Introducción a la Investigación de Operaciones
Facultad de Ingeniería – Universidad de la República Oriental del Uruguay

Grafos

Ejercicio 1.

a) El grafo en notación de conjuntos es el par (X,U) donde X es el conjunto de *nodos* y U es el conjunto de *arcos*. En este caso, $X = \{x_1,x_2,x_3,x_4,x_5,x_6,x_7,x_8\}$ y $U = \{(x_1,x_2), (x_1,x_4), (x_2,x_1), (x_2,x_3), (x_3,x_1), (x_3,x_4), (x_3,x_5), (x_4,x_6), (x_4,x_5), (x_4,x_2), (x_5,x_6), (x_6,x_5), (x_6,x_7), (x_7,x_8)\}$.

El orden del grafo es la cantidad de nodos, en este caso **8**.

b) Sucesores

$\boldsymbol{x_1}$: $\{x_2, x_4\}$ $\boldsymbol{x_4}$: $\{x_2, x_5, x_6\}$
$\boldsymbol{x_3}$: $\{x_1, x_4, x_5\}$ $\boldsymbol{x_8}$: ϕ

Predecesores

$\boldsymbol{x_2}$: $\{x_1, x_4\}$ $\boldsymbol{x_7}$: $\{x_6\}$
$\boldsymbol{x_6}$: $\{x_4, x_5\}$

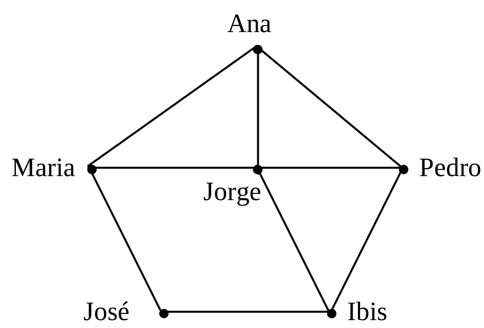
c) Adyacentes

$\boldsymbol{x_1}$: $\{x_2, x_4\}$ $\boldsymbol{x_4}$: $\{x_2, x_5, x_6\}$ $\boldsymbol{x_7}$: $\{x_8\}$
$\boldsymbol{x_2}$: $\{x_1, x_3\}$ $\boldsymbol{x_5}$: $\{x_6\}$ $\boldsymbol{x_8}$: ϕ
$\boldsymbol{x_3}$: $\{x_1, x_4, x_5\}$ $\boldsymbol{x_6}$: $\{x_5, x_7\}$

d) Un posible camino es: $(x_1,x_2), (x_2,x_3), (x_3,x_5), (x_5,x_6), (x_6,x_7)$. Hay otros.

Ejercicio 2.

En el grafo, a cada persona le corresponde un nodo y existe un arco entre dos nodos si las personas que estos representan se comunican en un día.



Notar que, según la letra, si la persona i habla con la persona j, entonces también la persona j habla con la persona i. Por lo tanto, un grafo no dirigido es suficiente para representar la información. Si ese no fuera el caso, sería más conveniente modelar la situación mediante un grafo dirigido.

Ejercicio 3.

Si G fuera un sólo árbol (es decir, si $t = 1$) tendría $n - 1$ aristas (por el teorema T.1. de las notas: árboles de n vértices tienen $n - 1$ aristas). Dado un árbol, para obtener t árboles debe eliminarse exactamente $t - 1$ aristas de las $n - 1$ que tiene el árbol original. Por lo tanto, la cantidad de aristas restantes es $n - 1 - (t - 1) = n - t$ aristas.

Ejercicio 4.

Partiendo del nodo 1, utilizaremos DFS para encontrar un árbol de cubrimiento.

Los adyacentes al nodo 1, son los siguientes nodos: 2, 3, 4, 7 y 8. Elegimos alguno de ellos, por ejemplo el 2 y agregamos la arista (1,2) al árbol. Tenemos $T = \{(1,2)\}$.

Partimos del último nodo agregado al árbol, es decir, el 2. Sus adyacentes son los nodos 1, 3, 5 y 6. El nodo 1 ya está cubierto (si agregáramos la arista (2,1) se formaría un ciclo), por lo tanto, seleccionamos el nodo 3. Tenemos ahora $T = \{(1,2), (2,3)\}$.

Los adyacentes al nodo 3 son los nodos 1, 2, 4, 5 y 8, de los cuales seleccionamos al 4 para ingresar al árbol: $T = \{(1,2), (2,3), (3,4)\}$.

Siguiendo el procedimiento, obtenemos $T = \{(1,2), (2,3), (3,4), (4,5), (5,6), (6,7)\}$.

Si partimos del último nodo agregado, el 7, notamos que todos sus adyacentes ya están en el árbol. Pero aún quedan nodos por cubrir (el 8). Por lo tanto, intentamos agregar los adyacentes al penúltimo nodo agregado a T, es decir, el 6. También ocurre que todos sus adyacentes están en el árbol. Siguiendo esta búsqueda hacia atrás, notaremos que el nodo 3 tiene como adyacente al 8, que no está en el árbol. Entonces, agregamos la arista (3,8) al árbol, obteniendo el resultado final: $T = \{(1,2), (2,3), (3,4), (4,5), (5,6), (6,7), (3,8)\}$.

Ejercicio 5.

<u>Paso Base</u>: $a_{ij}^{(1)} = 1$ si y sólo si existe un camino de largo 1 de s_i a s_j.

Prueba
Por ser A la matriz de adyacencia de G, $a_{ij} = 1$ si y sólo si la arista (s_i, s_j) está en G. Como $a_{ij}^{(1)} = a_{ij}$, se cumple que $a_{ij}^{(1)}$ si y sólo si (s_i, s_j) está en G. Como (s_i, s_j) es un camino de largo 1, queda probada la tesis.

<u>Paso Inductivo</u>:

Hipótesis: $a_{ij}^{(p-1)} = 1$ sii existe un camino de largo $p - 1$ de s_i a s_j.
Tesis: $a_{ij}^{(p)} = 1$ sii existe un camino de largo p de s_i a s_j.

Prueba
Por la definición de producto de matrices: $a_{ij}^{(p)} = a_{i1}^{(p-1)}a_{1j} + a_{i2}^{(p-1)}a_{2j} + \ldots + a_{in}^{(p-1)}a_{nj}$. Como los sumandos son todos 0 o 1, si $a_{ij}^{(p)} = 1$, entonces al menos uno de ellos vale 1.

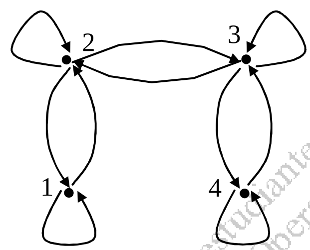
Es decir, existe k tal que $a_{ik}^{(p-1)}a_{kj} = 1$. Por propiedades del producto booleano, se cumple que $a_{ik}^{(p-1)} = a_{kj} = 1$.

Por la Hipótesis inductiva, $a_{ik}^{(p-1)} = 1$ implica que existe un camino C de largo $p - 1$ entre s_i y s_k. Además, por la definición de matriz de adyacencia, $a_{kj} = 1$ sii existe la arista (s_k, s_j) en el grafo. Si agregamos (s_k, s_j) a C obtenemos un camino de largo p de s_i a s_j, como queríamos.

Ejercicio 6.

Una matriz M (4×4) tridiagonal booleana y su grafo asociado son:

$$M = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix}$$



Como el grafo tiene una única componente fuertemente conexa, la matriz de alcance será la matriz de 4×4 con todas sus entradas siendo unos. Es triangularizable por bloques, con un solo bloque de unos.

Ejercicio 7.

Primero calculamos la matriz de alcance $(I \vee A)^5$, donde A es la matriz de adyacencia.

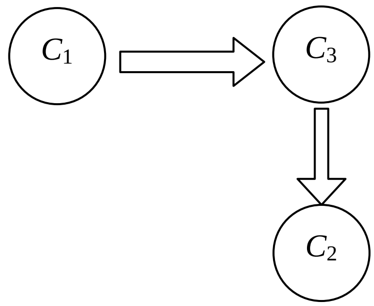
$$I \vee A = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \end{pmatrix} \qquad (I \vee A)^5 = \begin{array}{c} \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \end{array} \begin{array}{c} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{array}$$

Permutando las filas y columnas correspondientes a x_3 por x_6 podemos armar bloques de unos en la diagonal:

	x_1	x_2	$\boldsymbol{x_6}$	x_4	x_5	$\boldsymbol{x_3}$
x_1	1	1	1	1	1	1
x_2	1	1	1	1	1	1
$\boldsymbol{x_3}$	0	0	0	1	1	1
x_4	0	0	0	1	1	0
x_5	0	0	0	1	1	0
$\boldsymbol{x_6}$	1	1	1	1	1	1

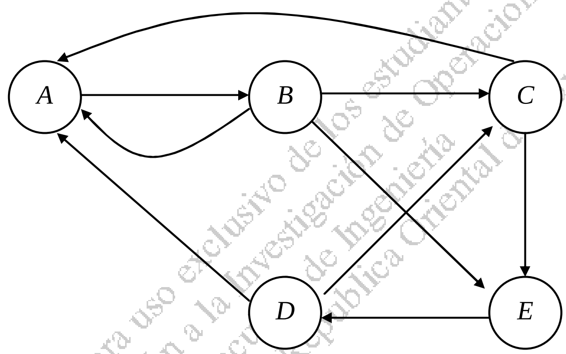
	x_1	x_2	$\boldsymbol{x_6}$	x_4	x_5	$\boldsymbol{x_3}$
x_1	1	1	1	1	1	1
x_2	1	1	1	1	1	1
$\boldsymbol{x_6}$	1	1	1	1	1	1
x_4	0	0	0	1	1	0
x_5	0	0	0	1	1	0
$\boldsymbol{x_3}$	0	0	0	1	1	1

Las Componentes Fuertemente Conexas son $C_1 = \{x_1, x_2, x_6\}$, $C_2 = \{x_4, x_5\}$ y $C_3 = \{x_3\}$. El grafo condensado es el siguiente:



Ejercicio 8.

Consideramos el grafo cuyos nodos son los tipos de cultivos y donde existe un arco de *i* a *j* si *j* puede ser sembrado inmediatamente después que *i*.



Como cada arco representa una secuencia válida de sembrados, un camino en este grafo también representa una secuencia válida. En particular, un camino hamiltoniano representa una posible manera de sembrar todos los cultivos. Por lo tanto, resolver el problema equivale a hallar todos los caminos hamiltonianos en el grafo.

Inicialmente tenemos las matrices:

$$M = \begin{pmatrix} 0 & AB & 0 & 0 & 0 \\ BA & 0 & BC & 0 & BE \\ CA & 0 & 0 & 0 & CE \\ DA & 0 & DC & 0 & 0 \\ 0 & 0 & 0 & ED & 0 \end{pmatrix} \quad \text{y} \quad \overline{M} = \begin{pmatrix} 0 & B & 0 & 0 & 0 \\ A & 0 & C & 0 & E \\ A & 0 & 0 & 0 & E \\ A & 0 & C & 0 & 0 \\ 0 & 0 & 0 & D & 0 \end{pmatrix}$$

Aplicando el operador de multiplicación latina una vez, obtenemos los caminos hamiltonianos de largo 2:

$$M(2) = ML\overline{M} = \begin{pmatrix} 0 & ABC & 0 & 0 & ABE \\ BCA & 0 & 0 & BED & BCE \\ 0 & CAB & 0 & CED & 0 \\ BCA & DAB & 0 & 0 & DCE \\ EDA & 0 & EDC & 0 & 0 \end{pmatrix}$$

Aplicando dos veces más el operador, llegamos a los caminos de largo 3 y 4:

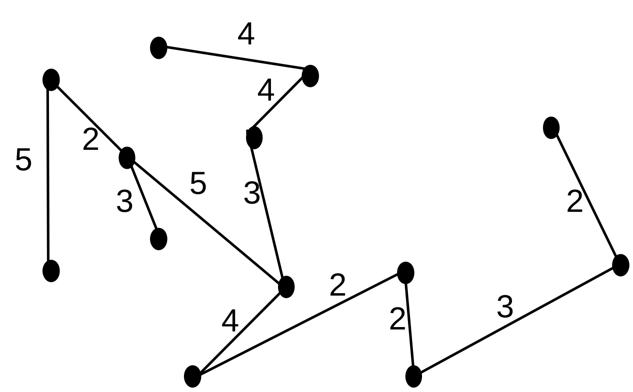
$$M(3) = M(2)L\overline{M} = \begin{pmatrix} 0 & 0 & 0 & ABED & ABCE \\ BEDA & 0 & BEDC & BCED & 0 \\ CEDA & 0 & 0 & 0 & CABE \\ 0 & DCAB & DABC & 0 & DABE \\ EDCA & EDAB & 0 & 0 & 0 \end{pmatrix}$$

$$M(4) = M(3)L\overline{M} = \begin{pmatrix} 0 & 0 & ABEDC & ABCED & 0 \\ \begin{matrix} BEDCA \\ BCEDA \end{matrix} & 0 & 0 & 0 & 0 \\ 0 & CEDAB & 0 & CABED & 0 \\ 0 & 0 & 0 & 0 & \begin{matrix} DCABE \\ DABCE \end{matrix} \\ 0 & EDCAB & EDABC & 0 & 0 \end{pmatrix}$$

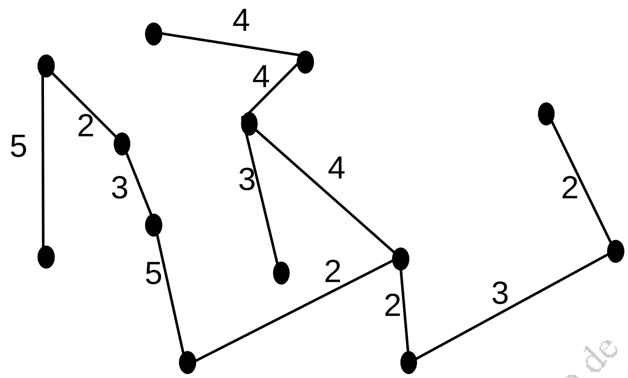
Los caminos en $M(4)$ son los ordenes posibles para la siembra de los distintos tipos de semillas.

Ejercicio 9.

Con el algoritmo de Kruskal obtenemos el siguiente árbol de cubrimiento mínimo:



Y con el algoritmo de Prim obtenemos:



En ambos casos, puede encontrarse otro esqueleto según como se elijan las aristas en cada iteración. Pero debe notarse que el costo de los esqueletos mínimos es siempre el mismo (si hubiera dos diferentes, uno sería mayor que el otro y ese no sería mínimo). El costo mínimo es 39.

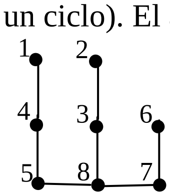
Ejercicio 10.

Consideramos un grafo completo cuyos nodos son las zonas y tal que el arco entre dos nodos *i* y *j* está ponderado con la distancia entre las zonas *i* y *j*.

Aplicamos el algoritmo de Kruskal sobre ese grafo. Para cada iteración mostramos las aristas que ya han sido elegidas (bosque actual) y las aristas de costo mínimo consideradas. Comenzamos con la arista de menos costo, la (1,4):

Paso	Bosque actual	Siguiente
1	(1,4)	**(7,8)**
2	(1,4) (7,8)	**(6,7)**
3	(1,4) (7,8) (6,7)	**(4,5)**
4	(1,4) (7,8) (6,7) (4,5)	**(5,8)**
5	(1,4) (7,8) (6,7) (4,5) (5,8)	**(2,3)**
6	(1,4) (7,8) (6,7) (4,5) (5,8) (2,3)	(4,8) forma un ciclo (5,6) forma un ciclo **(3,8)**
7	(1,4) (7,8) (6,7) (4,5) (5,8) (2,3) (3,8)	–

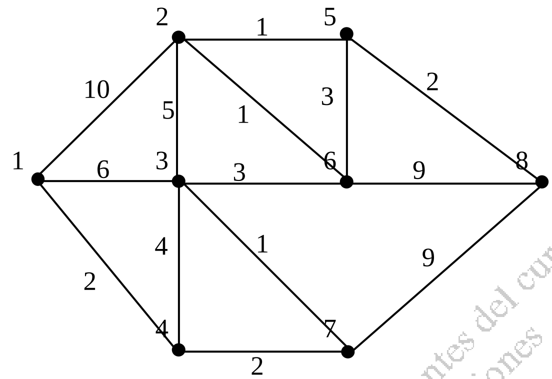
Notar que en el paso 7 se descartan las aristas (4,8) y (5,6) pues a pesar de tener menos costo que (3,8), forman un ciclo con las aristas ya elegidas (si las seleccionáramos no obtendríamos un árbol, pues tendríamos un ciclo). El árbol resultante es:



Y su costo es la distancia total: 5 Km.

Ejercicio 11.

Modelamos la situación como un grafo. Los nodos son los dados en la letra del ejercicio y un arco entre los nodos *i* y *j* existe cuando hay un ómnibus directo para viajar de *i* a *j*. Además, cada arco está ponderado con el tiempo que lleva hacer dicho viaje. Notar que los tiempos de viaje son simétricos y por lo tanto, puede utilizarse un grafo no dirigido.

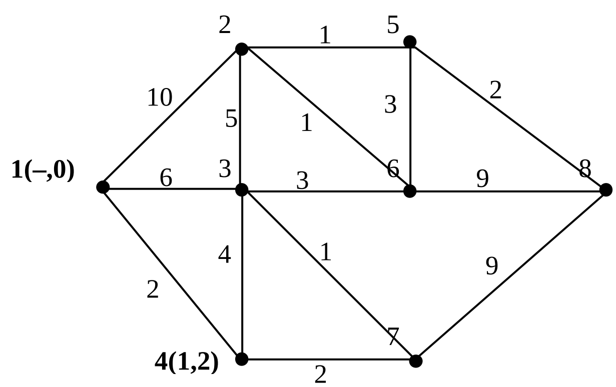


El problema se reduce a hallar un camino de costo mínimo entre 1 y 8. Aplicamos el algoritmo de Dijkstra para hallarlo. Al final del algoritmo tendremos etiquetado cada vértice *v* con (*p*, *l*) donde *p* es el predecesor en el camino más corto desde 1 y *l* es el largo de dicho camino.

Paso 1. Comenzamos etiquetando el vértice de origen **1** con **(–,0)**.

Paso 2. Analizamos los vértices 2, 3 y 4. De ellos, vamos a seleccionar *v* tal que *w*(*v*)+0 sea mínimo. Vemos que el mínimo se alcanza en 4 y vale 2. Por lo tanto etiquetamos **4** con **(1,2)**. Esto quiere decir: "el camino más corto de 1 a 4 tiene costo 2 y el predecesor de 4 en el camino es 1".

Paso 3. Tenemos un conjunto de aristas marcadas {1, 4} y un conjunto de aristas no marcadas {2, 3, 5, 6, 7, 8}. En este momento conocemos el camino más corto desde 1 hasta cada una de los vértices marcados.

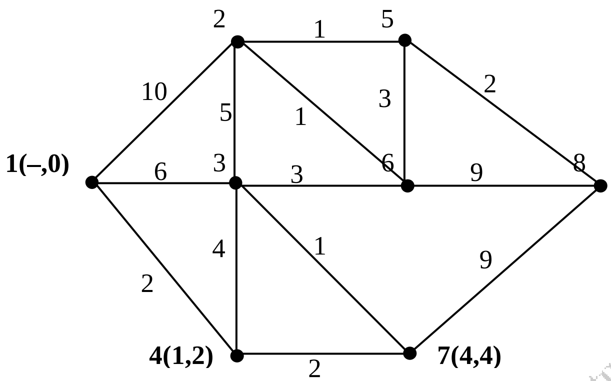


Debemos considerar los vértices no marcados que sean adyacentes a vértices marcados. O sea: 2, 3 y 7. Igual que en el paso 2, seleccionaremos el vértice *v* tal que *w*(*p*,*v*) + *d*(*p*) sea mínimo, donde *p* es un vértice marcado adyacente a *v* y *d*(*p*) es la segunda componente de la etiqueta de *p* (o sea el costo mínimo del camino de 1 a *p*).

Tenemos:

$w(1,2) + d(1) = 10$
$w(1,3) + d(1) = 6$
$w(4,7) + d(4) = 4$

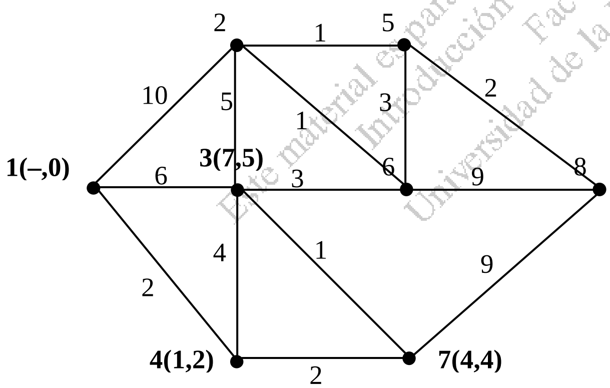
El mínimo se da en 7 y por lo tanto marcamos el vértice 7 con **(4,4)**.

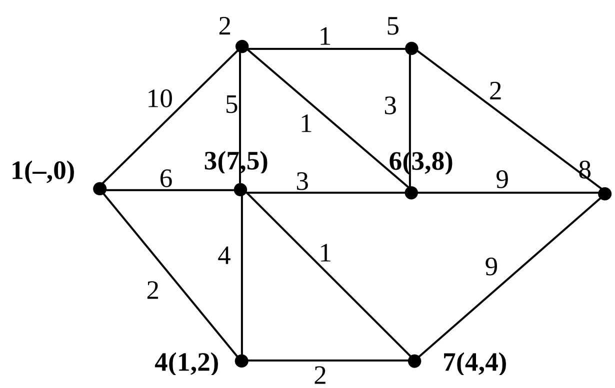


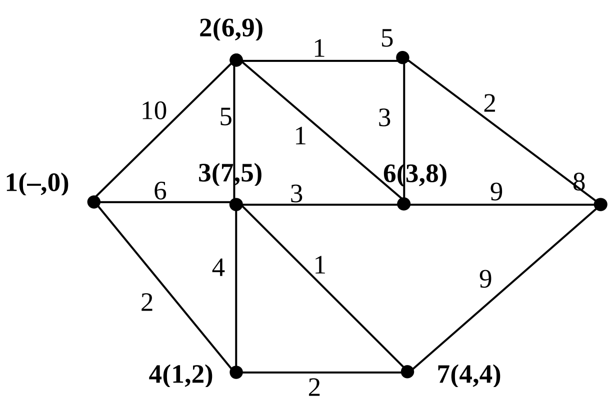
El algoritmo sigue de la misma forma, hasta poder etiquetar el vértice 8:

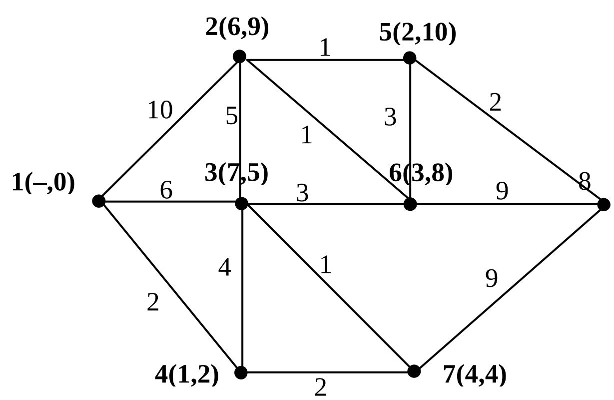
- buscamos v que minimiza $w(p,v) + d(p)$ siendo p un vértice marcado adyacente a v
- marcamos v con $(p, w(p,v)) + d(p)$

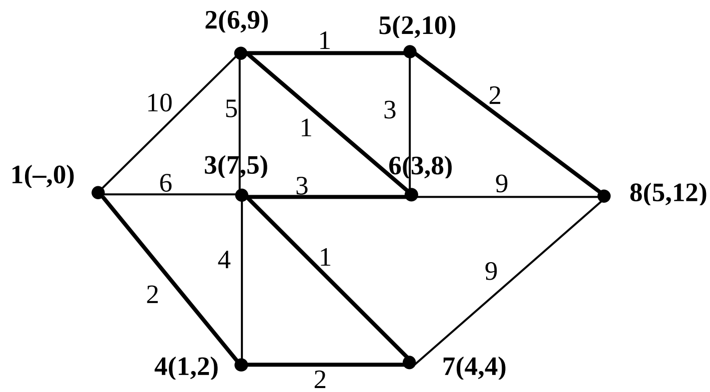
Mostramos como sigue el etiquetado de los vértices:











Ahora hemos marcado el vértice 8. Concluimos que el camino más corto de 1 a 8 tiene costo 12: La secuencia de vértices que el trabajador debe seguir se obtiene siguiendo la primera etiqueta a partir de 8. Es decir, 1–4–7–3–6–2–5–8.

Ejercicio 12.

a) El flujo marcado en la red no es un flujo *a*–*z*. Los nodos C y H no cumplen con la condición de conservación de flujos:

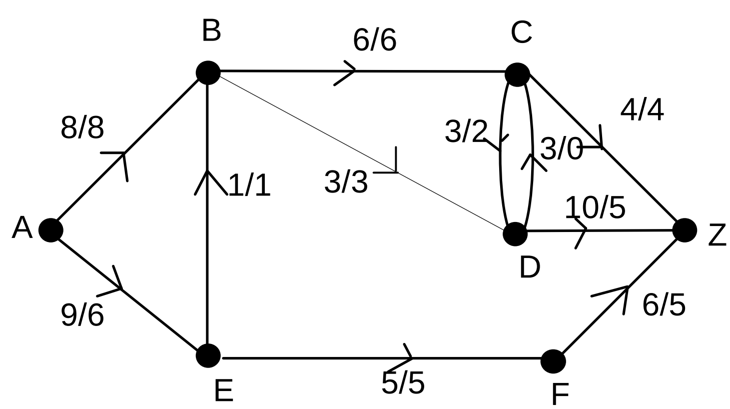
El flujo que llega a C es 7 y el flujo que sale es 5.
El flujo que llega a H es 7 y el flujo que sale es 9.

b) Un corte *a*–*z* es, por ejemplo: (P,P^c), con P = {A, B, C} y P^c = {D, E, F, G, H, I, J, K}.

Ejercicio 13.

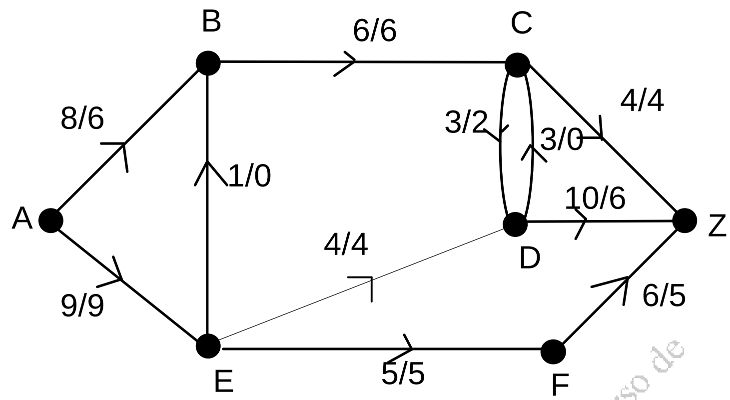
Analizaremos cada caso por separado. En cada caso supondremos que la capacidad del caño agregado es infinita.

Un flujo máximo para el caso en que se coloca el caño A es:



que logra transportar 14 unidades de flujo.

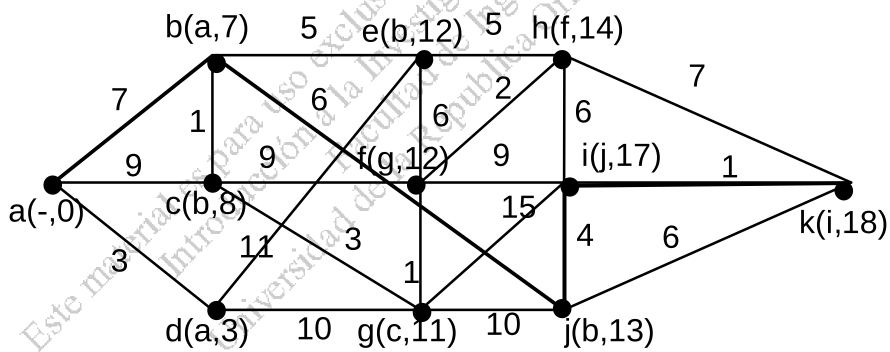
Un flujo máximo para el caso en que se coloca el caño B es:



que logra transportar 15 unidades de flujo. Por lo tanto, la mejor opción es colocar el caño B, de capacidad 4.

Ejercicio 14.

Etiquetamos los nodos en el siguiente orden: a(–,0), d(a,3), b(a,7), c(b,8), g(c,11), f(g,11), e(b,12), j(b,13), h(f,14), i(j,17), k(i,18). Resulta el siguiente grafo:

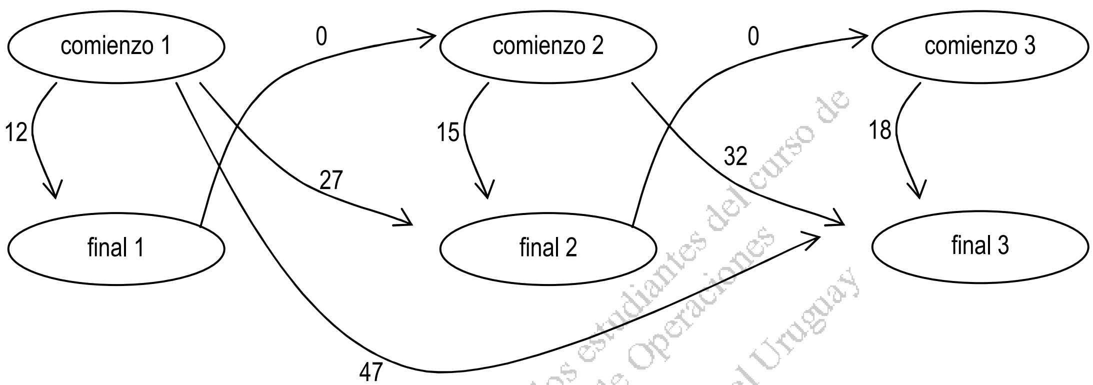


El camino mínimo entre *a* y *k* es (*a*, *b*, *j*, *i*, *k*) y su costo es 18.

Ejercicio 15.

Modelamos el problema sobre un grafo dirigido. Sus nodos son momentos del tiempo (comienzo y fin de un año). El costo de un arco del nodo *i* al nodo *j* indica el costo de comprar un tractor en el momento *i* y mantenerlo hasta el momento *j*. El problema se reduce a hallar un camino de costo mínimo entre comienzo 1 y final 3.

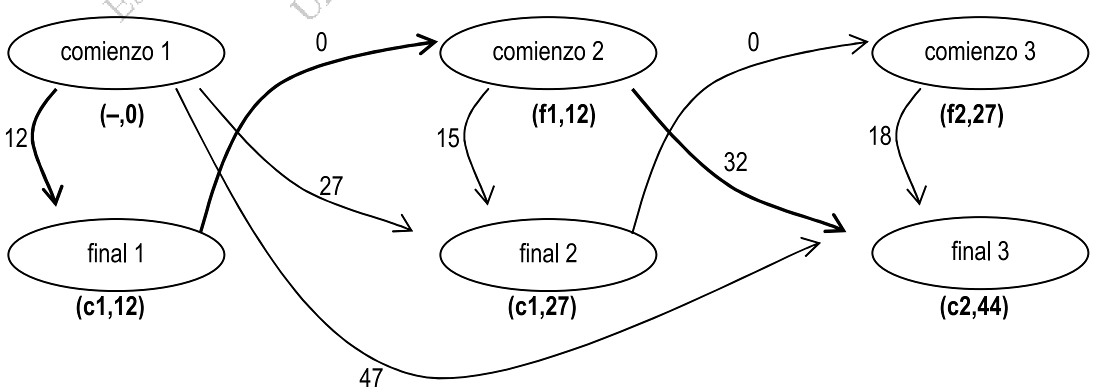
El grafo es el siguiente:



Notar que si en lugar de un nodo para el comienzo del año *i* y uno para su final, hubiera un solo nodo para el año *i*, no podríamos modelar la decisión de comprar un tractor al comienzo del año y cambiarlo a fin de año (diagonal de la matriz).

Los arcos de costo 0, indican que el final del año *i* y el comienzo del año *i* + 1, son el mismo momento, pero las compras se realizan al comienzo del año. Si estos arcos no estuvieran, habría un solo camino del comienzo del año 1 al final del año 3.

Utilizamos el algoritmo de Dijkstra para hallar el camino. El grafo con las etiquetas al final del algoritmo es el siguiente:



La solución de menor costo es comprar un tractor al comienzo del año 1, utilizarlo por todo ese año, comprar otro al comienzo del año 2 y utilizarlo durante los años 2 y 3. El costo total de la solución es 44.

Ejercicio 16.

Para hallar el flujo máximo se utilizará el Algoritmo de Ford y Fulkerson.

Comenzamos etiquetando el primer nodo (el nodo A) con $(–,\infty)$. A cada nodo q que se visita se le asigna la etiqueta $(p^{\pm},D(q))$ donde p es el nodo precedente a q en la cadena de flujo de a a q, el índice superior de p será + si el arco es (p,q) y – si es (q,p). $D(q)$ es la holgura mínima entre todos los arcos de la cadena que van de a a q. En un arco de orientación contraria al de la cadena, la holgura es la cantidad de flujo que puedo quitar.

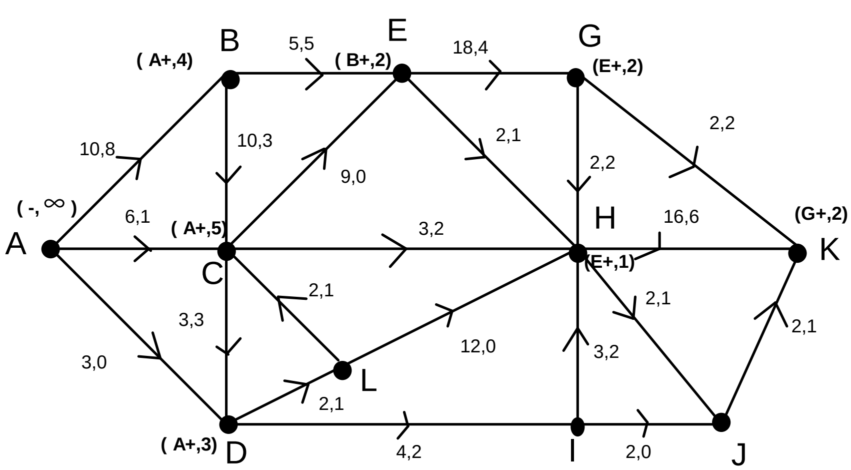
Se visitan los nodos sucesores de A.
Visitamos D. Como $k(A,D)-\varphi(A,D)>0$ y D no está marcado, lo marcamos con (A^{+},3).
Visitamos B y lo marcamos con la etiqueta (A^{+},4).
Visitamos C y lo marcamos con (A^{+},5)

Ahora que examinamos todos los arcos que salen de A, como K (el nodo terminal) no está marcado debemos elegir otro nodo para examinar y volver a examinar sus sucesores y sus predecesores. Elegimos B. El único predecesor de B es A, que ya está marcado. Analizamos sus sucesores E y C. Marcamos E con (B^{+},2). C ya está marcado así que no hacemos nada.

Como K no está marcado, elegimos otro nodo para examinar. Elegimos E. Sus predecesores son C y B que ya están marcados. Analizamos sus sucesores G y H. Marcamos G con (E^{+}, 2) y H con (E^{+}, 1).

Como K no está marcado elegimos otro nodo para examinar. Elegimos G. Su predecesor ya está marcado y el único sucesor que resta marcar el K. Entonces marcamos K con (G^{+},2)

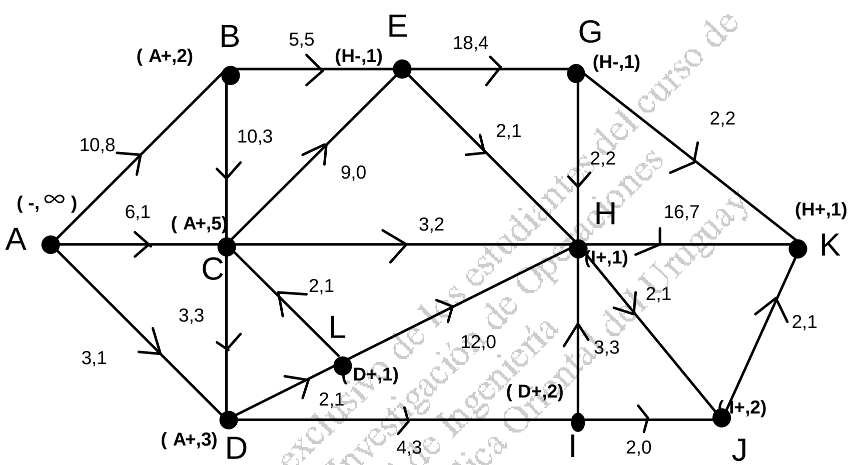
Como K está marcado tenemos una cadena de arcos con holguras: A–B–E–G–K. La hallamos retrocediendo desde K por la primer etiqueta como en el algoritmo de Dijkstra. Podemos aumentar el flujo de esa cadena en D(K) = 2 unidades, como se muestra en el siguiente grafo:



Ahora borramos todas las etiquetas y recomenzamos. Marcamos A con (–, ∞).

Marcamos B con (A^+,2), C con (A^+,5) y D con (A^+,3)
Elegimos D. Sus sucesores son L, I. Marcamos L con (D^+,1) e I con (D^+,2).
Elegimos I. Sus sucesores son H y J. Marcamos H con (I^+,1) y J con (I^+,2).
Elegimos H. Marcamos E con (H–, 1) y G con (H–, 1). Marcamos J con (H^+, 1) y K con (H^+, 1).

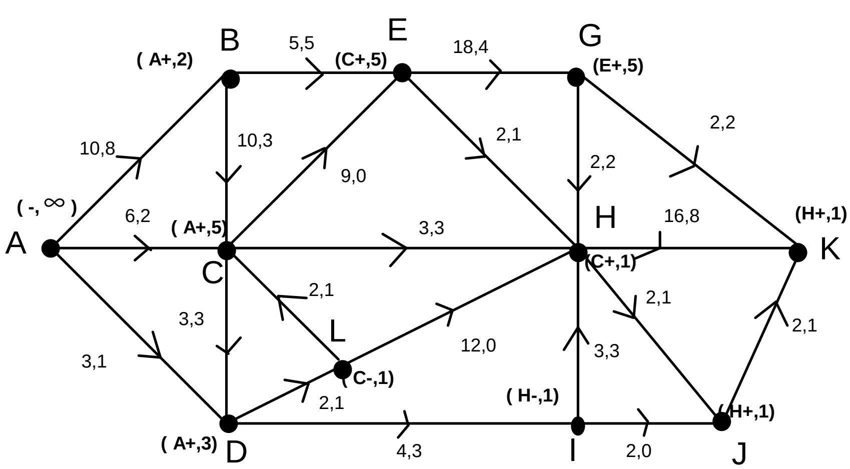
Como K está marcado tenemos una cadena de arcos con holguras: A–D–I–H–K. Podemos aumentar el flujo de esa cadena en D(K) = 1 unidad:



Recomenzamos marcando A con (–, ∞).

Marcamos B con (A^+,2), C con (A^+,5) y D con (A^+,2).
Elegimos C. Marcamos L con (C–,1). Marcamos E con (C^+,5) y H con (C^+,1).
Elegimos E. Marcamos G con (E^+,5).
Elegimos H. Marcamos I con (H^-,1). Marcamos J con (H^+,1) y K con (H^+,1).

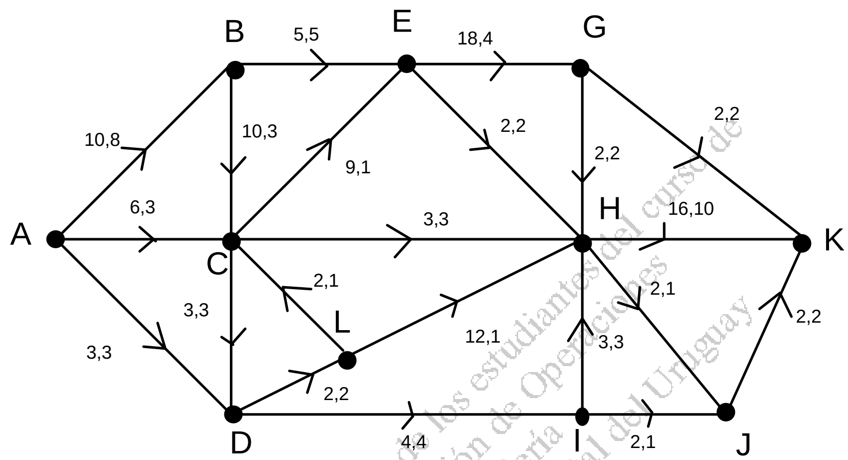
Tenemos una cadena de arcos con holguras: A–C–H–K. Podemos aumentar el flujo a través esa cadena en D(K) = 1 unidad, como se muestra en el siguiente grafo:



Continuando el algoritmo, podemos encontrar, las siguientes cadenas

A–C–E–H–K aumentando el flujo en 1 unidad
A–D–L–H–K aumentando el flujo en 1 unidad
A–D–I–J–K aumentando el flujo en 1 unidad

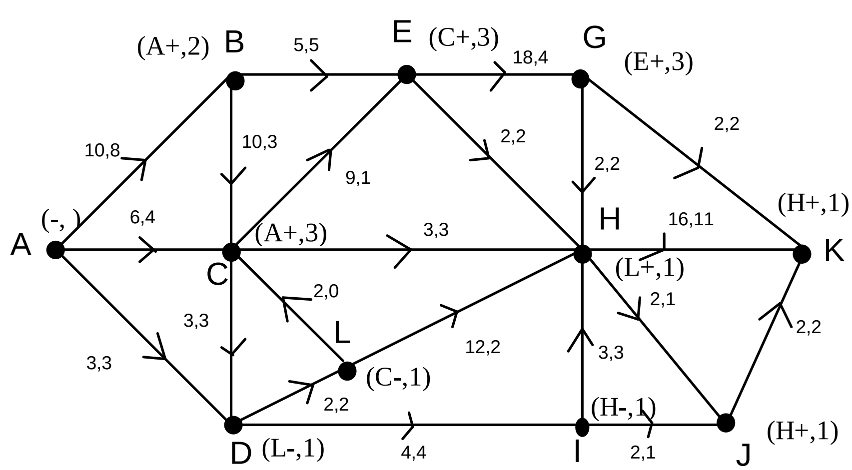
El grafo que resulta es el siguiente:



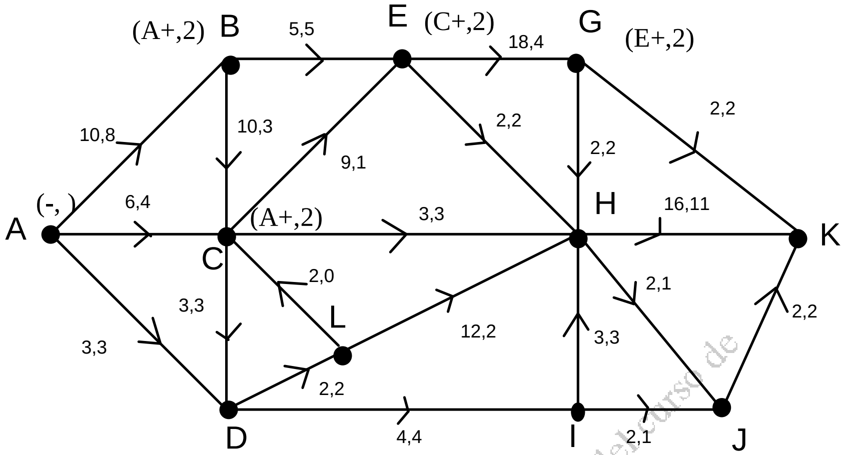
Borramos todas las etiquetas y seguimos iterando una vez mas.

Elegimos A. Marcamos B con (A^+,2) y C con (A^+,3).
Elegimos C. Marcamos L con (C^-,1) y E con (C^+,3).
Elegimos L. Marcamos D con (L^-,1) y H con (L^+,1).
Elegimos H. Marcamos I con (H^-,1), J con (H^+,1) y K con (H^+,1).

Hemos hallado A–C–L–H–K y aumentamos el flujo en 1 unidad. Notar que la arista (L,C) en realidad decrementará su valor en 1 (pues L está marcado con (C^-,1)). Obtenemos



Al intentar seguir iterando, llegaremos a la siguiente situación:

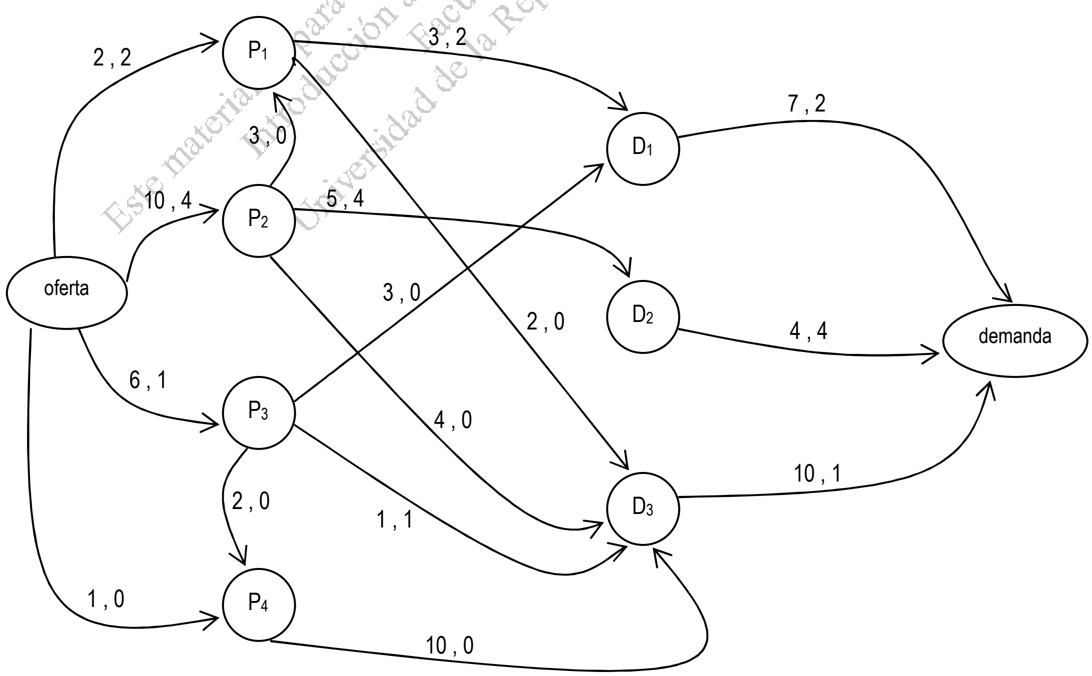


Aquí no podemos marcar más vértices, pues las aristas que salen de los vértices marcados están todas saturadas y las que entran a ellos no tienen flujo asignado.

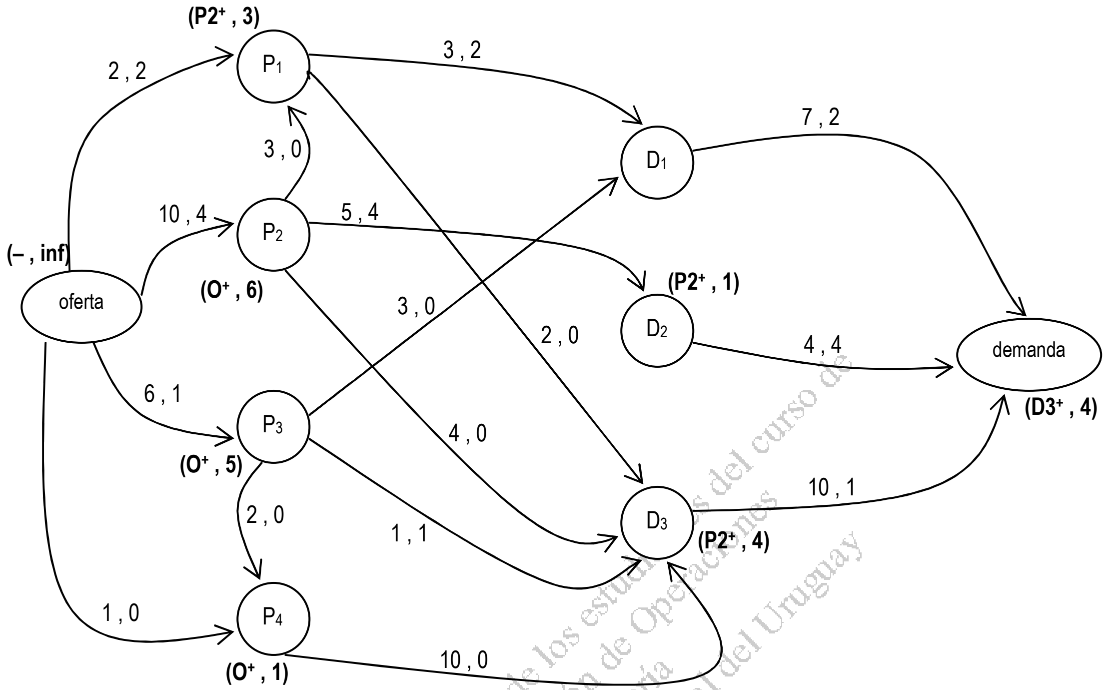
Por lo tanto, el flujo asignado es máximo, su valor es 15 y los vértices etiquetados {A,B,C,E,G} forman un corte mínimo.

Ejercicio 17.

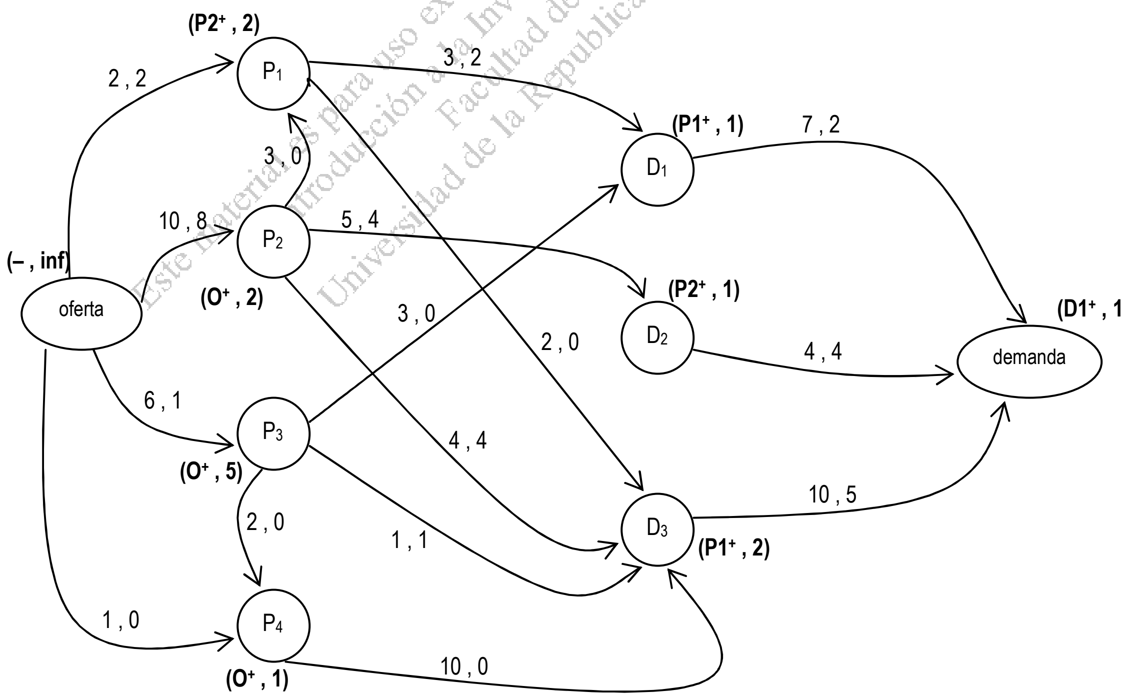
El problema modelado como una red de flujo (y una asignación inicial de flujos) es:



Aplicando una iteración del algoritmo de Ford y Fulkerson, llegamos a:

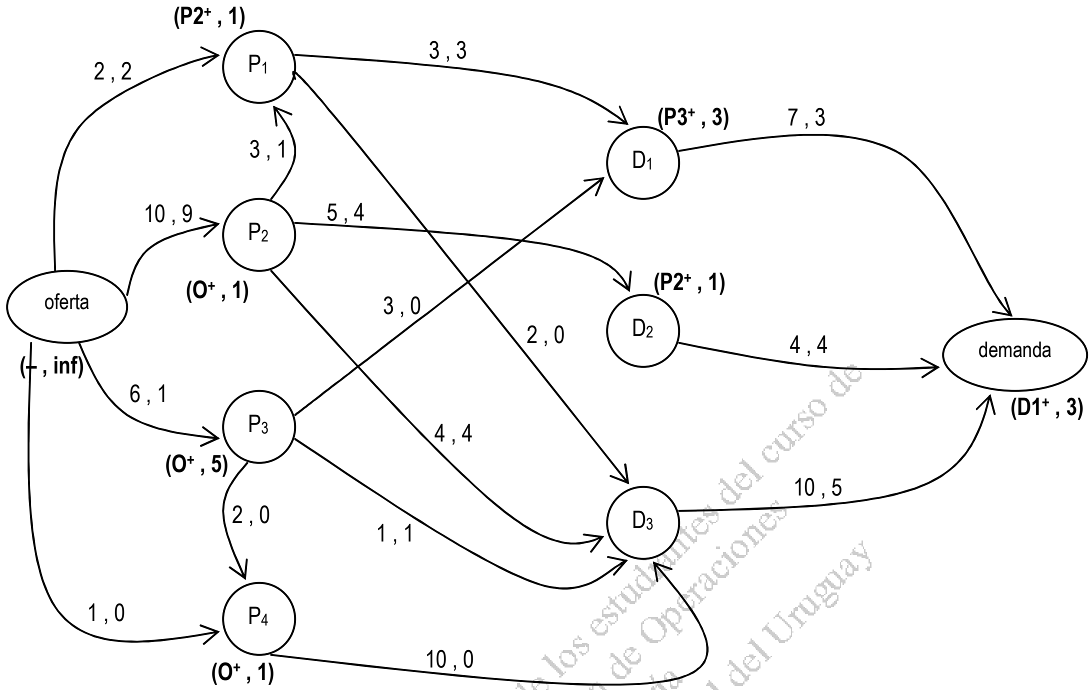


Encontramos O–P_2–P_3–D con holgura mínima 4. Incrementamos el flujo de esa cadena en 4 unidades y seguimos iterando. Llegamos a:

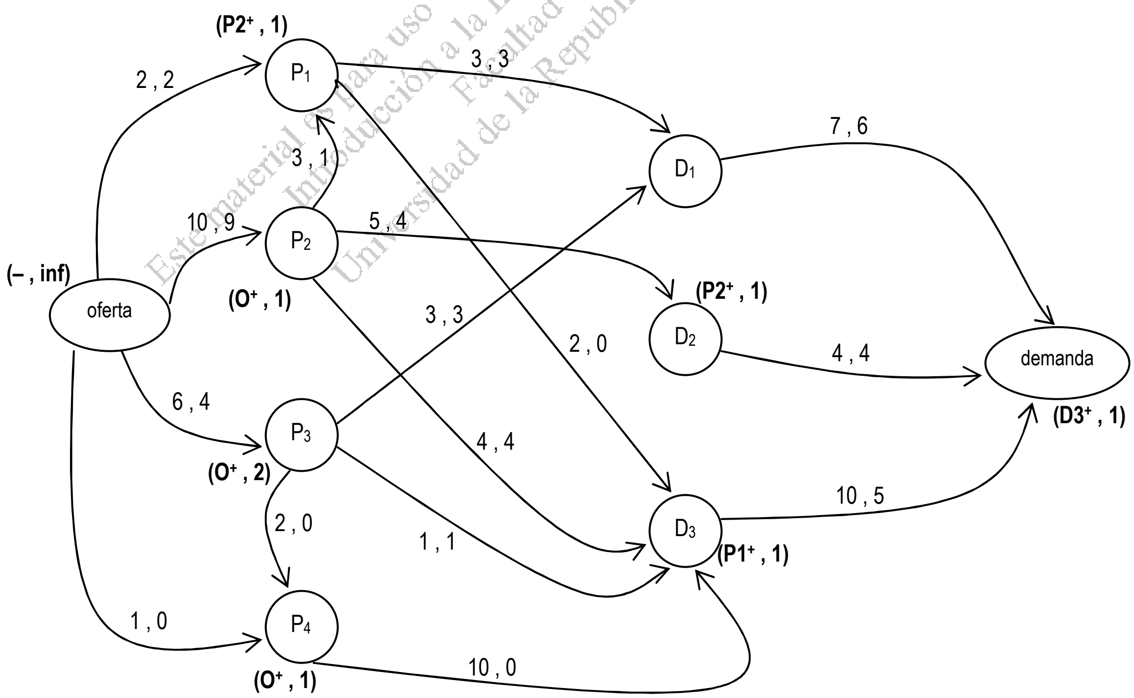


Hallamos O–P_1–D_1–D con holgura mínima 1. Incrementamos el flujo de esa cadena en 1 y volvemos a iterar.

Llegamos a:

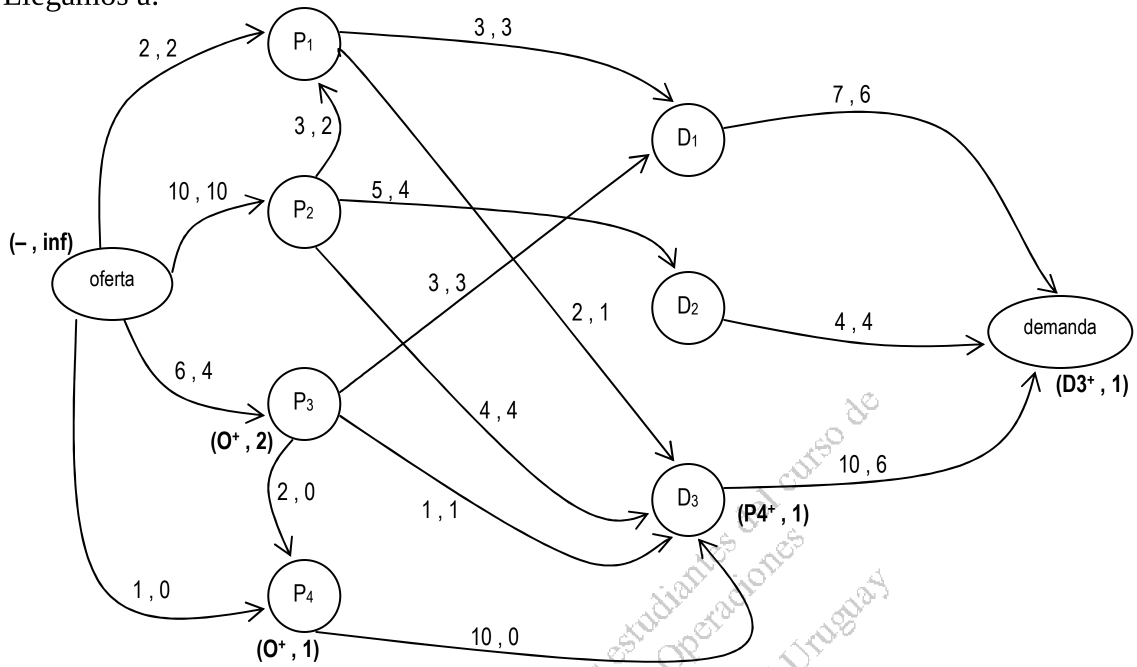


La cadena O–P_3–D_1–D tiene holgura mínima 3. Incrementamos el flujo a lo largo de ella en 3 unidades y seguimos iterando. Llegamos a:

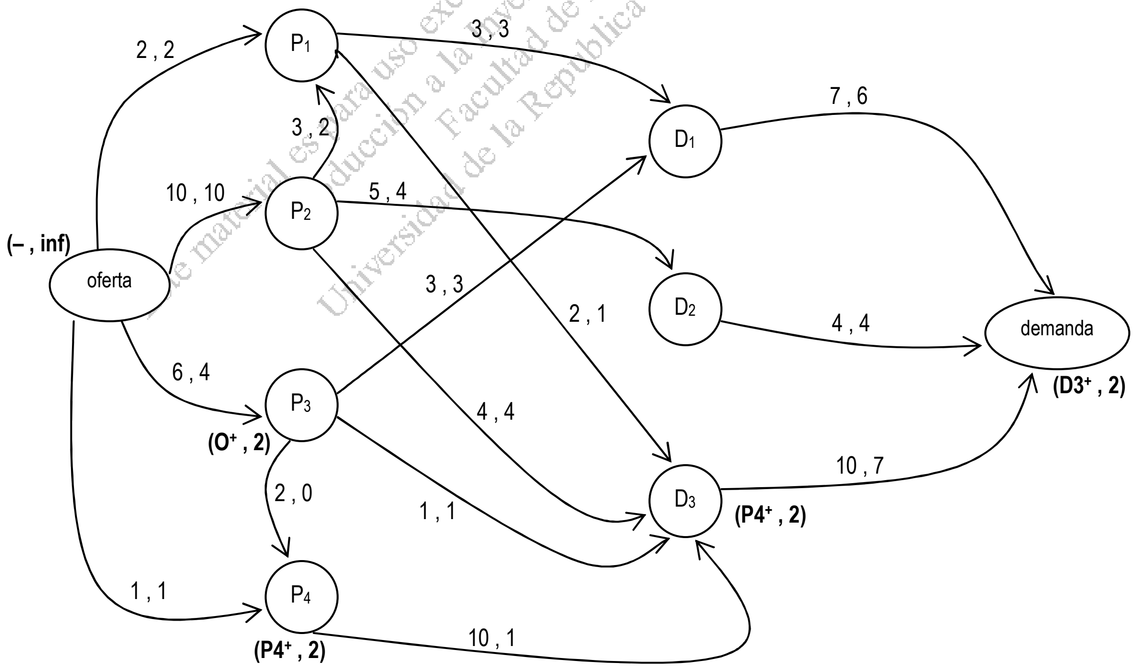


Ahora hallamos O–P_2–P_1–D_3–D con holgura mínima 1. Aumentamos el flujo a lo largo de esa cadena en 1 unidad y seguimos iterando.

Llegamos a:

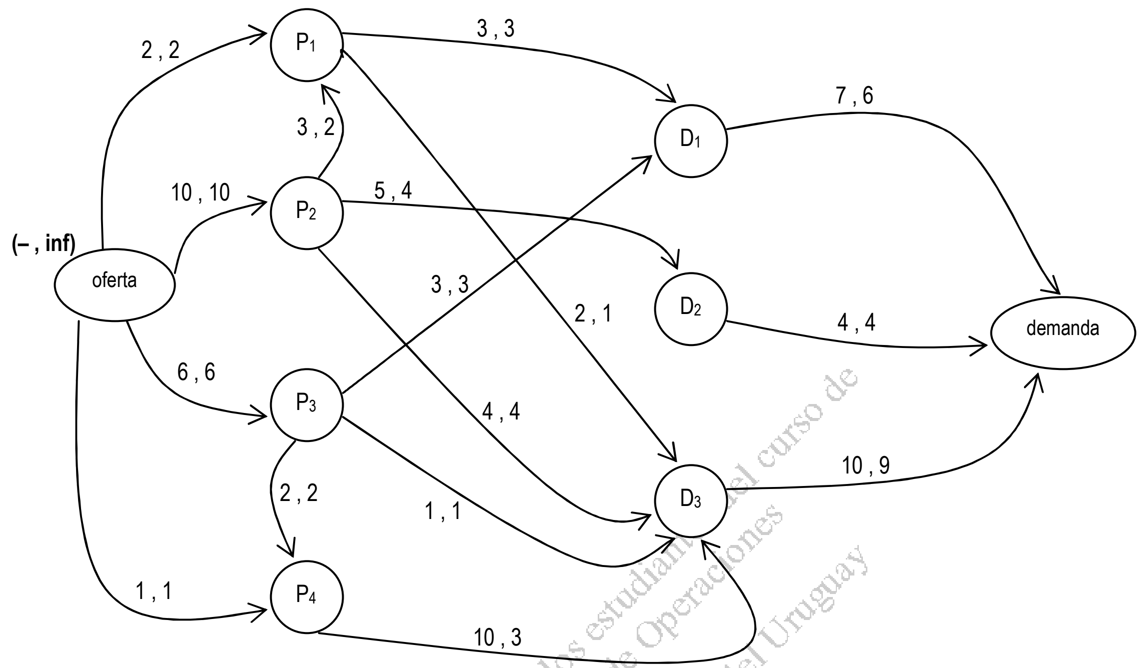


Tenemos la cadena O–P_4–D_3–O con holgura mínima 1. Aumentamos el flujo por ella en 1 unidad y seguimos. Llegamos a:



Hallamos O–P_3–P_4–D_3–D con holgura mínima 2. Incrementamos el flujo y seguimos.

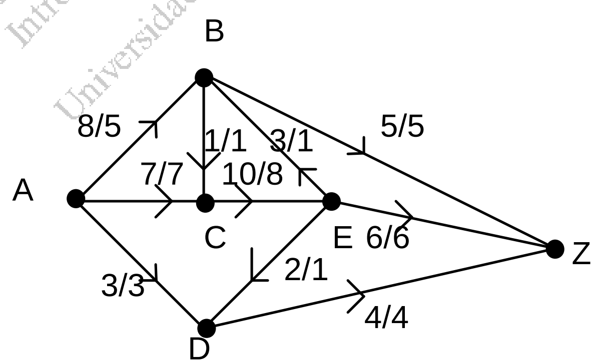
Llegamos a:



Ahora tenemos la fuente marcada. Todos los arcos salientes están saturados y no hay arcos incidentes. Por lo tanto no podemos seguir marcando. Hemos hallado un flujo máximo de valor 19.

Ejercicio 18.

El flujo máximo es



Un corte mínimo es {A, B, C, D, E}.