginal es NODO_MARGINAL.



. Estructura ARCO_SUCM {
DIR_NODO_INI
DIR_NODO_FINAL
MARCA
PESO
CAP_SUP
DIR_SIGUIENTE
}

En la estructura ARCO_SUCM se guarda la información de los arcos de salida de la red marginal (Arcos Sucesores), donde cada elemento representa lo siguiente:

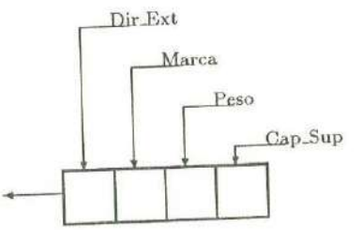
- DIR_NODO_INI. Nodo antecesor en la ruta dada por Floyd.
- DIR_NODO_FINAL. Nodo de llegada del arco.
- MARCA. Indica si pertenece al conjunto A_1 o a A_2.
  - Vale 1 si pertenece a A_1.
  - Vale 2 si está en A_2.
- PESO. Corresponde al costo del arco por unidad de flujo.
- CAP_SUP. Capacidad máxima de flujo que puede circular por el arco.
- DIR_SIGUIENTE. Da acceso al siguiente arco de la lista ordenada.



```
Estructura  ARCO_ANTM {
              DIR_EXT
              MARCA
              PESO
              CAP_SUP
              DIR_ANT
            }
```

En la estructura ARCO_ANTM se guarda la información de los arcos de llegada de la red marginal (Arcos Antecesores), donde cada elemento representa lo siguiente:

- DIR_EXT. Nodo de salida del arco.
- MARCA. Indica si pertenece al conjunto A_1 o a A_2.
  - Vale 1 si pertenece a A_1.
  - Vale 2 si está en A_2
- PESO. Corresponde al costo del arco por unidad de flujo.
- CAP_SUP. Capacidad máxima de flujo que puede circular por el arco.
- DIR_ANT. Da acceso al anterior arco de la lista ordenada.



- Circuito Negativo

Para guardar el ciclo que regresa el algoritmo de Floyd sobre la red marginal, también se utiliza una estructura especial.

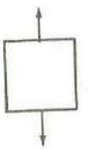
```
Estructura  CICLO {
              COMIENZO
              DIR_ANT
              DIR_SIGUIENTE
            }
```

En la estructura CICLO se guarda el nodo que se encuentra en el ciclo, donde cada elemento representa lo siguiente:

- COMIENZO. Nodo del ciclo.
- DIR_ANT. Acceso al anterior elemento del ciclo.

– DIR_SIGUIENTE. Elemento siguiente en el ciclo.

El nombre asignado al inicio y final de la lista del ciclo son CICLO_INI y CICLO_FINAL respectivamente.



El establecimiento de estas estructuras de datos es clave para la implementación, ya que tanto la red original, la red marginal y el circuito negativo se basan en estas unidades de información para construire de una manera fácil y de manejo eficiente y rápido de toda su información. El Algoritmo de Floyd aplicado a la red marginal está basado en éste tipo de estructuración de la red.

3.3.2 Procesamiento de la Información: Red Marginal y Circuito Negativo

El enlazar adecuadamente estas estructura de datos para crear las redes que se utilizarán es muy importante, ya que no basta poseer solamente la información de un nodo en particular, sino poder accesar libremente a la de otros nodos o arcos que también componen la red.

A continuación se muestra la representación de la red original del Ejemplo 1 que servirá como base para construir la red marginal asociada al flujo definido en ella, cabe aclarar que el valor de las cotas inferiores no se esncuentran especificados ya que todos poseen la cota inferior igual a cero.

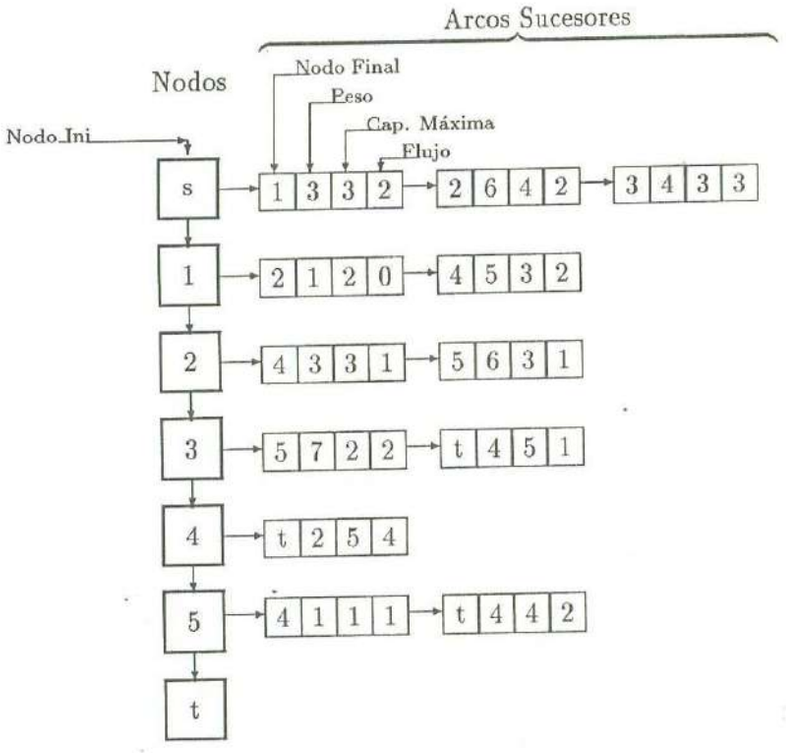


FIG.3.1 Representación de la Red Original
mediante Estructuras de Datos

- Red Marginal

La red marginal se crea a partir de la red original ya creada mediante la función denominada CREA_RED_MARGINAL(), y se realiza de la siguiente manera:

- Se va recorriendo sobre cada nodo de la red original, supongamos que se está en el nodo i.
- Se realiza también un recorrido sobre los arcos sucesores que salen de ese nodo, tomemos como nodo final del arco el nodo j.
- Se revisa la información de ese arco (i, j)
  - * Si Cap_Inf $<$ Flujo $<$ Cap_Sup, se agregan dos arcos en la red marginal.
    - El arco $(i, j) \in A_1$ con Cap_Sup = Cap_Sup - Flujo y Costo = Costo.
    - El arco $(j, i) \in A_2$ con Cap_Sup = Flujo - Cap_Inf y Costo = -Costo.
  - * Si Cap_Inf $<$ Flujo = Cap_Sup, se agrega un solo arco.
    - El arco $(j, i) \in A_2$ con Cap_Sup = Flujo- Cap_Inf y Costo = -Costo
  - * Si Flujo = Cap_Inf, se agrega un arco.
    - $(i, j) \in A_1$ con Cap_Sup = Cap_Sup - Flujo y Costo = Costo.
- Se continua el recorrido sobre los arcos sucesores del nodo hasta no haber.

– Se prosigue el recorrido sobre los nodos hasta agotarlos.

Cada que se agrega un arco en la red marginal, el proceso es similar alagregar arcos en la red original.

A continuación la distribución de la red marginal del Ejemplo 1 en la primera iteración con estructuras de datos.

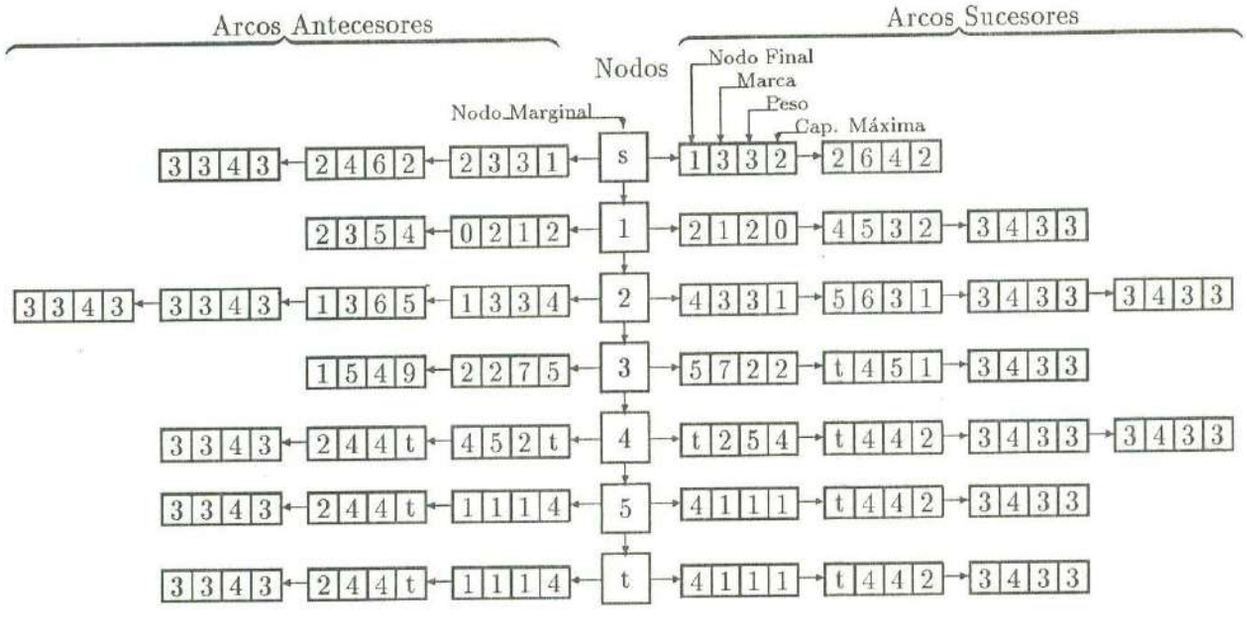


FIG.3.2 Red Marginal representada con Estructuras de Datos

- Circuito Negativo

Cuando se aplica el algoritmo de Floyd a la red marginal, y se encuentra en ella algún circuito negativo, el algoritmo de rutas mas corta crea una lista doblemente enlazada donde va guardando la información de los nodos que forman dicho circuito. La lista se va creando de la siguiente manera: En alguna iteración de Floyd se detecta un circuito de costo negativo que contiene a la ruta del nodo i al nodo j.

- Se busca si en el ciclo existe el arco en direcciôm (i, j).
  - * De haber existido dicho arco.
    1. Se llena la información de la primera estructura CICLO (CICLO_INI) con la información del nodo j.
    2. Se apunta CICLO_FINAL a esta estructura.
    3. Se recorre sobre la ruta encontrando el siguiente nodo k del ciclo.

4. Se liga una nueva estructura CICLO con la información de k al final de la lista doblemente enlazada.
5. Se reasigna la dirección de CICLO_FINAL.
6. Se regresa a (3) hasta encontrar el nodo i y agregarlo.

* De no haber encontrado arco directo de i a j se realizan los pasos a partir de (1) para el nodo i hasta encontrar j.

La figura siguiente nos muestra como se representa a la estructura del ciclo mediante Estructuras de datos. La información es del Ejemplo 1 con el ciclo detectado en la primera iteración.

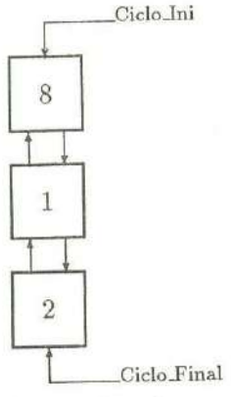


FIG.3.3 Diagrama del Circuito Negativo representado con Estructuras de Datos

3.3.3 Desarrollo del Algoritmo

Para que la descripción computacional del algoritmo que de mas clara, se desglosará en partes,que codificado significa utilizar funciones que laboren partes del programa, esto facilita la lectura de los códigos y esclarece el desarrollo general del programa.

La temática general del algoritmo se describe a continuación, el nombre de las funciones utilizadas en el código se encuentra entre corchetes.

Habiendo sido creada la red original, a partir de los datos leidos del archivo del problema previamente verificados, se itera de la siguiente manera:

- Mientras se detecten circuitos de costo negativo en la red marginal, se realiza lo siguiente:

1. Se crea la red marginal asociada al flujo actual en la red. [CREA_RED_MARGINAL()]
2. Se aplica el algoritmo de Floyd a la red marginal ya construida. [FLOYD()]
3. Se revisa si se detectó un circuito negativo.

- De haberse encontrado circuito, se continua con el paso 4.
- Si no hay circuito negativo, termina la iteración y se escribe la solución en un archivo. [GRABA_SOLUCION()]

4. Se Calcula la cantidad d máxima de flujo que se puede modificar a través del circuito. [CALCULA_D()]
5. Se modifica el flujo del circuito en la cantidad antes determinada. [MODIFICA_FLUJO()]
6. Se libera la memoria utilizada en crear la red marginal y el circuito negativo, a fin de garantizar memoria para crearlos en la siguiente iteración. [LIBERA_MEMORIA()]

El análisis de cada una de las funciones que actúan en cada parte del algoritmo se analizarán en el siguiente módulo. Cabe aclarar que la implementación que se utilizó del Algoritmo de Floyd no es la más eficaz, ya que en el desarrollo de la implementación se necesitan agregar arcos a la red marginal y con una revisión de ellos se puede omitir agregar algunos, en general, bajo algunos cambios la implementación de los algoritmos pueden hacerse más eficientes.

3.3.4 Modificación de Flujo

Lo referente a la modificación del flujo en la red, habiéndo encontrado un circuito negativo en la red marginal por Floyd, se divide en dos partes; la primera consiste en determinar la cantidad d que es aproiada para modifical el flujo de tal manera que provoque la máxima cantidad que es posible bajar el costo, la segunda parte es modificar el flujo en la red original, en los arcos que se encuentran en el circuito negativo.

Los pasos que se siguen en estas dos etapas son las siguientes, empezando por describir la primera parte [CALCULA_D()].

- A partir de la estructura del ciclo, se van leyéndo los arcos que se encuentran en él.
- Se accesa a la información del arco en la red marginal, para obtener el valor de la capacidad Superior de flujo.
- Finalmente, después de comparar las capacidades de los arcos del circuito, d es el valor del menor de esas capacidades.

La cantidad d es la apropiada para disminuir lo más posible el costo de la red, modificando el flujo, manteniendo el mismo valor deseado de flujo.

La modificación del flujo en la red original es la siguiente [MODIFICA_FLUJO()]:

- Se localiza el arco en la red marginal identificado como en el circuito.
- Se detecta si el arco es de aumento o disminución de flujo (es decir si pertenece a A_1 o a A_2).

- Se modifica el flujo como sigue:
  - Si f_{ij} simboliza el flujo actual de él y éste arco es de aumento se modifica $f_{ij} = f_{ij} + d$.
  - Si el arco era de disminución de flujo, queda $f_{ij} = f_{ij} - d$.
- Se avanza sobre el siguiente arco del ciclo hasta agotarlos.

De Esta manera se tiene la red con un flujo nuevo definido sobre ella, se calcula la red marginal asociada al nuevo flujo y se empieza una nueva iteración hasta que no se localicen circuitos negativos en las redes marginales, en ese momento se está en el óptimo y el flujo definido en la iteración anterior será el que dá el costo menor para ese valor de flujo.

Esto es a grandes rasgos la descripción de la implementación realizada del Algoritmo basado en eliminación de circuitos negativos.

En la siguiente sección se presenta la corridas de escritorio para el Ejemplo 1 resuelto con visualizando la red y con las estructuras de datos para poder comparar.

3.4 Corrida de Escritorio

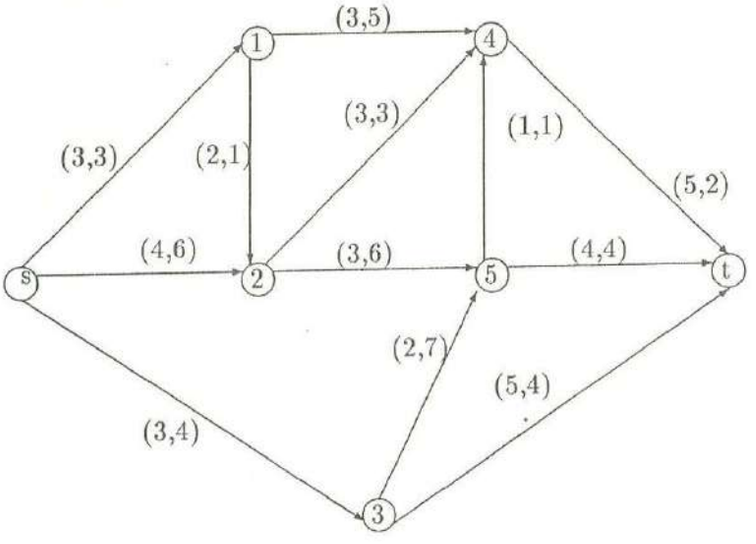
En adelante tomaremos la convención de que los recuadros de los arcos que se encuentren en línea cortada corresponden a arcos que el algoritmo de Floyd agrega a la gáfica y sirven para asignar la ruta más corta, o como es el caso, el circuito negativo. Los datos de las estructuras que se encuentren tachados y aparece el nuevo valor arriba, simbolizan cambio de peso de una ruta, este cambio también lo efectúa el Floyd al momento de iterar, recordemos que las marcas en los arcos nos indican si son arcos de aumento o de disminución de flujo, 1 si se puede incrementar y 2 lo contrario. Los nodos que son antecesores de las rutas mínimas detectadas por floyd se señalan en la parte de arriba del número de nodo en los arcos sucesores, si en alguno no aparece indicado significa que el antecesor de la ruta es el nodo del que salen dichos arcos. Al haberse aplicado el algoritmo de Floyd, éste detecta un circuito negativo en la revisión que efectúa sobre los arcos, los arcos que se encuentren encerrados en ovalos son aquellos donde fue detectado el circuito negativo, para extraer el circuito se hacen las siguientes revisiones:

- Se revisa si el arco sucesor (Nf) encerrado en ovalo tiene un nodo sucesor distinto del nodo inicial (N) del arco.
  - Si es así el ciclo se reconstruye con los nodos antecesores de la ruta de N a Nf.
  - En caso contrario el ciclo se busca sobre los antecesores de la ruta Nf a N.

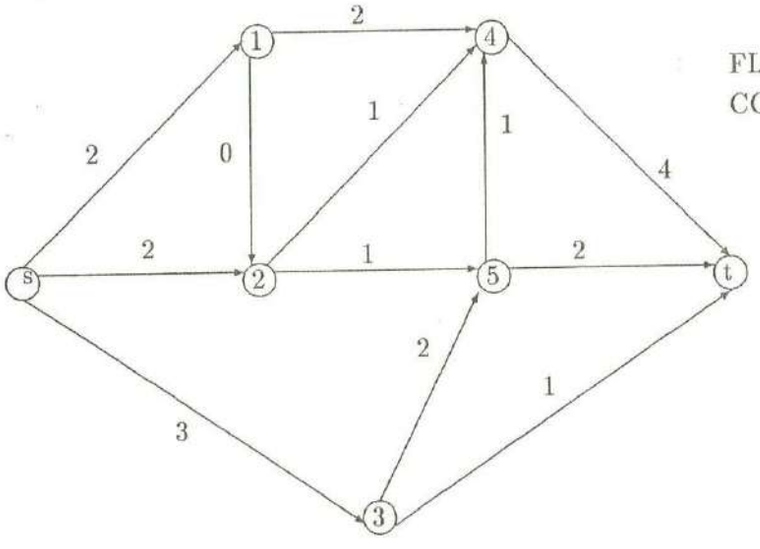
Los arcos representados con línea contínua pertenecen a la red marginal asociada a ese flujo.

3.4.1 Ejemplo 1

Determinemos el flujo a costo mínimo de valor 7 en la red de la ***Figura2.1*** mediante el algoritmo basado en eliminación de circuitos negativos.

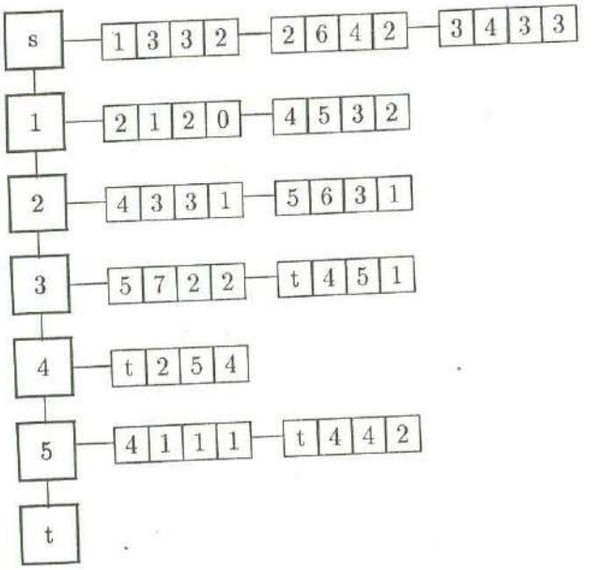


ITERACIÓN 1. Utilizando la red de la ***Figura2.2***, donde nos da una distribución del flujo de valor 7 y de costo 84.

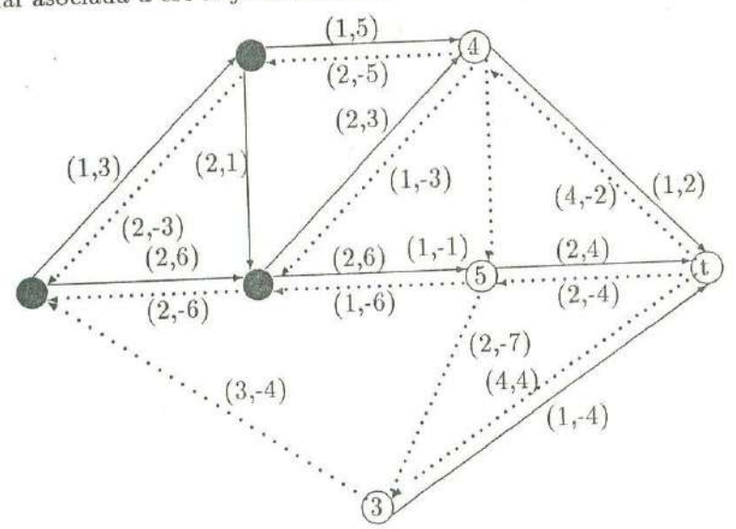


FLUJO = 7
COSTO = 84

La figura siguiente nos establece a la red del poblema mediante estructuras de datos



La red marginal asociada a ese flujo la retomamos de la ***Figura2.3***, y es:

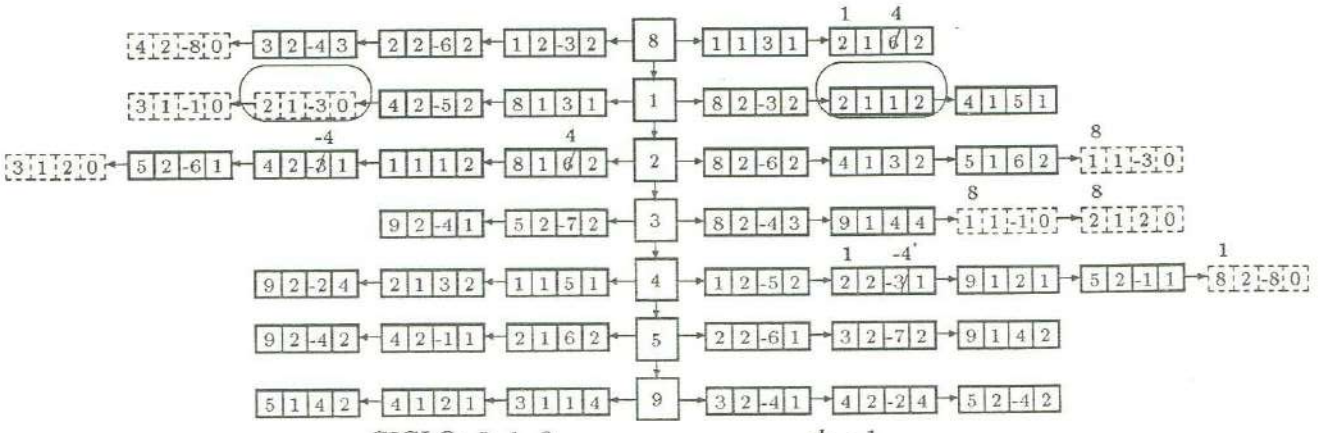


Identificando el ciclo **8, 1, 2** de costo **−2**, tenemos que:

$$d = \min\{q'_{81}, q'_{12}, q'_{28}\} = \{1, 2, 2\} = 1$$

la ***fig.*** siguiente muestra la red con estructuras de datos después de aplicar Floyd.

1a. ITERACION

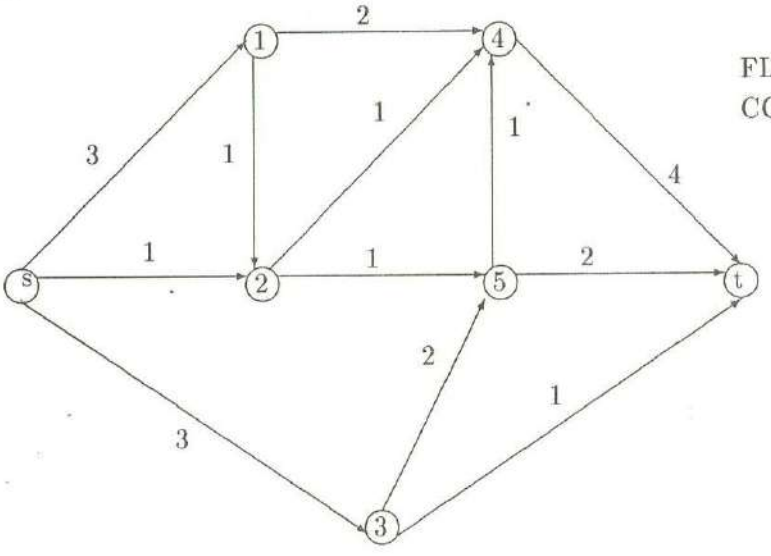


CICLO: 8, 1, 2 $\qquad$ $d = 1$

entonces

$$f_{s1} = f_{s1} + 1 = 2 + 1 = 3$$
$$f_{12} = f_{12} + 1 = 0 + 2 = 2$$
$$f_{s2} = f_{s2} - 1 = 1 - 1 = 0$$

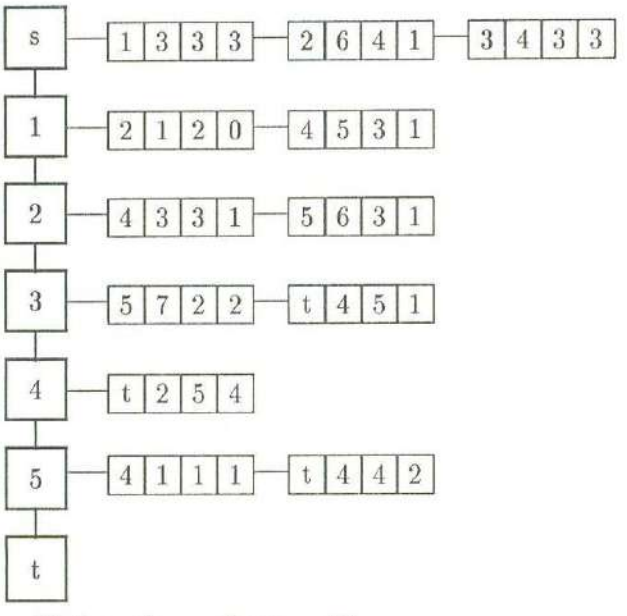
donde la red queda:



FLUJO = 7
COSTO = 82

FIG. 3.4 Nueva red después de la 1a. iteración

Realizando la actualización del flujo en la red original con estructuras de datos, nos queda:



Flujo = 7 Costo = 82

ITERACIÓN 2. Utilizando la red de la iteración anterior y encontrando la red marginal asociada a ella, tenemos:

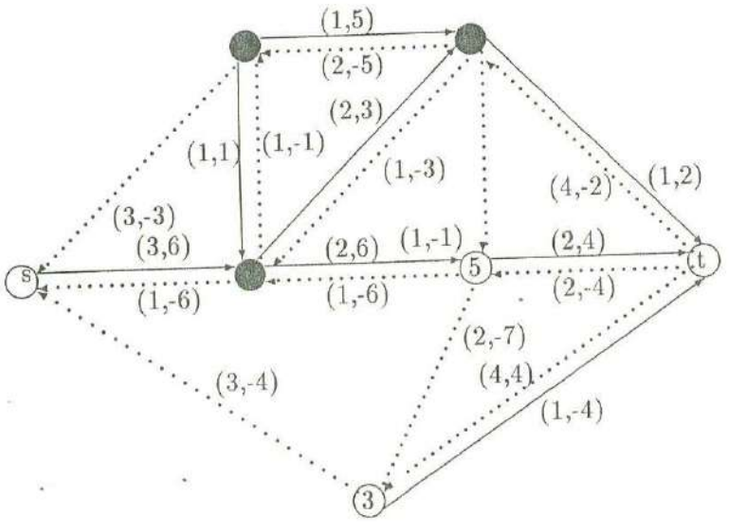


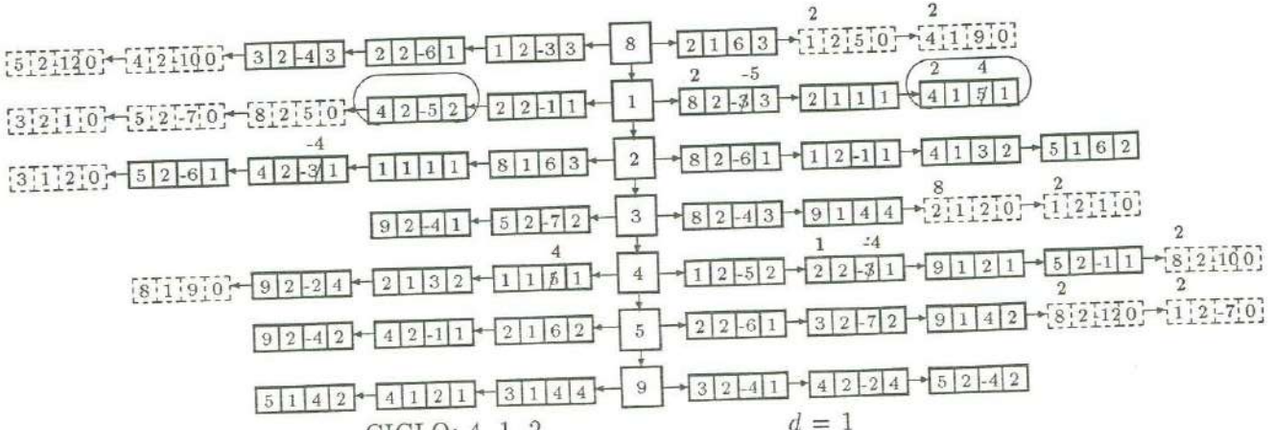
FIG. 3.5 Red marginal asociada al nuevo flujo de la iteración anterior

Identificando el ciclo 1, 2, 4 de costo -1, tenemos que:

$$d = \min\{q'_{12}, q'_{24}, q'_{41}\} = \{1, 2, 2,\} = 1$$

con red marginal después de aplicar el algoritmo de Floyd en estructuras de datos:

2a. ITERACION

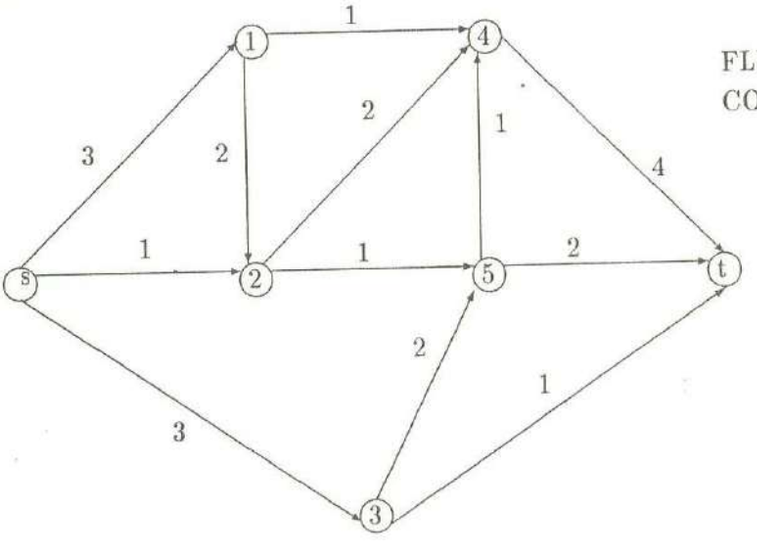


CICLO: 4, 1, 2 $\qquad$ $d = 1$

entonces

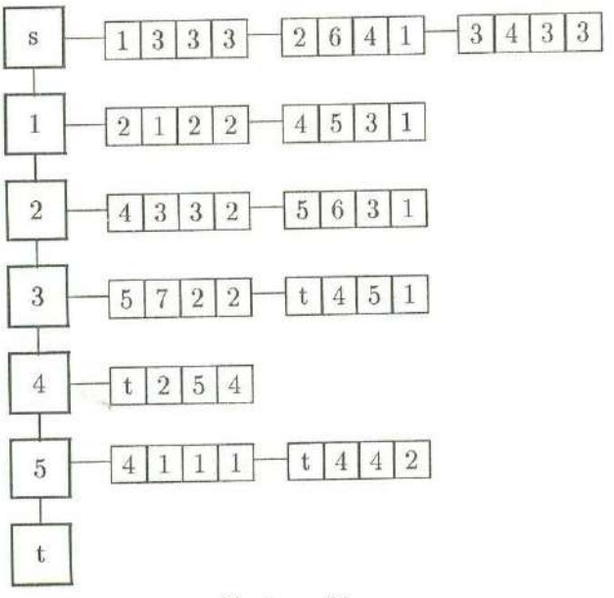
$$
\begin{aligned}
f_{12} &= f_{12} + 1 = 1 + 1 = 2 \\
f_{24} &= f_{24} + 1 = 1 + 1 = 2 \\
f_{14} &= f_{14} - 1 = 2 - 1 = 1
\end{aligned}
$$

donde la red queda:



FLUJO = 7
COSTO = 81

El flujo actualizado se presenta en la siguiente estructura de datos de la red original.



Flujo = 7 Costo = 81

ITERACIÓN 3. Nuevamente retomando la red de la iteración anterior y encontrando la red marginal asociada a ella, tenemos:

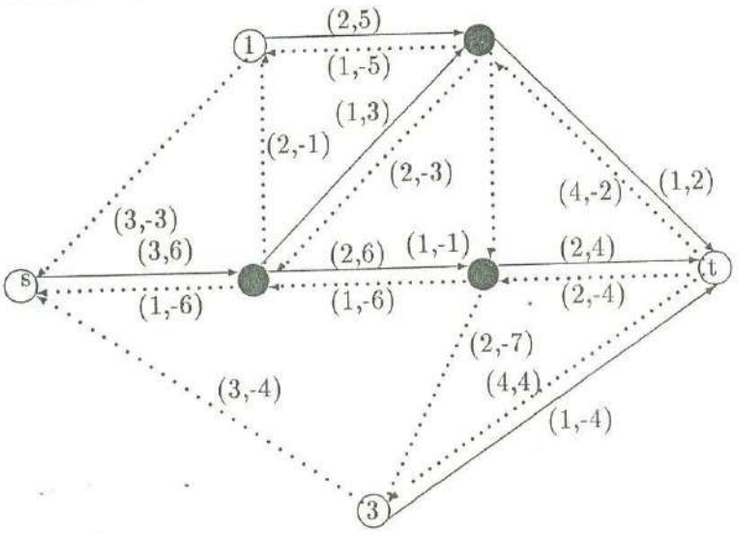
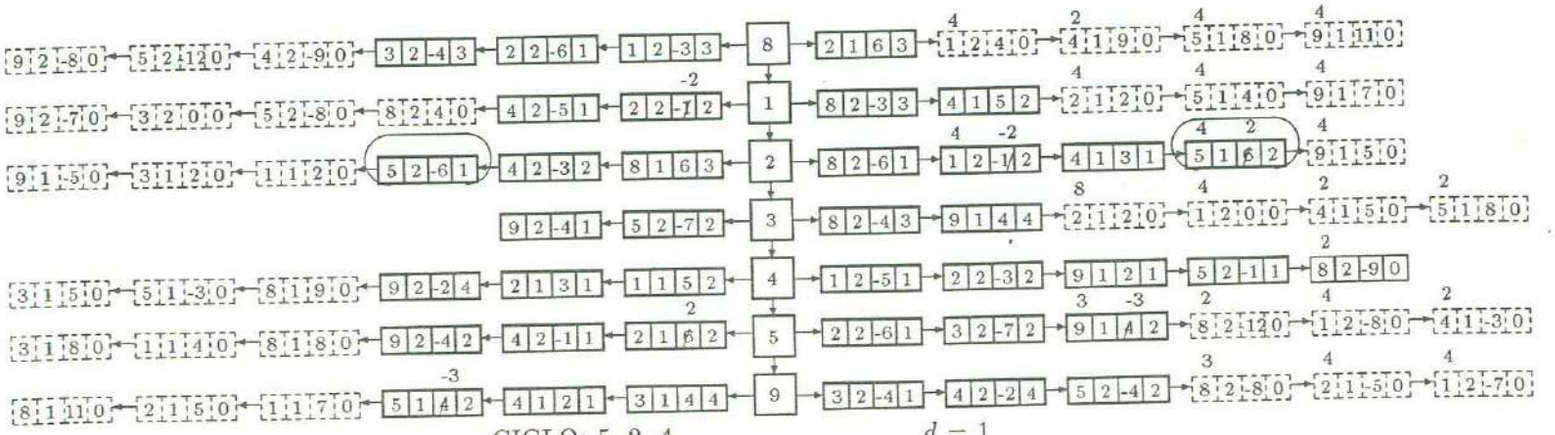


FIG. 3.6 Red marginal asociada al flujo de la iteración anterior

Identificando el ciclo **2, 4, 5** de costo -4, tenemos que:

$$d = \min\{q'_{24}, q'_{45}, q'_{52}\} = \{1, 1, 1\} = 1$$

3a. ITERACION

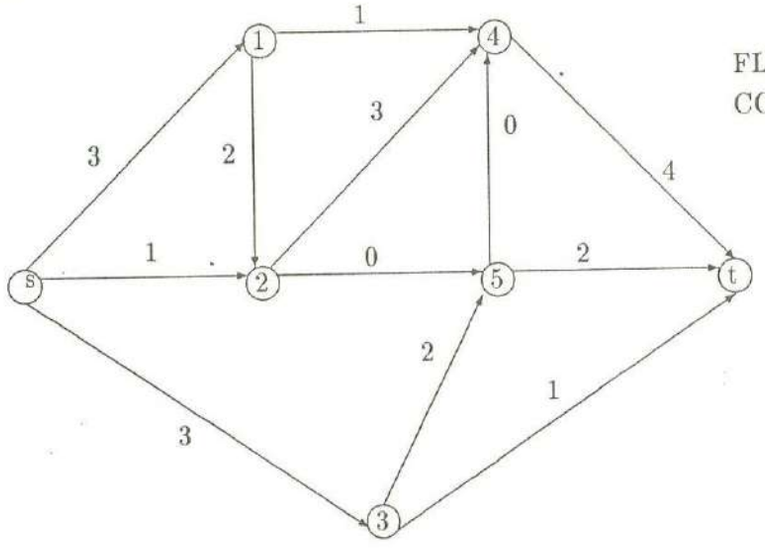


CICLO: 5, 2, 4 $\quad d = 1$

entonces

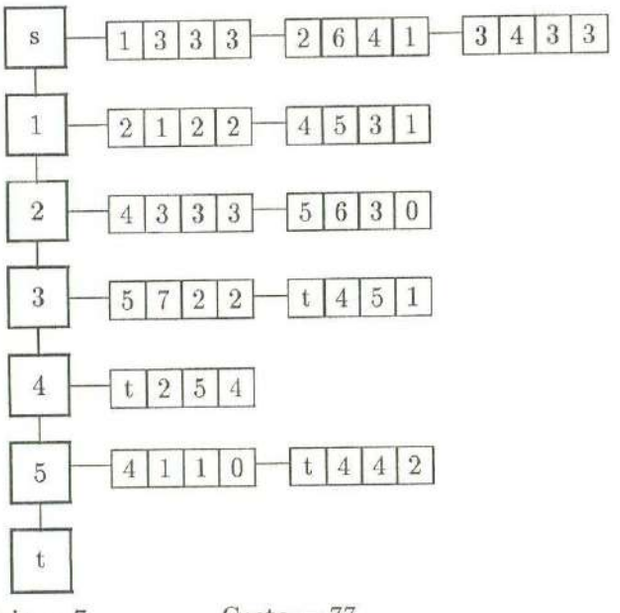
$$
\begin{aligned}
f_{24} &= f_{24} + 1 = 2 + 1 = 3 \\
f_{54} &= f_{54} - 1 = 1 - 1 = 0 \\
f_{25} &= f_{25} - 1 = 1 - 1 = 0
\end{aligned}
$$

donde la red queda:



FLUJO = 7
COSTO = 77

FIG. 3.7 Red de flujo después de la iteración 3



Flujo = 7 Costo = 77

ITERACIÓN 4. Con la red anterior y encontrando la red marginal asociada a ella, tenemos:

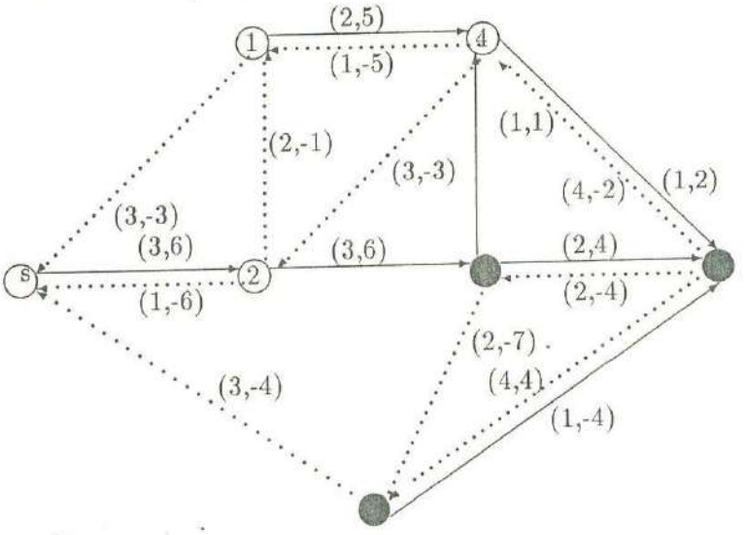
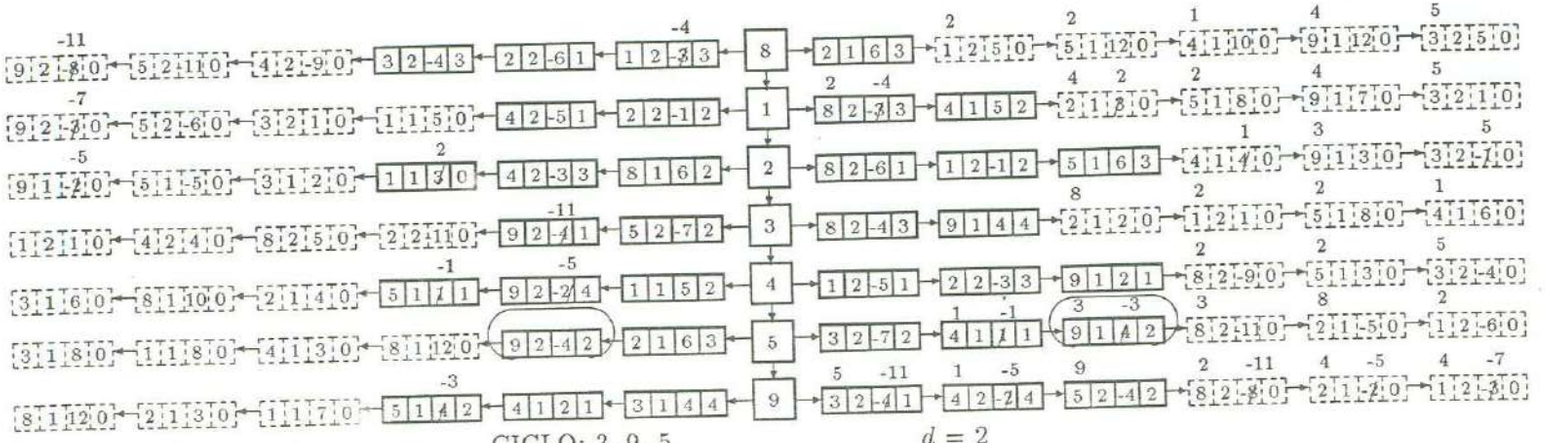


FIG. 3.8 Red Marginal de la distribución anterior de flujo

Identificando el ciclo $5, 3, t$ de costo -7, tenemos que:

$$d = \min\{q'_{53}, q'_{3t}, q'_{t5}\} = \{2, 4, 2\} = 2$$

4a. ITERACION

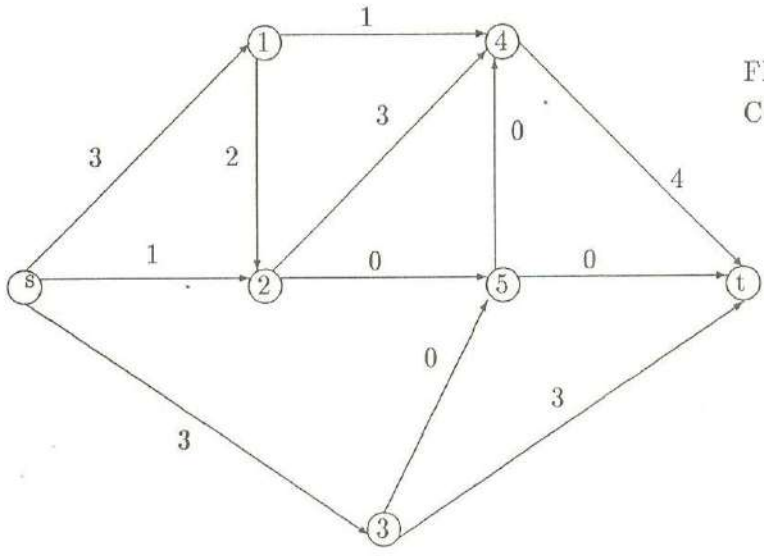


CICLO: 3, 9, 5 $d = 2$

entonces

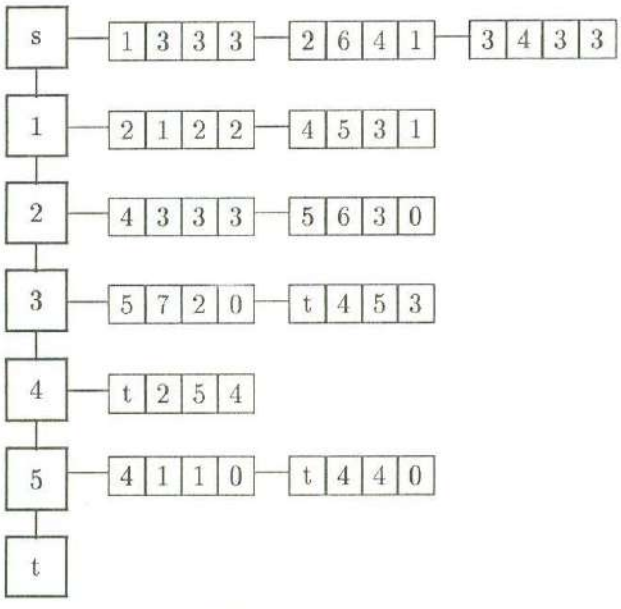
$$
\begin{aligned}
f_{35} &= f_{35} - 2 = 2 - 2 = 0 \\
f_{3t} &= f_{3t} + 2 = 1 + 2 = 2 \\
f_{5t} &= f_{5t} - 2 = 2 - 2 = 0
\end{aligned}
$$

donde la red queda:



FLUJO = 7
COSTO = 63

FIG. 3.9 Red de flujo después de la iteración 3



Flujo = 7 Costo = 63

ITERACIÓN 5. Con la red anterior y encontrando la red marginal asociada a ella, tenemos:

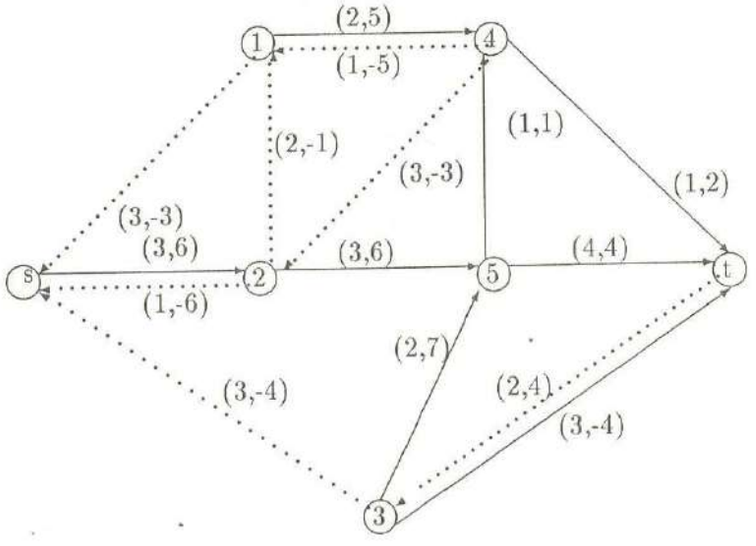
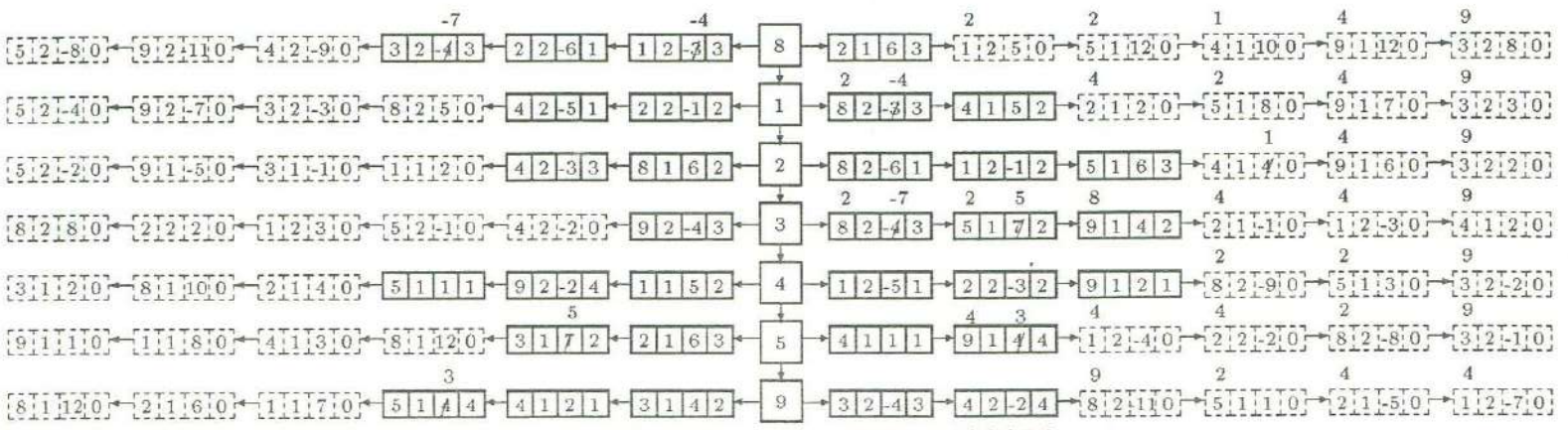


FIG. 3.10 Red Marginal de la distribución anterior de flujo

La siguiente reds marginal nos muestra la ausencia de circuitos negativos después de aplicar el algoritmo de Floyd.

5a. ITERACION



FLUJO = 7 COSTO = 63

Donde en ésta red no se encuentran circuitos negativos, por lo que la distribución anterior de flujos es la de mínimo costo. Obteniendo entonces un costo de **63** para un flujo de valor **7**.

Comparándo los valores obtenidos con la solución a mano y mediante la implementación se observa que da el mismo costo para el valor deseado de flujo.

Con la corrida de escritorio anterior se puede ver que para problemas con mayor número de nodos y arcos el aplicar el algoritmo a mano puede ser muy latoso y requiere tiempo, es por eso que es necesario formular computacionalmente el algoritmo.

En el capítulo siguiente se analizará el otro algoritmo que también resuelve el problema, éste, como ya se ha comentado, no requiere tener el valor de un flujo inicial si las cotas inferiores para todos los arcos es cero y se determinan rutas mas cortas en la red marginal, esto quiere decir que para su implementación no es posible auxiliarse del algoritmo de Floyd, el apropiado es el Algoritmo General de Dijkstra que calcula rutas mas cortas entre un par de vértices teniendo costos negativos en la digráfica. La implementación de este algoritmo no será igual que el aquí presentado, las estructuras utilizadas serán diferentes y de acorde a las necesidades.

Capítulo 4

Algoritmo basado en detección Rutas mas Cortas

En este capítulo se presentará un segundo algoritmo que también resuelve el problema de asignar un flujo de cierto valor en una red, tal que sea de costo mínimo. A diferencia del método anterior éste necesita de un flujo óptimo de valor menor al deseado y en cada iteración lo que hace es aumentar el flujo manteniendo la optimalidad.

La elección de cualesquiera de los dos métodos de solución dependerá de la naturaleza del problema, ya que algunas veces es más fácil encontrar un flujo factible de valor deseado, o bien uno de menor valor sabiendo que es de costo mínimo.

Para garantizar la efectividad del procedimiento basado en la determinación de rutas mas cortas en la red marginal se establecerá un teorema cuya demostración ilustra los pasos a seguir también en este capítulo se mostrará la corrida de escritorio para el ejemplo 1 resuelto anteriormente por el algoritmo de eliminación de circuitos negativos, estableciéndo los pasos aplicados en la red como con la estructura de datos que se utlilizó en la implementación computacional.

4.1 Condición de Optimalidad

El algoritmo del Capítulo anterior se basa en que dado un flujo f de valor v, se modifica el flujo sin cambiar su valor hasta tener la optimalidad. A este método se le conoce generalmente como el problema ***primal***, ya que conservando las condiciones originales se llega al costo mínimo. Existe otra alternativa de resolver el problema mediante el método ***dual*** que en cambio construye un flujo que sea óptimo de valor v empezando con un flujo de costo mimo $v_o < v$ e ir aumentando flujo a la red conservando la optimalidad hasta llegar al valor v del flujo. Generalmente el valor inicial de flujo que se toma es cero que es un flujo de costo mínimo, ya que la red marginal asociada a este flujo es igual a la red original.

El algoritmo está basado en el cálculo de rutas mas cortas en la red marginal y sobre la cadena

aumentante asociada a esta ruta incrementar unidades de flujo. La idea de la demostración es verificar que con este nuevo valor de flujo definido en la red sigue siendo de costo mínimo, viendo que no existen circuitos de costo negativo en la red marginal asociada a este nuevo flujo, de esta manera se tiene un flujo óptimo de mayor valor que el inicial, continuando con este procedimiento se aumenta flujo hasta llegar al valor deseado.

Teorema. Sea f un flujo de costo mínimo de valor v en una red R y P^* es la ruta mas corta (de costo mínimo) de s a t en la red marginal $R_f(f)$ y sea P la cadena aumentante en la red R correspondiente a P^*. Entonces $f + 1_0(P^*)$ es el flujo de costo mínimo de valor $v + 1$.

Demostración.

Claramente el flujo $f + 1_0(P^*)$ es de valor $v + 1$, ya que se aumenta una unidad de flujo a través de la cadena P^*.

Por otro lado, puesto que f es de costo mínimo, $R_f(f)$ no contiene circuitos negativos. Para demostrar que el flujo $f + 1_0(P^*)$ también es de costo mínimo, basta probar que la red marginal $R_f(f + 1_0(P^*))$ tampoco posee circuitos negativos.

Las redes marginales $R_f(f)$ y $R_f(f + 1_0(P^*))$ coinciden excepto, posiblemente, en los arcos de P^*. Consideremos la partición, de los arcos de P, en los conjuntos N y $X - N$, donde

- $N = \{(i,j) \mid f_{ij} < q_{ij}\}$
- $X - N = \{(i,j) \mid f_{ij} > 0\}$

Si $f_{ij} + 1_0(P^*) = f_{ij} + 1 < q_{ij}$, para todo $(i,j) \in N$ y $f_{ij} + 1_0(P^*) = f_{ij} - 1 > 0$, para todo $(i,j) \in X - N$, es claro que $R_f(f)$ y $R_f(f + 1_0(P^*))$ contienen los mismos arcos y por lo tanto $R_f(f + 1_0(P^*))$ no contiene circuitos negativos en este caso.

Ahora, si existe algún $(i,j) \in N$ tal que $f_{ij} + 1_0(P^*) = f_{ij} + 1 = q_{ij}$ entonces este arco, que pertenece a $R_f(f)$, no pertenece a $R_f(f + 1_0(P^*))$ y el arco (j,i) pertenece a esta última red. Supóngase que este arco pertenece a un circuito Φ de $R_f(f + 1_0(P^*))$, y sean los vértices de Φ : $i, 1, 2, \ldots, k, j, i$..

Puesto que P^* es una ruta mas corta de s a t en $R_f(f + 1_0(P^*))$, entonces el arco (i,j) es una ruta mas corta de i a j ya que pertenece a P^*. Por lo tanto en la red $R_f(f)$ se tiene que $c_{ij} \geq c_{i1} + c_{12} + \ldots + c_{kj}$; luego $c_f(\Phi) = c_{ij} - c_{i1} + c_{12} + \ldots + c_{kj} - c_{ij} \geq 0$ que es el costo del circuito, donde c_{ij} es el costo del arco (i,j) y por lo tanto Φ no es un circuito negativo.

Análogamente se prueba que $R_f(f + 1_0(P^*))$ no contiene circuitos negativos si existe algún arco $(i,j) \in X - N$ tal que $f_{ij} + 1_0(P^*) = f_{ij} - 1 = 0$.

■

Como ya se mencionó el flujo f de valor $v = 0$ es factible y de costo mínimo si las cotas inferiores para todos los arcos es cero, para este caso el problema se facilita ya que no se tiene que buscar algún flujo factible óptimo, si hay cotas inferiores diferentes de cero se comienza con un flujo óptimo que cumpla con dichas restricciones de arcos, el siguiente paso es determinar la ruta mas corta de s a t, en la red marginal; de nuevo es posible utilizar el algoritmo general de Dijkstra para encontrar arborescencias de rutas mas cortas para después enviar la máxima cantidad posible de flujo de la cadena obtenida de s a t y por el teorema anterior el nuevo flujo será de costo mínimo, procediendo igualmente hasta alcanzar el valor de flujo deseado.

Caba señalar que, si en alguna iteración no existe ruta alguna de s a t en la red marginal y el valor $v\prime$ 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 çadenas aumentantes de s a t en la red original y por tanto f es flujo máximo.

Por otro lado, si $v\prime$ mas la capacidad incremental de la cadena encontrada es mayor que v entonces solo será necesario enviar la cantidad $v - v\prime$ a través de esa cadena para determinar el flujo deseado.

4.2 Descripción del Algoritmo

A continuación se establecen los pasos a seguir para aplicar el algoritmo, al igual que en el algoritmo anterior dichos pasos están sustentados por la demostración del teorema anterior o algunos otros resultados vistos en el capítulo dos.

Paso 1 [Obtención de un flujo factible]. Estableścase el flujo factible f de costo mínimo en R.

Paso 2 [Red Marginal]. Constrúyase la red marginal, con respecto a f, $R\prime(f)$.

Paso 3 [Ruta mas corta de s a t en $R\prime(f)$]. Determínese la ruta mas corta P^* de s a t en $R\prime(f)$

Paso 4 [Actualización del flujo]. Sea $d = \min_{(i,j)\in P^*}\{q\prime_{ij}\}$

(i) Si el valor del flujo $f + d \circ (P^*)$ es igual a v, actualizar $f = f + d \circ (P^*)$ y terminar. En este caso f es el flujo a costo mínimo de valor v.

(ii) Si el valor del flujo $f + d \circ (P^*)$ es igual a v, actualizar $f = f + d \circ (P^*)$ y terminar. En este caso f es el flujo a costo mínimo de valor v.

(iii) Si el valor del flujo $f + d \circ (P^*)$ es mayor que v y $v\prime$ es el valor de f, actualizar $f = f + (v - v\prime) \circ (P^*)$ y terminar ya que f es el flujo requerido.

4.3 Implementación Computacional del Algoritmo

Como ya se ha comentado anteriormente es muy importante tener una buena implementación del algoritmo, y para ello se hizo uso de estructuras de datos para almacenar la información y poder

accesar a ella de una manera rápida. Al igual que en la implementación del algoritmo de circuitos negativos se crea la red marginal en una estructura aparte de la red original, para detectar la ruta mas corta entre el nodo fuente s y destino t se utiliza el algoritmo Generalizado de Dijkstra, ya que en la red marginal se pueden tener pesos negativos y éste algoritmo encuentra la arborescencia mínima aún habiendo costos negativos.

4.3.1 Manejo de la Información

Al igual que en el capítulo anterior en esta sección se definirán las estructuras que se utilizaron, cada una muestra la información que contiene, necesaria para el desarrollo del algoritmo en diversas etapas, la red original posee una lista ligada de nodos y en cada uno de ellos cuelga una lista de arcos, la forma en que se construye la red original es exactamente la misma que para el algoritmo anterior, solo se cambiaron algunos nombres por facilidad de manejo de variables. La red marginal se construye exactamente igual que en el capítulo anterior, es por eso que en esta sección se omitirá su descripción; para la recopilación de la ruta más corta encontrada por el Dijkstra se utiliza una lista doblemente ligada muy similar a la que se utilizó en el almacenamiento del circuito negativo del algoritmo anterior.

- Red Original

  Para crear la red original se definen las siguientes estructuras:

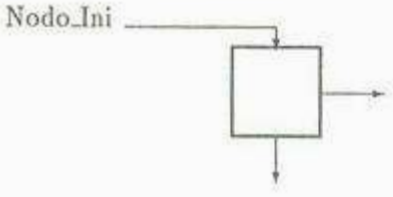
```
Estructura  NODO {
              NUMERO
              ARCO_INI
              SIG
            }
```

  En la estructura NODO se guarda la información de los nodos de la red, donde cada elemento representa lo siguiente:

  - NUMERO. Corresponde al número asignado al nodo en la red.
  - ARCO_INI. Nos permite tener acceso a la lista de los arcos de salida de ese nodo.
  - SIG. Da acceso al siguiente nodo de la lista ordenada.

  El nombre asignado al inicio de la lista de nodos de la red original es NODO_INI.

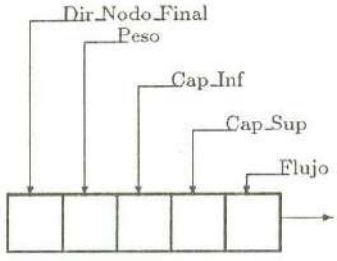
  La figura siguiente muestra un diagrama de la representación de un nodo mediante estructuras de datos.



Estructura ARCO {
NF
PESO
CAP_INF
CAP_SUP
FLUJO
SIG
}

En la estructura ARCO se guarda la información de los arcos de la red, donde cada elemento representa lo siguiente:

- NF. Nodo de llegada del arco.
- PESO. Corresponde al costo del arco por unidad de flujo.
- CAP_INF. Capacidad mínima de flujo que puede circular por el arco.
- CAP_SUP. Capacidad máxima de flujo que puede circular por el arco.
- FLUJO. Flujo que circula a través de ese arco.
- SIG. Da acceso al siguiente arco de la lista ordenada.



• Red Marginal

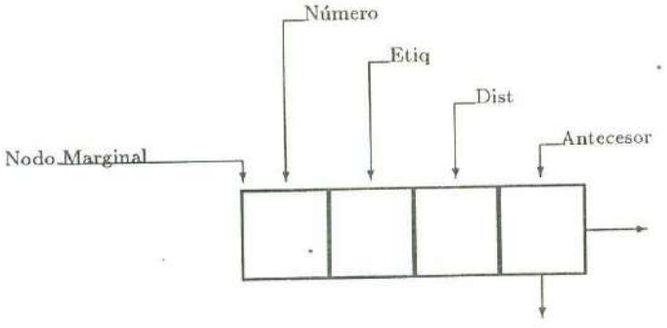
Para la red marginal definimos otro tipo de estructuras, adcuadas para poder aplicar el algoritmo General de Dijkstra y encontrar ciclos negativos.

Estructura NODOM {
NUMERO
ETIQ
DIST
ANTECESOR
ARCO_SUC
SIG
}

En la estructura NODOM se guarda la información de los nodos de la red marginal, donde cada elemento representa lo siguiente:

- NUMERO. Corresponde al número asignado al nodo en la red marginal.
- ETIQ. Etiqueta que sirve para la aplicación del Dijkstra.
- DIST. Distancia de la ruta mínima desde el nodo Fuente hasta el nodo en cuestión.
- ANTECESOR. Nodo antecesor en la ruta mínima desde el nodo fuente.
- ARCO_SUC. Acceso a la lista de arcos que salen del nodo.
- SIG. Apuntador al siguiente nodo de la lista.

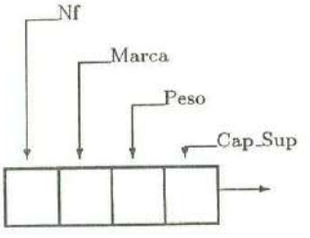
El nombre asignado al inicio de la lista de nodos de la red marginal es NODO_MARGINAL.




```
Estructura  ARCO_SUCM {
              NF
              MARCA
              PESO
              CAP_SUP
              SIG
            }
```

En la estructura ARCO_SUCM se guarda la información de los arcos de salida de la red marginal (ArcosSucesores), donde cada elemento representa lo siguiente:

- NF. Nodo de llegada del arco.
- MARCA. Indica si pertenece al conjunto A_1 o a A_2.
- PESO. Corresponde al costo del arco por unidad de flujo.
- CAP_SUP. Capacidad máxima de flujo que puede circular por el arco.
- SIG. Da acceso al siguiente arco de la lista ordenada.



- Ruta mas Corta

  Para guardar la ruta mas corta entre el nodo fuente s y el nodo destino t que regresa el algoritmo General de Dijkstra sobre la red marginal, también se utiliza una estructura especial.

```
Estructura  RUTA {
              COMIENZO
              ANT
              SIG
            }
```

  En la estructura RUTA se guarda el nodo que se encuentra en la ruta mas corta entre s y t, donde cada elemento representa lo siguiente:

  - COMIENZO. Nodo que se encuentra en la ruta.
  - ANT. Acceso al anterior elemento de la ruta.
  - SIG. Elemento siguiente en la ruta.

  El nombre asignado al inició y final de la lista de la ruta son RUTA_INI y RUTA_FINAL respectivamente.

Con las estructuras antes definidas se crean las redes original y marginal, así como una lista donde se guarda la información de los nodos que están en la ruta mínima entre s y t que regresa el algoritmo General de Dijkstra, la manera de enlazar estas estructuras se establecerá en la siguiente parte de ésta sección.

4.3.2 Procesamiento de la Información: Red Marginal y Ruta mas Corta

Como se había mencionado anteriormente es muy importante hacer un enlace correcto de las estructuras de datos, a fin de que el acceso a la información que se requiera sea muy fácil, la parte de la creación de la red marginal no se explica ya que es el mismo procedimiento visto en el capítulo anterior, salvo quizá algunos arreglos por la diferencia de estructuras utilizadas en los diferentes algoritmos.

- Red Marginal

  La creación de la red marginal a partir de la información de la red original se realiza mediante la función CREA_RED_MARGINAL(), ésta función es exactamente la misma que la presentada en el capítulo anterior, por lo que no se volverá a detallar la manera como se construye.

  A continuación se muestran la red original y marginal para el Ejemplo 1 en la primera iteración del algoritmo, a fin de ver como quedan establecidas con las estructuras y ligas adecuadas.

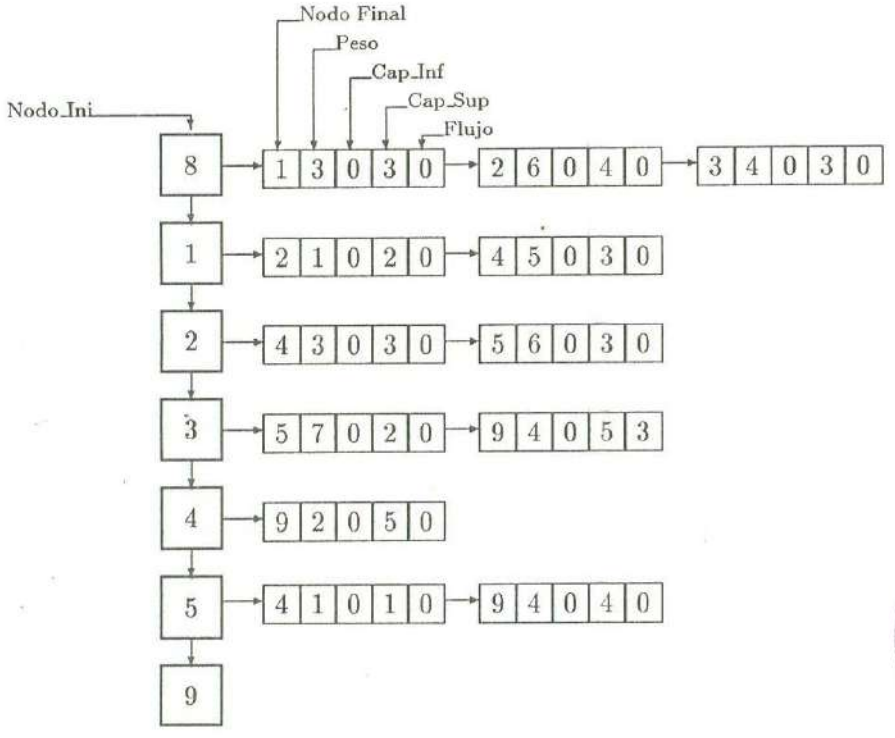


FIG. Red Original del Ejemplo 1 con un flujo igual a cero.

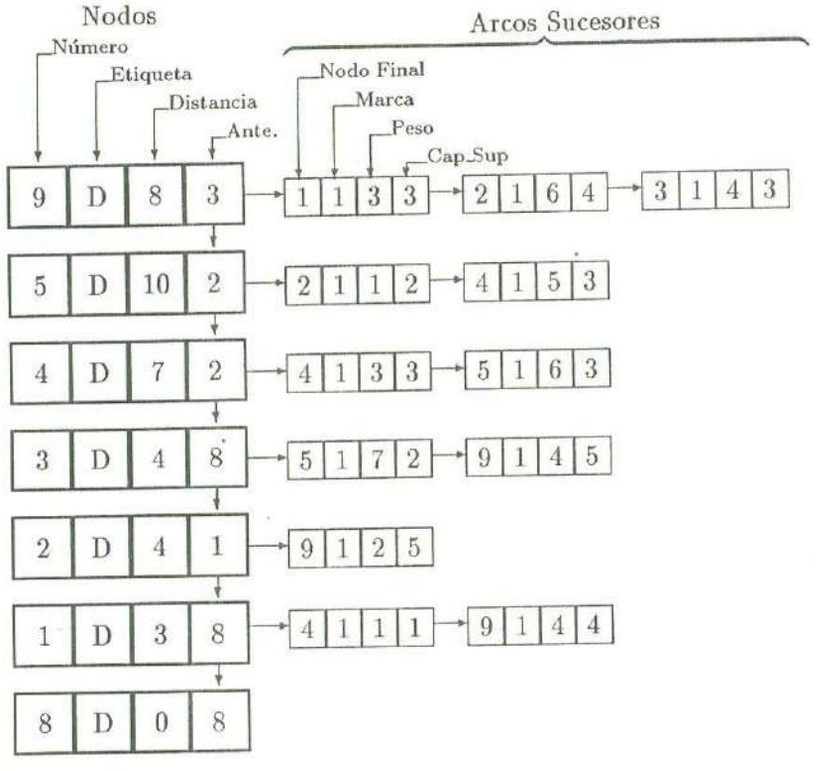


FIG. Red Marginal de la 1a. iteración del Ejemplo 1
representada con estructuras de datos.

- Ruta mas Corta

  Cuando se aplica el algoritmo Generalizado de Dijksta a la red maginal éste detecta la ruta mas corta entre el nodo fuente s y el nodo destino t, ésta información es guardada en una estructura doblemente ligada. La lista se va creando de la siguiente manera:

  1. Se busca en la lista de nodos al destino t.
  2. Se llena con la información del nodo la primera estructura RUTA (RUTA_INI).
  3. Se apunta RUTA_FINAL a esta estructura.
  4. Se toma el nodo k tal que es el nodo antecesor en la ruta del Destino.
  5. Se llena una estructura RUTA con la información de ése nodo y se efectuan las ligaduras agregando al final de la lista.
  6. Se apunta RUTA_FINAL a esta nueva estructura.
  7. Se toma el nuevo nodo que es antecesor en la ruta a k.
  8. Se va al paso 5 hasta encontrar el nodo Fuente.

De esta manera se tiene guardada la información de los nodos para poder utilizarse posteriormente en el desarrollo del algoritmo, en seguida se muestra la ruta detectada en la primera iteración del Ejemplo 1 con estructuras de datos.

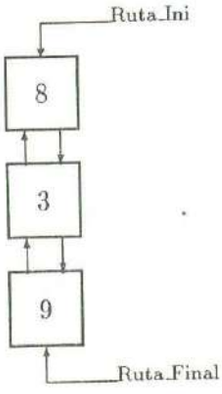


FIG. Diagrama de una Ruta representada
con Estructuras de Datos

4.3.3 Desarrollo del Algoritmo

El desglose del algoritmo, al igual que el anterior fue hecho mediante funciones, el desarrollo general del algoritmo se establece en esta parte, los nombres de las funciones utilizadas para efectuar tal labor se encuentran entre corchetes.

La información del problema por iterar es recogido de un archivo previamente preparado, a la captura de estos datos se va creando la red original de información verificando que verdaderamente el flujo dado sea factible bajo las restricciones de arcos establecidas, al valor de Flujo definido enla red le llamaremos Flujo_Total y al valor deseado de flujo Flujo_Deseado, de esta forma se comienza a iterar de la siguiente manera:

- Mientras el valor del Flujo_Total sea menor que el Flujo_Deseado se realiza lo siguiente:

  1. Se crea la red marginal asociada al flujo actual definido en la red. [CREA_RED_MARGINAL()]
  2. Se aplica el algoritmo Generalizado de Dijkstra a la red marginal ya creada.[DIJKSTRA_GENERAL()]
  3. Se revisa si se encontró la ruta mínima
     - De haberse encontrado ruta mínima se continua con el paso 4.
     - Si no existe ruta el Flujo_Total es máximo y se escribe la solución obtenida en un archivo.[GRABA_SOLUCIÓN]
  4. Se Calcula la cantidad d máxima de flujo que se puede aumentar a través de la ruta. [CALCULA_D]
  5. Se modifica el flujo a través de la ruta en la red original en la cantidad antes determinada bajo las siguientes condiciones. [MODIFICA_FLUJO()]

- Si la Flujo_Total + d < Flujo_Deseado. Continuar
- Si Flujo_Total + d = Flujo_Deseado. Parar y grabar la solución. [GRABA_SOLUCION()]
- Si Flujo_Total + d > Flujo_Deseado.
  Continuar con lamodificación de flujo para d = Flujo_Deseado - Flujo_Total.

6. Se libera la memoria utilizada en crear la red marginal y la lista de nodos en la ruta a fin de garantizar memoria disponible para la iteración posterior.[LIBERA_MEMORIA]

En esta parte se establecieron los rasgos generales del desarrollo del algoritmo, el estudio mas detallado de las labores que realiza cada una de las funciones antes mencionadas se explica en la siguiente parte de esta sección, para un análisis mas detallado de éstas y otras funciones que se uilizaron en la implementación del algoritmo ver el Anexo 2 que contiene los códigos fuetes de la implementación en Lenguaje C.

4.3.4 Modificación de Flujo

Cuando se ha detectado en la red marginal la uta de peso menor entre el nodo Fuente y el Destino se procede a hacer una modificación en el flujo definido en la red, este proceso se realiza en dos pasos principales, el primero de ellos consiste en determinar cual es la cantidad máxima que se puede aumentar de Flujo a través de la trayectoria, la segunda parte consiste en el cambio del flujo sobre la red original.

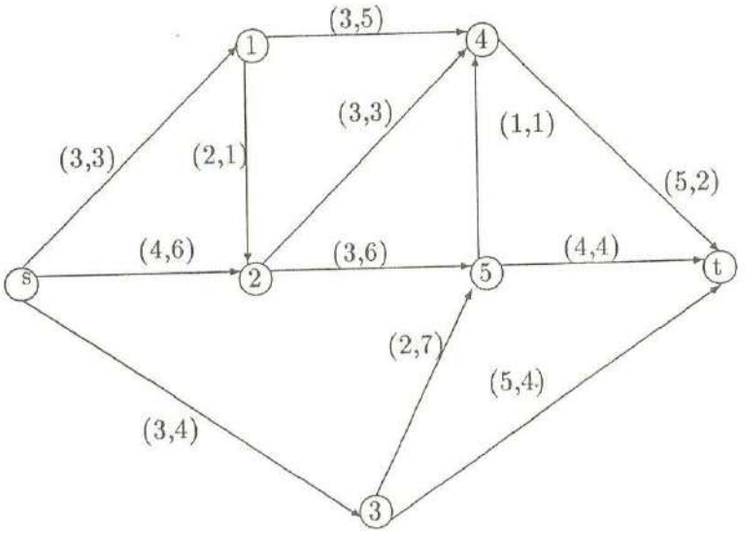
Los pasos a seguir en la siguiente parte se muestran a continuación, éstas labores las realiza la función CALCULA_D en la implementación.

- Con la lista que posee la ruta, se va leyendo los arcos que lo forman, ya que se tiene perfectamente determinados los extremos inicial y final de dichos arcos que pertenecen a la ruta mínima.
- Se accesa a la información del arco en cuestión en la red marginal para obtener el valor de la capacidad Superior de ése arco.
- Se realiza la misma búsqueda para todos los arcos de la red que se encuentran en la ruta.
- Después de analizar todos los valores de las capacidades superiores se toma como d a la menor de ellas.

La cantidad d encontrada es la apropiada para aumentar lo mas posible el flujo a través de la ruta mínima, ya que recordemos que esta ruta está asociada a una cadena incremental en la red original.

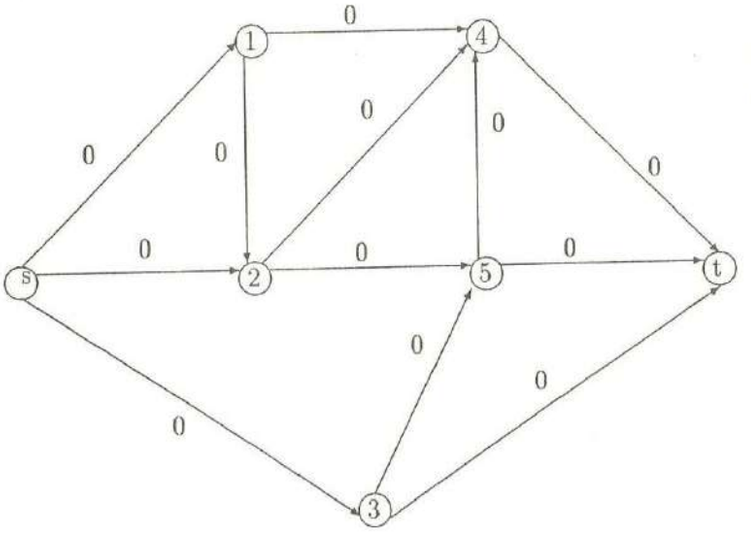
Para la segunda parte, ya sabiéndo la cantidad óptima de flujo por aumentar en la iteración se realiza el cambio de flujo en la red original mediante la función MODIFICA_FLUJO:

- Se localiza cada arco de la ruta en la red original.



Los valores en los arcos son (Capacidad, Costo)

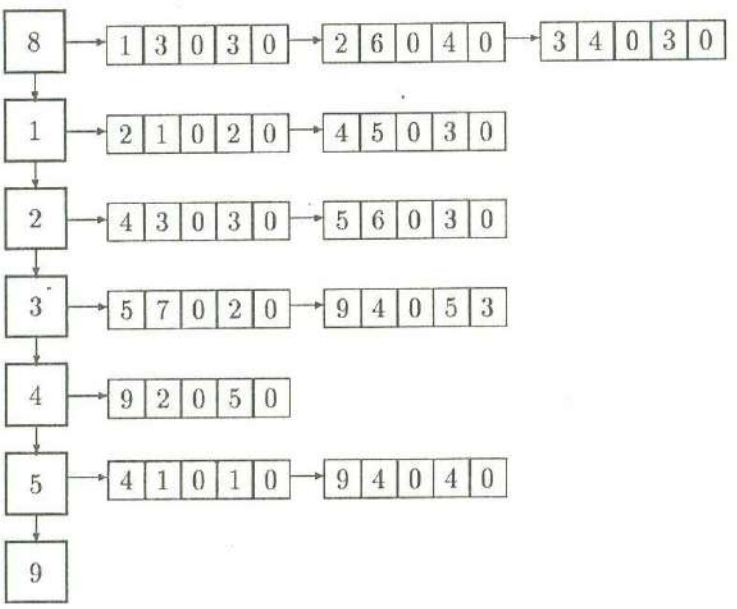
ITERACIÓN 1. Tomando el flujo de valor **o**, donde en cada arco el flujo es también **o**, vemos que es factible, y de costo mínimo ya que su red marginal asociada es la original y no posee ciclos negativos.



FLUJO = 0
COSTO = 0

FIG. 2.30 Red de flujo constante de valor o

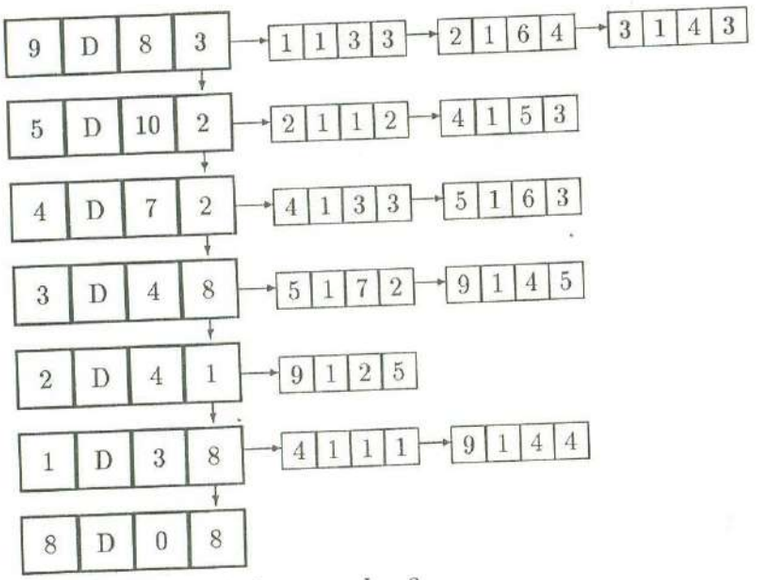
La red representada con estructuras de datos queda:



Como ya se mencionó en ésta primera iteración la red original y la marginal coinciden, encontrando en ella la ruta más corta se tiene que es: $P^* : s, 3, t$ de costo 8. En este caso

$$d = \min\{q'_{s3}, q'_{3t}\} = \{3, 5\} = 3$$

La red marginal después de aplicar el algoritmo Generalizado de Dijkstra se muestra en la siguiente figura.



Ruta: 8, 3, 9 $\quad d = 3$

y actualizando el flujo $f = f + 3 \circ (P^*)$; es decir, se aumentan 3 unidades al flujo a través de los arcos $(s, 3)$, $(3, t)$, la siguiente red lo muestra

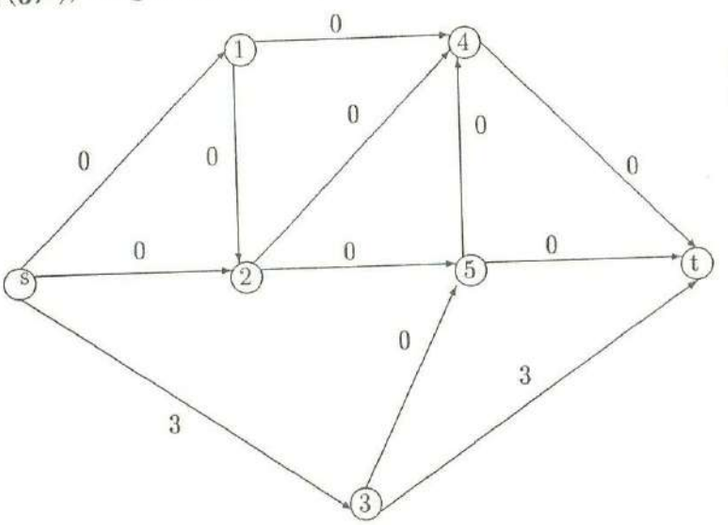
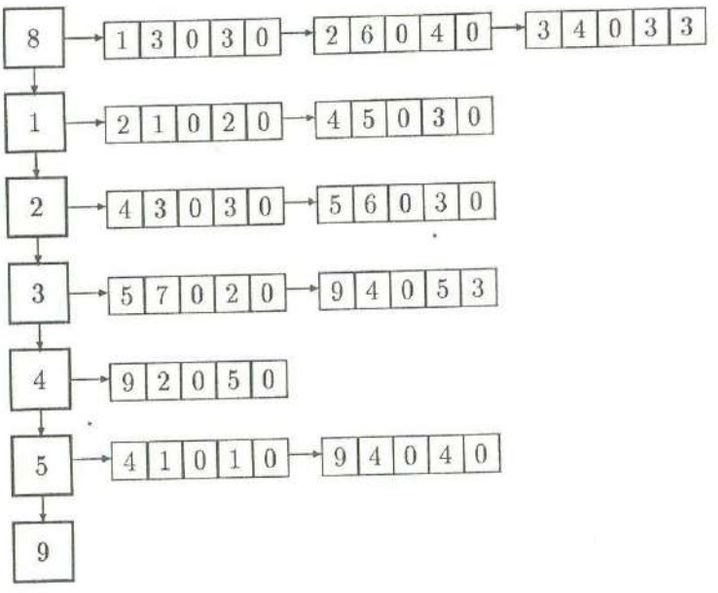


FIG. 2.31 Red de nuevo flujo de valor 3

Expresado con estructuras de datos nos queda:



ITERACIÓN 2. La red marginal de la red de la iteración anterior se muestra en la siguiente figura.

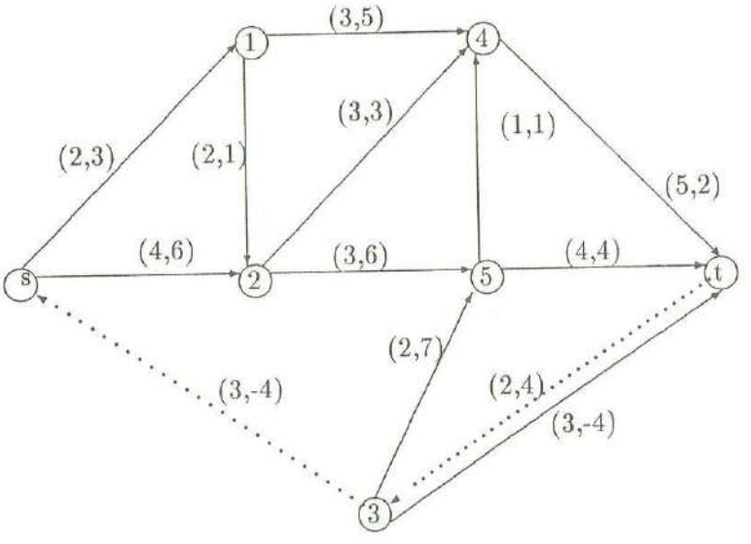
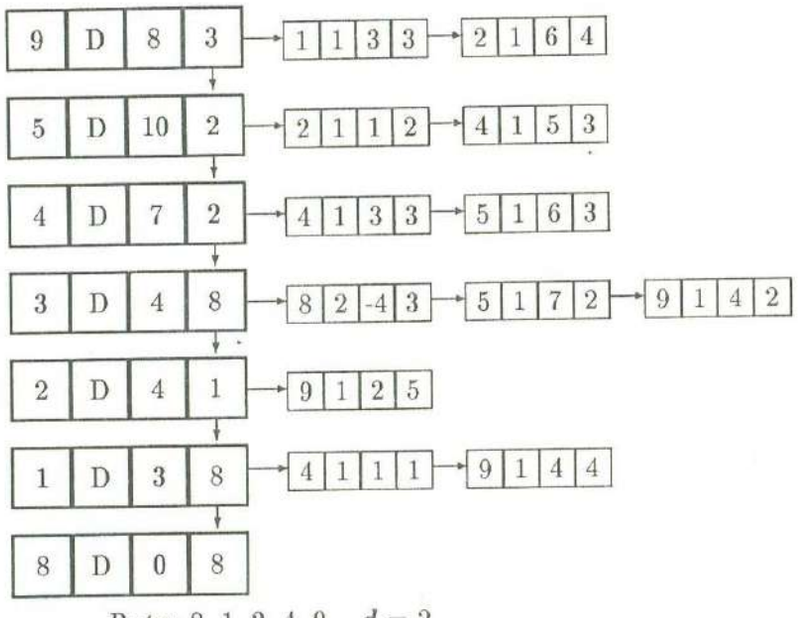


FIG. 2.32 Red Marginal asociada al flujo anterior

La ruta mas corta entre s y t es: P^* : s, 1, 2, 4, t de costo 9. Luego

$$d = \min\{q^t_{s1}, q^t_{12}, q^t_{24}, q^t_{4t}\} = \{3, 2, 3, 5\} = 2$$

y con estructuras de datos tenemos:



Ruta: 8, 1, 2, 4, 9 $\quad d = 2$

Se actualiza $f = f + 2 \circ (P^*)$ definiéndose, de este modo, el flujo mostrado en la siguiente figura.

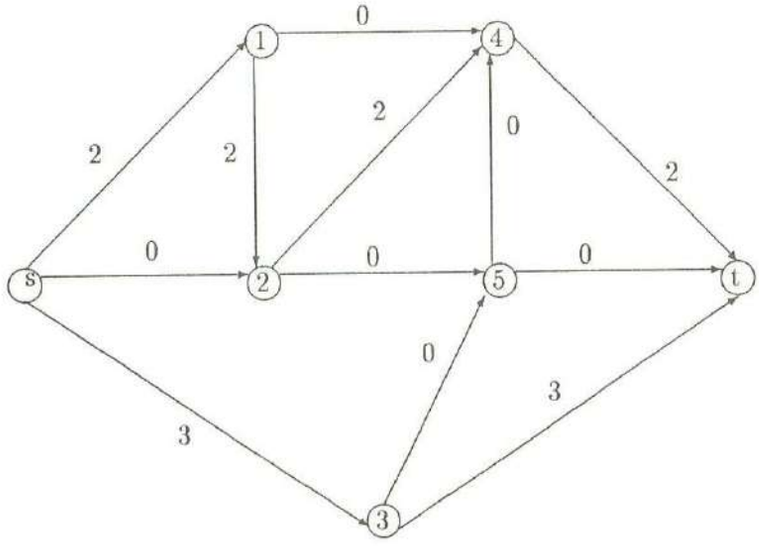
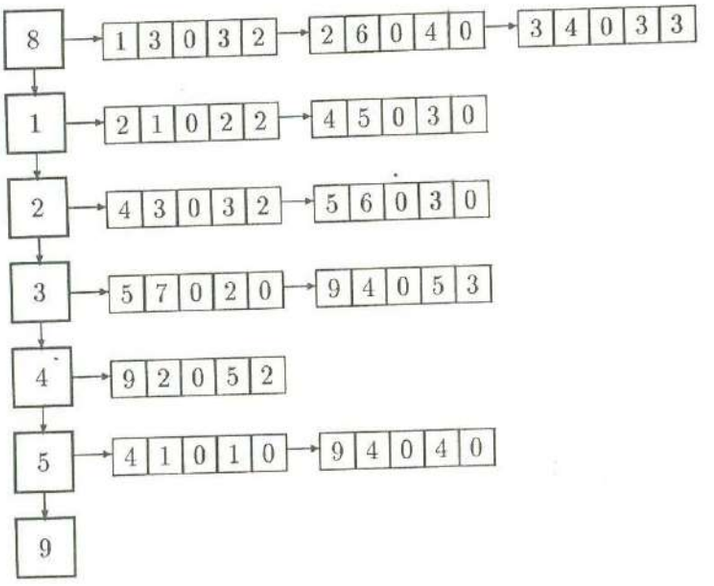


FIG. 2.33 Flujo resultado de la segunda iteración

El flujo actualizado se presenta en la siguiete lista.



ITERACIÓN 3. La *fig.2.34* corresponde a la nueva red marginal.

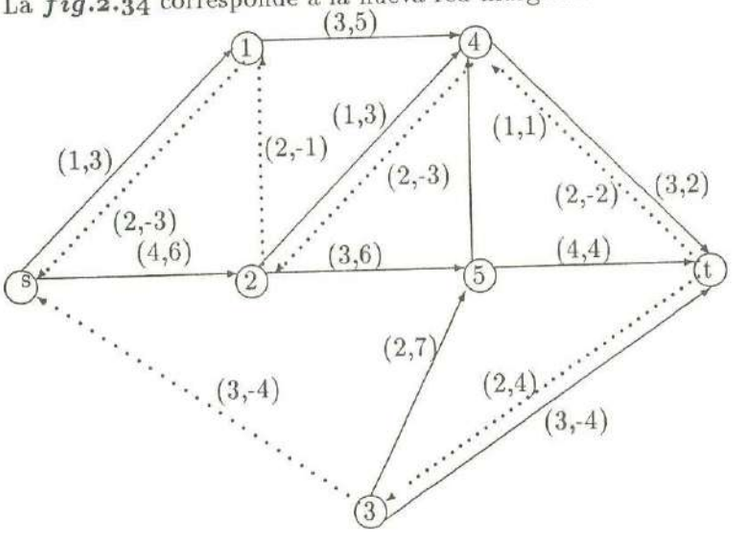
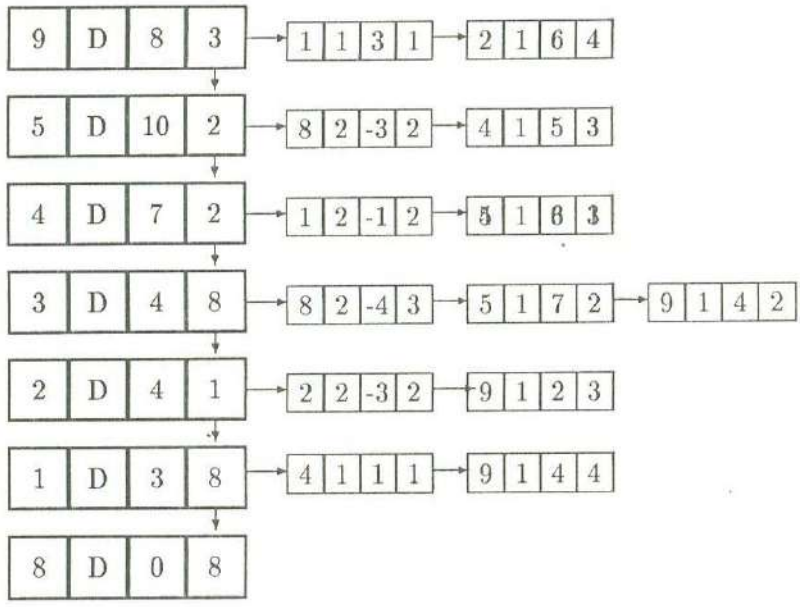


FIG. 2.34 Red Marginal para la iteración número 3

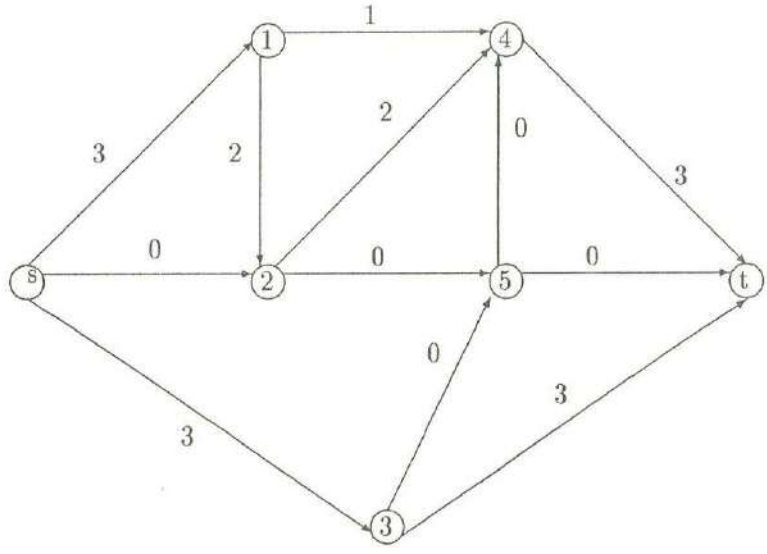
P^* resulta ser: s, 1, 4, t de costo 10, de donde

$$d = \min\{q'_{s1}, q'_{14}, q'_{4t}\} = \{1, 3, 3\} = 1$$



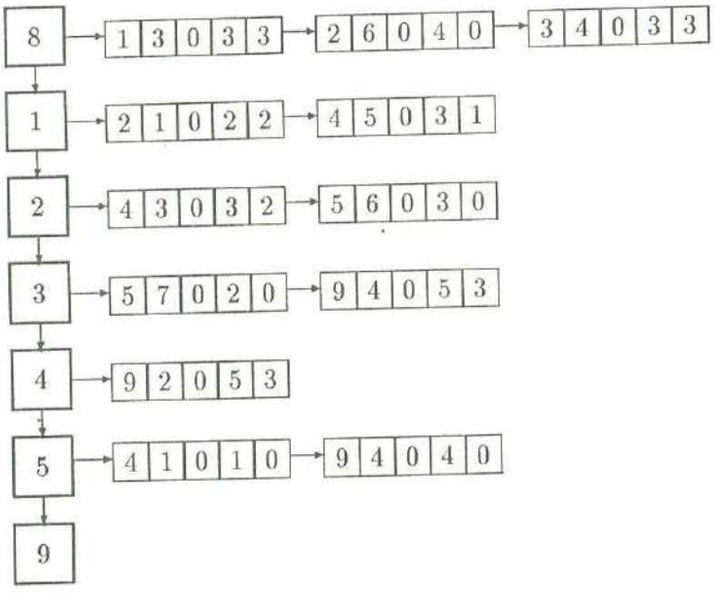
Ruta: 8, 1, 4, 9 $\quad d = 1$

El flujo $f + 1 \circ (P^*)$ se muestra en la *fig.2.33*



FLUJO = 6
COSTO = 52

FIG. 2.35 Nuevo flujo para la red del ejemplo 1



ITERACIÓN 4. La red marginal asociada al flujo de la figura anterior es mostrada en la siguiente figura.

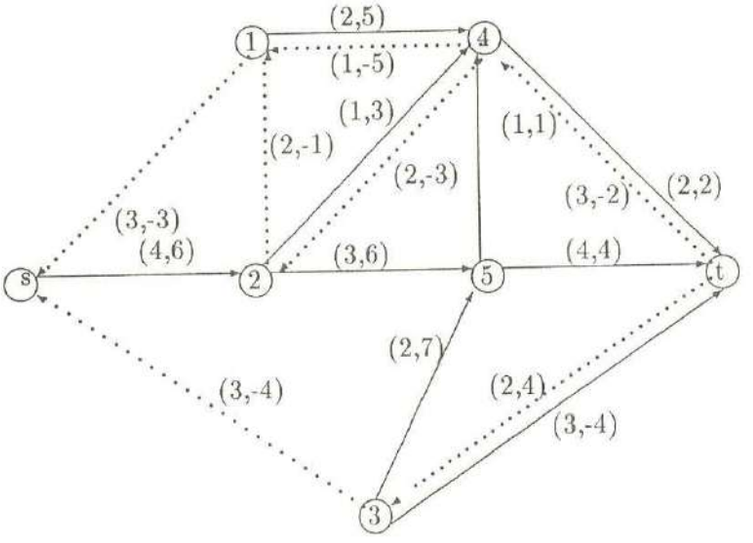
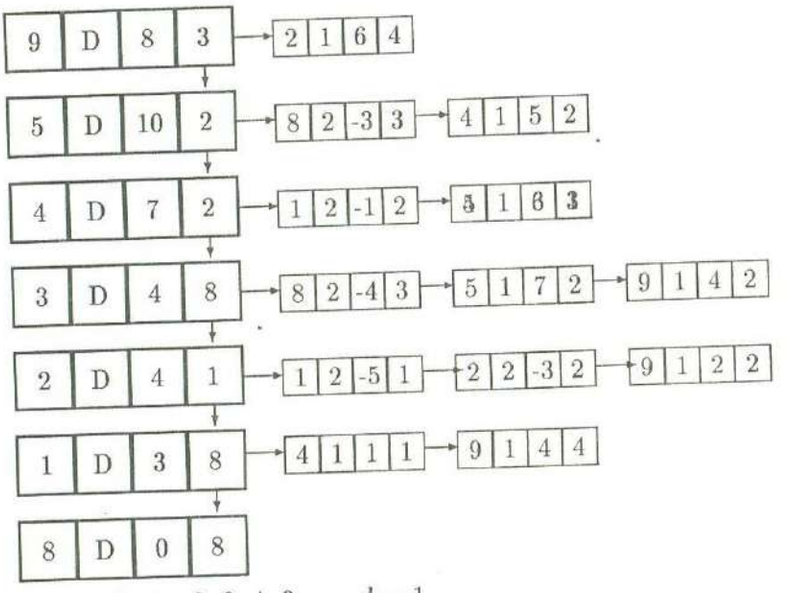


FIG. 2.36 Red Marginal actualizada

P^* es: s, 2, 4, t de costo 11. De aquí

$$d = \min\{q'_{s2}, q'_{24}, q'_{4t}\} = \{4, 1, 2\} = 1$$

La red marginal asignada a esta iteración es la siguiente.



Ruta: 8, 2, 4, 9 $\quad d = 1$

Se actualiza $f = f + 1 \circ (P^*)$. De este modo se obtiene el flujo de valor 7 deseado que se muestra en la ***fig.2.35***. que es de costo 63, siendo el mínimo.

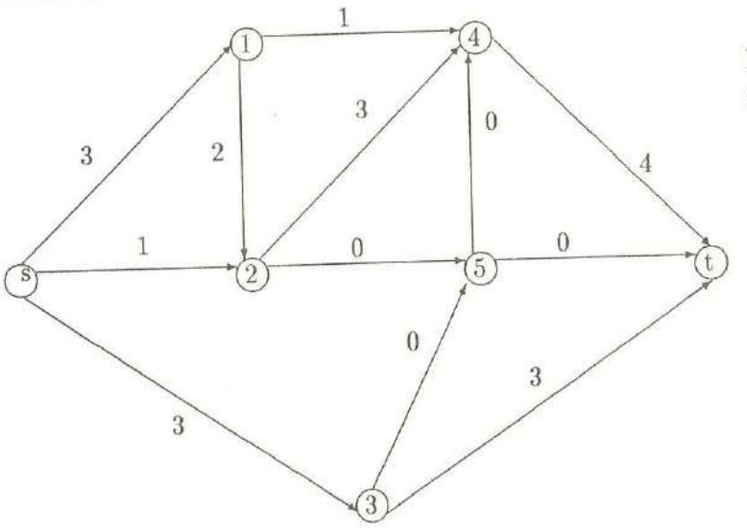
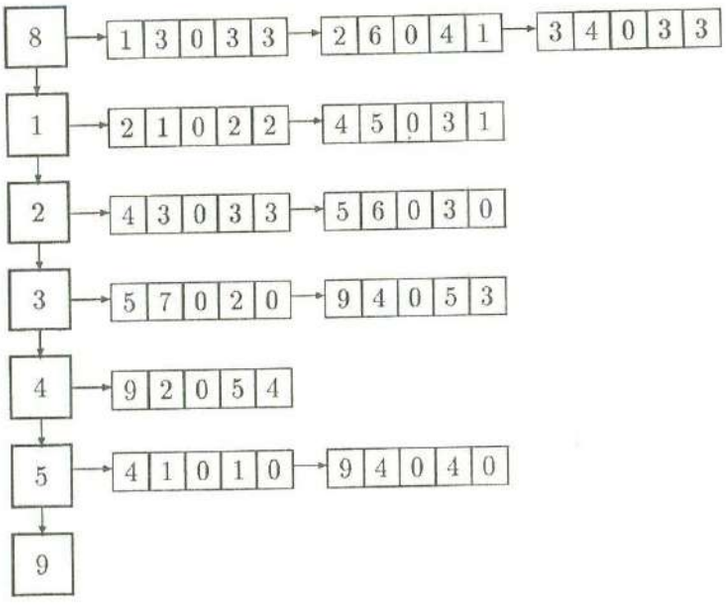


FIG. 2.37 Distribución de flujo de valor 7 de costo mínimo

La red con el flujo de valor deseado de costo mínimo se muestra en las siguientes estructuras de datos.



Así se tiene un flujo de valor 7 definido en la red y de costo mínimo, verificando con los resultados al iterar con el algoritmo de eliminación de circuitos negativos se puede observar que la distribución de flujo es la misma para los dos y óptimo.

Como se puede observar la implementación de cualesquiera de los dos algoritmos es sumamente sencilla, las funciones que se utilizaron resultan de una manera natural al seguir los pasos del algoritmo. La eficiencia y rapidez de las implementaciones radica en gran medida por el tipo de estructuras utilizadas y la versatilidad del lenguaje para accesar a la información de una manera rápida.

Cuando se tenga un problema por resolver mediante flujo constante a costo mínimo, la decisión de utilizar uno u otro algoritmo dependerá en gran medida de la naturaleza del mismo, ya que como se comentaba anteriormente para algunos problemas se puede conseguir un flujo del valor deseado que sea factible y entonces es inmediato que el algoritmo que se utilizará en este caso sea el de eliminación de cicuitos negativos, en cambio cuando en el problema sea fácil encontrar el flujo de cierto valor menor que el flujo deseado y garantizando que es de costo mínimo el algoritmo de rutas mas cortas es una buena elección, esta última situación en la práctica no se presenta muy a menudo, ya que tener un flujo factible de valor menor que el deseado puede ser fácilmente encontrado para algunas redes, pero garantizar la optimalidad no siempre es sencillo, para este caso una opción pudiera ser hacer una combinación de ambos algoritmos, teniendo ya la distribución

de valor menor aplicar el algoritmo de eliminación de circuitos negativos para obtener así un flujo óptimo de ese valor, para después estar en condición de aplicar el método de rutas mas cortas y así tener el flujo óptimo para el problema que se tenga. Otra opción es modificar el Ford-Fulkerson que da una primera solución factible agregándole que los incrementos sean por rutas mas cortas para que la primera solución factible sea óptima.

Conclusiones

Con la elaboración de éste y otros tres trabajos más[1] se amplió el PAREIMM a problemas de Flujos, quedando un contenido de ocho diferentes algoritmos. Dentro de los algoritmos que trabajaba en esta primera versión se encuentran los de Prim y Kruskal para árbol de mínima expansión, Floyd y Dijkstra Generalizado para el cálculo de Rutas más cortas, el algoritmo de Ford y Fulkerson para encontrar flujo máximo y sus Generalizaciones, los algoritmos para el problema de Flujo Constante a Costo mínimo, que se presentan en éste trabajo, y el Símplex especializado en redes. Con esto se cumple otro de los objetivos trazados y realizados que fue la creación de una serie de monografías completas sobre ésta área que servirán de consulta y apoyo a futuros estudiantes interesados en ésta área.

Los proyectos de trabajo a futuro son eficientar y ampliar los programas elaborados haciendo un análisis más detallado de la complejidad de los algoritmos y las estructuras de datos utilizadas para obtener futuras versiónes profesionales, así como el incluir en el PAREIMM representación gráfica de redes pequeñas.

[1] [6], [7] y [8]

Apéndice A

Teoría de Gráficas

En éste Anexo se presentará un panorama básico de Teoría de Gráficas, donde se muestran los resultados mas importantes que se utilizan para analizar el problema de flujo constante a costo mínimo. Las demostraciones a los teoremas que se enuncien no se presentarán, una referencia donde se encuentran son [1] y [5].

Uno de los conceptos básicos e importante a la vez, es el de gráfica y digráfica, el cual posteriormente se extiende para establecer el de red que es sumamente utilizado.

A.1 Gráficas y Digráficas

Definición. Una ***Gráfica*** $G = (X, A, f)$ consta de:

1. Un conjunto $X = X(G)$ cuyos elementos se llaman *vértice* (puntos o nodos) de G.
2. Un conjunto $A = A(G)$, el conjunto de las *aristas* (o líneas) de G.
3. Una función $f : A(G) \longrightarrow X(G) \times X(G)$, la función de *incidencia* de G.

Siempre que nos refiramos a una gráfica supondremos de antemano que es finita.

Si a es una arista de G y $f(a) = \{u, w\}$ se dice que u y w son los *extremos* o *vértices terminales* de a. Una arista cuyos extremos coinciden es un ***lazo***. Dos aristas a y a' son paralelas si tienen los mismos extremos, es decir si $f(a) = f(a')$.

Definición. Una ***Digráfica*** (o gráfica dirigida) es una terna $D = (X, A, f)$ formada por:

1. Unconjunto $X = X(G)$ cuyos elementos se llaman *vértice* (puntos o nodos) de G.

2. Un conjunto $A = A(G)$, el conjunto de *arcos* de G.

3. Una función $f : A(G) \longrightarrow X(G) \times X(G)$, la función de *incidencia* de G.

Si a es un arcos de G y $f(a) = (u, v)$ se dice que u es el extremo inicial y v el extremo final del arco a.

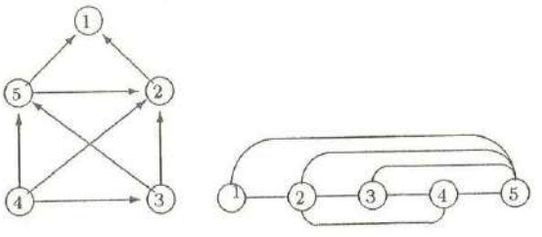
Obsérvese que la única diferencia, aunque importante, entre ambas definiciones, es que en una digráfica los extremos de los arcos están ordenados, mientras que en una gráfica no. Las figuras **A.1** y **A.2** nos muestran un ejemplo de gráfica y digráfica.

Una aclaración importante que el término arista o línea se utiliza en gráficas y arco para digráficas, aunque aveces se hace referencia a uno u otro en fórma indistinta.

A.2 Representación Geométrica de una Gráfica

Una cuestión importante es el hecho de como poder representar geométricamente a una gráfica, la representación que se utiliza es una consecuencia natural de su definición.

Los nodos de la gráfica se representan mediante puntos o círculos y las aristas por líneas entre nodos. Si es gráfica dirigida los arcos se representan poniendo una flecha en la dirección de llegada. Ejemplos:



A.3 Subgráficas

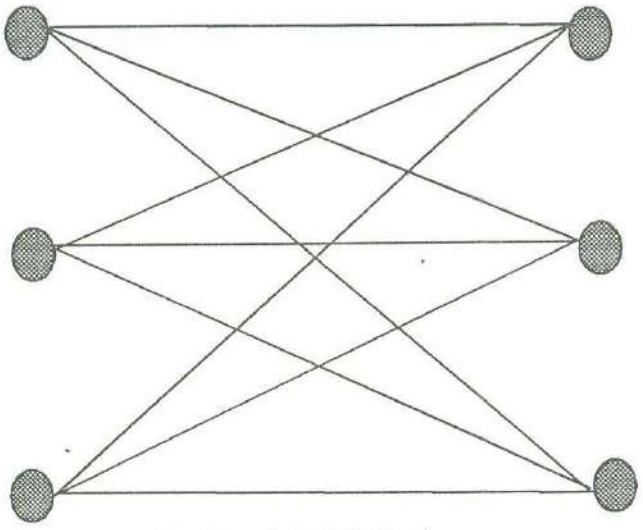
Otros de los conceptos importantes que surgen de una manera muy intuitiva es el de subgráfica, es muy común su utilización en el cálculo de soluciones en los problemas de rutas más cortas y en el de cálculo de flujos óptimos.

Definición. Sea $G = (X, A, f)$ y $G\prime = (X\prime, A\prime, f\prime)$ dos gráficas. Se dice que $G\prime$ es subgráfica de G $(G\prime \subset G)$ si $X\prime \subset X$, $A\prime \subset A$ y $f(a) = f\prime(a)$ para toda arista a de G. Se dice que es una subgráfica

(a) *Generadora* de G cuando $X = X\prime$.

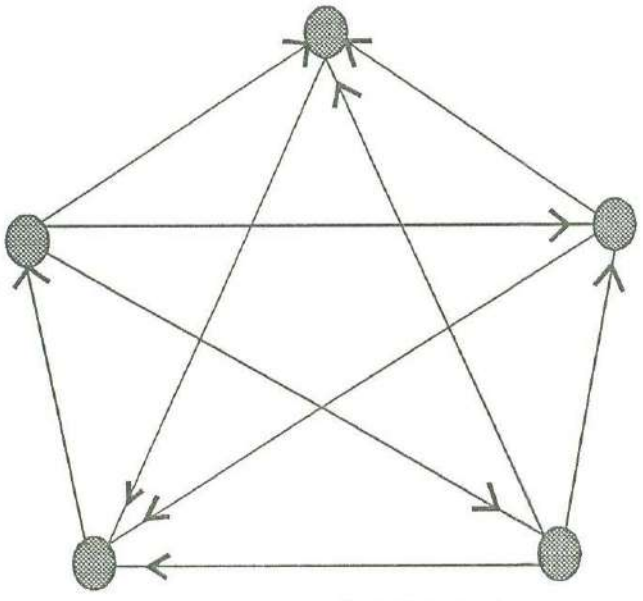
(b) *Inducida* de G $(G\prime \subset G)$ cuando $f(a) \subset X\prime \times X\prime$ implica que $a \in A\prime$.

GRAFICA Y DIGRAFICA



GRAFICA

FIG. A.1 Representación de una Gráfica



DIGRAFICA

FIG. A.2 Representación de una digráfica

Si $G\prime = (X\prime, A\prime, f\prime)$ es subgráfica de G diremos que $X\prime$ induce $G\prime$ en G.

Entonces las gráficas generadoras de una gráfica G dada son aquellas que poseen los mismos nodos que la original y quizá no todos los arcos. Otra manera de verlo es que si se agregan algunos arcos a la gráfica inducida se obtiene la gráfica original, es por eso que se le denomina Generadora. En cambio la gráfica inducida es aquella que posee un subconjunto de nodos de la gráfica original y todas las aristas entre esos nodos que se encuentren también en la gráfica original.

En las figuras $A.3$ y $A.4$ se muestra una gráfica generadora y una inducida.

A.4 Grado de un nodo

Con la representación geométrica que se hace de una gráfica es claro ver que para cada nodo es diferente el número de aristas que tienen como extremo ese nodo, al igual en las digráficas el número de arcos de entrada o de salida de un nodo puede no ser el mismo para todos los nodos, surgiendo entonces el concepto de grado de un nodo tanto de una gráfica como para una digráfica.

Definición. Si v es un vértice de la Gráfica G, se llama *grado* de v y se denota $\Gamma(v)$ al número de aristas de G que tienen a v como extremo.

Definición. Si D es una digráfica y v un vértice, se llama *grado de entrada* de v y se denota $\Gamma^+(v)$ al número de arcos cuyo extremo final es v y se llama *grado de salida* de v y se denota $\Gamma^-(v)$ al número de arcos cuyo extremo inicial es v.

Denotemos $\Gamma(v, v)$ al número de arcos con extremo inicial u y extremo final v.

Proposición. Si D es una digráfica, u,v son dos vértices y a el número de arcos, se verifican las propiedades siguientes:

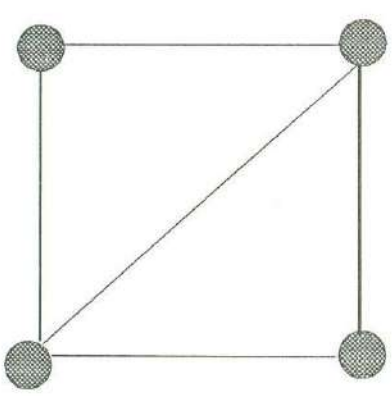
$$(a)\quad \Gamma^+(v) = \sum_{u \in X} \Gamma(u, v); \quad \Gamma^-(v) = \sum_{u \in X} \Gamma(v, u)$$

$$(b)a = \sum_{v \in X} \Gamma^+(v) = \sum_{v \in X} \Gamma^-(v)$$

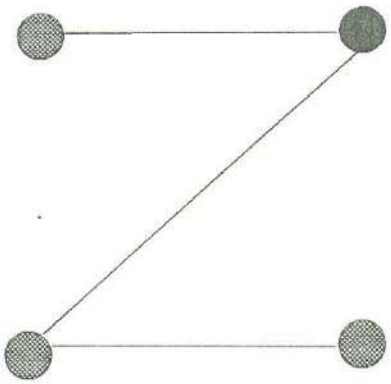
Ejemplos:

El siguiente ejemplo nos muestra el grado de cada nodo como gráfica, el grado de entrada y de salida de la digráfica definida.

SUBGRAFICAS

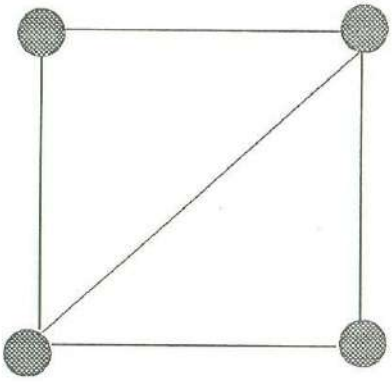


GRAFICA

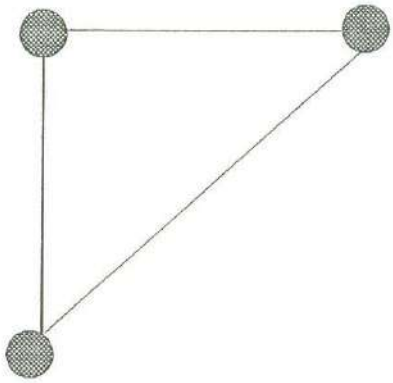


GRAFICA
GENERADORA

FIG. A.3 Ejemplo de Gráfica Generadora

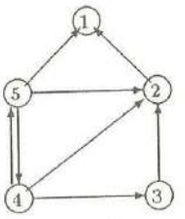


GRAFICA



GRAFICA
INDUCIDA

FIG. A.4 Ejemplo de Gráfica Inducida



$\Gamma(1) = 2 \quad \Gamma^+(1) = 2 \quad \Gamma^-(1) = 0$
$\Gamma(2) = 4 \quad \Gamma^+(2) = 3 \quad \Gamma^-(2) = 1$
$\Gamma(3) = 3 \quad \Gamma^+(3) = 1 \quad \Gamma^-(3) = 1$
$\Gamma(4) = 4 \quad \Gamma^+(4) = 1 \quad \Gamma^-(4) = 3$
$\Gamma(5) = 4 \quad \Gamma^+(5) = 1 \quad \Gamma^-(5) = 3$

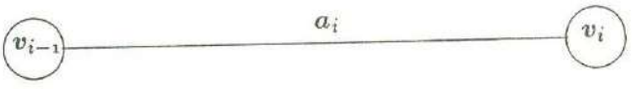
A.5 Camino, Paseo, Trayectoria, Ciclo y Circuito

La definición de Camino, Paseo, Trayectoria, Ciclo y Circuito en gráficas o digráficas según corresponda es muy importante para el problema de flujo Constante a Costo Mínimo, ya que el primer algoritmo se basa en gran medida en la detección de Circuitos de peso negativo en la red Marginal asociada al flujo definido en una red, éstos circuitos en la red marginal corresponden a ciclos de la red original. En el segundo de los algoritmos es muy utilizados los conceptos de Camino, Paseo y Trayectoria, ya qe para obtener el flujo óptimo en una red se calculan trayectorias mínimas en la red marginal asociada a un flujo.

Definición. Sea G una gráfica. Un ***Camino*** es una sucesión alternante

$$(v_0, a_1, v_1, ..., v_{n-1}, a_n, v_n)$$

de vértices y aristas de G en donde cada arista a_i tiene por extremo al vértice v_{i-1} que le precede y tal que le sigue v_i



Si γ es un camino, la longitud de γ, $l(\gamma)$ es el número de aristas no necesariamente distintas que figuran en γ.

Si el camino tiene un solo vértice y no tiene aristas, su longitud es cero, y se dice de él que es nulo.

Se dirá que γes un v_0v_n – ***camino*** (o camino de v_0 a v_n) que tiene como extremo v_0 y v_n.

Un camino no nulo cuyos extremos son iguales se llama ***Cerrado***.

Un camino que no repite aristas se llama ***Paseo*** o camino simple.

Un camino que no repite vértices se llama ***Cadena*** o camino elemental.

Nótese que toda cadena es un paseo pero no recíprocamente.

Definición. Un ***ciclo*** es un camino cerrado elemental de longitud positiva.

Definición. Sea D una digráfica. Un *Camino Dirigido* en D es una sucesión de vértices y arcos.

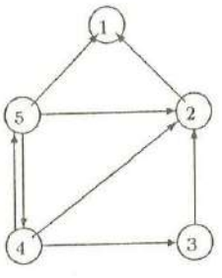
$$(v_0, a_1, v_1, \ldots, v_{n-1}, a_n, v_n)$$

tal que v_{i-1} y v_i son, respectivamente, los extremos inicial y final del arco a_i.

Un camino dirigido que no repite vértices se llama *trayectoria.*

Definición. Un *Circuito* es una trayectoria cerrada que no repite arcos.

Ejemplos:



- $\gamma_1 = (1, a_{21}, 2, a_{32}, 3, a_{43}, 4, a_{42}, 2, a_{32}, 3)$ $\quad l(\gamma_1) = 5$. Es un camino.
- $\gamma_2 = (1, a_{51}, 5, a_{54}, 4, a_{45}, 5, a_{52}, 2)$ $\quad l(\gamma_2) = 4$ Es un paseo.
- $\gamma_3 = (4, a_{42}, 2, a_{32}, 3)$ $\quad l(\gamma_3) = 2$ Es una cadena.
- $\gamma_4 = (1, a_{21}, 2, a_{32}, 3, a_{43}, 4, a_{54}, 5, a_{51}, 1)$ $\quad l(\gamma_4) = 5$ Es un ciclo.
- $\gamma_5 = (4, a_{43}, 3, a_{32}, 2, a_{21}, 1)$ $\quad l(\gamma_5) = 3$ Es una trayectoria.
- $\gamma_6 = (5, a_{54}, 4, a_{45}, 5)$ $\quad l(\gamma_6) = 2$ Es un circuito.

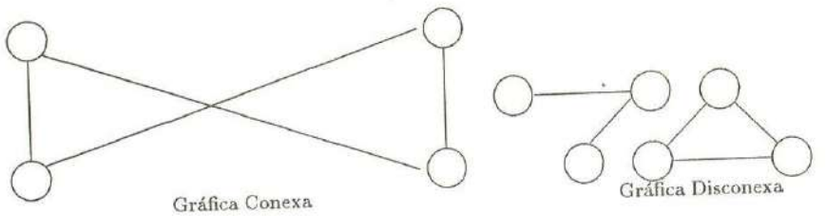
Teorema. Sea P un camino dirigido simple entre un nodo s y un t, entonces se puede descomponer en una trayectorias de s a t y n circuitos elementales de vértices de P.

La demostración de éste teorema es muy importante para el teorema de descompusición de un flujo sobre trayectorias y circuitos elementales, la idea de la prueba es muy intuitiva. Dado que la trayectoria sobre la cual se trabaja es simple, es decir que no repite arcos en ella, se construye un nuevo camino quitándo todos aquellos circuitos que posee P, claramente el nuevo camino dirigido P^* será una trayectoria, ya que no repetirá vértices y será elemental, en los circuitos eliminados

Definición. Dos vértices u y v de una gráfica G se dice que están conectados cuando existe un camino en G de extremos u y v.

Definición. Una gráfica G es ***conexa*** si, y solo si para todo par de vértices u y v de G existe un camino en G que conecta u y v.

Ejemplos:



A.8 Arboles

En los problemas de ruta mínima el concepto de árbol y arborescencia es de suma importancia, ya que las soluciones están caracterizadas por ellos y en general la teoría que se deriva de éste tipo de problemas gira en torno a la definición y propiedades de árboles y arborescencias. Otro uso que no es menos importante es en la implementación de algoritmos, ya que muchos de los conceptos de estructuras de datos que eficientizan la implementación de un algoritmo se basan en las propiedades de los árboles.

Definición. Una gráfica G se dice que es un ***arbol*** si, y sólo si se verifican las siguientes condiciones.

(a) G es conexa;

(b) G no posee ciclos (acíclica).

Definición. Un árbol A se dice que es un árbol generador de la gráfica G si A es subgráfica de G tal que su conjunto de vértices coincide en el de G.

Proposición. Todo grafo conexo posee un árbol generador

Proposición. Un árbol con n vértices tiene $n-1$ aristas

Proposición. Todo árbol con mas de un vértice posee al menos dos vértices de grado 1.

REPRESENTACION DE UNA ARBORESCENCIA

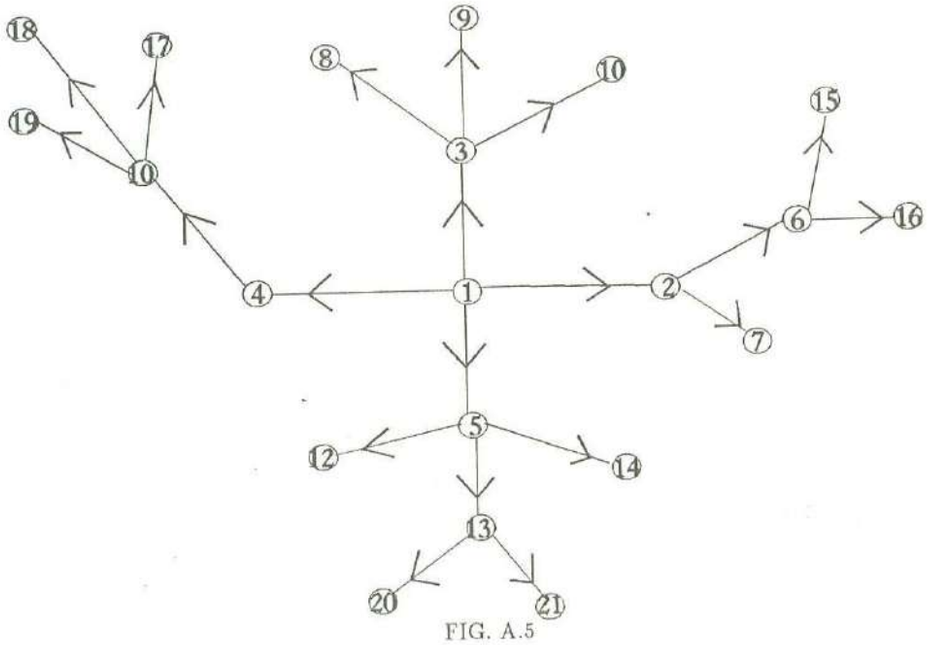
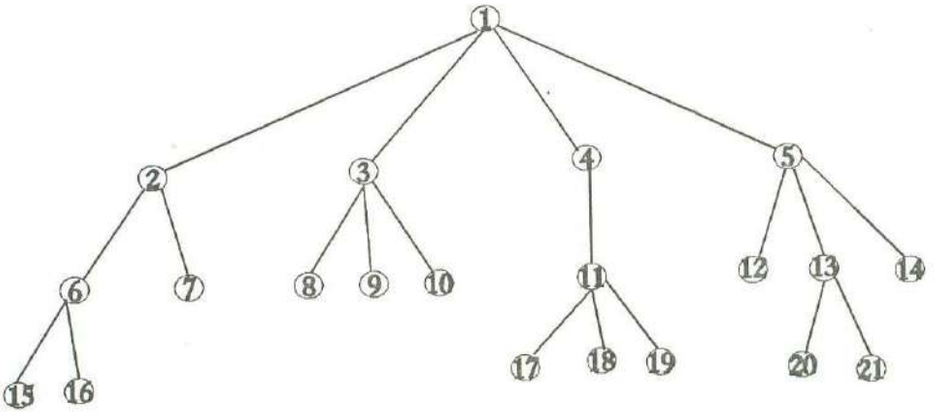


FIG. A.5

Representacion grafica de una arborescencia



Mejor representacion grafica de una arborescencia

FIG. A.6

Apéndice B

Manual del Usuario

B.1 ¿Cómo se prepara la información?

La información que se utiliza para la corrida de los programas se obtiene de un archivo que previamente debe estar preparado. El archivo con la información puede llevar el nombre que se desee, bajo al restricción normal de poseer un nombre con a lo mas ocho caracteres; la terminación del archivo debe ser .SOL, para que pueda ser leido. Ejemplo: EJEMPLO1.SOL

La información tiene que estar distribuida de la siguiente manera:

- Un primer renglón con solamente el número 3.
  Este número sirve para identificación dentro del programa.
- La información de los arcos se pone por renglón de la siguiente manera

  NODO_INICIAL NODO_FINAL CAP_MINIMA CAP_MAXIMA COSTO FLUJO
- Ya que se hayan capturado todos los arcos se pone un renglón que contiene cinco ceros.
- La última línea contiene la información siguiente:

  NUMERO_DE_NODOS NUMERO_DE_ARCOS PESO FLUJO

En cada renglón la información va separadapor espacios.

A continuación se muestra como ejemplo de como se distribuye la información del archivo de la gráfica del ejemplo 1 para que los programas los puedan leer.

```
3
8 1 0 3 3 2
8 2 0 4 6 2
8 3 0 3 4 3
1 2 0 2 1 0
1 4 0 3 5 2
2 4 0 3 3 1
2 5 0 3 6 1
3 5 0 2 7 2
3 9 0 5 4 1
5 4 0 1 1 1
4 9 0 5 2 4
5 9 0 4 4 2
0 0 0 0 0
7 12 84 7
```

Al empezar la corrida del programa, lee la información del archivo y crea una red.

B.2 ¿Cómo se corren los programas?

Consideremos que todos los datos de un problema que se quiere resolver ya se encuentran en un archivo adecuado con terminación .PRB, supongamos que se está trabajando con el Ejemplo 1 mostrado anteriormente, el archivo va a cambiar si se desea aplicarle el programa basado en eliminación de circuitos negativos o el de detección de rutras más cortas, a continuación se mostrará la manera de utilizar cada uno de los problemas mostrando el archivo del ejemplo 1 para cada problema.

- Algoritmo de Flujo Constante a Costo Mínimo por medio de Eliminación de Circuitos Negativos

  Como se comentó anteriormente para correr éste programa se necesita tener todos los datos en un archivo distribuidos correctamente, los datos para el Ejemplo 1 con un flujo inicial de valor 7 se encuentran en el archivo PROB2.PRB el cual se utilizará como ejemplo. Lo datos siguientes nos muestran la infoirmación de PROB2.PRB en un archivo.

```
3
8 1 0 3 3 2
8 2 0 4 6 2
8 3 0 3 4 3
1 2 0 2 1 0
1 4 0 3 5 2
2 4 0 3 3 1
2 5 0 3 6 1
3 5 0 2 7 2
3 9 0 5 4 1
5 4 0 1 1 1
4 9 0 5 2 4
5 9 0 4 4 2
0 0 0 0 0
7 12 84 7
```

Para correr el programa hay que cerciorarse de que el archivo .PRB se encuentre en el disco de trabajo de la computadora, para ejecutar un problema se teclea:

- El nombre del programa (CICLOS).
- Seguido y separado de un espacio el nombre del problema con extensión .PRB
- Posteriormente los números de los nodos Fuente y Destino de la red.

Esquematizado quedaría de la siguiente manera:

$*: \textit{CICLOS} \quad [\textit{NOMBRE}].\textit{PRB} \quad [\#\textit{FUENTE}] \quad [\#\textit{DESTINO}]$

Ejemplificado para el PROB2.PRB se tendría:

$*: \textit{CICLOS} \quad \textit{PROB2.PRB} \quad 8 \quad 9$

Si los argumentos en línea no se encuentran completos el programa presenta el siguiente mensaje:

Uso: CICLOS problema.prb Fuente Destino
La extensión .PRB es obligada para el nombre del archivo
que contiene al problema. Tal archivo debe tener una
primera linea que solo contenga el número 1,
la información de los arcos será
NODO_INICIAL NODO_FINAL CAP_INF CAP_SUP COSTO FLUJO
una linea al final de la información de los arcos
que contenga 5 ceros (0 0 0 0 0).
por último un renglón con la siguiente información
NUMERO_DE_NODOS NUMERO_DE_ARCOS PESO FLUJO

Si los argumentos si fueron dados completos el programa empieza a correr y manda letreros si la solución no se pudo dar por alguna causa, o bien indica el archivo donde fue grabada la solución, en la sección posterior se detallará la forma de interpretar la solución o resultados. Antes de dar la solución manda el mensaje siguiente:

FLUJO A COSTO MINIMO POR ELIMINACION DE CIRCUITOS NEGATIVOS (CICLOS)

Versión 1.0, 1993 DEPARTAMENTO DE MATEMATICAS

UNIVERSIDAD DE SONORA

- Algoritmo de Flujo Constante a Costo Mínimo basado en la detección de Rutas más Cortas

Para este caso consideremos que la información del Ejemplo 1 para aplicar éste algoritmo se encuentra en un archivo PROB2R.PRB, queriéndose encontrar el flujo de costo mínimo entre el nodo **8** y **9** con un flujo de valor **7** partiendo de una distribución de flujo igual a cero, ya que para todos los arcos la cota mínima de arcos es igual a cero. Los datos siguientes muestran el archivo PROB2R.PRB listo para ser ejecutado.

```
3
8 1 0 3 3 0
8 2 0 4 6 0
8 3 0 3 4 0
1 2 0 2 1 0
1 4 0 3 5 0
2 4 0 3 3 0
2 5 0 3 6 0
3 5 0 2 7 0
3 9 0 5 4 0
5 4 0 1 1 0
4 9 0 5 2 0
5 9 0 4 4 0
0 0 0 0 0
7 12 0 7
```

Para correr el programa hay que cerciorarse de que el archivo .PRB se encuentre en el disco de trabajo de la computadora, para ejecutar un problema se teclea:

- El nombre del programa (RUTAS).
- Seguido y separado de un espacio el nombre del problema con extensión .PRB
- Posteriormente los números de los nodos Fuente y Destino de la red.
- Al final se coloca el valor de flujo que se desea en la red.

Esquematizado quedaría de la siguiente manera:

$*:$ ***RUTAS*** $[NOMBRE].PRB$ $[\#FUENTE]$ $[\#DESTINO]$ $[\#FLUJO]$

Resolviendo el PROB2.PRB, el archivo solución queda escrito de la siguiente manera:

```
3
8 1 3.000000 3
8 2 6.000000 1
8 3 4.000000 3
1 2 1.000000 2
1 4 5.000000 1
2 4 3.000000 3
2 5 6.000000 0
3 5 7.000000 0
3 9 4.000000 3
4 9 2.000000 4
5 4 1.000000 0
5 9 4.000000 0
0 0 0 0
7 12 63.000000 7
```

y para el PROB2R.PRB nos quedaría un archivo igual al anterior.

En el caso de que no haya solución para alguno de los dos programas, éstos envian mensajes de acuerdo a la falla existente, estos son:

- CICLOS
  - **ERROR: Archivo** [Nombre] **no encontrado**: Este letrero surge cuando el programa no encontró un archivo con ese nombre .
  - **ERROR: Archivo** [Nombre] **no abierto**: Este letrero surge cuando el programa no pudo abrir el archivo con ese nombre .
  - **ERROR: Archivo** [Nombre] **no adecuado**: Este letrero surge cuando el programa detectó incompatibilidad en el tipo de archivo para aplicar el algoritmo.
  - **ERROR: Archivo** [Nombre] **vacío**: Este letrero surge cuando el programa detectó que el archivo con el nombre dado se encontraba vacío.
  - **ERROR: Archivo** [Nombre] **demasiado grande**: Este letrero surge cuando el archivo contiene un programa muy grande para el propósito educativo de ésta versión.
  - **ERROR: Memoria insuficiente para continuar en** [Nombre] : Este letrero surge cuando el programa detectó escasez de memoria para seguir iterando.
- RUTAS
  - **ERROR: Archivo** [Nombre] **no encontrado**: Este letrero surge cuando el programa no encontró un archivo con ese nombre .
  - **ERROR: Archivo** [Nombre] **no abierto**: Este letrero surge cuando el programa no pudo abrir el archivo con ese nombre .

- **ERROR: Archivo** [Nombre] **no adecuado**: Este letrero surge cuando el programa detectó incompatibilidad en el tipo de archivo para aplicar el algoritmo.
- **ERROR: Archivo** [Nombre] **vacío**: Este letrero surge cuando el programa detectó que el archivo con el nombre dado se encontraba vacío.
- **ERROR: Archivo** [Nombre] **demasiado grande**: Este letrero surge cuando el archivo contiene un programa muy grande para el propósito educativo de ésta versión.
- **ERROR: Memoria insuficiente para continuar en** [Nombre] : Este letrero surge cuando el programa detectó escasez de memoria para seguir iterando.
- **No hay ruta mínima, hay ciclo negativo**: Este mensaje es mandado por el algoritmo General de Dijkstra y como se indica no existe ruta mínima, ya que existe un circuito negativo.
- **El Nodo** [Nombre] **no existe** : Indica que en algúna búsqueda no se encontró el nodo en cuestión.
- **Memoria insuficiente para la primera fase**: Insuficiencia de memoria para aplicar la segunda fase del algoritmo General de Dijkstra.
- **No hay arborescencia para el nodo** [Nombre]: Indica que no se pudo encontrar una arborescencia con el nodo Fuente como raiz.
- **Del Nodo** [Nombre] **no salen arcos** En la búsqueda de una arborescencia que cuelga del Nodo fuente no se encuentran arcos que cuelguen de él.

Así se presentaron los aspectos básicos para la corrida de los dos programas que resuelven el problema de Flujo Constante a Costo Mínimo.