Figura 1

FACULTAD DE INGENIERÍA
DIMEI
DEPTO. DE INGENIERÍA DE SISTEMAS
PROGRAMA DE POSGRADO EN IDEO

NOTAS DE CLASE
OPTIMIZACIÓN COMBINATORIA
SEMESTRE: 2024-1
DRA. PATRICIA E. BALDERAS CAÃS

Contenido temático

1. INTRODUCCIÓN
2. GRAFOS
3. PROBLEMAS Y ALGORITMOS
4. ÁRBOLES ÓPTIMOS Y RUTAS
5. PROBLEMAS DE FLUJO MÁXIMO
6. PROBLEMAS DE FLUJO A MÍNIMO COSTO
7. EMPAREJAMIENTOS ÓPTIMOS
8. PROBLEMA DEL CARTERO CHINO
9. PROBLEMA DE LA MOCHILA
10. PROBLEMA DEL AGENTE VIAJERO

1. INTRODUCCIÓN

La optimizacimbinatoria es una rama de la optimización en matemáticas aplicadas y en ciencias de la computación, relacionada a la investigación de operaciones, teoría de algoritmos y teoría de la complejidad computacional. Tambistá relacionada con otros campos, como la inteligencia artificial e ingeniería de software. Los algoritmos de optimización combinatoria resuelven instancias de problemas que se creen ser difíciles en general, explorando el espacio de soluciones (usualmente grande) para estas instancias. Los algoritmos de optimización combinatoria logran esto reduciendo el tamaectivo del espacio, y explorando el espacio de búsqueda eficientemente.

Los algoritmos de optimización combinatoria a menudo son implementados en lenguajes como C y C++ entre otros softwares inteligentes, en lenguajes de programación lógicos tales como Prolog, o incluso en lenguajes multi-paradigma tales como Oz.

Mediante el estudio de la teoría de la complejidad computacional es posible comprender la importancia de la optimización combinatoria. Los algoritmos de optimización combinatoria se relacionan comúnmente con problemas NP-hard. Dichos problemas en general no son resueltos eficientemente, sin embargo, varias aproximaciones de la teoría de la complejidad sugieren que ciertas instancias (ej. "peque instancias) de estos problemas pueden ser resueltas eficientemente. Dichas instancias a menudo tienen ramificaciones prácticas muy importantes.

El contenido de estas Notas de Clase está organizado de acuerdo con la figura:

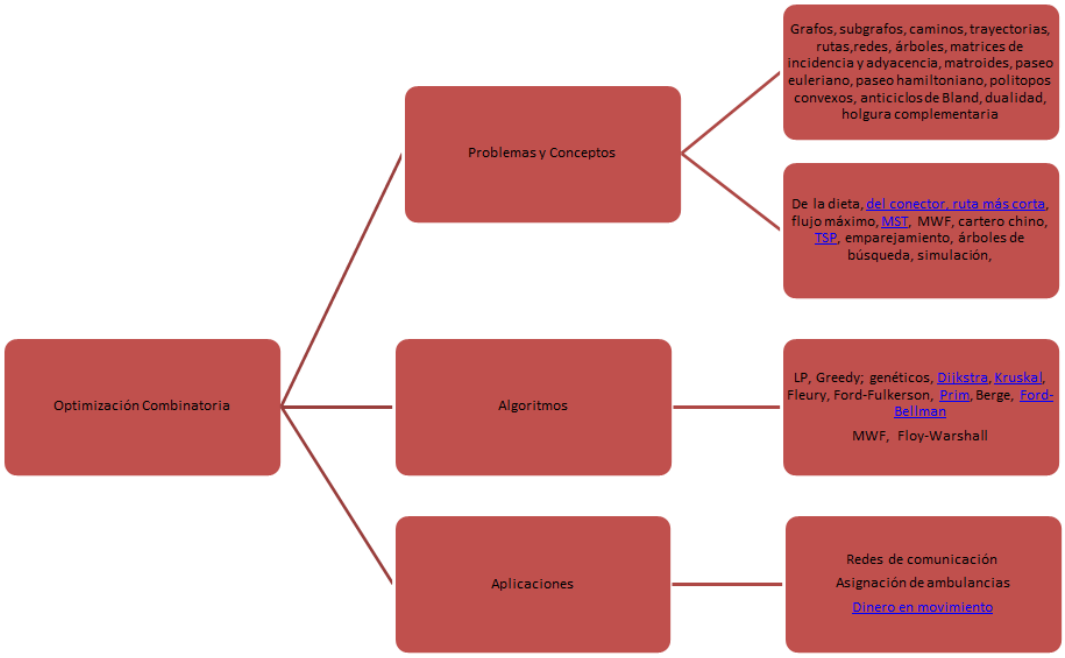

Figura 2

2. GRAFOS

2.1. Grafos y árboles

Definition 1 *Una gráfica G es una terna ordenada*

$$(V(G), E(G), \psi_G) \tag{1}$$

que consiste de un conjunto no vacío $V(G)$ de vices, un conjunto $E(G)$, ajeno a $V(G)$, de aristas o lados, y una función de incidencia ψ_G, que asocia a cada lado de G, un par vices, de G, no ordenado y no necesariamente distintos.
Si e es una arista, u y v dos vices tales que

$$\psi_G(e) = uv$$

entonces se dice que e une a u y v, en este caso los vices u y v se llaman extremos de e.

Definition 2 *El grado de un vice* $\deg(v)$*, es el número de aristas de las cuales es un extremo, o que inciden en*

Example 3 *Sea* $(V(G), E(G), \psi_G)$*, donde*

$$\begin{aligned} V(G) &= \{v_1, v_2, v_3\} && (2)\\ E(G) &= \{e_1, e_2\} && (3)\\ \psi_G(e_1) &= v_1 v_2 && (4)\\ \psi_G(e_2) &= v_2 v_3 && (5)\\ & && (6) \end{aligned}$$

la gráfica (el grafo) correspondiente es

Example 4 *Sea* $(V(H), E(H), \psi_H)$*, donde*

$$\begin{aligned} V(H) &= \{v_1, v_2, v_3, v_4, v_5\}\\ E(G) &= \}e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8\}\\ \psi_G(e_1) &= v_1 v_2\\ \psi_G(e_2) &= v_2 v_3\\ \psi_G(e_3) &= v_3 v_3\\ \psi_G(e_4) &= v_3 v_4\\ \psi_G(e_5) &= v_2 v_4\\ \psi_G(e_6) &= v_4 v_5\\ \psi_G(e_7) &= v_2 v_5\\ \psi_G(e_8) &= v_2 v_5 \end{aligned}$$

la gráfica (el grafo) correspondiente es

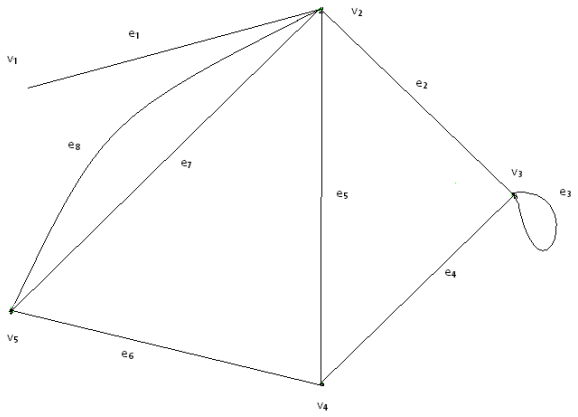

Figura 3: grafica2

Example 5 *Sea* $H = (V(HG), E(H), \psi_H)$, *donde*

$$\begin{aligned} V(H) &= \{u, v, w, x, y\} \\ E(H) &= \{a, b, c, d, e, f, g, h\} \\ \psi_H(a) &= uv \\ \psi_H(b) &= uu \\ \psi_H(c) &= vw \\ \psi_H(d) &= wx \\ \psi_H(e) &= vx \\ \psi_H(f) &= wx \\ \psi_H(g) &= ux \\ \psi_H(h) &= xy \end{aligned}$$

la gráfica (el grafo) correspondiente es

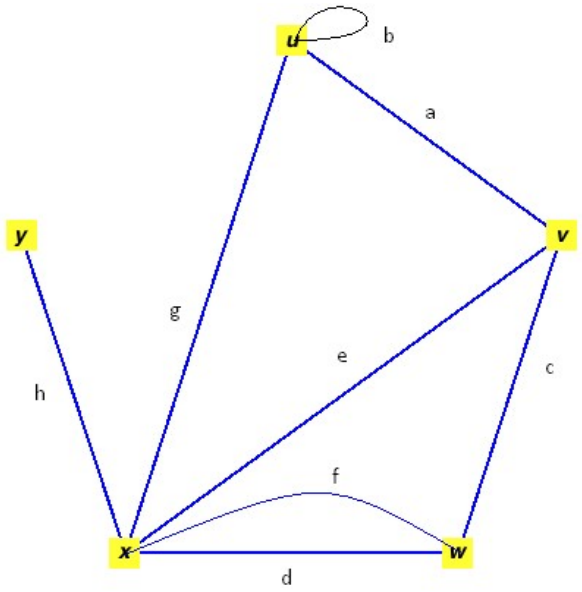

Figura 4

La gráfica muestra la relación de incidencia. No es única la forma de mostrar el grafo. La posición relativa de los vices no tiene significado. Las gráficas cuyas aristas sólo se intersectan en los vices se llaman planas, por representarse en un plano de manera simple.

Exercise 6 *Dibujar las siguientes gráficas de manera que sean planas.*

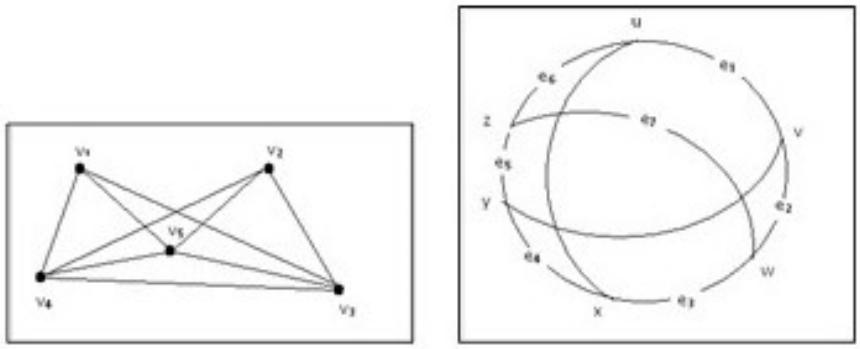

Figura 5

Definition 7 *Dos vices que son incidentes a un lado común se llaman adyacentes. Así como los lados son adyacentes cuando tienen un vice en común.*

Definition 8 *Una arista con extremos idicos, es decir, el mismo vice, se llama lazo.*

Definition 9 *Una arista con extremos distintos se llama vínculo.*

Definition 10 *Una gráfica es finita si sus dos conjuntos que la definen, V y E son finitos.*

Definition 11 *Una gráfica con un sólo vice se denomina trivial y el resto de las gráficas no triviales.*

Definition 12 *Una gráfica se denomina simple si no tiene lazos y dos vínculos no tienen el mismo par de vices.*

Definition 13 *Con la notación $\nu(G)$ y $\varepsilon(G)$, nos referimos a la cardinalidad de los conjuntos $V(H)$ y $E(H)$, respectivamente.*

Definition 14 *Dos gráficas son idicas $G = H$, si*

$$\begin{aligned} V(G) &= V(H) \\ E(G) &= E(H) \\ \psi_G &= \psi_H \end{aligned}$$

Definition 15 *Dos gráficas son isomórficas $G \simeq H$ si existen biyecciones, entre los vices y las aristas,*

$$\begin{aligned} \theta &: V(G) \to V(H) \\ \phi &: E(G) \to E(H) \end{aligned}$$

tales que

$$\psi_G(e) = uv \Leftrightarrow \psi_H(\phi(e)) = \theta(u)\,\theta(v)$$

el par de mapeos (θ, ϕ) se llama isomorfismo entre G y H. El isomorfismo puede indicarse mediante los mapeos de incidencia.

Example 16 *Se presentan ambas gráficas de forma que se aprecie rápidamente el isomorfismo.*

G	H
$\theta(v_1) = y$	$\phi(e_1) = h$
$\theta(v_2) = x$	$\phi(e_2) = g$
$\theta(v_3) = u$	$\phi(e_3) = b$
$\theta(v_4) = v$	$\phi(e_4) = a$
$\theta(v_5) = w$	$\phi(e_5) = e$
	$\phi(e_6) = c$
	$\phi(e_7) = d$
	$\phi(e_8) = f$

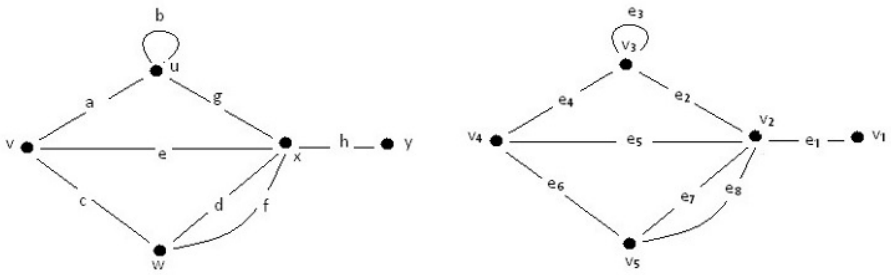

Figura 6

Una gráfica sin etiquetas es la representante de todas las gráficas isomórficas a ella.

Definition 17 *Una gráfica se llama completa cuando cada par de vices distintos está unido por una arista.*

Notation 18 *Salvo isomorfismo, sólo hay una gráfica completa con n vices, la cual se representa por K_n.*

Example 19 *Gráfica completa con cinco vices: K_5*

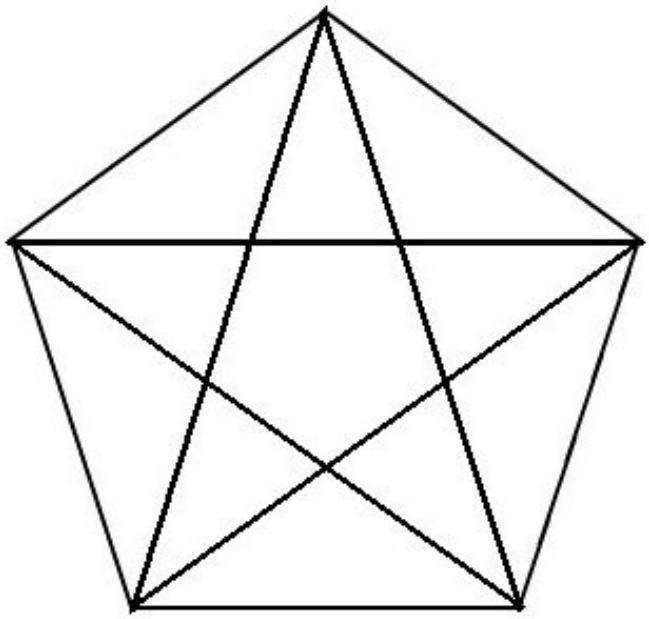

Figura 7

Definition 20 *Una gráfica vacía es una sin aristas.*

Definition 21 *Una gráfica bipartita es una cuyo conjunto de vices se puede partir en dos subconjuntos X y Y, de modo que, cada arista tiene un extremo en X y un extremo en Y.*

Notation 22 *La partición se denota por (X,Y).*

Definition 23 *Una gráfica bipartita completa es una gráfica simple bipartita con partición (X,Y), en la cual cada vice de X está unido con un vice de Y.*

Notation 24 *Si $\nu(X) = m$ y $\nu(Y) = n$, la gráfica bipartita completa de denota por $K_{m,n}$.*

2.2. Matrices de incidencia y adyacencia

Definition 25 *Para toda gráfica G, con* $V(G) = \{v_1, \ldots, v_m\}$ *y* $E(G) = \{e_1, \ldots, e_n\}$*, hay una matriz de dimensión* $m \times n$*, denominada matriz de incidencia, definida por*

$$M(G) = [m_{ij}]$$

donde m_{ij} *es el número de veces que el vice* v_i *y la arista* e_j*, son incidentes. Nótese que* $m_{ij} = 0, 1, 2$

Example 26 *Matriz de incidencia*

?	v_1	v_2	v_3	v_4	v_5
v_1	1	1	0	0	1
v_2	1	1	1	0	0
v_3	0	0	1	1	0
v_4	0	0	0	1	0
v_5	0	0	0	0	1

Definition 27 *La matriz de adyacencia es una matriz de dimensión* $m \times m$*,* $A(G) = [a_{ij}]$*, donde* a_{ij} *es el número de aristas que unen a los vices* v_i *,* v_j*.*

Example 28 *Gráfica y matriz de adyacencia*

$$\begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \end{array} \begin{array}{c} \begin{array}{cccc} v_1 & v_2 & v_3 & v_4 \end{array} \\ \begin{bmatrix} 0 & 2 & 1 & 1 \\ 2 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix} \end{array}$$

Propiedades. Sean M y A las matrices de incidencia y adyacencia de la gráfica G, respectivamente.

1. La suma de todas las celdas de una columna de M es 2
2. La suma de todas las celdas de una columna de A es el grado del vice, esto es, el número de aristas que lo tienen por extremo.

Definition 29 *Sobre subgráficas*

1. Una gráfica H es una subgráfica de G, $H \subseteq G$, si

$$\begin{array}{rcl} V(H) & \subseteq & V(G) \\ E(H) & \subseteq & E(G) \\ & & \psi_H \text{ es la restricción de } \psi_G \text{ a } E(H) \end{array}$$

   En este caso, tambie dice que G es una supergráfica de H.

2. Cuando $H \subseteq G$, pero $H \neq G$ se dice que H es una subgráfica propia de G.
3. Una expansión H (subgráfica o supergráfica) de G, se da cuando $V(H) = V(G)$.
4. Quitando los lazos y dejando sólo una arista, de las que halla en cada par de vices adyacentes, se tiene una subgráfica expansión simple de G denominada la gráfica simple subyacente de G.

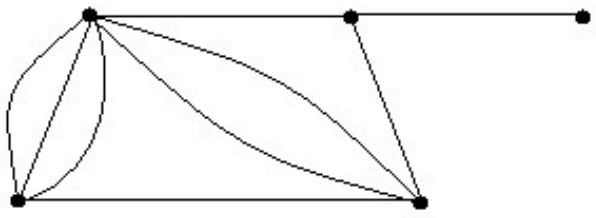

Figura 8

5. Un subgrafo de recubrimiento H de G, es un subgrafo tal que

$$V(H) = V(G)$$

Definition 30 *Supongamos que $V' \subseteq V$, no vacío. La gráfica de G cuyo conjunto de vices es V' y el conjunto de aristas es está formado por aquellas de $E(G)$, que tienen sus dos extremos en V', se llama subgráfica de G inducida por V' y se denota por $G\left[V'\right]$.*

Definition 31 *La subgráfica $G\left[V|V'\right] = G - V'$, de G es la subgráfica que se obtiene removiendo los vices de V' junto con las aristas incidentes.*

En particular, si $V' = \{v\}$, $G - v$ significa $G - \{v\}$.

Definition 32 *Supongamos que E' es un subconjunto no vacío de E. La subgráfica de G cuyo conjunto de vices es el conjunto de extremos en E' y cuyo conjunto de aristas es E', se llama la subgráfica de G inducida por E', la cual denotamos por $G\left[E'\right]$, esto es, la subgráfica de G inducida por el conjunto de aristas E'.*

Definition 33 *La subgráfica expansión de G cuyo conjunto de aristas es $E|E'$, se denota simplemente por $G - E'$, la cual se obtiene de G al remover las aristas en E'.*

Definition 34 *Similarmente, la gráfica obtenida de G, en la que se agregan las aristas de E', se denota por $G + E'$.*

Definition 35 *Dos subgráficas G_1 y G_2 se dicen ajenas, si no tienen vices en común.*

Definition 36 *Se dicen ajenas en cuanto a las aristas, si no tienen aristas en común.*

Definition 37 *La unión $G_1 \cup G_2$, de G_1 y G_2, es la subgráfica donde el conjunto de vices es $V(G_1) \cup V(G_2)$ y el conjunto de aristas es $E(G_1) \cup E(G_2)$.*

Definition 38 *En el caso de gráficas ajenas, la unión se suele representar por $G_1 + G_2$.*

Definition 39 *La intersección $G_1 \cap G_2$, se define análogamente, solo se pide que, al menos, tengan un vice en común, para que la intersección sea no vacía.*

De la relación entre la suma de los grados y el número de aristas, en una gráfica, se tiene el siguiente:

Corollary 40 *En cualquier gráfica, el número de vices de grado impar es par.*

Definition 41 *Una gráfica G es k-regular si*

$$d(v) = k$$

para todo $v \in V(G)$.

Example 42 *Las gráficas completas K_n y las gráficas bipartitas completas $K_{m,n}$ son regulares. Tambios k-cubos. En gráfica se tiene el 3-cubo, con 8 vices.*

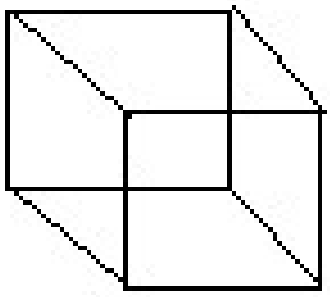

Figura 9

2.3. Caminos, trayectorias y rutas

Definition 43 *Un camino (walk) en una gráfica G es una sucesión finita, no nula, de vices y aristas, alternadas. El camino de v_0 a v_k, denotado por (v_0, v_k), es un camino con $k+1$ vices y k aristas:*

$$W = v_0 e_1 v_1 e_2 v_2 \dots e_k v_k$$

cuyos tinos son sucesivamente, vices y aristas, tal que, los extremos de e_i son v_{i-1} y v_i, con $1 \leq i \leq k$. El origen es v_0, el final es v_k, y los vices internos son $v_1 \dots v_{k-1}$. La longitud del camino es k.

Example 44 *En la gráfica se resalta el camino*

$$W = v_1 e_1 v_2 e_2 v_3 e_3 v_4 e_4 v_5 e_5 v_3 e_6 v_2 e_7 v_6$$

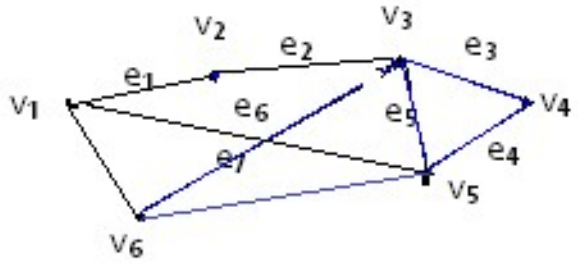

Figura 10

Definition 45 *Un camino cerrado tiene longitud positiva y su origen y fin son el mismo vice.*

Definition 46 *Un camino euleriano es un camino que pasa por cada arista una y solo una vez.*

Definition 47 *Un ciclo o circuito euleriano es un camino cerrado que recorre cada arista exactamente una vez.*

El problema de encontrar dichos caminos fue discutido por primera vez por Leonhard Euler, en el famoso problema de los puentes de Königsberg.

Definition 48 *Un paseo (trail) es un camino donde las aristas son diferentes.*

Definition 49 *Un trayecto o ruta (path) es un paseo donde todos los vices son distintos.*

Definition 50 *Un trayecto cerrado con origen y vices internos, distintos es un ciclo.*

Example 51 *Se ilustra un trayecto cerrado y un ciclo:*

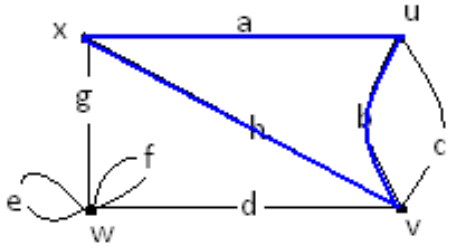

Figura 11

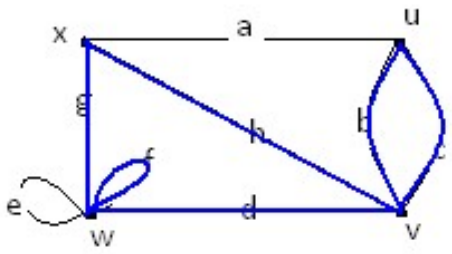

Figura 12

Definition 52 *Un grafo es conexo si, para cualquier par de sus vices u y v, exite una* (u,v)*-trayectoria en el grafo.*

Definition 53 *Una subgráfica H es maximal respecto a una propiedad, si H cumple la propiedad y no es subgráfica propia de otro grafo que cumpla la propiedad.*

Definition 54 *Un subgrafo H es minimal con respecto a una propiedad, si H cumple la propiedad y no existe un subgrafo propio de H que cumpla la propiedad.*

Definition 55 *Una componente de una gráfica es un subgrafo conexo maximal.*

Corollary 56 *Una gráfica es conexa si tiene exactamente una componente.*

Definition 57 *Una gráfica con más de una componente se denomina disconexo.*

Definition 58 *Una gráfica es bipartita si el conjunto de sus vices se puede partir en dos subconjuntos tales que cada arista tiene un extremo en cada uno de esos subconjuntos.*

Theorem 59 *Una gráfica es bipartita si y sólo si contiene ciclos no impares.*

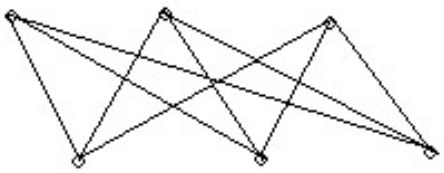

Figura 13

Example 60

Definition 61 *Un árbol es un grafo acíclico conexo.*

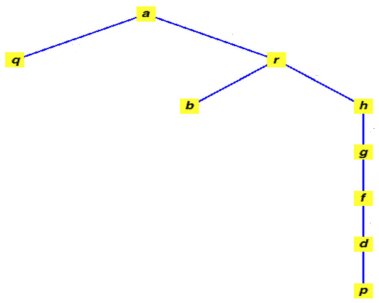

Figura 14

Example 62

Proposition 63 *Si T es un árbol entonces para cualquier par, u y v, de sus vices, existe una única trayectoria* (u,v).

Proposition 64 *Si T es un árbol no trivial, entonces existen al menos dos vices distintos de grado uno.*

Proposition 65 *Si T es un árbol con n vices, entonces el número de aristas es* $n-1$.

Definition 66 *Un puente en un grafo es una arista e tal que el número de componentes de* $G-e$ *es mayor que el número de componentes de G. En particular, si G es conexo entonces e es un puente de G si y solo si* $G-e$ *es disconexo.*

Example 67 *La arista c es un puente de la gráfica siguiente.*

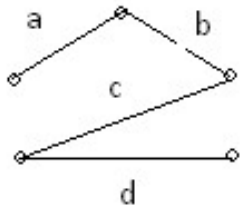

Figura 15

3. PROBLEMAS Y ALGORITMOS

3.1. Problema del conector

Se tienen n ciudades y se quiere disena red de comunicación conectando a todas ellas. Para ciertos pares de ciudades se puede construir una vía de comunicación directa entre ellas, y se conoce el costo de tal vía. El problema del conector consiste en determinar quias de comunicación directa deben construirse, con el fin de que todas la n ciudades queden conectadas a un costo total mínimo.

Es decir, dado un grafo conexo G y una función de costo $c : E(G) \to \Re$, tal que $c(e) > 0$ para toda $e \in E(G)$, el problema del conector consiste en hallar un subgrafo de recubrimiento conexo H tal que

$$c(H) = \sum_{e \in E(H)} c(e) \tag{7}$$

sea mínimo.

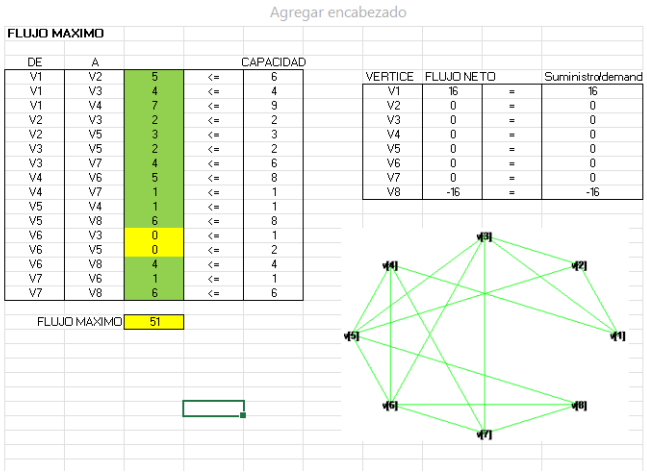

Figura 17: Solución hoja de cálculo

3.2. Problema del flujo máximo

Example 68 *Para la red que se muestra a continuación, use el algoritmo de la ruta de flujo máximo para encontrar el patrón de flujo que da el flujo máximo desde la fuente (source) hasta el sumidero (sink), dado que la capacidad del arco desde el nodo i al nodo j es el número del nodo i más cercano a lo largo del arco entre estos nodos.*

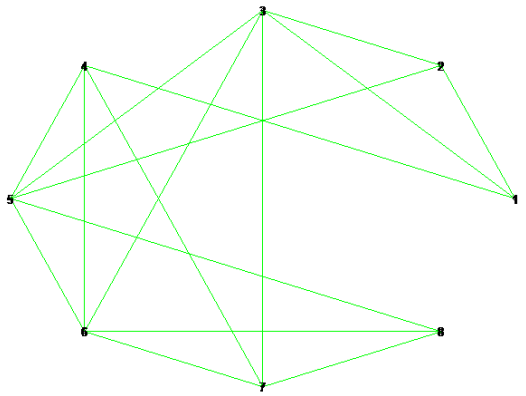

Figura 16

Exercise 69 *Reproduce en una hoja de cálculo el problema de la figura y dibuja la ruta máxima hallada.*

3.3. Ruta más corta

Cuando cada arista e de G tiene asociado su peso $w(e)$, tenemos la siguiente:

Definition 70 *Una gráfica G, junto con sus pesos en las aristas se denomina gráfica ponderada (weighted graph).*

Definition 71 *Si H es una subgráfica de una gráfica ponderada G, el peso*

$$w(H) = \sum_{e \in H} w(e)$$

es decir, el peso de H es la suma de los pesos de sus aristas.

Muchos problemas de optimización de cantidades, tienen que ver con encontrar una subgráfica de una gráfica ponderada, de cierto tipo con mínimo (máximo) peso, como el problema de la ruta más corta, que dice que dada una red (ferroviaria) que conecta a varios pueblos o ciudades, se busca determinar la ruta más corta, entre dos pueblos en particular, en la red.

Lo que significa, encontrar en una gráfica ponderada, una ruta de mínimo peso, que conecte dos vices u_0 y v_0. Los pesos representan las distancias por la vía entre dos pueblos distintos, conectados directamente, es decir entre vices distintos adyacentes, por tanto con distancia no negativa.

Example 72 *En la gráfica se resalta la subgráfica que es la (u_0, v_0) ruta de mínimo peso*

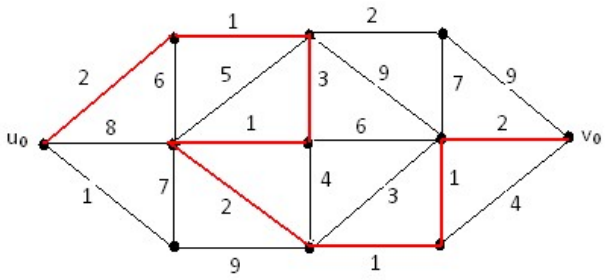

Figura 18

Notation 73 *La longitud de una subgráfica ponderada se representa por $w(H)$.*

Definition 74 *El mínimo peso de una (u,v) ruta es la distancia entre u y v, la cual se denota por $d(u,v)$.*

En las definiciones anteriores y en la formulación del problema de la ruta más corta, se asume que la gráfica *G* es simple.

Example 75 *Usted necesita hacer un viaje en automóvil a un pueblo que usted nunca antes ha visitado. Por consiguiente, está estudiando un mapa para determinar la ruta más corta a su destino. Dependiendo de la ruta que usted elija, hay otros cinco pueblos (llámelos A, B, C, D, E) a trave los cuales usted podría pasar en el camino. El mapa muestra la distancia en millas a lo largo de cada camino que conecta directamente dos pueblos sin ningún pueblo intermedio. Estos números se resumen en la tabla siguiente, donde un guión indica que no existe camino que conecte directamente estos dos pueblos, sin pasar por cualquier otro pueblo.*

	Millas entre los Pueblos Cercanos					
Pueblo	*A*	*B*	*C*	*D*	*E*	*Destino*
Origen	40	60	50	–	–	–
A		10	–	70	–	–
B			20	55	40	–
C				–	50	–
D					10	60
E						80

Figura 19

1. Formule este problema como un problema de ruta más corta dibujando una red donde los nodos representen pueblos, los arcos representen caminos y los números indican la longitud de cada arco en millas.
2. Identifique la ruta más corta.
3. Si cada número en la tabla representara su costo (en U.M.) por manejar su automóvil de un pueblo al próximo, ¿daría ahora la respuesta en la parte b su ruta a costo mínimo?
4. Si cada número en la tabla representara su tiempo (en minutos) por manejar su automóvil de un pueblo al próximo, ¿daría la respuesta en la parte b a su ruta a tiempo mínimo?

Definition 76 *Un algoritmo es un conjunto ordenado de operaciones sistemáticas que permite hacer un cálculo y hallar la solución de un tipo de problemas.Tambiuede decirse que es un conjunto de instrucciones o reglas definidas y no-ambiguas, ordenadas y finitas que permite solucionar un problema, realizar un cómputo, procesar datos o llevar a cabo otras tareas o actividades (Real Academia de la Lengua Española, https://dle.rae.es/algoritmo, fecha de consulta 02/09/2021)*

Definition 77 *árbol de expansión o recubrimiento es un subgrafo de expansión o recubrimiento acíclico.*

Algorithm 78 *Dijkstra*

1. *Sea*

$$\begin{aligned} l(u_0) &= 0 \\ L(v) &= \infty,\ v \neq u_0 \\ S &= \{u_0\} \\ i &= 0 \end{aligned}$$

2. *Para cada* $v \in \bar{S}_i$, *remplace* $l(v)$ *por* $\min\{l(v), l(u_i) + w(u_i v)\}$. *Calcule*

$$\min_{v \in \bar{S}_i} \{l(v)\}$$

y sea u_{i+1} *un vice para el cual este mínimo se alcanza. Sea* $S_{i+1} = S_i \cup \{u_{i+1}\}$.

3. *Si* $i = \nu - 1$, *pare. Si* $i < \nu - 1$, *remplace i por* $i+1$ *y vaya al paso 2.*

Cuando finaliza el algoritmo, la distancia de u_0 a v está dada por el valor final de $l(v)$. Si el inters determinar la distancia a un vice específico v_0, nos detenemos en cuanto algún u_j es igual a v_0.

En el siguiente diagrama se presenta el algoritmo de Dijkstra.

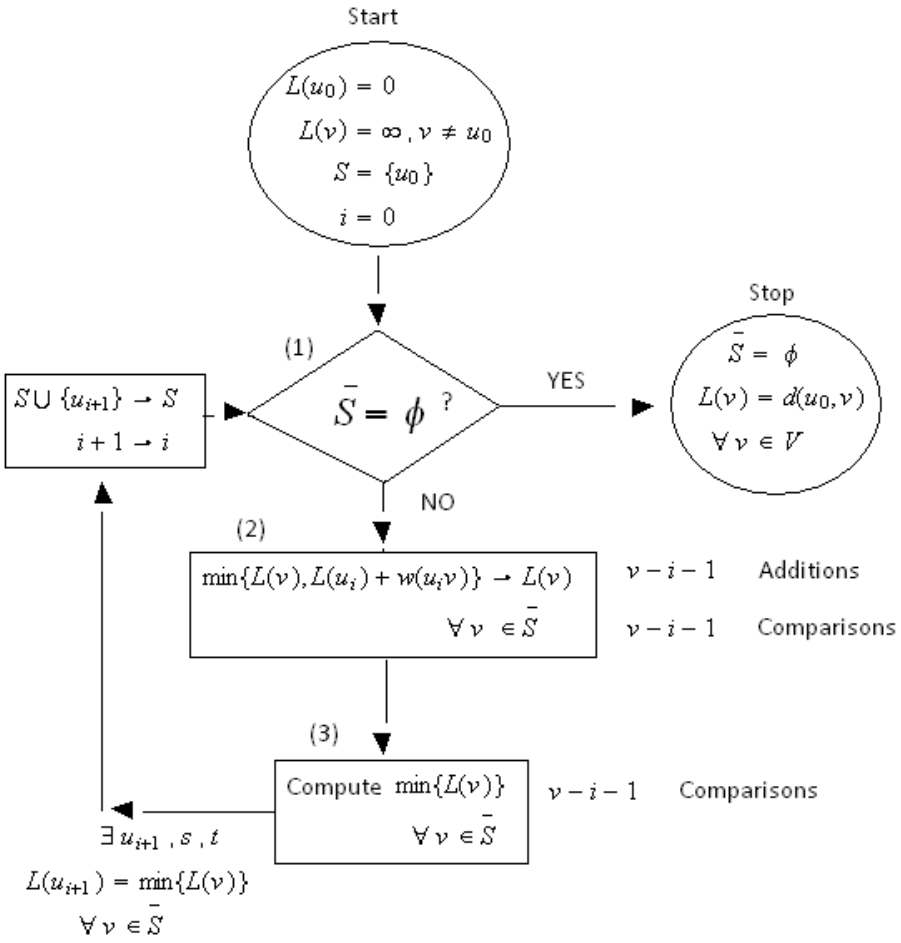

Figura 20

3.4. árboles de expansión a mínimo costo (MST)

Definition 79 *Arbol de expansión*

Example 80 *Encuentre el árbol de expansión a costo mínimo MST, de la red con nodos A, B, C, D, E, F y G. Con la asignación de pesos para las aristas, como se ve en la figura.*

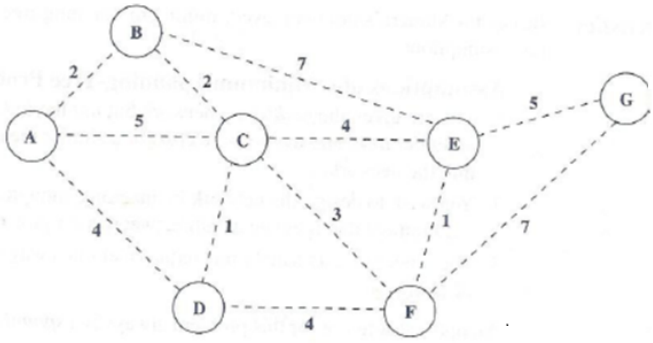

Figura 21: Red ponderada para el ejemplo

Example 81 *Encontrar el MST por el algoritmo Kruskal, para el grafo*[1]

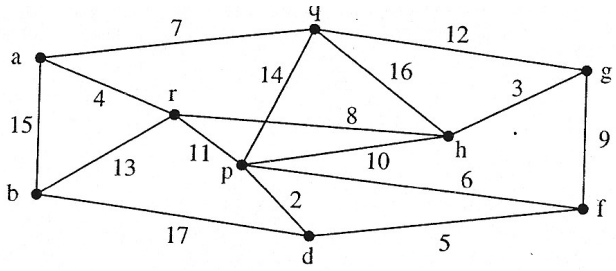

Figura 22

Solution 82 *A continuación se muestra el MST, cuya longitud es 51.*

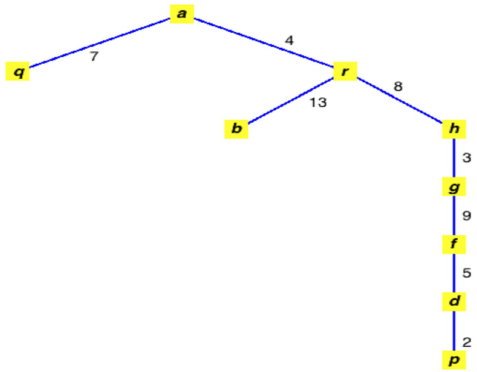

Figura 23

Exercise 83 *(Problema 3.20, Hillier, at.al. 2002) Un avión carguero tiene tres compartimientos para guardar la carga: al frente, al centro y atrás. Estos compartimientos tienen límites de capacidad, tanto en el peso como en el espacio, como se resume en seguida.*

[1] Tomado de Cook, W. et.al. (1997) Combinatorial Optimization, p. 17

Compartimiento	*Capacidad de Peso*	*Capacidad de Espacio*
	(Tons)	*(Pies Cúbicos)*
Frente	*12*	*7,000*
Centro	*18*	*9,000*
Atrás	*10*	*5,000*

Además, el peso de la carga en los compartimientos respectivos debe tener la misma proporción que la capacidad de peso de ese compartimiento para mantener el equilibrio del avión.

Se han propuesto las siguientes cuatro cargas para su embarque en un próximo vuelo considerando el espacio disponible. Puede aceptarse cualquier parte de estas cargas. El objetivo es determinar cuánto (si es posible) de cada carga debe aceptarse y cómo distribuir cada una entre los compartimientos para maximizar la utilidad total del vuelo.

Carga	*Peso*	*Volumen*	*Utilidad*
	(Tons)	*(Pies Cúbicos)*	*(\$/ Tons)*
1	*20*	*500*	*320*
2	*16*	*700*	*400*
3	*25*	*600*	*360*
4	*13*	*400*	*290*

1. *Este es un problema mixto de distribución de recursos pero tambincluye un número pequeño de restricciones de requerimientos fijos. Construya una tabla de parámetros que incluya todo excepto las restricciones de requerimientos fijos. Luego, construya por separado una tabla de parámetros con las restricciones de requerimientos fijos.*

2. *Formule y resuelva un modelo de programación lineal en una hoja de cálculo.*

3. *Exprese el modelo en forma algebraica.*

CONTINUAR AQUI CON LA SOLUCION DEL EJERCICIO

3.5. árboles de expansión mínima (MST) y programación lineal

Hay una conexión interesante entre los árboles de expansión mínima y la programación lineal, esto es, hay un problema de programación lineal para el cual cada árbol de expansión mínima proporciona una solución óptima. Considere el siguiente problema de programación lineal:

$$\begin{array}{lll} \text{Minimizar:} & c^T(x) & \\ \text{Sujeta a:} & & \\ & x(\gamma(S)) \leq |S|-1 \quad \text{para todo } S & \phi \neq S \subset V \\ & x(E) = |V|-1 & \\ & x_e \geq 0 \quad \text{para todo} & e \in E \end{array}$$

Theorem 84 *Sea x^0 el vector característico de un MST con respecto a los costos c_e. Entonces x^0 es una solución óptima del problema anterior.*

Proof 85 *Se reformula el problema para tener un manejo más fácil, como sigue*

$$\begin{array}{lll} \textit{Minimizar:} & c^T x & \\ \textit{Sujeta a:} & & \\ & x(A) \leq |V| - \kappa(A) \quad \textit{para todo} & A \subset E \\ & x(E) = |V|-1 & \\ & x_e \geq 0 \quad \textit{para todo} & e \in E \end{array}$$

La afirmación es que ambos problemas tienen las mismas soluciones factibles, y de ahí que las mismas soluciones óptimas. Cada restricción de la forma

$$x(\gamma(S)) \leq |S|-1$$

es consecuencia de

$$x(A) \leq |V| - \kappa(A)$$

tomado $A = \gamma(S)$. Nótese que

$$\kappa(\gamma(S)) \geq |V \setminus S| + 1$$

Recíprocamente, cada restricción

$$x(A) \leq |V| - \kappa(A)$$

es consecuencia de una combinación de las restricciones

$$\begin{array}{l} x(\gamma(S)) \leq |S| - 1 \quad \textit{para todo } S \quad \phi \neq S \subset V \\ x(E) = |V| - 1 \end{array}$$

Sea $A \subset E$ y $S_1, \ldots S_k$ los conjuntos de nodos de las componentes de la subgráfica (V,A). Entonces

$$x(A) \leq \sum_{i=1}^{k} x(\gamma(S_i)) \leq \sum_{i=1}^{k} (|S_i| - 1) = |V| - k$$

Es suficiente probar que x^0 es óptima para el problema reformulado, es más, que es suficiente probar que esto es cierto cuando x^0 es el vector característico del árbol de expansión T generado por el algoritmo Kruskal. La optimalidad del vector característico se prueba mostrando que dicho algoritmo puede usarse para calcular una soluci ón factible del problema dual que satisface la holgura complementaria con x^0. Para facilitar la construcción del problema dual, remplazamos el objetivo de minimizar $c^T x$, por maximizar $-c^T x$. De ahí que el problema dual es

$$\begin{array}{lll} \textit{Minimizar:} & \sum_{A \subseteq E} (|V| - \kappa(A)) y_A & \\ \textit{Sujeta a:} & & \\ & \sum (y_A : e \in A) \geq -c_e & \textit{para todo} \quad e \in E \\ & y_A \geq 0 & \textit{para todo} \quad A \subset E \end{array}$$

Nótese que no se requiere que y_E sea no-negativo. Sea $e_1, \ldots, e_m$ el orden en el cual el algoritmo Kruskal considera los arcos. Sea

$$R_i = \{e_1, \ldots, e_i\}$$

para $1 \leq i \leq m-1$ y

$$y^0_{R_i} = c_{e_{i+1}} - c_{e_i}$$

para $1 \leq i \leq m-1$ y

$$y^0_{R_m} = -c_{e_m}$$

Se sigue del orden en los arcos que

$$y^0_A \geq 0$$

para $A \neq E$. De la primera restricción del problema dual, con $e = e_i$ se tiene

$$\sum \left(y^0_A : e \in A\right) = \sum_{j=i}^{m} y^0_{R_j} = \sum_{j=i}^{m} \left(c_{e_{i+1}} - c_{e_i}\right) - c_{e_m} = -c_{e_i} = c_e$$

En otras palabras, todas las restricciones se cumplen con igualdad. Falta mostrar que y^0 es una solución factible del problema dual, y que por holgura complementaria, la condición

$$x^0_e > 0$$

implica que la igualdad en la respectiva restricción dual, se satisface. Finalmente, falta verificar que

$$y^0_A > 0$$

implica que x^0 satisface la primera restricción, con igualdad. Para ello, se sabe que

$$A = R_i$$

para alguna i. Si la restricción no se cumple con igualdad, entonces hay una arista en R_i, cuya adición a

$$T \cap R_i$$

reduce el número de componentes de

$$(V, T \cap R_i)$$

Pero una arista de ese tipo tendría sus extremos en dos componentes distintas de $(V, T \cap R_i)$, de ahí que se habría agregado a T, debido al algoritmo Kruskal. Por lo tanto, x^0 y y^0 satisfacen las condiciones de holgura complementaria. De donde se tiene que x^0 es una solución óptima del problema reformulado y el problema original.

Remark 86 *Dado que cualquier ábol de expansión que proporciona una soluci ón óptima del problema de programación lineal debe ser un MST, y debido a que la demostración anterior solo utiliza el hecho de que T fue generado por el algoritmo Kruskal, esa prueba es una prueba de que dicho algoritmo calcula un MST.*

Problem 87 *DINERO EN MOVIMIENTO*[2] *Jake Nguyen pasa nerviosamente una mano a trave su pelo, alguna vez peinado finamente, se afloja al mismo tiempo su corbata de seda perfectamente anudada y se frota sus manos sudadas en sus, alguna vez pantalones impecablemente planchados. Hoy no ha sido ciertamente un buen d ía.*
Durante los últimos meses, Jake había oído rumores que circulaban en Wall Street - rumores de los labios de banqueros inversionistas y corredores de la bolsa conocidos por su franqueza. Hacían murmuraciones acerca de un próximo colapso de la economía japonesa - las murmuraciones eran porque pensaban que haciendo públicos sus temores acelerarían el derrumbamiento.
Hoy, sus mismos temores se han hecho realidad. Jake y sus colegas se reúnen alrededor de una pequeña televisión dedicada exclusivamente al canal de Bloomberg (canal 315 de Sky). Jake mira fijamente con escepticismo y escucha los horrores que tienen lugar en el mercado japon El mercado japonstá arrastrando todos los mercados financieros de los demás países Asiático Orientales en su vertiginosa caída. Jake se queda paralizado. Como gerente de la inversión extranjera Asiática para Grant Hill Associates, una pequeña empresa de inversión de la Costa Oriental que se especializa en la comercialización del dinero. Jake lleva la responsabilidad personal por cualquier impacto negativo de este colapso. Y Grant Hill Associates experimentará impactos negativos.
Jake no había hecho caso de las murmuraciones que advertían un derrumbamiento japon En cambio, abía aumentado en gran manera el riesgo de Grant Hill Associates de quedar fuera del mercado japon Debido a que el mercado japone había comportado mejor de lo esperado durante el último año, Jake había aumentado las inversiones en Japón de 2.5 millones de a 15 millones de dólares, hace sólo un mes. En ese momento, la paridad del dólar era de 80 yens.
No más. Jake comprende que la devaluación de hoy del yen significa que un dólar equivaldrá a 125 yens. El podrá liquidar las inversiones sin pida en yens. La pida de dólares cuando convierta los yens en dólares americanos sería enorme. Jake respira profundamente, cierra los ojos y mentalmente se prepara para reparar los severos daños.
La meditación de Jake es interrumpida por una potente voz que lo llama desde una esquina de su gran oficina. Grant Hill, el presidente de Grant Hill Associates, vocifera "¡Nguyen, esto es un infierno!"
Jake da un brinco y mira renuentemente hacia la esquina de la oficina donde aparece la figura de Grant Hill hecho una furia. Jake entonces se alisa el cabello, se aprieta la corbata, y camina aprisa por la oficina.
Grant Hill le sale al paso a Jake, lo mira fijamente a los ojos y le sigue gritando, No me diga ni una sola palabra, Nguyen! No hay pretextos; simplemente arregle este desastre! ¡Saque todo nuestro dinero de Japón! ! Mi intestino me dice que e es sólo el principio! Invierta el dinero en bonos americanos confiables! ¡AHORA! Y no olvide sacar nuestro dinero en efectivo de Indonesia y Malasia!
Jake tiene bastante sentido común para no hacer ningún comentario. nclina su cabeza, gira sobre los tacones de sus zapatos, y prácticamente sale volando de la oficina.
Seguramente atrás de su escritorio, Jake empieza a formular un plan para sacar sus inversiones de Japón, Indonesia y Malasia. Sus experiencias de invertir en mercados extranjeros le han enseñado que jugar con millones de dólares, así como sacar dinero de un mercado extranjero es casi tan importante como sacar dinero del mercado. Los socios de la banca de Grant Hill Associates cobran diferentes honorarios por la transacción de convertir un tipo de dinero en otro y remitir grandes sumas de dinero alrededor del mundo. Y ahora, las cosas han empeorado, los gobiernos en el Asia Oriental han impuesto límites muy estrictos en la cantidad de dinero que un individuo o compañía puede cambiar del dinero domico a una divisa particular y pueda sacarlo del país. El objetivo de esta dramática medida es reducir la salida de inversiones extranjeras de esos países para prevenir un colapso total de las economías en la región. Debido a las posiciones en efectivo de Grant Hill Associates de 10.5 billones de rupias en Indonesia y 28 millones de ringgits en Malasia, junto con los yenes, no está claro cómo deben convertir esas divisas en dólares.
Jake quiere encontrar el mdo más efectivo en costos, para convertir estas posiciones en dólares. En el sitio Web de su compañía, uede encontrar siempre los tipos de cambio, en todo, momento para la mayoría de las monedas en el mundo, vea la tabla:

Tipos de cambio de divisas

[2]Tomado y adaptado de Hillier, F., Hillier, M., Hillier, and Lieberman, G. (2001) Caso 6.2

DE / A	*Yen*	*Rupia*	*Ringgit*	*US Dolar*	*C Dolar*	*Euro*	*Libra*	*Peso*
Yen japones	*1*	*50*	*0.04*	*0.008*	*0.01*	*0.0064*	*0.0048*	*0.0768*
Rupia hindu		*1*	*0.0008*	*0.00016*	*0.0002*	*0.000128*	*0.00096*	*0.001536*
Ringgit malasio		*1*	*0.2*	*0.25*	*0.16*	*0.12*		*1.92*
US Dolar				*1*	*1.25*	*0.8*	*0.6*	*9.6*
C Dolar					*1*	*0.64*	*0.48*	*7.68*
Euro						*1*	*0.75*	*12*
Libra britanica							*1*	*16*
Peso mexicano								*1*

EJEMPLO ALGORITMO KRUSKAL

Usando el comando de MAPLE

```
> restart : with(GraphTheory) :
> V := [A, B, C, D, E, F, G, H, J];
```

$$V := [A, B, C, D, E, F, G, H, J] \quad (1)$$

```
> P := {[{A, B}, 15], [{A, C}, 4], [{A, D}, 13], [{A, E}, 7], [{A, F}, 11], [{A, G}, 14], [{A,
    H}, 17], [{A, J}, 6], [{B, C}, 5], [{B, D}, 2], [{B, E}, 10], [{B, F}, 3], [{B, G}, 9],
    [{B, H}, 8], [{B, J}, 16], [{C, D}, 7], [{C, E}, 6], [{C, F}, 4], [{C, G}, 10], [{C, H},
    9], [{C, J}, 8], [{D, E}, 4], [{D, F}, 9], [{D, G}, 6], [{D, H}, 10], [{D, J}, 5], [{E,
    F}, 7], [{E, G}, 12], [{E, H}, 11], [{E, J}, 6], [{F, G}, 9], [{F, H}, 5], [{F, J}, 3],
    [{G, H}, 4], [{G, J}, 7], [{H, J}, 5]};
```

$$P := \{[\{A, B\}, 15], [\{A, C\}, 4], [\{A, D\}, 13], [\{A, E\}, 7], [\{A, F\}, 11], [\{A, G\}, 14], [\{A, H\}, 17], [\{A, J\}, 6], [\{B, C\}, 5], [\{B, D\}, 2], [\{B, E\}, 10], [\{B, F\}, 3], [\{B, G\}, 9], [\{B, H\}, 8], [\{B, J\}, 16], [\{C, D\}, 7], [\{C, E\}, 6], [\{C, F\}, 4], [\{C, G\}, 10], [\{C, H\}, 9], [\{C, J\}, 8], [\{D, E\}, 4], [\{D, F\}, 9], [\{D, G\}, 6], [\{D, H\}, 10], [\{D, J\}, 5], [\{E, F\}, 7], [\{E, G\}, 12], [\{E, H\}, 11], [\{E, J\}, 6], [\{F, G\}, 9], [\{F, H\}, 5], [\{F, J\}, 3], [\{G, H\}, 4], [\{G, J\}, 7], [\{H, J\}, 5]\} \quad (2)$$

```
> R := Graph(V, P);
```

$$R := \text{Graph 1: an undirected weighted graph with 9 vertices and 36 edge(s)} \quad (3)$$

```
> DrawGraph(R);
```

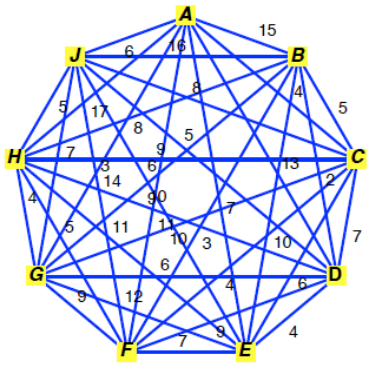

Figura 24

La tabla anterior indica que, por ejemplo, 1 yen japonquivale a 0.008 dólares americanos. Haciendo unas llamadas telefónicas, conoce los costos de la transacción que su compañía debe pagar por las grandes transacciones de divisas durante estos tiempos críticos, vea la tabla 2.

A / De	*Yen*	*Rupia*	*Ringgit*	*Dólar Americano*	*Dólar Canadiense*	*Euro*	*Libra*	*Peso*
Yen Japonés	---	0.5	0.5	0.4	0.4	0.4	0.25	0.5
Rupia Indonés		---	0.7	0.5	0.3	0.3	0.75	0.75
Ringgit			---	0.7	0.7	0.4	0.45	0.5
Malasiano				---	0.05	0.1	0.1	0.1
Dólar					---	0.2	0.1	0.1
Americano						---	0.05	0.5
Dólar							---	0.5
Canadiense								---
Euro C.E.E.								

Figura 26: Tabla 2. Costo por transacción (porcentaje)

```
> T := MinimalSpanningTree(R);
        T := Graph 2: an undirected weighted graph with 9 vertices and 8 edge(s)   (4)
> DrawGraph(T);
```

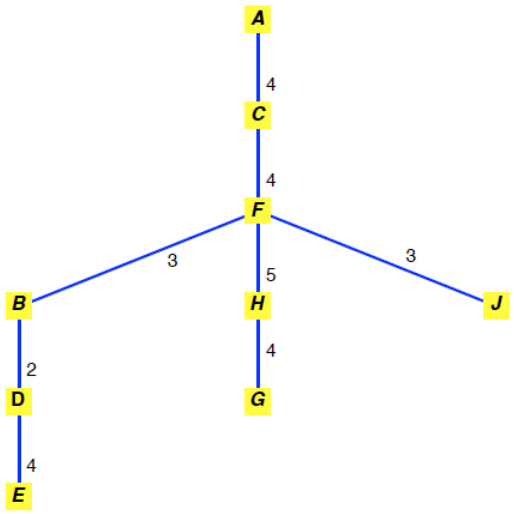

Figura 25

Jake nota que al cambiar una moneda por cualquier otra se produce el mismo costo de la transacción que para una conversión inversa. Finalmente, Jake averigua las cantidades máximas de dineros domicos que su compañía tiene permitido convertir a otros tipos de moneda en Japón, Indonesia y Malasia, vea la tabla 3.

A / De	Yen	Rupia	Ringgit	Dólar Americano	Dólar Canadiense	Euro	Libra	Peso
Yen Japonés	---	5,000	5,000	2,000	2,000	2,000	2,000	4,000
Rupia Indonés	5,0003 ,	----	2,000	200	200	1,000	500	200
Ringgit Malasiano	000	4,500	----	1,500	1,500	2,500	1,000	1,000

Figura 27: Tabla 3 Límites de transacciones en miles de dólares equivalentes

El problema de Jake se formula como un problema de flujo a costo mínimo. La red se muestra a continuación. Hay tres nodos de suministro – Yen, Rupia y Ringgit. Hay un nodo de demanda – el nodo Dólares Americanos.

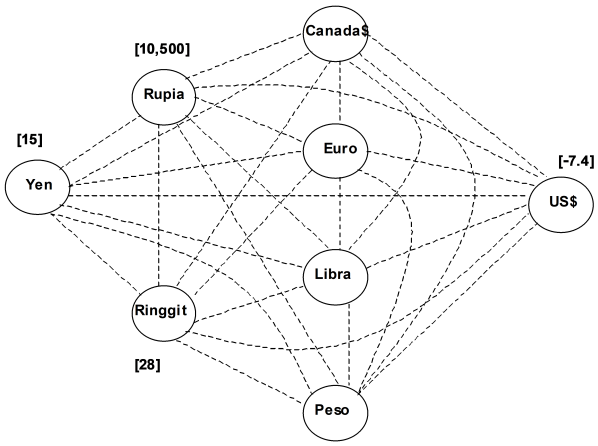

Figura 28

Exercise 88 *Complete la información faltante en la red anterior y encuentre la ruta de costo mínimo de los tres orígenes (yen, rupia y ringgit) al destino (USD).*

Solution 89 *La gráfica queda como sigue*

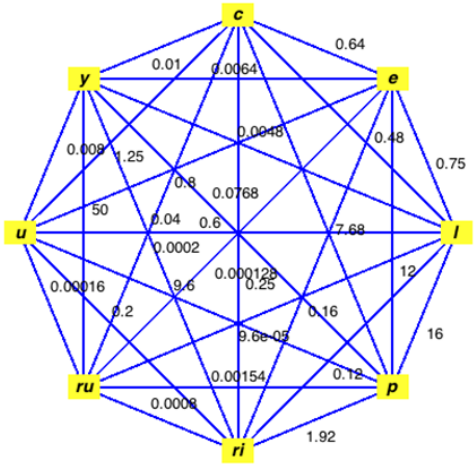

Figura 29

4. PROBLEMA DE EMPAREJAMIENTOS óPTIMOS

[3]

Definition 90 *Dada una gráfica $G = (V,E)$ n emparejamiento M es un subconjunto de aristas o lados con la propiedad de que ningún par de lados comparten un nodo.*

Definition 91 *El problema del emparejamiento consiste en encontrar el emparejamiento máximo.*

La cardinalidad de un emparejamiento máximo denominado completo o perfecto es

$$\frac{|V|}{2}$$

[3]Tomado de Papadimitriou, C. and Steiglitz K., p. 218

Considse una gráfica G junto con un emparejamiento fijo M de G, los lados de M se denominan emparejados y el resto libres.

Sea $[v,u]$ un lado emparejado entonces u es la pareja de v. Los nodos que no son incidentes en ningún lado emparejado se denominan expuestos. Los nodos restantes est?n emparejados.

Definition 92 *Una trayectoria $p = [u_1,u_2,...,u_k[$ se llama alternante si dos lados $[u_1,u_2]$ y $[u_3,u_4]$ son libres, mientras que $[u_2,u_3]$, $[u_4,u_5]$,..., $[u_{2j-1},u_{2j}]$ son emparejados.*

Definition 93 *Los vices que caen en una trayectoria alternante comienzan con uno expuesto y tiene rango impar en esta trayectoria se llaman exteriores y los de rango par se llaman interiores.*

Se dice que el emparejamiento en este caso es perfecto cuando su cardinalidad es

$$|V|/2$$

Otro problema es que dados los pesos de los lados, el reto es encontrar un emparejamiento de peso total m?ximo.

En el emparejamiento siguiente se marcan en gris dos vices exteriores. El resto son vices interiores

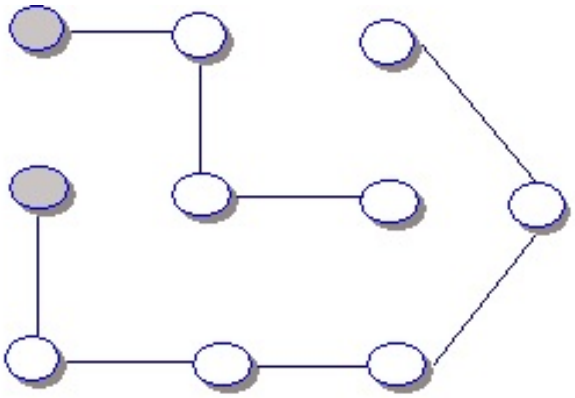

Figura 30

Figura 31

Example 94 *Encuentre un emparejamiento aumentado para la siguiente gráfica*

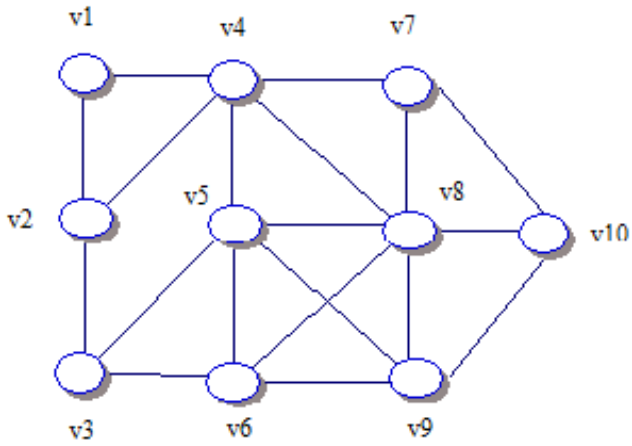

Figura 32

Solution 95 *Una solución quedaría como sigue:*

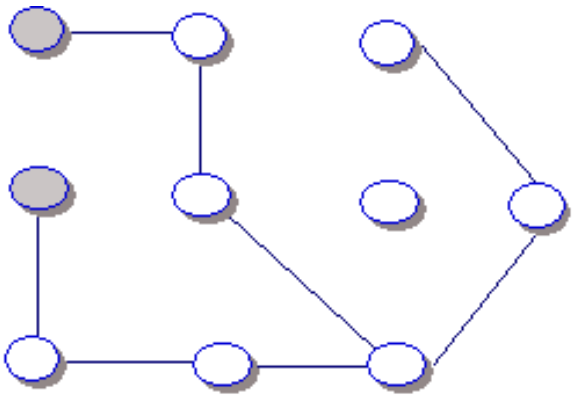

Figura 33

La trayectoria $p1 = [v_1, v_2, v_3, v_5, v_4, v_8[$ *es alternante y la trayectoria* $p2 = [v_1, v_4, v_5, v_6, v_8, v_7, v_9, v_{10}[$ *es aumentada porque los nodos* v_1 *y* v_9 *son expuestos.*

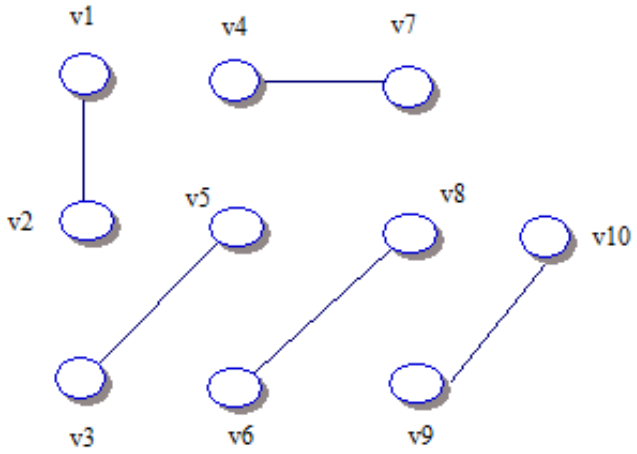

Figura 34

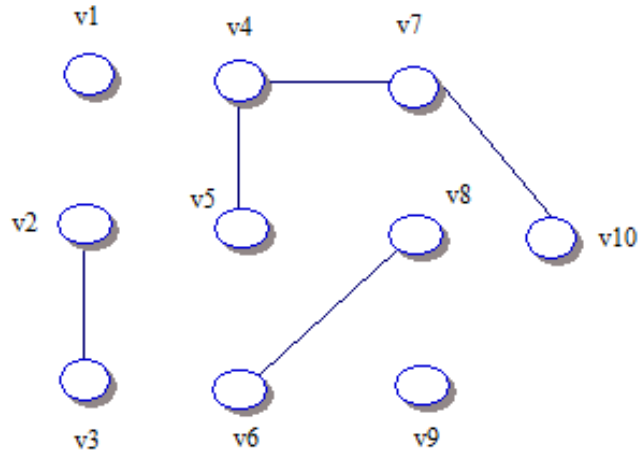

Figura 35

Lemma 96 *Sea P el conjunto de v?rtices en una trayectoria aumentada M en la gr?fica G bipartita con respecto al emparejamiento M. Entonces el emparejamiento* $M' = M \oplus P$ *es un emparejamiento con cardinalidad* 1

Exercise 97 *Encuentre una trayectoria aumentada para el emparejamiento mostrado en la definición anterior.*

Theorem 98 *Un aumento M en una gráfica G es máxima si y solo si no hay una trayectoria aumentada en G con respecto a M*

Remark 99 *Debido a que el problema de emparejamiento para gr?ficas bipartitas es un caso especial del problema de emparejamiento, se aplica el teorema anterior y se resuelve descubriendo repetidamente una ruta de aumento p con respecto a la coincidencia actual M, aumentando el emparejamiento actual a* $M \oplus P$

5. PROBLEMA DEL CARTERO CHINO [4]

El trabajo de un cartero es recoger el correo en la oficina postal, entregarlo a los destinatarios, y regresar a la oficina postal. Se espera que recorra cada calle de su área al menos una vez, de modo que camine lo menos posible.

Definition 100 *En una gráfica ponderada, se define el peso del paseo ponderado*

$$v_0 e_1 v_1 \dots e_n v_0$$

como

$$\sum_{i=1}^{n} w(e_i)$$

El problema del cartero chino es encontrar un paseo con mínimo peso en una gráfica conectada ponderada, con pesos no-negativos, óptimo.

Si la gráfica *G* es euleriana, es decir que contiene un paseo euleriano, entonces cualquier paseo euleriano de *G* es un paseo óptimo. Recordamos que un paseo se denomina euleriano si recorre cada arco o arista exactamente una vez.

El problema en cuestión se resuelve con el algoritmo Fleury, el cual construye un paseo euleriano a partir de un trayecto, sujeto a la única condición de que, en cada etapa, un arco puente de la subgráfica no recorrida se toma si y sólo si no hay alternativa.

[4] Tomado de Bondy and Murty, p. 62

5.1. Algoritmo Fleury

El algoritmo de Fleury se usa para mostrar la ruta de Euler o el circuito de Euler a partir de un gr?fico dado. En este algoritmo, comenzando desde un borde, intenta mover otros v?rtices adyacentes eliminando los bordes anteriores. Usando este truco, el gr?fico se vuelve m?s simple en cada paso para encontrar la ruta o circuito de Euler.

1. Escoger un vice arbitrario v_0 y establecer el trayecto

$$W_0 = v_0$$

2. Supóngase que el trayecto

$$W_i = v_0 e_1 v_1 \dots e_i v_i$$

se ha escogido, entonces elegir

$$e_{i+1}$$

de

$$E \setminus \{e_1, \dots e_i\}$$

de modo tal que
i) e_{i+1} es incidente con v_i
ii) A menos que no haya una alternativa, e_{i+1} no es una arista de corte (puente) de

$$G_i = G - \{e_1, \dots e_i\}$$

3. Detenerse cuando el paso 2 no puede implementarse más.

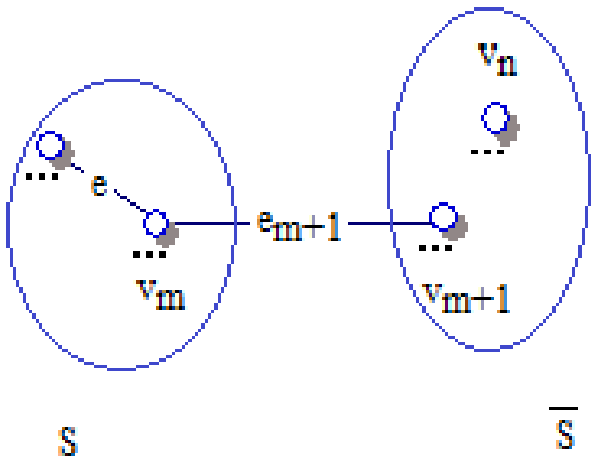

Figura 36

Por su definición el algoritmo de Fleury construye un trayecto en G.

Theorem 101 *Si G es euleriana, entonces un trayecto en G construido mediante el algoritmo de Fleury es un paseo euleriano de G.*

Example 102 *Se ilustra un ejemplo implementado con lenguaje C++, para el grafo:*

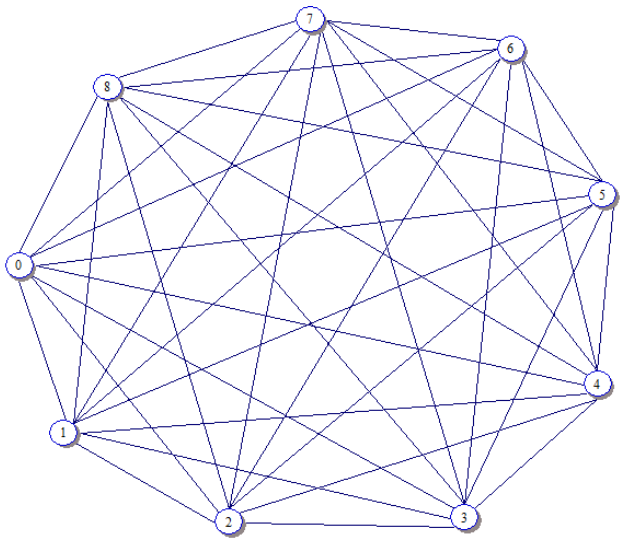

Figura 37

Las figuras 35 y 36 muestran la compilación en C++

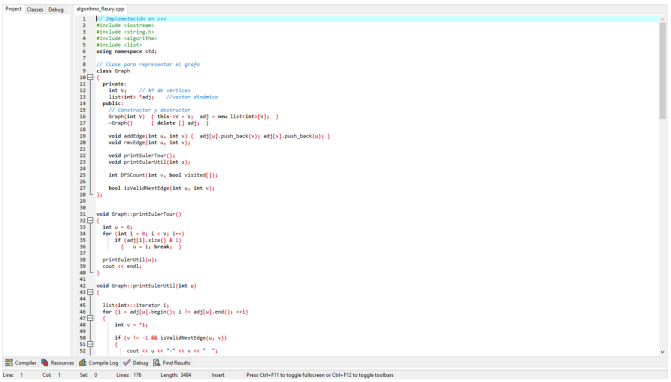

Figura 38

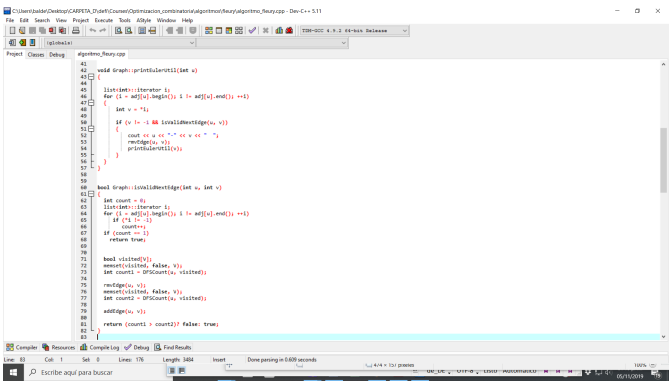

Figura 39

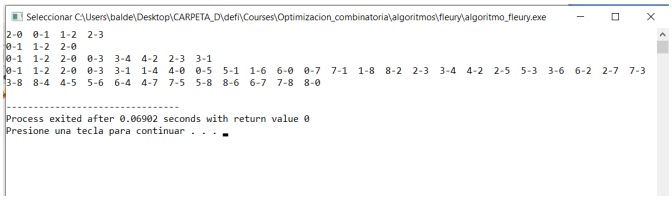

Figura 40

La ejecución del programa en C++ muestra cuatros circuitos eulerianos los cuales se ven en colores diferentes, vse las Figuras 38 y 39.

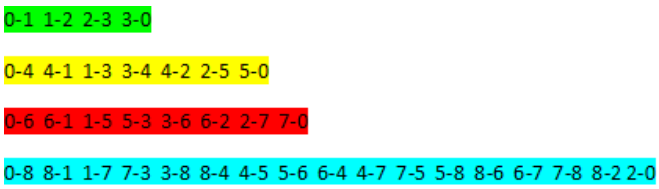

Figura 41

Otra solución se ve en la figura 40.

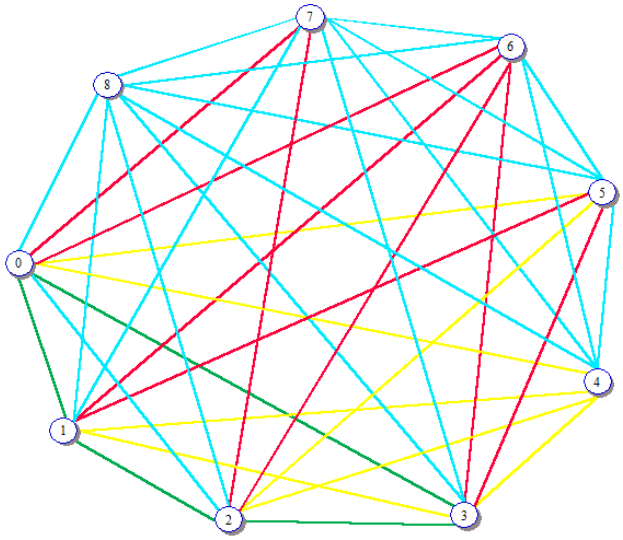

Figura 42

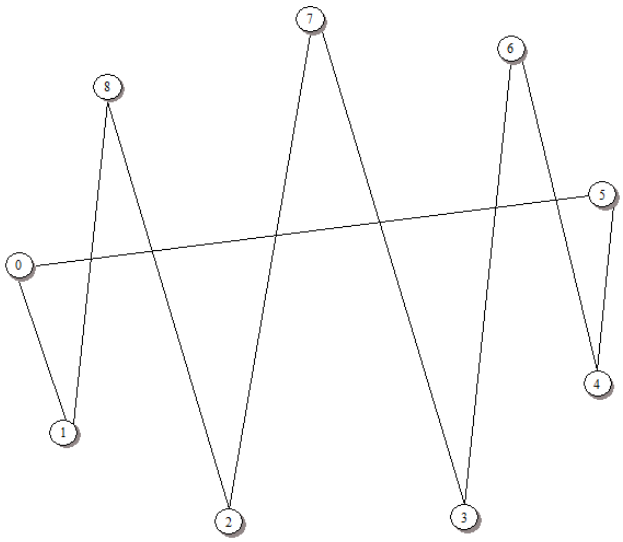

Figura 43

6. PROBLEMA DE LA MOCHILA

El problema de la mochila, comúnmente abreviado por KP (del inglnapsack problem) es un problema de optimización combinatoria, es decir, que busca la mejor solución entre un conjunto finito de posibles soluciones a un problema. Modela una situación análoga al llenar una mochila, incapaz de soportar más de un peso determinado, con todo o parte de un conjunto de objetos, cada uno con un peso y valor específicos. Los objetos colocados en la mochila deben maximizar el valor total sin exceder el peso máximo.

6.1. Fomulación

Maximizar: $\sum_{i=1}^{n} v_i x_i$

Sujeta a:

$\sum_{i=1}^{n} w_i x_i \leq W$

$0 \leq x_i \leq q_i$

Si $q_i = 1$ para $i = 1, 2, ..., n$ se dice que se trata del problema de la mochila 0-1.

Si uno o más de los q_i es infinito entonces se dice que se trata del problema de la mochila no acotado tambilamado a veces problema de la mochila entera.

En otro caso se dice que se trata del problema de la mochila acotado.

El siguiente ejemplo ilustra el caso de tres compartimientos.

Example 103 *(Problema 3.20, Hillier, at.al. 2002) Un avión carguero tiene tres compartimientos para guardar la carga: al frente, al centro y atrás. Estos compartimientos tienen límites de capacidad, tanto en el peso como en el espacio, como se resume en seguida.*

Compartimiento	*Capacidad de Peso (Tons)*	*Capacidad de Espacio (Pies Cúbicos)*
Frente	12	7,000
Centro	18	9,000
Atrás	10	5,000

Figura 44

Además, el peso de la carga en los compartimientos respectivos debe tener la misma proporción que la capacidad de peso de ese compartimiento para mantener el equilibrio del avión.

Se han propuesto las siguientes cuatro cargas para su embarque en un próximo vuelo considerando el espacio disponible.

Carga	*Peso (Tons)*	*Volumen (Pies Cúbicos)*	*Utilidad ($/Ton)*
1	20	500	320
2	16	700	400
3	25	600	360
4	13	400	290

Figura 45

Puede aceptarse cualquier parte de estas cargas. El objetivo es determinar cuánto (si es posible) de cada carga debe aceptarse y cómo distribuir cada una entre los compartimientos para maximizar la utilidad total del vuelo.

1. *Este es un problema mixto de distribución de recursos pero tambincluye un número peque restricciones de requerimientos fijos. Construya una tabla de parámetros que incluya todo excepto las restricciones de requerimientos fijos. Luego, construya por separado una tabla de parámetros con las restricciones de requerimientos fijos.*
2. *Formule y resuelva un modelo de programación lineal en una hoja de cálculo.*
3. *Exprese el modelo en forma algebraica.*

7. PROBLEMA DEL AGENTE VIAJERO

Consiste en visitar cierto número de ciudades, una sola vez y regresar al punto de partida. Conocidos los tiempos de viaje entre cada par de ciudades, se quiere hacer un itinerario para que se visite cada ciudad sólo una vez y emplear el menor tiempo posible en regresar al punto de partida. En tinos de grafos, se desea encontrar un ciclo hamiltoniano con peso mínimo en una gráfica completa ponderada. Un ciclo de esa naturaleza se le denomina óptimo. A diferencia del problema de la ruta más corta y del problema del conector, no se conoce un algoritmo eficiente para resolver el problema del agente viajero. Por lo que, se desea encontrar una solución razonablemente buena (no necesariamente óptima).

Un enfoque posible es encontrar un primer ciclo hamiltoniano C, despuuscar otro de menor peso que pueda sustituir al primero y así suscesivamente. Considere un ciclo hamiltoniano

$$C = v_1 v_2 \dots v_n v_1$$

Para todo i y j tales que

$$1 < i+1 < j < n$$

se puede obtener un nuevo ciclo hamiltoniano

$$C_{ij} = v_1 v_2 \dots v_i v_j v_{j-1} \dots v_{i+1} v_{j+1} v_{j+2 \dots} v_n v_1$$

quitando las aristas:

$$v_i v_{i+1}$$

y

$$v_j v_{j+1}$$

pero agregando

$$v_i v_j$$

y

$$v_{i+1} v_{j+1}$$

como se muestra en la siguiente figura:

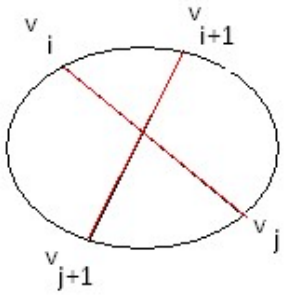

Figura 46: Tomado de Bondy and Murty (1976)

Figura 47

Si, para algunos i y j

$$w(v_i v_j) + w\left(v_{i+1} v_{j+1}\right) < w(v_i v_{i+1}) + w\left(v_j v_{j+1}\right)$$

el ciclo C_{ij} mejora, reduce el peso de C.

Despue realizar una sucesión de modificaciones como la explicada, se llegará a un ciclo que ya no pueda mejorarse por este mdo. Este ciclo final, no será óptimo, pero se supone razonablemente que será mucho mejor. Para mayor precisión, el procedimiento anterior puede realizarse varias veces, iniciando en con ciclos diferentes, en cada ocasión.

Example 104 *Considere la siguiente tabla de distancias (en millas), por avión, entre seis ciudades principales, Londres, Mexico City, New York, Paris, Peking y Tokyo*[5]

	L	MC	NY	PA	PE	T
L	–	56	35	2	51	60
MC	56	–	21	57	78	70
NY	35	21	–	36	68	68
PA	2	57	36	–	51	61
PE	51	78	68	51	–	13
T	60	70	68	61	13	–

Use el enfoque explicado, iniciando con el ciclo:

$$L, MC, NY, PA, PE, T$$

para encontrar un ciclo hamiltoniano desde L, de menor peso.

Solution 105 *La gráfica correspondiente a la tabla de distancias se ve en la siguiente figura:*

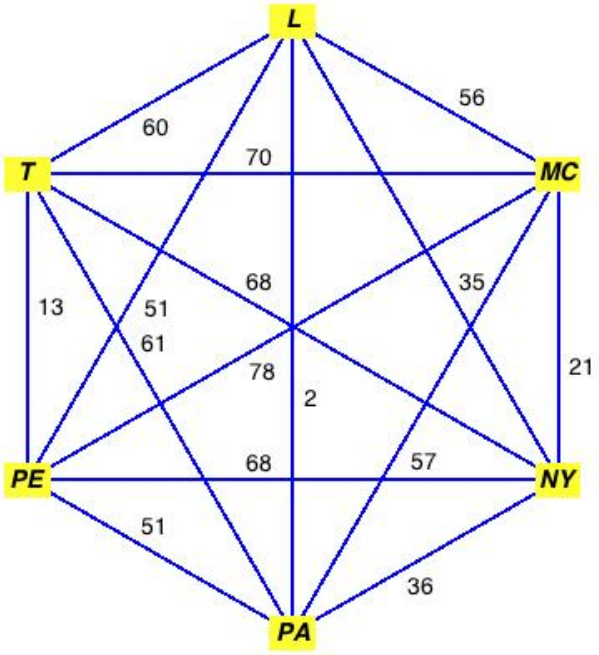

Figura 48: Red distancias en millas

Figura 49

el paseo hamiltoniano se resalta en la figura 48.

[5] Adaptado de Bondy and Murty (1976)

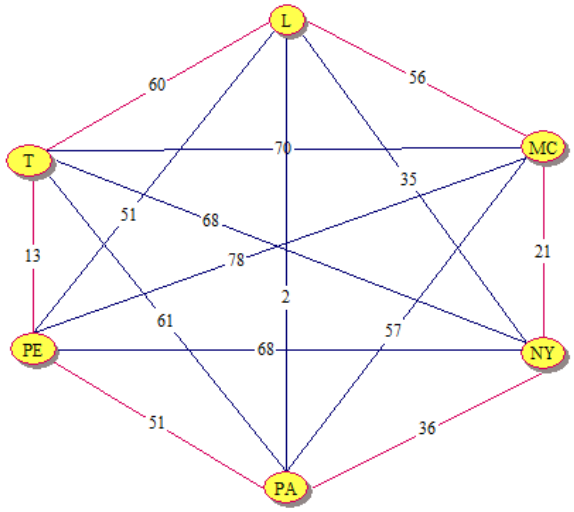

Figura 50: Solución

Figura 51

el cual se obtiene de las iteraciones.

7.1. Algoritmo inversión de sub-recorrido

Exercise 106 *Mejorar el camino euleriano NY - MC - L - T - PE - PA - NY con longitud total de 237 considerando la tabla de distancias siguiente*[6]*:*

Salida	*NY*	
	MC	*21*
	L	*56*
	T	*60*
	PE	*13*
	PA	*51*
	NY	*36*
Total		*237*

[6]Hillier and Lieberman, 2005, p. 623

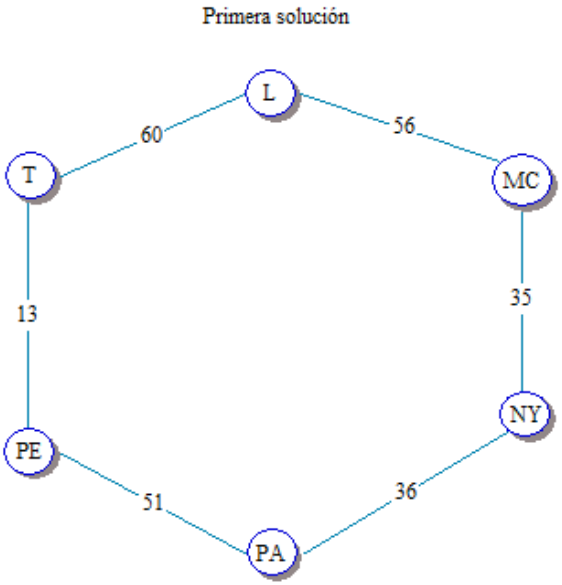

Figura 52

Solution 107

El recorrido inicial se mejora intercambiando los nodos PE y PA para generar el recorrido NY - MC - PE - T - L - PA - NY con longitud total de 224.

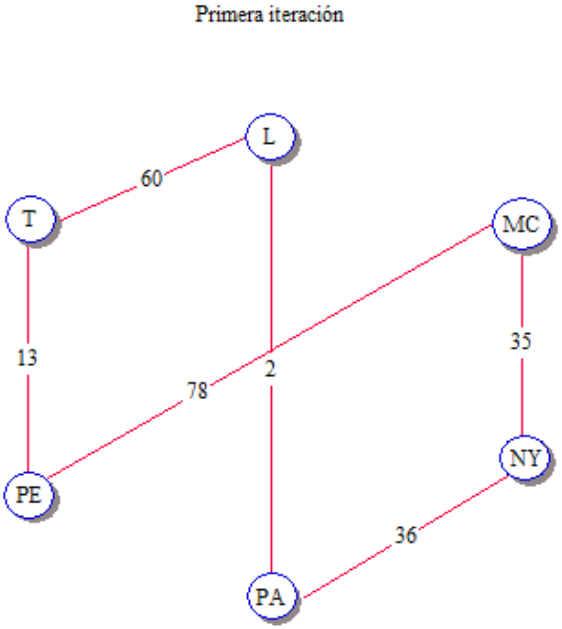

Figura 53

Salida	*NY*	
	MC	*35*
	PE	*78*
	T	*13*
	L	*60*
	PA	*2*
	NY	*36*
Total		*224*

La segunda iteración se obtiene intercambiando PA con MC con la que se logra una longitud total de 206: NY - PA - L - T - PE - MC - NY

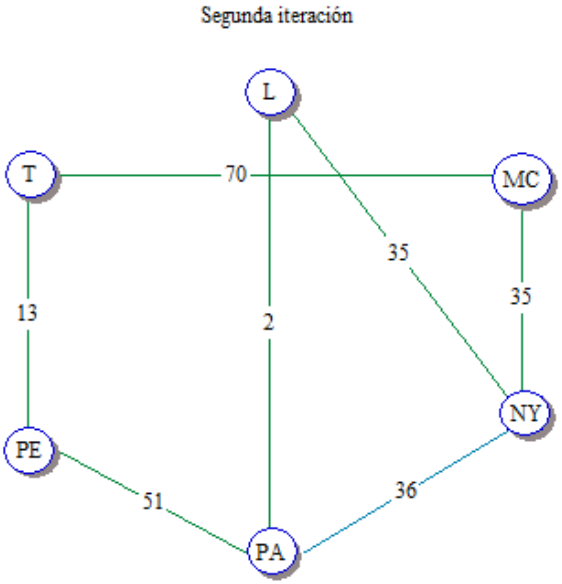

Figura 54

Salida	*NY*	
	L	*35*
	PA	*2*
	PE	*51*
	T	*13*
	MC	*70*
	NY	*35*
Total		*206*

Exercise 108 *Aplicar el algoritmo por inversión de sub-recorrido, al paseo que se inicia y termina en L: L- MC - NY - PA - PE - T - L*

Solution 109

i	*Paseo hamiltoniano*	*Aristas que se quitan*	*Aristas que se agregan*
0	L, MC, NY, PA, PE, T, L		
1	LPA, NY, MC, PE, T, L	LMC , $PAPE$	LPA , $MCPE$
2	L, PA, NY, MC, T, PE, L	TL , $MCPE$	LPE , MCT
3	L, PA, PE, T, MC, NY, L	LNY , $PAPE$	NYL , $PAPE$

7.2. Búsqueda Tabú

Resumen adaptado de Hillier and Lieberman (2005)
Inicio: con una solución inicial de ensayo.
Iteración: se usa un procemiento de búsqueda local apropiado para definir los movimientos factibles hasta el vecino local de la solución de ensayo actual. Se elimina de la consideración cualquier movimiento de la actual lista tabú a menos que el movimiento produzca una mejor solución que la mejor solución de ensayo encontrada hasta aquí.
Despue determina cuáles de los movimientos restantes proporcionan la mejor solución. Adoptar esta solución como la siguiente solución de ensayo sin importar si es mejor o peor que la actual solución de ensayo.
Si la lista tabú se llenó, borrar el miembro más *antigüo* de la lista tabú para proveer mayor flexibilidad a los futuros movimientos.
Regla de paro: Use algún criterio de parada tal como un número fijo de iteraciones, o una cantidad fija de tiempo de CPU, o un número fijo de iteraciones consecutivas sin que se mejore el mejor valor de la función objetivo. Tambietener en cualquier iteración donde no haya movimientos factibles en la vecindad local de la actual solución de ensayo. Aceptar la mejor solución de ensayo en cualquier iteración como la solución final. El algoritmo de búsqueda tabú, genera preguntas para las cuales los autores resumen algunas respuestas.

Preguntas / Respuestas

1. ¿Quocedimiento de búsqueda debería usarse? **/** En cada iteració escoger el mejor vecino inmediato de la actual solución de ensayo que se ha sacado de su estatus tabú.

2. ¿Cómo el procedimiento define la estructura del vecino que especifica que vecinos son inmediatos, es decir, elegibles en una sola iteración de cualquier solución de ensayo actual. **/** Estructura de la vecindad. Un vecino inmediato de la solución de ensayo actual es una que se alcanzó agregando solo un enlace y borrando uno de los otros ensayos en el ciclo que se forma por esta adición. El enlace borrado debe provenir de un ciclo a fin de que se conserve un árbol de expansión (ST).

3. ¿Cuál es la forma en las que los movimientos tabú deberían representarse en la lista tabú. **/** Forma de los movimientos tabú. Enliste los enlaces que no se borran.

4. ¿Cuál movimiento tabú debería agregarse a la lista tabú **/** Adicione un movimiento tabú en cada iteracción despue escoger el enlace por agregarse a la red y tambigregue este enlace en la lista tabú

5. ¿Qunto un movimiento tabú debe conservarse en la lista tabú **/** Tamaño máximo de la lista tabú: dos. Siempre que un movimiento tabú sea agregado, borre el más *antigüo*

6. ¿Qugla de paro debería usarse? **/** Regla de paro: parar despue 3 iteracciones consecutivas en las que no hay un mejoramiento del mejor valor hallado. Tambiarar en cualquier iteracción donde la actual solución de ensayo no tiene vecinos inmediatos, que no fueron sacados por su estatus tabú.

Example 110 *Se ilustra con el grafo de la figura siguiente*[7]*:*

[7] Hillier and Lieberman, 2005, p. 622

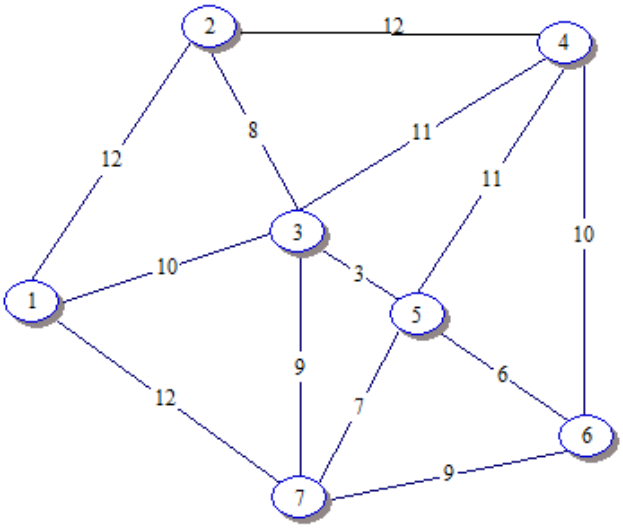

Figura 55

Una solución de ensayo inicial es 1 - 2 - 3 - 4 - 5 - 6 - 7 - 1 con distancia 69.
Iteración 1

1. *Seleccionar subtour para revertir, 3 - 4*
2. *Enlaces borrados 2 - 3 y 4 - 5 Enlaces agregados 2 - 4 y 3 - 5*
3. *Lista Tabú 2 - 4 y 3 - 5 Nueva solución de ensayo 1 - 2 - 4 - 3 - 5 - 6 - 7 - 1 con distancia 65*

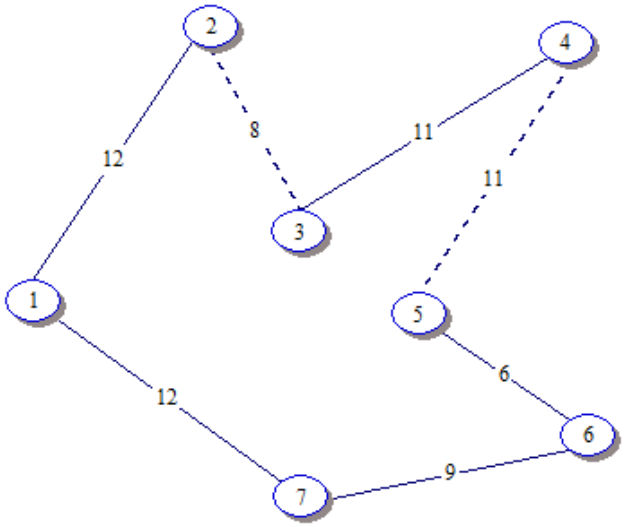

Figura 56: Enlaces que se quitan

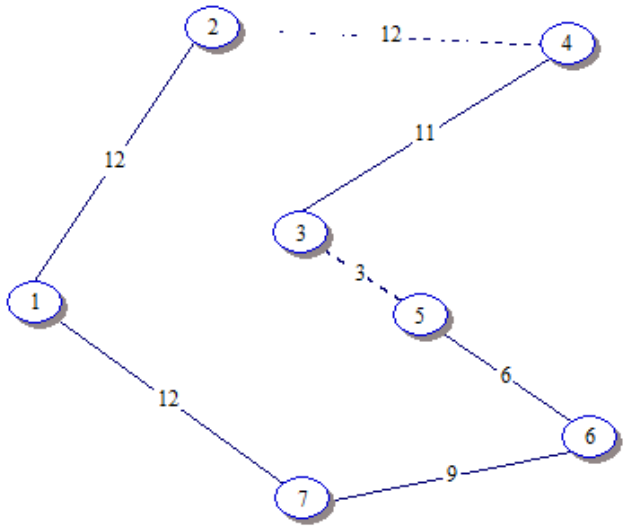

Figura 57: Enlaces que se agregan

Iteración 2

1. *Seleccionar subtour para revertir, 3 - 5 - 6 a 6 -5 - 3*
2. *Enlaces borrados 3 - 4 y 6 - 7 Enlaces agregados 2 - 4 y 3 - 5*
3. *Lista Tabú 2 - 4 , 3 - 5 , 4 - 6 y 3 - 7 Nueva solución de ensayo 1 - 2 - 4 - 6 - 5 - 3 - 7 - 1 con distancia 64*

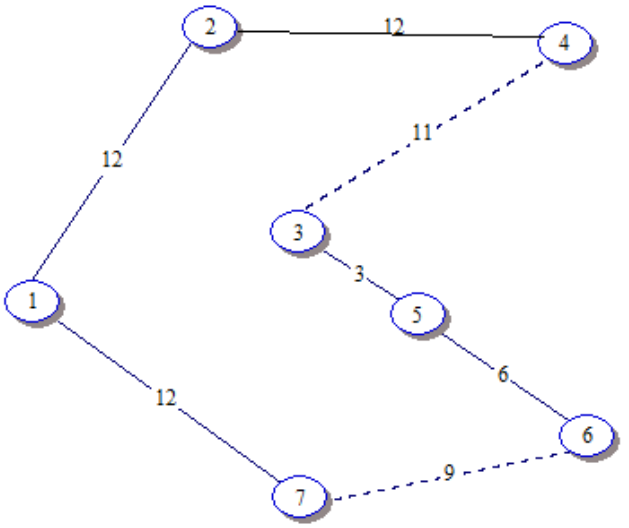

Figura 58: Enlaces que se borran

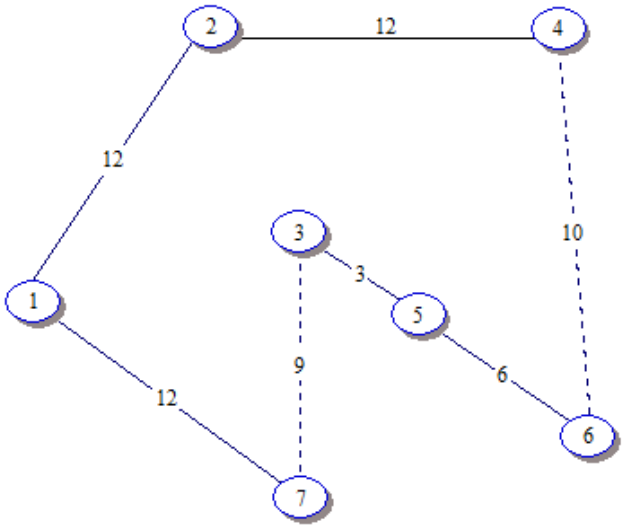

Figura 59: Enlaces que se agregan

8. Referencias y bibliografía

- Balderas, P. y Sánchez, G. (coords., 2011) Ingeniería de Sistemas. Investigación e Intervención. Mco: Plaza y Vald Facultad de Ingeniería, UNAM, 67 – 92
- Bazaraa, M., Jarvis, J., and Sherali, H. (1998) Programación lineal y flujo de redes. Mco LIMUSA, 2a.
- Bondy, J. A. and Murty, U.S.R. (1982) Graph Theory with Applications. N.Y. North-Holland, 5a.
- Chartrand, G. (1977) Introductory Graph Theory. New York: Dover Publications, Inc. pp. 294
- Cook, W., Cunningham, W. , Pulleyblank, W., and Schrijver, A. (1998) Combinatorial optimization. New York : J. Wiley.
- Espinosa Armenta, R. (2010) Matemáticas Discretas. Mco:_Alfaomega, pp. 467
- Harary, F. (1969) Graph Theory. Massachussets: Addison-Wesley, pp. 274
- Harary, F. Norman, F. y Cartwright D. (1965, 1978) Structural Models: an Introduction to the Theory of Directed Graph. New York: John Wiley & Sons, pp. 415
- Hillier, F and Lieberman,G. (2005) Introduction to Operations Research, 8th edition. New York: McGraw-Hill.
- Ortega, James M. (1987) Matrix Theory. A second course. The University Series in Mathematics. New York: Plenum Press, pp. 262
- Murthy, D., Page, N. & Rodin, E. (1990) Mathematical Modelling. A tool for Problem Solving Engineering, Physical, Biological and Social Sciences. N.Y.: Pergamon Press.
- Slomson, Alan (1991)An Introduction to Combinatorics. London: Capman & Hall/CRC

8.1. BIBLIOGRAFÍA ADICIONAL

- Pierluigi Crescenzi, Viggo Kann, Magnús Halldorsson, Marek Karpinski, Gerhard Woeginger, A Compendium of NP Optimization Problems.
- Christos H. Papadimitriou, and Kenneth Steiglitz; Combinatorial Optimization: Algorithms and Complexity; Dover Pubns; (paperback, Unabridged edition, July 1998) ISBN 0486402584.