01168
2ej. 4

NOMBRE DE TESIS

OPTIMIZACION COMBINATORIA CON ENFOQUE MATROIDAL

CRÉDITOS ASIGNADOS A LA TESIS 12 (DOCE)

APROBADO POR EL JURADO

PRESIDENTE:	Dr. José Jesús Acosta Flores
VOCAL	Dr. Sergio Fuentes Maya
SECRETARIO	M en I Antonio Olivera Salazar
SUPLENTE:	Luis Andres Buzo Dr. Andres Buzo de la Peña
SUPLENTE:	M en I Gonzalo Negroe Pérez

I N T R O D U C C I O N

En los últimos años el desarrollo de la Investigación de Operaciones y Ciencias de la Computación ha sido explosivo en términos de retroalimentación de ideas y técnicas para la solución de problemas actuales. Un caso típico de esta interacción es sin duda la Optimización Combinatoria y la extensa variedad de problemas prácticos que plantea y resuelve.

El término combinatoria se refiere al estudio de los ordenamientos o selección de un conjunto finito de objetos. Este término ha sido popularizado al aplicar técnicas para determinar la probabilidad de un cierto subconjunto de acuerdo a la forma axiomática. Una nueva linea de investigación relacionada con el término combinatoria ha ganado particular interés. En este contexto se desea determinar el mejor arreglo o subconjunto de elementos de un conjunto en estudio, de acuerdo a un cierto criterio o función de mérito. Específicamente la optimización combinatoria está definida por una problemática que a primera vista parece sencilla pues el esquema del problema que analiza es sencillo de definir:

Dados un conjunto finito E de elementos; un conjunto finito S de subconjuntos de E y una función $f: S \to \mathbb{R}$ lo que se pretende es determinar el elemento s^* que pertenece a S tal que

$$f(s^*) = \min \{f(s) \mid s \in S\}$$

El proceso trivial de enumeración del conjunto de los casos, cuya solución se antoja sencilla, si usamos el método de comparación exhaustiva de los elementos de acuerdo a f, aunque teóricamente es posible, conduce a procedimientos de una eficacia desastrosa cuando el número de elementos de E es grande, lo cual es inaceptable. Un ejemplo clásico de esta situación es el problema del agente viajero, quien tiene que visitar n ciudades y desea establecer un itinerario que le permita pasar una sóla vez por cada una de las ciudades y finalmente llegar al punto de partida, minimizando la distancia recorrida. Usualmente este problema se formula de la siguiente manera: dada una red R(X,E,d); donde X es el conjunto de nodos que representan las ciudades, E es el conjunto de aristas que unen un par de nodos y d(e) una función positiva que asocia a cada arista de E su longitud, se desea determinar la trayectoria con nodo inicial y final idénticos que pase por todos los nodos una sóla vez y de longitud mínima.

El problema se plantea de la siguiente manera: determinar $s^* \in S$ tal que:

$$f(s^*) = \min \{f(s) \mid s \in S\}$$

donde

$$f(s) = \sum_{e \in S} d(e);$$

$S = (s/s$ es la trayectoria con el mismo nodo inicial y final)

Sabemos que si n es el número de ciudades, entonces $(n-1)!$ es el número de itinerarios a comparar si $n = 100$, $99!$ es un número muy grande.

En contraparte y dentro de este mismo campo de la Optimización Combinatoria existen problemas, que debido a su peculiar estructura, pueden ser resueltos por algoritmos eficaces los cuales se consideran en la literatura especializada como "fáciles". Un ejemplo de uno de estos problemas es el de la determinación del árbol de expansión de peso máximo en una gráfica el cual tiene el siguiente planteamiento:

Dada una gráfica simple y conectada $G:(V.E)$ donde V es el conjunto de nodos y E el de aristas, y una función positiva de ponderación $w: E \to \mathbb{R}$, determinar la gráfica que no forme ciclos y conecte a todos los nodos. Un procedimiento consiste en ir tomando las aristas de mayor peso, siempre y cuando no formen ciclo, hasta formar el árbol. Esto tiene un trabajo Computacional de $(|V|-1)$* iteraciones lo cual es una evidencia de la eficacia de tal proceso. Esto es debido a que este problema tiene inherente una estructura algebraica denominada matroide.

*$|V|$ cardinalidad del conjunto V.

El estudio de la teoría de matroides empieza en la década de los treintas, cuando Van der Waerden propone un sistema de postulados para la dependencia lineal algebráica. Sin embargo es Whitney en 1935 quien usa y define el término de Matroide concibiéndolo como una generalización de la estructura algebráica de matrices. Por tanto mucho del lenguaje de esta teoría está basada en el Algebra lineal. El mismo Whitney introduce el concepto dual de una matroide y observa que el subconjunto de aristas de una gráfica que no tiene circuitos tiene las mismas propiedades de independencia que los subconjuntos linealmente independientes lo que lo lleva a incluir términos de teoría de gráficas.

Es Jack Edmonds quien impulsa la Optimización Combinatoria al aplicar el algoritmo glotón a aquellos problemas de esta área que tengan una estructura fundamental de carácter matroidal. Otros autores encontraron diversas ramificaciones en areas como Geometría y teoría de rejillas. Algunos artículos representativos del desarrollo de esta teoría son los de Birkhoff, Maclanie y Dilworh. En geometría combinatoria están los artículos de Crapo y Rota, en teoría de gráficas está Tutte y en optimización combinatoria, Jack Edmonds.

El objetivo de este trabajo es analizar los fundamentos y resultados básicos de matroides así como su aplicación a la optimización combinatoria con especial énfasis a aquéllos problemas que se resuelven usando la técnica del algoritmo glotón.

Este trabajo se desarrolla como sigue: En el capítulo uno se define el problema de Optimización Combinatoria y se presentan ejemplos típicos, en particular aquellos cuyo método de solución resulta eficaz. Asimismo se discute brevemente el importante concepto de complejidad algorítmica.

El capítulo dos revisa los conceptos básicos de matroides, sus definiciones, equivalencias, operaciones básicas y el concepto de dualidad. Así como la manera en que se usan en algunos problemas de gráficas.

El capítulo tres presenta el algoritmo glotón con aplicaciones sencillas a los problemas de determinación de vectores linealmente independientes, en matrices, el árbol de expansión y finalmente sus variaciones.

El capítulo cuatro discute el problema de acoplamiento en gráficas bipartitas, teoría transversal, transversales, transversales comunes y para cada una, sus representaciones como matroides.

El capítulo cinco plantea dos aplicaciones del algoritmo glotón para encontrar el árbol de expansión de peso máximo o mínimo, específicamente los problemas de síntesis de redes y la construcción de la red de Steiner.

En el capítulo seis se dan las conclusiones.

En el apéndice A se dan los conceptos básicos de gráficas mientras que en el apéndice B se plantea el problema de satisfacción. En el apéndice C se presentan los listados de programas de algunos de los algoritmos glotones que se manejaron en el trabajo, finalmente en el apéndice D se da una explicación intuitiva de la definición de matroide.

C A P I T U L O I

OPTIMIZACIÓN COMBINATORIA

El término combinatorio se refiere al estudio de los ordenamientos o selección de un conjunto finito de objetos y, resuelve problemas sobre la existencia y enumeración de tales arreglos. Recientemente una nueva linea de investigación combinatoria ha ganado particular interés, pues se desea determinar cuál es el mejor arreglo de acuerdo a un cierto criterio. La problemática que se define de esta forma es la Optimización Combinatoria, que parece en primera instancia muy sencilla, el esquema de un problema de este campo es el siguiente: dado un conjunto finito de elementos, y una ponderación que afecte a cada elemento, se desea determinar el arreglo o subconjunto óptimo. Esta problemática tiene un sentido matemático y un gran interés operacional, debido a que su importancia práctica necesitó de la creación de herramientas matemáticas y algorítmicas dando una relación teórico-práctica intensa. En este capítulo se define formalmente el problema de Optimización Combinatoria.

El capítulo se desarrolla como sigue: En la primera sección se proporciona la definición formal del problema de optimización combinatoria y se ilustra con ejemplos típicos. En la segunda sección se plantean los problemas "fáciles" de optimización combinatoria y en la última se da un breve bosquejo de Complejidad Algorítmica.

1.1 El problema de optimización combinatoria.

Una clase importante de problemas de optimización la constituyen los problemas discretos o bien problemas con un número finito de alternativas factibles. Esta clase de problemas se caracteriza por la generalidad con que se trata a la función objetivo y es estudiada en esta sección bajo el nombre de problemas de optimización combinatoria. Dicho problema se define a partir de un conjunto finito S y una función $f:S \rightarrow R$ y lo que se pretende es determinar el elemento $\bar{s} \epsilon S$ tal que

$$f(\bar{s}) = \min \{f(s) \mid s \epsilon S\}$$

Acerca de la función f, diremos que no es diferenciable con lo cual no es posible recurrir a las técnicas del cálculo, para la determinación del mínimo de la función.

En problemas de optimización donde el conjunto S tiene pocos elementos, un método para encontrar el mínimo de $f(s)$ será el exhaustivo; sin embargo hacemos notar que no debemos desorientarnos por la aparente trivialidad de la definición, ya que recurriendo al problema del agente viajero en una red que tiene 101 ciudades, decir que basta con hacer una lista de itinerarios y escoger el mejor, es prácticamente imposible calcular.

Otra manera de plantear el problema de optimización combinatoria es considerando una función objetivo separada de las restricciones, es decir: Si existe un conjunto finito E y una función $c:E \to R$ tal que

. $S \subset 2^E$

. $f(s) = \sum_{e \in s} c(e)$.

Otra forma equivalente es plantearlo como un problema de P.L.* en variable bivalente. Considere una matriz A de orden mxn, un m-vector columna b, un n-vector renglón c, un n-vector x que contiene las variables de decisión, entonces se desea:

$$\text{Max } z = c\,x$$

s.a

$$Ax \leq b$$

$$x_j \in [0,1]$$

y dado que las variables solo pueden tomar valores 0 ó 1, no es posible aplicar los métodos conocidos para resolver problemas de programación lineal con variables reales.

Algunos problemas típicos son:

* P.L. Programación Lineal.

El problema de traslado.

El problema consiste en determinar un itinerario de longitud mínima para trasladarse de una ciudad A a una ciudad B. Se supone que se cuenta con un mapa de carreteras al que se asocia una gráfica $G = (X,E)$ donde X es el conjunto de nodos, los cuales representan los cruceros del mapa, y E es el conjunto de aristas que unen a cada par de nodos. La gráfica del mapa de carreteras se supone simétrica, esto es podemos viajar en ambos sentidos.

Si a cada arista $e \in E$ se asocia su "longitud" $d(e)$, entonces se tiene la red $R = (X,E,d)$.

El problema del traslado se puede plantear como el problema de la ruta más corta del nodo A al nodo B, de la siguiente manera: dados dos nodos A y B, de la red $R = (X,E,d)$ encontrar una sucesión de aristas donde el extremo inicial de la primera arista es A, y el extremo final de la última arista es B, de longitud mínima. La longitud del camino es igual a la suma de las longitudes de los arcos que componen la secuencia.

El problema queda planteado de la siguiente forma, dados:
$S = \{S_i \mid S_i$ es una ruta de A a B$\}$, y

$$f(s) = \sum_{e \in S_i} d(e)$$

Se desea encontrar: $\hat{s} \in S$ tal que $f(\hat{s}) = \min \{f(s) \mid s \in S\}$.

El problema del agente viajero.

Un agente viajero tiene que visitar n ciudades y desea establecer un itinerario que le permita pasar una sola vez por cada una de las ciudades y finalmente llegar al punto de partida, minimizando la distancia recorrida.

Este problema se puede formular de la siguiente manera: dada una red R(X,E,d) donde X es el conjunto de nodos que representan las ciudades, E es el conjunto de aristas que unen un par de nodos y d(e) una función positiva que asocia a cada arista $e \varepsilon E$ su longitud, se desea encontrar una trayectoria con nodo inicial y final idénticas que visite todas las ciudades y tenga longitud mínima.

El problema se plantea de la siguiente manera:

Sean $S = \{S_i \mid S_i$ es trayectoria con el mismo nodo inicial y final$\}$ y,

$$f(s) = \sum_{e \varepsilon S} d(e)$$

se desea encontrar $\hat{s} \varepsilon S$ tal que:

$$f(\hat{s}) = \min \{f(s) \mid s \varepsilon S\}$$

El problema de la mochila.

Considere n objetos, con un peso a_j y un valor c_j para el j-ésimo objeto (j=1,...,n). Se desea seleccionar un subconjunto de esos objetos cuyo peso total sea inferior o igual a un número dado y que el valor de los objetos sea máximo, es decir, determinar $J \subset \{1,2,...,n\}$ tal que $\sum_{j \in J} c_j$ sea máxima con la restricción $\sum_{j \in J} a_j \leq b$.

Si se asocia a cada objeto una variable x_j que pueda tomar el valor 1 si $j \in J$, y 0 si no lo es. La formulación con programación entera de este problema será

$$\max z = \sum c_j x_j$$

sujeto a

$$\sum a_j x_j \leq b$$

$$x_j = \{0,1\} \qquad j=1,2,...,n$$

observe que este problema tiene una función objetivo separada.

Este problema conserva su nombre más por razones tradicionales que representativos de la realidad, ya que no toma en cuenta las utilidades cruzadas que se presentan al llenar una mochila de viaje sin embargo el modelo tiene aplicabilidad en otros problemas.

1.2 Los problemas fáciles de Optimización Combinatoria.

El problema del agente viajero y el de la mochila, presentan una dificultad común, la de encontrar un algoritmo que los resuelva de manera eficiente, cuando el número de ciudades o artículos sea muy grande, ya que todos los métodos conocidos para su solución (exhaustivo, ramificación y acotamiento) tienen una tasa de crecimiento del tiempo computacional exponencial en n (número de ciudades). Aunque existen algoritmos heurísticos de aproximación con una tasa de orden polinomial en n y experimentalmente se ha observado que funcionan bien, para ver algunos de estos procesos ver [4] y [9].

Por fortuna no todos los problemas de optimización combinatoria poseen esa dificultad, y a continuación se presenta una serie de problemas distintos que tienen en común un algoritmo que los resuelve de manera eficaz, lo cual los hace en este sentido fáciles.

La característica, de este tipo de problemas, que consiste en ser resuelto por un algoritmo que esencialmente es el mismo, a pesar de que el plantemaiento y tipo de problema es muy diferente, no se debe a una casualidad, es el marco teórico-abstracto de la teoría de matroides lo que permite que puedan ser resueltos por tales algoritmos.

Para los problemas que a continuación se presentan, se resuelven intuitivamente, con el objeto de construir el procedimiento de solución.

Un problema de semiacoplamiento.

Sea $A = [a_{ij}]$ una matriz de orden mxn no negativa. Suponga que se desea determinar un subconjunto de elementos de la matriz cuyo peso sea máximo, sujeto a la restricción de que no se seleccione dos elementos del mismo renglón. Una manera de escribir este programa es como sigue:

$$\text{Max } Z = \sum\sum a_{ij} x_{ij}$$

sujeto a

$$\sum_{j=1}^{n} x_{ij} \leq 1 \qquad i=1,2,\ldots,m$$

$$x_{ij} \in [0,1]$$

Una forma intuitiva de resolver el problema consiste en seleccionar uno y sólo un elemento de mayor peso de cada renglón. Específicamente si A es:

$$A = \begin{bmatrix} 4 & 6 & 4 & 5 \\ 3 & 8 & 1 & 6 \\ 2 & 9 & 2 & 10 \\ 1 & 2 & 3 & 18 \end{bmatrix}$$

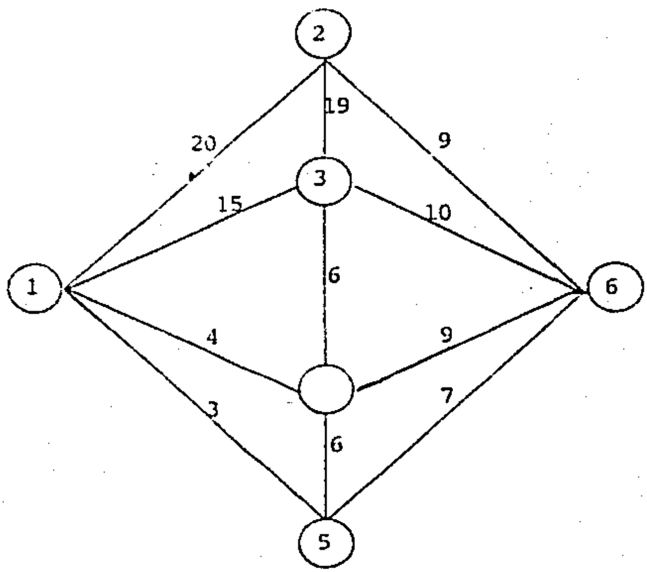
Entonces los elementos con círculo son los seleccionados es decir s = (6, 8, 10 y 18) ∴ $\hat{\bar{x}} = (x_{12} = 1, x_{22} = 1, x_{34} = 1, x_{44} = 1)$.

El árbol de peso máximo.

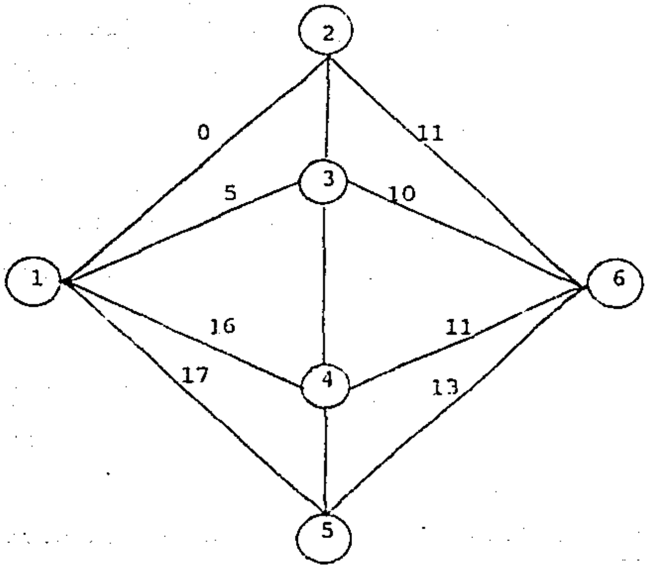
Una red de televisión desea rentar canales de video, de tal manera que las estaciones en varias ciudades puedan formar una red conectada. Cada cable (i,j) que conecta a una ciudad i con una ciudad j tiene un costo por renta c_{ij}. ¿Como construir la red de cableado a un costo total mínimo?.

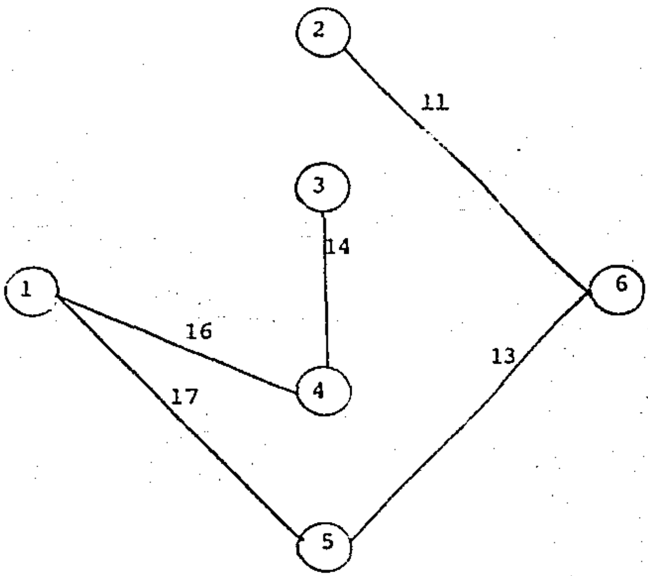
Claramente este problema se puede resolver determinando el árbol de expansión mínima o replantearlo y tratarlo como un problema de árbol de peso máximo, haciendo $w_{ij} = N - C_{ij}$ donde N es el número más grande y, entonces encontrar el árbol de peso máximo. El algoritmo de solución consiste en ir seleccionando las aristas (que representan los canales de video) de mayor peso pero sin que se formen ciclos.

Considere la gráfica siguiente:



A continuación se presenta la gráfica transformada, y la gráfica solución.



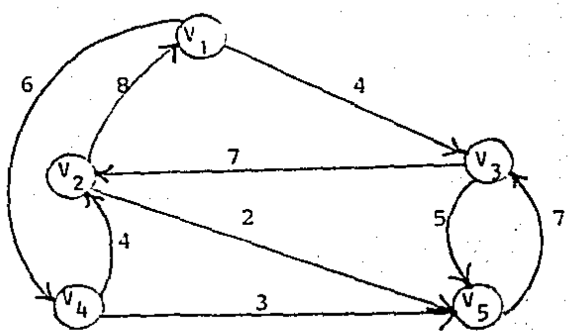


Un problema particular.

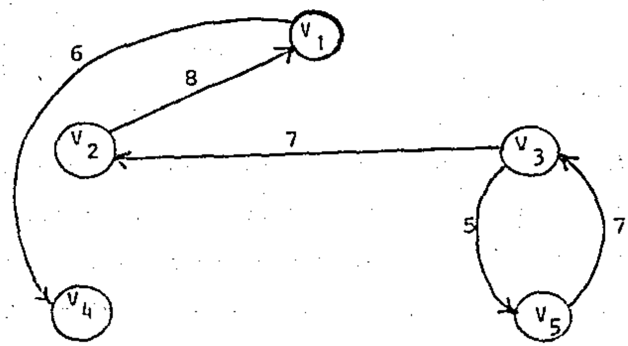
Dada una digráfica D = (V,A) donde V es el conjunto de nodos y A el conjunto de arcos; y una función de ponderación positiva $W:A \to \mathbb{R}$, se desea encontrar un subconjunto B de A con peso más grande tal que no haya dos arcos de B que tengan el mismo nodo final. Este es el problema de optimización combinatorio asociado con una pareja de conjuntos (A,F) donde un subconjunto B de A está en F, si y sólo si no contiene dos arcos de B que finalicen en el mismo nodo.

El procedimiento intuitivo para resolver este problema consiste en aplicar para todos los nodos la siguiente instrucción: Seleccionar el arco con mayor peso que finalice en el nodo v_i (i=1,...,n) y ningún otro arco podrá ser seleccionado. Al terminar de seleccionar se obtendrá el conjunto B de A óptimo.

Considere la digráfica D = (V,A)



Aplicando el procedimiento descrito anteriormente se obtiene:



Un problema de secuenciación.

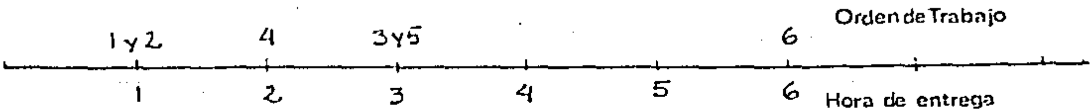
Un cierto número de trabajos van a ser procesados por una sóla máquina. Todos requieren el mismo tiempo de procesamiento, digamos una hora. A cada trabajo se le asigna su hora de entrega y su correspondiente penalización en caso de no estar terminado. El problema es: ¿En qué orden deben entrar los trabajos a la máquina de tal manera que se minimice el costo total de penalización.

De nuevo se intuye que para resolver este problema, podemos proceder a seleccionar el trabajo con mayor penalización y con tiempo de entrega menor; si dos trabajos se tienen que entregar a la misma hora, evidentemente se seleccionará el de mayor penalización.

Supongamos que se tienen los siguientes trabajos con horas de entrega y penalización descritos:

Orden de trabajo	Hora de entrega	Penalización
1	1	10
2	1	9
3	3	7
4	2	6
5	3	4
6	6	2

o más esquemáticamente:



Aplicando el método intuitivo: procederemos a escoger el trabajo uno, en lugar del trabajo 2, ya que el uno tiene penalización mayor. Luego se elige el trabajo cuatro, luego el tres y no el cinco y finalmente el trabajo seis. De esta manera la secuenciación es: 1, 4, 3, y 6, ahorrándose 25 de penalización y solo pagando 13 por los trabajos no entregados a tiempo.

La intensión de presentar estos problemas es observar que tienen como punto común; el algoritmo de solución, que consiste en ir tomando el componente de mayor peso que satisfaga ciertas restricciones. Pues bien este es el esquema del algoritmo glotón o greedy en inglés que fué desarrollado por Jack Edmonds, y cuyo nombre se debe a que consiste en "comer" los elementos de E en un cierto orden sin jamás poner en tela de juicio una selección realizada (lo comido, comido está).

Este algoritmo funciona cuando una familia de soluciones factibles S de un problema de Optimización Combinatoria con una función objetivo separada es hereditaria o cerrada bajo la inclusión es decir si $s \in S$, $s' \subset s \Rightarrow s' \in S$.

Lo que significa que si un conjunto es solución factible cualquier subconjunto del mismo también lo es.

Asi para los problemas de Optimizacion Combinatoria donde se tiene a E un conjunto finito de elementos; $E = \{e_1, \ldots, e_n\}$, una función $f: E \to \mathbb{R}^+$ y $S = \{S \subseteq E \mid S$ es una solución factible$\}$ donde S es cerrado bajo la inclusión, el principio del algoritmo glotón puede ser presentado de la siguiente manera:

<u>Principio del Algoritmo Glotón</u>.

$E = \{e_1, e_2, \ldots, e_n\}$, $f(e_i)$ $i = 1, \ldots, n$.

$(S = \{S \subseteq E \mid S_k = (e_1, \ldots, e_k),\ K \leq n, \mid s \in S,\ s' \subset s \Rightarrow s' \in S\})$

(I es el conjunto de solución óptima)

1. $I = \phi$
2. Para $i = 1, \ldots, n$ hacer

   Si $I \cup \{e_i\} \in S$ entonces $I = I \cup \{e_i\}$

   FIN PARA

FIN ALGORITMO.

Es importante observar que en el paso dos es donde se construye el conjunto óptimo de solución y tanto el conjunto solución S como la pregrunta ¿$I \cup \{e_i\} \in S$?. Están determinados por las restricciones del problema y hacen que este algoritmo sea diferente para cada aplicación.

1.3 Complejidad Algorítmica.

El objetivo de esta sección es dar un sentido preciso de algoritmo eficaz del problema de optimización combinatoria y a los términos de problemas "fácil" y "difícil".

Los algoritmos pueden ser evaluados por una variedad de criterios. En el caso de los problemas de optimización combinatoria, se está interesado en la tasa de crecimiento del tiempo o espacio requerido para resolver problemas "grandes". Para definir cuando un problema es grande o no, se asocia a un problema un entero llamado la dimensión del problema, y corresponde al número de datos de entrada del problema. Esquemáticamente se dice que un algoritmo es eficaz si el número de operaciones necesarias para resolver el problema es acotado por una función polinomial de la dimensión del problema. De aquí es posible deducir directamente que una definición de problemas "fáciles" son aquellos que se resuelven por un algoritmo eficaz.

La teoría de la complejidad algorítmica muestra que una clase amplia de problemas de optimización combinatoria (como el del agente viajero, mochila y otros) tienen una nivel idéntico con respecto a la existencia de un algoritmo polinomial. Es decir si existe un algoritmo polinomial que permite resolver uno sólo de estos problemas, entonces existe un algoritmo para cada uno de ellos que los resuelve.

A este tipo de problemas se les denomina como NP-completos. La existencia de tal equivalencia le da validez a la siguiente conjetura: Los problemas pertenecientes a esta clase son intrínsecamente complejos o bien los problemas NP completos son difíciles.

Para medir la eficacia de un algoritmo dado se establece una relación entre la duración de ejecución del mismo definida por T(n) y la dimensión n del problema, es decir se plantea la siguiente cuestión: ¿Cuáles son las funciones que relacionan la dimensión del problema con el número de operaciones elementales (flops)* del algoritmo de solución, que nos determina cuando un algoritmo es eficaz?.

Si un tiempo de proceso algorítmico para una entrada de dimensión n es cn^2 para alguna constante c, se dice que el tiempo de complejidad de este algoritmo es $O(n^2)$ (de orden n^2) más precisamente: dadas dos funciones $f,g:N \to N$, se dice que f es $O(g)$ si existe una constante c tal que

$$|f(n)| \leq c\ |g(n)|\ \forall\ n \in N$$

Una función f es "polinomial" si es $O(g)$ y g es un polinomio en n, o existen dos constantes c y k tales que

$$f(n) \leq cn^k \quad \forall\ n \in N$$

* (asignación, suma, resta).

Un algoritmo es polinomial si el número de operaciones elementales necesarias para resolver un problema de dimensión n es una función polinomial en n. Un algoritmo es eficáz si y sólo si es polinomial. De esta manera, para calcular la complejidad de un algoritmo, se tiene que contar las operaciones por realizar.

Se podría sospechar que el tremendo incremento en las velocidades de las computadoras podría decrementar la importancia de la eficiencia de los algoritmos desafortunadamente esto no es así, supóngase que se tienen cinco algoritmos $A_1, \ldots, A_5$ con el siguiente tiempo de complejidad

Algoritmo	Complejidad
A_1	n
A_2	$n\log n$
A_3	n^2
A_4	n^3
A_5	2^n

Suponga que una unidad de tiempo es igual a una millonésima de segundo.

El algoritmo A_1 puede procesar en un segundo una entrada de tamaño 1000, mientras que A_5 puede procesar en un segundo una entrada de a lo más 9. En el cuadro se presentan las dimensiones de problemas que pueden ser resueltos en un segundo, un minuto y una hora por cada algoritmo.

Algoritmo	Complejidad	1 seg.	1 min.	1 hora
A_1	n	1000	$6x10^4$	$3.6x10^6$
A_2	nlogn	140	4893	$2.4x10^5$
A_3	n^2	31	244	1897
A_4	n^3	10	39	153
A_5	2^n	9	15	21

Observe que el algoritmo más eficiente es A_1 y el peor es A_5.

Se dice que los algoritmos polinomiales son, de hecho más eficaces que los no polinomiales, con excepción del método simplex que al no ser polinomial es eficaz experimentalmente.

Los problemas NP, NP-Duros y NP-Completos.

La clase de problemas NP, (donde el símbolo NP no quiere decir no polinomiales) es aquella que contiene a los problemas que pueden ser resueltos en tiempo polinomial por un algoritmo no determinístico.

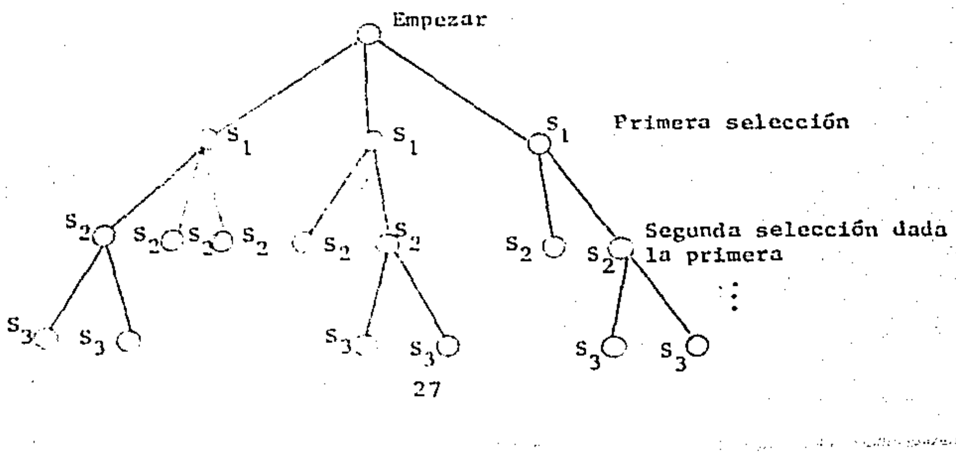
Los algoritmos no determinísticos son una construcción arbitraria y a diferencia de los determinísticos no pueden ser implementados en una computadora, debido a que ningún implemento físico es capaz de tener un comportamiento no determinístico no acotado. Sin embargo el concepto de algoritmo no determinístico tiene un gran interés teórico, pues permite definir la calse NP de manera más rigurosa. Un algoritmo no determinístico contiene la instrucción "selección", la cual opera sobre un conjunto finito, pero no se especifica como esta selección es efectuada y, en cada selección realizada la respuesta puede ser cierta o falsa. De este modo los algoritmos no determinísticos se caracterizan por un lado por saber si existe al menos una manera de efectuar la selección que conduce a la respuesta cierta en un mínimo tiempo y, por el otro se caracterizan en que sólo pueden resolver los de "reconocimientos" estos problemas sólo resuelven sobre la existencia o no de la solución, pero no calcula soluciones. Sin embargo todo problema de Optimización combinatoria es suceptible de reducción a uno de reconocimiento usando este último como vehículo para demostrar si es NP o no.

A continuación se presentan dos ejemplos de algoritmos no determinísticos usando tres instrucciones: SELECCION, FALSO Y CIERTO. La instrucción de Selección(s) es una función de valores múltiples, cuyos valores son los elementos de un conjunto finito S; en cada conjunto S_k estan las alternativas de solución del problema y esta restricción puede explorarlas todas simultaneamente, creando "copias" de si misma para cada alternativa.

La instrucción FALSO detiene la ejecución para esa copia del proceso cuando la selección es incorrecta. La instrucción CIERTO detiene la ejecución de todas las instancias del algoritmo e indica el resultado satisfactorio.

El primer ejemplo es un algoritmo no deterministico que determina cuando en un problema se tiene que hacer una busqueda exhaustiva de un conjunto de posibles soluciones, tiene una solución.

Esquemáticamente tal problema se puede representar por un árbol de búsqueda de soluciones parciales.



Ejemplo 1. Algoritmo no determinístico para determinar cuando un problema tiene solución.

```
Empezar
     K: = 1
     Calcule S_1
     Mientras S_k ≠ φ hacer:
               A_k: = SELECCION (S_k)
               Si (a_1,a_2,...,a_k) es una solución entonces CIERTO
                                                              Si no
                                                              K: = K+1
                                                              Calcule S_k
     //en (a_1,a_2,...,a_{k-1}) no hay solución //
     FIN Mientras
     FALSO
FIN.
```

Este algoritmo por ser no determinístico es solo de reconocimiento, pues solo informa si el problema tiene solución o no, pero no la calcula, y además es importante observar que está acotado polinomialmente pues las operaciones "calcule s_k" y pruebas ¿$S_k = \phi$? y ¿$(a_1, a_2, \ldots, a_k)$ es solución? pueden ser hechos en tiempo polinomial sobre los datos de entrada asi como cada ruta computacional.

Ejemplo 2. Algoritmo no determinístico acotado polinomialmente para determinar cuando, el problema del agente viajero tiene solución con un costo de a lo mas b. (C es la matriz de orden nxn de distancias).

```
EMPEZAR
     S = {2,3,....,n}
     a_1: = 1
     K: = 1
     Cost: = 0
     Mientras S ≠ ∅ hacer
          K: = K+1
          a_k:= SELECCION(S)
          S: = S - {a_k}
          Cost:= cost + C [a_k-1, a_k]
     FIN MIENTRAS
     Cost:= cost + C[an,1]
     Si Cost ≤ b entonces CIERTO
                 si no    FALSO
FIN ALGORITMO.
```

Es evidente que estas construcciones algorítmicas son arbitrarias y hacen caso omiso de consideraciones usuales para las construcciones de algoritmos determinísticos, pero su importancia en la teoria de complejidad algorítimica es relevante pues se puede afirmar que todo problema que se resuelva por un algoritmo no determinístico acotado polinomialmente pertenece a la clase NP.

A partir de definir a la clase de problemas NP como los que se pueden resolver teóricamente en tiempo polinomial por un algoritmo no determinístico es posible deducir que los problemas P que se resuelven por algoritmos determinísticos son un caso particular de los NP es decir P CNP.

Para la mayoria de los problemas de la clase NP no se sabe decir si éstos pueden o no ser resueltos por un algoritmo polinomial. Lo único que se sabe es que no se conoce algoritmo polinomial para resolverlos.

Existe una clase amplia de problemas que son equivalentes desde el punto de vista de la existencia de un algoritmo polinomial para resolverlos en este sentido: si uno sólo de estos problemas puede ser resuelto por un algoritmo polinomial, entonces todos lo pueden.

Para describir esta clase de equivalencia, se define lo que es una reducción polinomial P.: Un problema P_1 se reduce en tiempo polinomial a P_2 si existe un algoritmo para P_1 que recurre a un algoritmo de P_2 y este es polinomial, entonces P_1 tambien es polinomial. Si P_1 se reduce a P_2 y $P_2 \in P$ entonces $P_1 \in P$*.

Un problema es NP-duro, si todo problema de la clase NP puede reducirse polinomialmente a él, y un problema es NP-completo si es NP y NP duro.

Por lo tanto para probar que un problema es NP-duro sólo es necesario probar que es reductible a alguno NP-duro, la dificultad es establcer como punto de partida, que algún problema particular es NP-duro, el cual se puede usar para probar que otros son de la clase NP-duros.

Es Cook quien asegurando la existencia de un problema NP-completo fundamentó la teoría de la complejidad algorítmica pues demuestra que un problema matemático de satisfacción (ver apendice B) es NP-completo y que los problemas de Optimización combinatorio tales como el agente viajero, mochila y otros se pueden reducir polinomialmente al de satisfacción y por tanto son NP completos.

Finalmente podemos clasificar los problemas segun su complejidad algoritmica en "fáciles" y "difíciles": dentro de los primeros hacemos una subclasificación, es decir los "más fáciles"

---

*P - Polinomio

como son: semiacoplamiento, árbol de expansión mínima o máxima y todos aquellos que se resuelven con el algoritmo glotón. Entre los fáciles mencionaremos a el problema de la ruta más corta, flujo máximo etc. y entre los dificiles se tienen a: el clásico y multicitado agente viajero, la red puntos de Steiner, mochila, pandilla, etc.

CAPITULO 2

CONCEPTOS BASICOS DE MATROIDES

Un aspecto importante de los problemas de optimización combinatoria discutidos en el capítulo anterior es su estructura algebraica peculiar. Dicha estructura puede asociarse con el concepto de matroide postulado por Whitney en 1935 como una generalización del concepto de dependencia lineal y sus implicaciones en diversos campos de las matemáticas aplicadas. En los inicios de la teoría matroidal, el énfasis se centraba en aspectos abstractos y de matemáticas puras, sin embargo, los avances recientes de la investigación de operaciones han demostrado su aplicabilidad a problemas reales.

En este capítulo se discuten los aspectos básicos de la teoría de matroides que se usan en capítulos posteriores. El capítulo se desarrolla como sigue: En la primera sección se proporciona la definición básica y se ilustra por medio de ejemplos. En la segunda sección se muestran las formas equivalentes de definir una matroide. En la tercera sección se discuten las operaciones con matroides mientras que en la última se discute el concepto de dualidad.

2.1 Definición básica.

Una manera de motivar los conceptos y la definición básica de matroides es empezar con algunas observaciones de algebra lineal relacionadas con independencia de vectores. Sea

$$E = \begin{bmatrix} 1 & 3 & -6 & 1 \\ 2 & 5 & -10 & 4 \end{bmatrix}$$

y denote por e_i la i-ésima columna de E. Los siguientes subconjuntos de columnas de E son linealmente independientes

$$\{e_1\},\ \{e_2\},\ \{e_3\},\ \{e_4\},\ \{e_1,e_2\},$$
$$\{e_1,e_3\},\ \{e_1,e_4\},\ \{e_2,e_4\},\ \{e_3,e_4\}.$$

En particular se observa que si S es un conjunto de columnas linealmente independientes lo mismo es cierto de cualquier subconjunto de S. También se verifica que si A y B son conjuntos de columnas linealmente independientes de cardinalidad uno y dos respectivamente, entonces existe una columna e_k de $B - A$ tal que $A \cup \{e_k\}$ consiste de vectores linealmente independientes.

Las observaciones anteriores son generalizables y equivalen al siguiente resultado: Sea A matriz mxn. Entonces

P1. Si S es un conjunto de columnas linealmente independientes de A lo mismo es cierto de cualquier subconjunto de S.

P2. Sean I_p e I_{p+1} conjuntos de columnas linealmente independientes de A con p y p+1 elementos, respectivamente. Entonces existe una columna de $I_{p+1} - I_p$, que denotamos por e_k, tal que $I_p \cup \{e_k\}$ es un conjunto de columnas linealmente independientes.

El primer postulado es inmediato y se sigue de las propiedades de independencia lineal. El segundo queda demostrado en el teorema siguiente.

Teorema 1. Sea A una matriz de orden mxn. Denote por a_i el i-ésimo vector columna de A. Sean

$$I_p = \{a_{i_1}, a_{i_2}, \ldots, a_{i_p}\}$$

$$I_{p+1} = \{a_{j_1}, a_{j_2}, \ldots, a_{j_{p+1}}\}$$

subconjuntos de columnas de A linealmente independientes. Entonces existe $a_{j_k} \in I_{p+1} - I_p$ tal que $I_p \cup \{a_{j_k}\}$ consiste de columnas linealmente independientes.

Demostración. Supongamos que esta afirmación es falsa. Entonces todos los elementos de I_{p+1} pertenece al subespacio generado por los elementos de I_p y esto es una contradicción; pues se tendrían p+1 vectores linealmente independientes en un subespacio de dimensión p. Por lo tanto, existe un vector, columna $a_{j_k} \in I_{p+1} - I_p$ con la propiedad señalada.

Dada una matriz A de orden mxn podemos formar la pareja de conjuntos (E,F) donde E es el conjunto de columnas de A y F es una familia de subconjuntos de E linealmente independientes que satisfacen P1) y P2). Estos resultados permiten introducir los siguientes conceptos:

Un sistema de subconjuntos S consiste de un par de conjuntos (E,F) donde F es un conjunto de subconjuntos de E cerrado bajo la operación de inclusión*. El sistema de subconjuntos se denota por S = (E,F) y a los elementos de F se les dice independientes.

Una matroide, es un sistema de subconjuntos con propiedades particulares y se define formalmente como: Una matroide M = (E,F) es un sistema de subconjuntos donde E contiene un número finito de elementos y F es tal que satisface.

M1) $\phi \varepsilon F$

M2) Si $X \varepsilon F$ y $Q \subset X \Rightarrow Q \varepsilon F$

M3) Si I_p, $I_{p+1} \varepsilon F$ con cardinalidad p y p+1 respectivamente, entonces existe $x \varepsilon I_{p+1} - I_p$ tal que $I_p \cup \{x\} \varepsilon F$

Algunos ejemplos clásicos de matroides se describen a continuación.

---

* Si $P \varepsilon F$ y $P' \subseteq P$ entonces $P' \varepsilon F$

Ejemplo 2.1 (Matroide uniforme). Sean los conjuntos finitos $E = \{e_1, e_2, \ldots, e_n\}$, $F = \{I \mid I \subset E$ donde $|I| \leq k$, $k \leq n\}$ entonces $M = (E, F_k)$ es una matroide.

La verificación de esta aseveración es como sigue: Las primeras dos propiedades que definen una matroide se satisfacen directamente. Para demostrar la tercera propiedad observe que si se tienen dos conjuntos I_p, I_{p+1} de F se tienen dos casos: i) $I_p \cap I_{p+1} = \phi$ y ii) $I_p \cap I_{p+k} \neq \phi$. Si i) se satisface entonces $I_p \cup \{x\} \in F$ para cualquier $x \in I_{p+1}$. Si ii) se satisface, entonces existe al menos un elemento y en I_{p+1} tal que $y \in I_{p+1} - I_p$. Por lo tanto $I_p \cup \{y\} \in F$, o bien se concluye la tercera propiedad de una matroide.

Ejemplo 2.2 (Matroide de matriz). Sea A una matriz de orden mxn. Sea E el conjunto formado por las columnas de A y

$$F = \left\{ I \subseteq E \mid I \text{ es un conjunto de columnas linealmente independientes de } A \right\}$$

entonces $M = (E,F)$ es una matroide.

La verificación de la primera y segunda propiedad es directa mientras que la tercera se sigue del teorema 1.

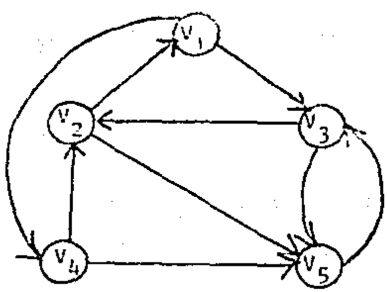
Ejemplo 2.3 (Matroide gráfica). Sea $G = (U, E)$ una gráfica no-dirigida donde U es el conjunto de vértice y E conjunto de aristas. Sea $F = \{I \subseteq E \mid I$ no tienen ciclos$\}$, entonces $M=(E,F)$ es una matroide.

Los dos primeros postulados de matroide se cumplen para el sistema de subconjuntos $M = (E,F)$. Con el propósito de verificar el tercer postulado denote por I_p y I_{p+1}, dos subconjuntos de aristas que no forman ciclo y que contienen p y p+1 aristas, respectivamente. Sea $\bar{v}$ un vértice que es extremo de una arista de I_{p+1} y ninguna arista de I_p incide en él. Denote por $\bar{e}$ la arista de I_{p+1} que incide en $\bar{v}$. Es inmediato que $\bar{e} \in I_{p+1} - I_p$ y el conjunto de aristas $I_p \cup \{\bar{e}\}$ no tiene ciclos. Por lo tanto $M = (E,F)$ es una matroide.

A continuación se presenta un ejemplo más de matroide, y aunque no verificamos las condiciones que definen a una matroide, se presentan dos ilustraciones concretas.

<u>Ejemplo 2.4</u> (Matroide partición). Sea E un conjunto finito y $\Pi = \{E_1, E_2, \ldots, E_k\}$ una partición de E, esto es, una colección de subconjuntos ajenos que cubren a E. Sea $F = \{I \subseteq E \mid I \cap E_j$ $j = 1, \ldots, k$ tiene a lo más un elemento$\}$. Entonces $M = (E,F)$ es una matroide.

<u>Ejemplo 2.5</u> (continúa). Considere la digráfica:



donde $D = (V, A)$ con V conjunto de nodos de la gráfica y A es el conjunto de arcos. Sea $E = A$ donde

$$A = \{(V_1, V_3), (V_1, V_4), (V_2, V_1), (V_2, V_5), (V_3, V_2)$$

$$(V_3, V_5), (V_4, V_2), (V_4, V_5), (V_5, V_3)\}$$

si hacemos $\Pi = \{E_1, E_2, E_3, E_4, E_5\}$ donde $E_1 = \{(V_2, V_1)\}$, $E_2 = \{(V_3, V_2), (V_4, V_2)\}$, $E_3 = \{(V_1, V_3), (V_5, V_3)\}$ $E_4 = \{V_1, V_4)\}$, $E_5 = \{(V_2, V_5), (V_3, V_5), (V_4, V_5)\}$, entonces

$$F = \{I \subset E \mid |I \cap E_j| \leq 1, \; j = 1, 2, \ldots, 5\}$$

Un conjunto I de F podría ser: $I = \{(V_2, V_1), (V_3, V_2), (V_5, V_3), (V_1, V_4), (V_3, V_5)\}$. Este ejemplo está asociado con el problema clásico del agente viajero.

Otro ejemplo de matroide partición es el relacionado con el problema de semiacoplamiento. Específicamente. Sea $W = [w_{ij}]$ una matriz de orden mxn no negativa. Se desea escoger el subconjunto de máximo peso de elementos sujeto a la restricción que no haya dos elementos del mismo renglón. Aquí $E = \{w_{ij}\}$ y la partición a considerar es:

$$\Pi = \{R_1, R_2, \ldots, R_m\} \text{ donde } R_i = \{w_{i1}, w_{i2}, \ldots, w_{in}\}$$

entonces

$$F = \{I \subset E \mid |I \cap R_i| \leq 1 \quad i = 1, 2, \ldots, m\}.$$

Para el ejemplo particular (1,3,1) el subconjunto independiente de elementos de máximo peso es $I = (6, 8, 10, 18)$.

2.2. Caracterizaciones equivalentes de una matroide

En la sección anterior se estableció el concepto de matroide como una generalización del concepto de independencia lineal del algebra lineal o bien, del concepto de árbol usado en gráficas. Una vez definida esta estructura conviene establecer algunas de sus caracterizaciones equivalentes usando los conceptos de base (semejante al algebra lineal), rango, operador cerradura y circuito.

Empezaremos la discusión con una observación. Sea $M = (E,F)$ una matroide y sea X, Y, elementos de F, esto es, cada uno de ellos es un subconjunto independiente. Denote por $|X|$, $|Y|$, la cardinalidad de X, Y, respectivamente y suponga que la cardinalidad de X es menor que la de Y. El subconjunto $Y - X$ es independiente pues es un subconjunto de Y. Conviene preguntarse si existe un subconjunto Z de $Y - X$ tal que $Z \cup X$ sea independiente y tenga la misma cardinalidad de Y. La respuesta es afirmativa. Sin embargo, antes de demostrar tal aseveración y con el propósito de motivar adicionalmente los conceptos de independencia de conjuntos que se manejan posteriormente, mencionaremos que es posible tener un conjunto Z de $Y - X$ tal que la cardinalidad del subconjunto independiente $Z \cup X$ sea mayor que la de Y.

Ejemplo. Considérese la matroide $M = (E,F)$ donde

$$E = \left\{ \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \quad \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \quad \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \right\}$$

y F consiste de todos los posibles conjuntos de elementos de E tales que los vectores correspondientes sean linealmente independientes. Sean

$$X = \left\{ \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \right\} \quad Y = \left\{ \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \right\}$$

y observe que $X \cap Y = \phi$. Asimismo, si hacemos $Z = Y$ se tiene que $Z \subset Y - X$ y $Z \cup X$ es un subconjunto independiente (pues consiste de vectores linealmente independientes) y su cardinalidad es tres, mientras que la cardinalidad de Y es dos.

En virtud del ejemplo anterior podemos decir que X,Y, son subconjuntos independientes de una matroide tales que $|X|<|Y|$ es posible que exista un subconjunto independiente Z en $Y - X$ tal que $Z \cup X$ sea independiente y $|Z \cup X| > |Y|$. En este caso podemos eliminar algunos elementos de Z, para obtener un conjunto Z_0 tal que $Z_0 \cup X$ sea independiente y tenga la misma cardinalidad de Y.

En el teorema 2 se demuestra que dados dos subconjuntos independientes X, Y de una matroide tal que $|X| < |Y|$ entonces existe un subconjunto Z_0 de $Y - X$ tal que $Z_0 \cup X$ es independiente y tiene la misma cardinalidad que Y. La importancia de este resultado esta ligada al concepto de base de matroide que definimos a continuación.

Un subconjunto X de S se dice maximal respecto a una propiedad P si X satisface dicha propiedad y no existe otro subconjunto de S que contenga propiamente a X y satisfaga la propiedad P. Una base B de una matroide $M = (E,F)$ es un subconjunto maximal independiente de E. Equivalentemente, B es un subconjunto independiente y no existe otro subconjunto independiente que lo contenga propiamente. Conviene observar que si X, Y son bases de una matroide necesariamente tienen el mismo número de elementos. Una manera de demostrar esta aseveración es como sigue: Suponga que $|X| < |Y|$. Entonces existe un subconjunto Z_0 de $Y - X$ tal que $Z_0 \cup X$ es independiente y $|Z_0 \cup X| = |Y|$. Sin embargo, esto contradice la maximalidad del conjunto X; pues es una base.

En términos de algebra lineal, el resultado anterior equivale a decir que en un espacio vectorial X todas sus bases tienen el mismo número de vectores; en términos de una gráfica conectada todos los árboles de expansión tienen el mismo número de aristas.

<u>Teorema 2</u> Sea $M = (E,F)$ uma matroide y sea X, Y elementos de F tales que $|X| < |Y|$. Entonces, existe un conjunto Z_0 contenido en $Y-X$ tal que $Z_0 \cup X$ pertenece a F y tiene la misma cardinalidad que Y.

<u>Demostración</u>. Sea $Z_0 \subseteq Y - X$ el conjunto tal que $Z_0 \cup X$ es independiente y $|Z_0 \cup X| \geq |Z \cup X|$ para todo $Z \subseteq Y - X$ tal que $Z \cup X$ es independiente. Si $|Z_0 \cup X| \geq |Y|$ el resultado es inmediato; pues, si acaso, eliminamos algunos elementos de Z_0 para obtener lo que deseamos. Suponga que $|Z_0 \cup X| < |Y|$. Entonces, existe un subconjunto Y_0 de Y tal que $|Y_0| = |Z_0 \cup X| + 1$ Dado que Y_0 es independiente (pues es subconjunto de Y), sabemos que existe un elemento y de $Y_0 - (Z_0 \cup X)$ tal que - $Z_0 \cup X \cup \{y\}$ es independiente y tiene la misma cardinalidad que Y_0. Sin embargo, es inmediato que y pertenece a $Y - X$ y que $Z_0 \cup \{y\}$ pertenece a $Y - X$. Asimismo $|Z_0 \cup X| < |Z_0 \cup \{y\} \cup X|$ y se contradice la maximalidad de Z_0. Esto termina la prueba.

Una consecuencia inmediata de este resultado es dado a continuación y es inmediata en el caso de bases de un espacio vectorial.

<u>Corolario 1</u>. Sea $M = (E,F)$ una matroide. Entonces todas las bases de una matroide tiene la misma cardinalidad.

Un aspecto importante del concepto de base de una matroide, es que dichas bases satisfacen un conjunto reducido de propiedades (Teorema 3) y reciprocamente, es posible definir una matroide usando un conjunto de bases que satisfagan las propiedades mencionadas (Teorema 3). Especificamente, podemos definir una matroide usando la pareja de conjuntos (E,β) donde β es el conjunto de bases de la matroide.

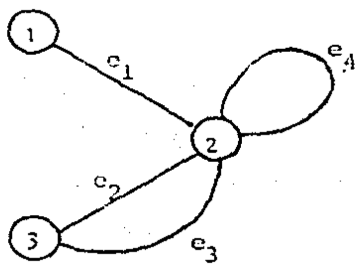
<u>Ejemplo 2</u>. Considerese la matroide $M = (E,\beta)$ donde

$$E = \left\{ \begin{bmatrix} 1 \\ 2 \end{bmatrix}, \begin{bmatrix} 3 \\ 5 \end{bmatrix}, \begin{bmatrix} -6 \\ -10 \end{bmatrix} \right\}$$

y β, conjunto de bases, es dada por

$$\beta = \left\{ \left\{ \begin{bmatrix} 1 \\ 2 \end{bmatrix}, \begin{bmatrix} 3 \\ 5 \end{bmatrix} \right\}, \quad \left\{ \begin{bmatrix} 1 \\ 2 \end{bmatrix}, \begin{bmatrix} -6 \\ -10 \end{bmatrix} \right\} \right\}$$

<u>Ejemplo 3</u>. Considerese la gráfica



y defina la matroide gráfica $M = (E,\beta)$ donde $E = \{e_1, e_2, e_3, e_4\}$ y $\beta = \left\{\{e_1, e_3\} \quad \{e_1, e_2\}\right\}$.

Teorema 3. Sea β el conjunto de bases de una matroide $M = (E,F)$. Entonces se cumple que

i) $\beta \neq \phi$ y no existe conjunto en β que este contenido propiamente en otro conjunto de β.

ii) Si B_1 y B_2 estan en β y $e_1 \varepsilon B_1$, entonces existe $e_2 \varepsilon B_2$ tal que $(B_1 - \{e_1\}) \cup \{e_2\}$ pertenece a β.

Recíprocamente, si (E,β) es una estructura finita que satisface las propiedades anteriores, entonces $M = (E,F)$ donde $F = \{I \mid I \leq B$ para algun $B\varepsilon\beta\}$ es una matroide.

Demostración. Las propiedades i y ii son inmediatos del teorema 2 y las debidas a una matroide. El resultado recíproco es como sigue: la colección de conjuntos F satisface la propiedad de ser no-vacia y ser cerrada respecto a la inclusión. Sean X, Y elementos de F cuya cardinalidad es P y $p + 1$, respectivamente. Sean B_1 y B_2 las bases o elementos de β tales que $X \subset B_1$, $Y \subset B_2$. Específicamente, $X = \{x_1, x_2, \ldots, x_p\}$; $B = \{x_1, x_2, \ldots, x_p, c_{p+1}, \ldots, c_N\}$; $Y = \{y_1, y_2, \ldots, y_p, y_{p+1}\}$ y, $B_2\{y_1, y_2, \ldots, y_{p+1}, r_{p+2}, \ldots, r_N\}$. Hagamos $B_1 - \{c_N\}$. Entonces, existe $z\varepsilon B_2$ tal que $(B_1 - \{c_N\}) \cup \{z\}$ pertenece a β, O bien es una base. Si $z\varepsilon Y$ terminamos pues el conjunto $X \cup \{z\}$ es independiente. De otra manera, podemos repetir el proceso anterior con $\underline{B}_1 = (B_1 - \{c_N\}) \cup \{z\}$ terminamos o bien repetiremos el proceso con pequeños cambios, pero únicamente un número finito de veces. Esto termina la prueba.

Otra manera equivalente de caracterizar una matroide es usando el concepto de o función rango r definida en el conjunto finito E como

$$r(A) = \max \{ |X| \ | X \subseteq A, X \in F \}$$

donde $A \subseteq E$. Note que la función rango r tiene como dominio el conjunto potencia 2^E y como contradominio los números enteros positivos. El rango de E se dice que es el rango de la matroide.

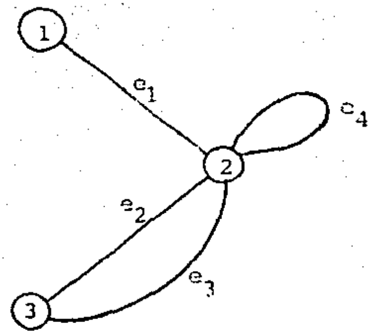
Observe que si tratamos con la matroide matriz, entonces la función rango r equivale a determinar el máximo subconjunto de vectores que es linealmente independiente o bien el concepto de rango de una matriz (o submatriz) usado en algebra lineal. Un ejemplo sencillo de esta observación es como sigue: sea la matroide $M = (E, r(A))$ donde

$$E = \left\{ \begin{bmatrix} 1 \\ 2 \end{bmatrix} \begin{bmatrix} 3 \\ 5 \end{bmatrix} \begin{bmatrix} -6 \\ -10 \end{bmatrix} \begin{bmatrix} 1 \\ 4 \end{bmatrix} \right\}$$

entonces $r(A) = 2$ si $|A| > 2$, esto es, el subconjunto de vectores contenidos en A es mayor que dos o bien $r(A) = |X|$ donde X es el máximo subconjunto de vectores linealmente independientes en A si $|A| \leq 2$.

En el caso de matroide uniforme con $n = 4$ y $k = 2$, la función rango r es sencilla de definir, pues equivale a lo siguiente: $r(A) = 2$ si $|A| > 2$, o bien $r(A) = |A|$ si $|A| \leq 2$ donde $A \subseteq E$.

Un ejemplo mas de la aplicación del concepto de la función rango se tiene en el caso de la matroide asociada a la gráfica $G = (N,E)$ dada como



esto es, la matroide $M = (E,r(A))$ donde $E = \{e_1,e_2,e_3,e_4\}$ y $r(A) = |V(A)| - K(A)$ donde A es cualquier subconjunto de aristas de G, V(A) los vértices de A y K(A) denota el número de componentes en la subgráfica generada por A. Específicamente:

$$r(A) = \begin{cases} 0 & \text{para } A = \{e_4\}; \\ 1 & \text{para } A = \{e_i\} \text{ ó } \{e_i,e_4\} \ i=1,2,3 \\ & \quad \text{y } A = \{e_2,e_3\} \text{ ó } \{e_2,e_3,e_4\}; \\ 2 & \text{para } A = \{e_1,e_2\},\ \{e_1,e_3\}, \{e_1,e_2,e_3\} \\ & \qquad \{e_1,e_2,e_4\},\ \{e_1,e_3,e_4\} \\ & \qquad \{e_1,e_2,e_3,e_4\} \end{cases}$$

El rango de la matroide es $|V(G)| - 1 = 2$ donde $V(G)$ son todos los vértices de G.

En el siguiente teorema se muestran algunas de las propiedades básicas de la función rango y en el corolario correspondiente se justifica su equivalencia con el concepto de matroide.

<u>Teorema 4</u>. Considere una matroide $M = (E,F)$ y una función rango r sobre E. Entonces dados X, A y B subconjuntos arbitrarios de E; y, Z elementos arbitraios de E se satisface:

a. $r(\phi) = 0$

b. $r(X) \leq r(X \cup \{y\}) \leq r(X) + 1$

c. Si $r(X \cup \{y\}) = r(X \cup \{Z\}) = r(X)$ implica que

$r(X \cup \{y\} \cup \{Z\}) = r(X)$

d. $r(A \cup B) + r(A \cap B) \leq r(A) + r(B)$.

<u>Demostración</u> <u>a</u> Es inmediato. <u>b</u> sea $|A| = r(X)$ y $|B| = r(X \cup \{y\})$ donde $A \subset X$ y $B \subset X \cup \{y\}$. Observe que $|A| \leq |B|$; pues si $|A| > |B|$ implicaria que B no es maximal independiente de $X \cup \{y\}$. Notese que $|B| \leq |A| + 1$ teniéndose dos posibilidades: $(A \cup \{y\}) \notin F$ lo que implica que $r(X \cup \{y\}) = r(X)$ y por tanto $|B| = |A|$ o bien $(A \cup \{y\}) \in F$ lo que implica $r(X \cup \{y\} = r(X) + 1$ y por tanto $|B| = |A| + 1$ y $r(X \cup \{y\} \leq r(X) + 1$.

Sea $r(X) = |A|$ donde $A \subseteq X$. Como $r(X \cup \{y\}) = |A|$ entonces $A \cup \{y\} \notin F$ de la misma forma $A \cup \{Z\} \notin F$ entonces $A \cup \{y\} \cup \{Z\} \notin F$

y se implica que $r(A \cup \{y,z\}) = |A| = r(X)$.

d. Sea $X \in F$, $X \subseteq A \cap B$, $r(A \cap B) = q$, $|X| = q$; Sea $Y \in F$, $Y \subseteq A \cup B$, $r(A \cup B) = p$ $|Y| = p$ de aquí $X \subseteq Y$, $|X| \leq |y|$ y por el teorema 2 existe $Z \subseteq Y - X$ tal que $|W \cup Z| = |y|$, si $Z = V \cup W$ donde $V \subseteq A - B$, $W \subseteq B - A$ entonces $Y = X \cup V \cup W$ y X, V, W son ajenos por pares y $X \cup V \in F$, $X \cup V \subseteq A$, $X \cup W \in F$ y $X \cup W \subseteq B$. Entonces

$$r(A) + r(B) \geq |X \cup V| + |X \cup W|$$

$$= 2|X| + |U \cup W|$$

$$= 2|X| + |U \cup W|$$

$$= |Y| + |X|$$

$$= r(A \cup B) + r(A \cap B).$$

Corolario 4.1 Si r es una función sobre un conjunto finito E que satisface las propiedades del teorema, entonces $M=(E,F_{(r)})$ es una matroide donde

$$F_{(r)} = \{I \mid r(I) = |I|\}$$

Otro de los conceptos que caracterizan una matroide es el de circuitos. Antes de proceder a definir el concepto de circuito diremos que un subconjunto X de S es un subconjunto <u>minimal</u> de X. Si X posee una cierta propiedad P y no hay un conjunto contenido propiamente en X y que además posea dicha propiedad. Un circuito de una <u>matroide</u> $M = (E,F)$ es un subconjunto minimal dependiente. La colección de circuitos de M la denotaremos por C.

<u>Ejemplo 1</u>. Considere la matroide uniforme definida por $M = (E,C)$ con $n = 4$, $k = 2$ donde $E = \{e_1,e_2,e_3,e_4\}$.

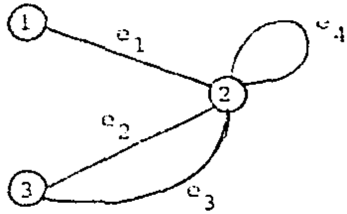
$$C = \{\{e_1,e_2,e_3\}, \{e_1,e_2,e_4\}, \{e_2,e_3,e_4\}\}.$$

<u>Ejemplo 2</u>. Considere la matroide de matriz definida por $M = (E,C)$ donde:

$$E = \left\{ \begin{bmatrix} 1 \\ 2 \end{bmatrix} \begin{bmatrix} 3 \\ 5 \end{bmatrix} \begin{bmatrix} -6 \\ -10 \end{bmatrix} \right\}$$

$$C = \left\{ \begin{bmatrix} 3 \\ 5 \end{bmatrix} \begin{bmatrix} -6 \\ -10 \end{bmatrix} \right\}$$

<u>Ejemplo 3</u>. Considere la matroide gráfica definida por $M=(E,C)$ donde: $E = \{e_1,e_2,e_3,e_4\}$ y $C = \{\{e_4\}, \{e_2,e_3\}\}$



La colección $\mathcal{C}$ de circuitos de una matroide M, tiene las siguientes propiedades:

a. Si C es un circuito entonces $r(C) = |C| - 1$
b. Si C es un circuito entonces $|C| \leq r|E| + 1$
c. Una matroide sin circuitos tiene sólo una base
d. Cada subconjunto propio de un circuito es independiente.

A continuación se presenta otra definición de matroides con base en sus circuitos y en el teorema 5.

<u>Definición 2</u>. Sea E un conjunto finito y $\mathcal{C}$ una familia de subconjuntos de E, entonces $M = (E, \mathcal{C})$ es una matroide si y sólo si satisface:

(1) Si $c_1 \neq c_2 \in \mathcal{C}$ implica que $c_1 \nsubseteq c_2$ ni $c_2 \nsubseteq c_1$.

(2) Si $c_1 \neq c_2 \in \mathcal{C}$ tal que $Z \in c_1 \cap c_2$ implica la existencia de $c_3 \subset (c_1 \cup c_2) - \{Z\}$ tal que $c_3 \in \mathcal{C}$.

La justificación de esta definición se sigue del teorema 5. De dicho teorema también se infiere que si A es un subconjunto independiente en $M = (E, F)$ entonces, para todo $x \in E-A$, el conjunto $A \cup \{x\}$ tiene a lo mas un circuito.

Teorema 5. Sea C familia de circuitos de E, entonces

a. Si $c_1 \neq c_2 \in C$ entonces $c_1 \not\subseteq c_2$ y $c_2 \not\subseteq c_1$

b. Si $c_1 \neq c_2 \in C$ y $Z \in c_1 \cap c_2$ implica que existe un circuito $c_3 \subseteq (c_1 \cup c_2) - \{Z\}$.

Demostración a. Es inmediato. b. Suponga que $c_1 \neq c_2$ son circuitos, tales que no existe circuito $c_3 \subseteq (c_1 \cup c_2) - \{Z\}$. Entonces dicho conjunto es independiente y

$$r((c_1 \cup c_2) - \{Z\}) = |c_1 \cup c_2| - 1;$$

Sin embargo $r(c_1) = |c_1| - 1$; $r(c_2) = |c_2| - 1$ y $r(c_1 \cap c_2) = |c_1 \cap c_2|$. Aplicando la desigualdad de la función rango

$$\begin{aligned} r(c_1) + r(c_2) &\geq r(c_1 \cup c_2) + r(c_1 \cap c_2) \\ &\geq r((c_1 \cup c_2) - \{Z\}) + r(c_1 \cap c_2) \end{aligned}$$

o bien

$$\begin{aligned} |c_1| + |c_2| - 2 &\geq |c_1 \cup c_2| - 1 + |c_1 \cap c_2| \\ |c_1| + |c_2| - 2 &\geq |c| + |c_2| - 1 \\ -2 &\geq -1 \end{aligned}$$

que es una contradicción y existe $c_3 \subseteq (c_1 \cup c_2) - \{Z\}$ con $Z \in c_1 \cap c_2$. ■

Una matroide puede caracterizarce por medio de sus bases, función rango y su colección de circuitos. Una forma adicional para caracterizar una matroide es a través del concepto de operador cerradura. El operador cerradura de un conjunto $A \subseteq E$ es el superconjunto maximal de A que tiene el mismo rango de A; donde el <u>superconjunto maximal</u> se establece por la función: S_p con dominio y contradominio el conjunto potencial de E definida por:

$$S_p(A) = \{x \in E \mid r(A \cup \{x\}) = r(A),\ A \subseteq E\}.$$

Y una matroide se puede caracterizar por la pareja $M = (E, S_p(A))$. La justificación de esta equivalencia no se demuestra, pero puede consultarse en Welsh [ ].

<u>Ejemplo 1</u>. Considere una matroide uniforme con $n = 4$, $k = 2$ definida por $M = (E, S_p(A))$ donde $E = \{e_1, e_2, e_3, e_4\}$, y

$$S_p(A) = \begin{cases} |A| \text{ si } |A| < 2 & (A \subseteq E) \\ E \text{ si } |A| \geq 2 \end{cases}$$

<u>Ejemplo 2</u>. Considere la matroide de matriz definida por $M = (C, S_p(A))$ donde:

$$E = \left\{ \begin{bmatrix} 1 \\ 2 \end{bmatrix}, \begin{bmatrix} 3 \\ 5 \end{bmatrix}, \begin{bmatrix} -6 \\ -10 \end{bmatrix}, \begin{bmatrix} 1 \\ 4 \end{bmatrix} \right\}$$

y $S_p(A) = \{x \in E \mid r(A \cup \{x\}) = r(A)\}$ especificamente, si denotamos como a_i el i-ésimo elemento de E tenemos:

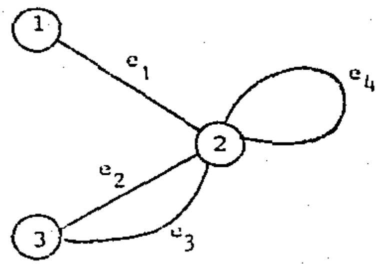
$$S_p(A) = \begin{cases} \{a_1\} \text{ si } A = \{a_1\} \\ \{a_4\} \text{ si } A = \{a_4\} \\ \{a_2, a_3\} \text{ si } A = \{a_2\} \text{ ó } \{a_3\} \text{ ó } \{a_2, a_3\} \\ \quad E \quad \text{para otros casos de } A,\ A \subseteq E. \end{cases}$$

$S_p(E) = E.$

Ejemplo 3. Considere la matroide gráfica de la figura definida por $M = (E, S_p(A))$ donde $E = \{e_1, e_2, e_3, e_4\}$ y $S_p(A) = \{x \in E \mid r(A \cup \{x\}) = r(A)\}$ especificamente

$$S_p(A) = \begin{cases} \emptyset & A = \{e_4\} \\ e_i, e_4 & A = \{e_i\}\ i = 1,2,3 \\ e_2, e_3, e_4 & A = \{e_2, e_3\} \\ e_1, e_2, e_3, e_4 & \text{otros casos de } A,\ A \subseteq E \end{cases}$$

entonces $S_p(E) = E$.



2.3 Operaciones con matroides.

En esta sección se discuten las operaciones con matroides. A través de estas operaciones se generan otras matroides. Las operaciones que se discuten son: aniquilamiento o restricción, contracción, truncamiento, suma directa y suma.

Sea $M = (E,F)$ una matroide y $T \subset E$. Se define la matroide que aniquila a $E-T$ o que se restringe a T como: $M|T = (T, F(E|T))$, donde $F(E|T) = \{X \mid X \subset T \text{ y } X \in F\}$.

Ejemplo 1. Considere la matroide de matriz $M = (E,F,C)$ donde

$$E = \left\{ \begin{bmatrix} 1 \\ 2 \end{bmatrix} \quad \begin{bmatrix} 3 \\ 5 \end{bmatrix} \quad \begin{bmatrix} -6 \\ -10 \end{bmatrix} \right\}$$

sea e_i el elemento i-ésimo de E, entonces:

$$F = \left\{ \emptyset,\ \{e_1\},\ \{e_2\},\ \{e_3\},\ \{e_1,e_2\},\ \{e_1,e_3\} \right\}$$

$$C = \left\{ \{e_2,e_3\} \right\}$$

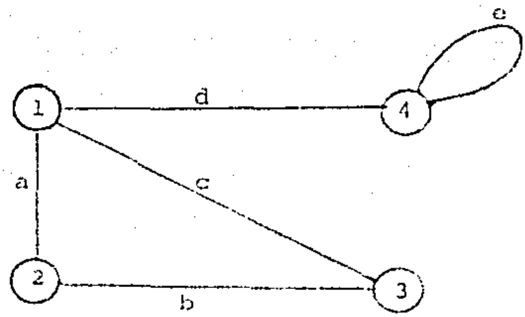
Sea $T = \{e_1\}$ entonces $M|T = (T, F(E|T))$ donde

$$F(E|T) = \left\{ \emptyset,\ \{e_2\},\ \{e_3\} \right\}$$

es decir:

$$F(E|T) = \left\{ \emptyset,\ \left\{ \begin{bmatrix} 3 \\ 5 \end{bmatrix} \right\} \ \left\{ \begin{bmatrix} -6 \\ -10 \end{bmatrix} \right\} \right\}$$

<u>Ejemplo 2</u>. Sea la gráfica $G = (V,E)$



Si se define al conjunto F como el conjunto de subgráficas de G que no contienen ciclo, se tiene la matroide $M = (E,F,C)$ donde: $E = \{a, b, c, d, e\}$ y

$$F = \Big\{\emptyset, \{a\}, \{b\}, \{c\}, \{d\}, \{a,b\}, \{a,c\}, \{a,d\},$$

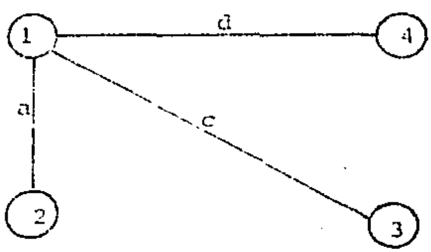
$$\{b,c\}, \{b,d\}, \{a,c,d\}, \{a,b,d\}\Big\},$$

$$C = \Big\{\{e\}, \{a,b,c\}\Big\}$$

Sea $T = \{a,c,d\}$ entonces $M|T = (T, F(E|T))$ donde

$$F(E|T) = \Big\{\emptyset, \{a\}, \{c\}, \{d\}, \{a,c\}, \{a,d\}, \{a,c,d\}\Big\},$$

y la gráfica asociada es



como se observa se aniquiló $\{b,e\}=E-T$, o bien se restringió a T.

Las propiedades de una matroide restringida son:

a. Si x es dependiente en M y $X \subseteq T$ entonces c es dependiente en $M|T$.

b. El rango de $M|T$ es $r(M|T) = r(T)$.

c. Los circuitos de $M|T$ son los circuitos de M contenidos en T.

Sea $M = (E,F)$ una matroide y sea $T \subseteq E$, se define la matroide contraida a T, denotada por $M.T = (T,F.T)$ donde $F(M.T) = \{X \subset T \mid$ existe una base $Y \subset E-T$ en M tal que $X \cup Y \in F\}$.

Ejemplo 1. Considere la matroide matriz del ejemplo anterior. Sea $T = \{a_1\}$, $E - T = \{a_2, a_3\}$ y $B_{E-T} = \left\{\{a_2\},\ \{a_3\}\right\}$, para formar $F(M.T)$, se tiene que probar que $\{a_1\}$ venido a las bases de E-T es independiente en $M = (E,F)$. Se observa que

$$\{a_1\} \cup \{a_2\} \in F \quad y \quad \{a_1\} \cup \{a_3\} \in F$$

por tanto $F(M,T) = \{\phi,\ \{a_1\}\}$ es decir

$$F(M,T) = \left\{\phi \quad \left\{\begin{bmatrix} 1 \\ 2 \end{bmatrix}\right\}\right\}.$$

Ejemplo 2. Considérese la gráfica $G = (V,E)$ del ejemplo de matroide gráfica anterior. Sea $T = \{a,c,d\}$, $E-T = \{b,e\}$ y $\beta_{E-T} = \{b\}$. Para formar $F(M,T)$, se tiene que probar para cada subconjunto de T unido a $\{b\}$ si pertenece o no a F. Se observa que

$\{x\} \subset T$; $\{x\} \cup \{b\} \in F$ implica que $\{x\} \in F(M.T)$ para $x = a,c,d$

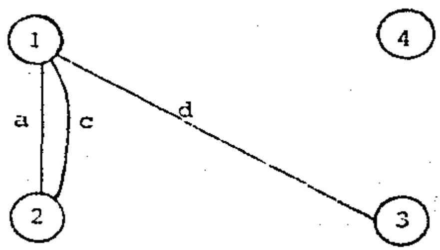
$\{x,y\} \subset T$; $\{x,y\} \cup \{b\} \in F$ implica que $\{x,y\} \in F(M.T)$ para

$\{x,y\} = \{a,d\}$ o $\{c,d\}$. Pero $\{a,c\} \subset T$ y

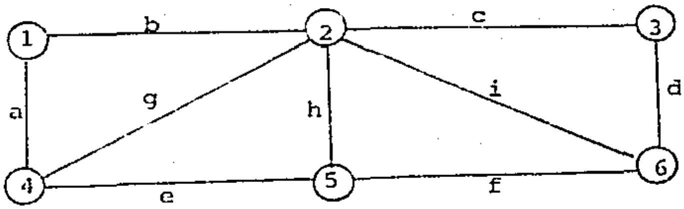
$\{a,c\} \cup \{b\} \notin F$ por tanto $\{a,c\} \notin F(M,T)$.

en resumen:

$F(M,T) = \{\phi, \{a\}, \{c\}, \{d\}, \{a,d\}, \{c,d\}\}$, y la gráfica asociada a $M(T, F(M.T))$ es:



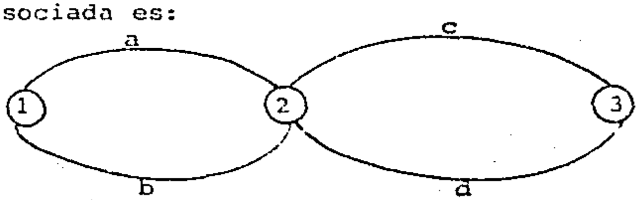
Ejemplo 3. Considérese la gráfica $G = (V,E)$, $E = \{a, b, c, d, e, f, g, h, i,\}$



La matroide asociada es $M = (E,F)$ donde $F = \{A \subseteq E \mid A$ no contiene ciclo$\}$. Sea $T = \{a,b,c,d,e,f\}$, entonces $E-T = \{g,h,i\}$, $B_{E-T} = \{g,h,i\}$. La matroide contraida a T es $M = (T, F(M.T))$ donde $F(M.T) = \{X \subseteq T \mid$ Existe una base $y \in E-T$ de M tal que $x \cup y \in F\}$, específicamente

$$F(M,T) = \left\{\phi,\ \{a\},\ \{b\},\ \{c\},\ \{d\},\ \{a,c\},\ \{a,d\},\ \{b,c\},\ \{b,d\}\right\}$$

cuya gráfica asociada es:



Sea $M = (E,F)$ una matroide y $K \leq r(E)$ k entero positivo. Una matroide $M_k = (E,F_k)$ truncada a k, tiene su familia de conjuntos independientes como $F_k = \{X \mid X \in F \text{ y } |X| \leq K\}$, $r(A) = \min \{K, r(A)\}$.

Ejemplo 1. Considere la matroide de matriz $M = (E,F)$ donde

$$E = \left\{ \begin{bmatrix} 1 \\ 2 \end{bmatrix} \begin{bmatrix} 3 \\ 5 \end{bmatrix} \begin{bmatrix} -6 \\ -10 \end{bmatrix} \right\}$$

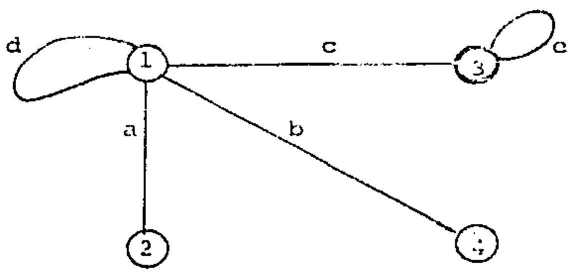
Sea e_i el i-ésimo elemento de E, entonces

$$F = \left\{ \phi,\ \{e_1\},\ \{e_2\},\ \{e_3\},\ \{e_1,e_2\},\ \{e_1,e_3\} \right\}$$

el rango de la matroide es $r(M) = r(E) = 2$. Sea $k=1$, entonces la matroide truncada a 1 es $M_1 = (E_1 F_1)$ con $r(M_1)=1$ y

$$F_1 \quad \left\{ \phi,\ \{e_1\},\ \{e_2\},\ \{e_3\} \right\}.$$

Ejemplo 2. Sea $G = (V,E)$, $E = \{a,b,c,d,e\}$

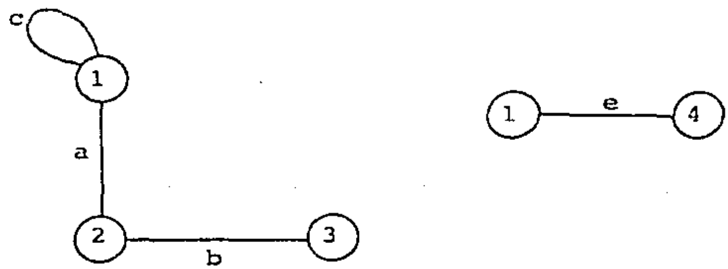


cuya matroide asociada es $M = (E,F)$, donde $F = \{A \subseteq E \mid A$ no contiene ciclo$\}$, rango $= r(M) = r(E) = 3$. Sea $k = 2$, entonces la matroide truncada a 2 es $M_2 = (E_1 F_2)$ con $r(M_2) = 2$ y

$$F_2 = \left\{ \phi,\ \{a\},\ \{b\},\ \{c\},\ \{a,b\},\ \{a,c\},\ \{b,c\} \right\}$$

Finalmente sean $M' = (E', F')$ y $M'' = (E'', F'')$ dos matroides donde $E' \cap E'' = \phi$, se define la <u>suma directa</u> $M' \oplus M'' = (E' \cup E'', F)$ donde $F = \{A \cup B \mid A \subset F',\ B \subset F''\}$.

<u>Ejemplo 1</u>. Sean G' y G" las siguientes gráficas



entonces las matroides asociadas respectivas son:

$M' = (E', F')$ donde $E' = \{a,b,c\}$, $F' = \{\phi, \{a\}, \{b\}, \{a,b\}\}$

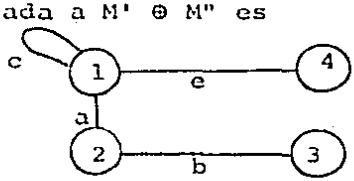
y $M' = (E'', F'')$ donde $E'' = \{e\}$, $F' = \{\phi, \{e\}\}$.

La matroide generada por la suma directa es:

$$M' \oplus M'' = (E' \cup E'', F);\ E' \cup E'' = \{a,b,c,e\}$$

y $F = \{\phi, \{a\}, \{b\}, \{e\}, \{a,b\}, \{a,e\}, \{b,e\}, \{a,b,e\}\}$

La gráfica asociada a $M' \oplus M''$ es



La operación de suma de matroides se define como la anterior sin pedir que $E' \cap E'' = \phi$.

2.4 Dualidad

En Programación Lineal el concepto de dualidad consiste en asociar a todo problema lineal un correspondiente problema lineal dual. En el caso de matroides, dada una matroide $M = (E,F)$ existe una matroide dual definida como $M^* = (E,F^*)$ donde cada base de M^* es el complemento de una de M que se le denomina cobase de M, y viceversa; en estos términos la familia de subconjuntos independientes de la matroide dual M^* se define como $F^* = \{I^* \mid \text{existe } B^* \varepsilon \beta^* \text{ tal que } I^* \subseteq B^*\}$ y β^* es el conjunto de bases de M^*. Esta definición se justifica en el teorema 6 (Whitney) y da entrada a propiedades y relaciones tales como que los circuitos y rango de M^* son los correspondientes cocircuitos y corrango de M y viceversa, lo cual permite una manipulación y aplicación de las matroides más amplia.

En términos de un conjunto de vectores linealmente independientes, el concepto de matroide dual se observa en el siguiente ejemplo:

Ejemplo. Sea $M(E,\beta)$ una matroide de matriz donde

$$E = \left\{ \begin{bmatrix} 1 \\ 2 \end{bmatrix} \begin{bmatrix} 3 \\ 5 \end{bmatrix} \begin{bmatrix} -6 \\ -10 \end{bmatrix} \begin{bmatrix} 1 \\ 4 \end{bmatrix} \right\}$$

$$\beta = \left\{ \left\{ \begin{bmatrix} 1 \\ 2 \end{bmatrix} \begin{bmatrix} 3 \\ 5 \end{bmatrix} \right\} \left\{ \begin{bmatrix} 1 \\ 2 \end{bmatrix} \begin{bmatrix} -6 \\ -10 \end{bmatrix} \right\} \left\{ \begin{bmatrix} 1 \\ 2 \end{bmatrix} \begin{bmatrix} 1 \\ 4 \end{bmatrix} \right\} \right.$$

Ahora escoja un elemento $e \in B_{p+1} - (B_p \cup I^*_p)$. $B_p + e$ contiene un único circuito en M. Sea $e' \neq e$ un elemento de ese circuito, el conjunto $B'_p = B_p \cup \{e\} - \{e'\}$ es una base de M disjunta de I^*_p. Si $I^*_{p+1} - (I^*_p \cup B'_p) \neq \phi$, entonces es el caso 1. Si $I_{p+1} - (I_p \cup B'_p) = \phi$ repetimos este argumento con B'_p en lugar de B_p hasta que se obtenga una base que nos lleva al caso 1, esto ocurre en un número finito de iteraciones.

Existe una variedad interesante de resultados que relacionan a las matroides duales entre ellas, a saber:

a) $(M^*)^* = M$

b) Si $x \in E$ y $\{x\} \notin F$ implica $x \in B^*$ para toda $B^* \in \beta^*$

c) El rango de $M^* = r^*(M^*) = |E| - r(M)$

que pueden interpretarse como sigue: en relación a la primera propiedad, podemos observar que es semejante a la de programación lineal, esto es el dual del dual es el primal, además se sigue de la definición de B^*, pues $B^* = E - B$ para toda $B \in \beta$ entonces $B = E - B^*$ y $(B^*)^* = E - B^* = B$.

En la segunda propiedad se observa que si existe un elemento de E que sea dependiente entonces estará en todas las bases de la matroide dual, esto se observa claramente en el ejemplo de la matroide gráfica, pues siendo $e_4 \in E$ y e_4 un ciclo, entonces e_4 esta en las dos bases de la matroide dual.

$$\left\{\begin{bmatrix}3\\5\end{bmatrix}\ \begin{bmatrix}1\\4\end{bmatrix}\right\}\ \left\{\begin{bmatrix}-6\\-10\end{bmatrix}\ \begin{bmatrix}1\\4\end{bmatrix}\right\}\Bigg\}$$

entonces $M^* = (E,\mathcal{B}^*)$ donde

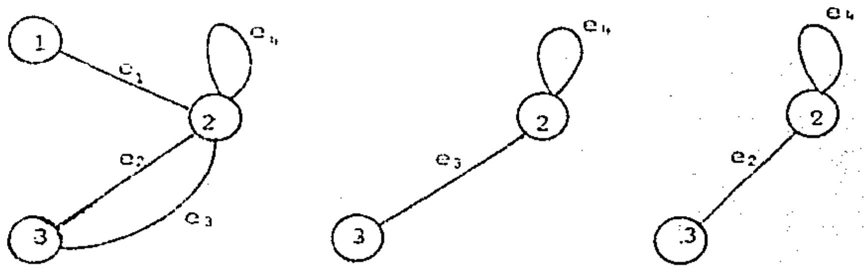
$$\left\{\begin{bmatrix}-6\\-10\end{bmatrix}\ \begin{bmatrix}1\\4\end{bmatrix}\right\}\ \left\{\begin{bmatrix}3\\5\end{bmatrix}\ \begin{bmatrix}1\\4\end{bmatrix}\right\}\ \left\{\begin{bmatrix}3\\5\end{bmatrix}\ \begin{bmatrix}-6\\-10\end{bmatrix}\right\}$$

$$\left\{\begin{bmatrix}1\\2\end{bmatrix}\ \begin{bmatrix}-6\\-10\end{bmatrix}\right\}\ \left\{\begin{bmatrix}1\\2\end{bmatrix}\ \begin{bmatrix}3\\5\end{bmatrix}\right\}$$

y $F^* = \{I^* \mid \text{existe } B^*\varepsilon\mathcal{B}^* \text{ tal que } I^* \subseteq B^*\}$.

En el caso especial de una matroide gráfica su matroide dual será una cográfica. Si la gráfica es conectada, sus árboles son las bases de M y los coarboles (cortes) son las bases de M^*. Hay que notar que no necesariamente es cierto que un co-ciclo es el complemento de un ciclo y en matroides tampoco es necesario que el complemento de un circuito sea un co-circuito.

Ejemplo. Sea $M = (E,\mathcal{B})$ una matroide gráfica donde $E = \{e_1,e_2,e_3,e_4\}$ y $\mathcal{B}$ $\{\{e_1,e_2\},\{e_1,e_3\}\}$ entonces $M^* = (E,\mathcal{B}^*)$ donde $\mathcal{B}^* = \{\{e_3,e_4\},\{e_2,e_4\}\}$ $F^* = \{\phi,\{e_2\},\{e_3\},\{e_2,e_4\},\{e_3,e_4\}\}$



Teorema 6. Sea $M = (E,F)$ una matroide cuya familia de bases es $\beta = \{B \mid B \text{ es base de } M\}$ entonces $\beta^* = \{E-B \mid B\varepsilon\beta\}$ es la familia de bases de una matroide denotada por $M^* = (E,F^*)$ donde $F^* = \{I^* \mid \text{existe } B^*\varepsilon\beta^* \text{ tal que } I^* \subseteq B^*\}$.

Demostración. Los axiomas M1), M2) y M3) son satisfechos por M^*, más aun $F^* \neq \phi$ puesto que $\phi\varepsilon F^*$. Sean I^*p, I^*_{p+1} dos conjuntos en F^* con p y $p+1$ elementos respectivamente. Sea B_p y B_{p+1} dos bases de M disjuntas de I^*_p, I^*_{p+1} respectivamente entonces se presentan dos casos:

Caso 1. Suponga $I^*_{p+1} - (I^*_p \cup B_p) \neq \phi$ y sea $e \varepsilon I^*_{p+1} - (I^*_p \cup B^*_p)$ entonces $I_p \cup \{e\}$ es disjunto de B_p, entonces $I^*_p \cup \{e\} \varepsilon F^*$ y M1 se demuestra.

Caso 2. Suponga $I^*_{p+1} - (I^*_p \cup B_p) = \phi$ primero deseamos demostrar que $B_{p+1} - (B_p \cup I^*_p)$ es no vacio. Suponga $B_{p+1}-(B_p \cup I^*_p)=\phi$ es decir $B_{p+1} \subseteq B_p \cup I^*_p$ entonces se tiene las siguientes relaciones:

$$(B_{p+1} - I^*_p) \cup (I^*_{p+1} - I^*_p) \subseteq B_p$$

$$(B_{p+1} \cap I^*_p) \cup (I^*_{p+1} \cap I^*_p) \subseteq I^*_p$$

$$B_{p+1} \cup I^*_{p+1} \subseteq B_p \cup I^*_p$$

de donde $$|B_{p+1}| + p+1 \leq |B_p| + p$$

$$p+1 \leq p!!$$

por tanto $$B_{p+1} - (B_p \cup I^*_p) \neq \phi$$

La tercera propiedad que relaciona los rangos de un par de matroides duales se justifica en el teorema 7, y podemos obtener los rangos para los dos ejemplos anteriores; en el ejemplo de matroides duales de matriz se tiene:

$$r^*(M^*) = |E| - r(M)$$
$$r^*(M^*) = 4 - 2 = 2.$$

En el ejemplo de matroides duales gráficas se tiene igualmente

$$r^*(M^*) = |E| - r(M)$$
$$= 4 - 2 = 2.$$

Teorema 7. La función rango de una matroide de M y su dual M* estan relacionadas por:

$$r^*(A) = |A| + r(E - A) - r(E) \qquad A \subseteq E$$

Demostración. El rango de A en M* está determinado por una base de M con un número mínimo de elementos en A. La máxima cardinalidad de un conjunto independiente de M, disjunto de A, es r(E - A), tal conjunto está contenido en A. El número de elementos en A no contenidos en esta base es:

$$|A| + r(E - A) - r(E).$$

Corolario 7.1 $r^*(M^*) = |E| - r(M)$; $r(E) = r(M)$ y $r^*(E) = r^*(M^*)$ si $A = E$ entonces $r(E) = |E| - r(E)$, entonces $r^*(M^*) = |E| - r(M)$.

Finalmente mencionaremos la relación entre las operaciones de restricción y contracción con respecto a la dualidad, la aplicación de la truncación a una matroide corresponde a la contracción de su dual y viceversa es decir:

Para alguna matroide $M = (E,F)$ y un subconjunto $T \subseteq E$ se cumple

$$(M \mid T)^* = M^*.T$$

$$(M \, . \, T)^* = M^* | T$$

CAPITULO 3

EL ALGORITMO GLOTÓN

Los problemas de Optimización Combinatoria que involucran la estructura de una matroide, pueden ser resueltos por el enfoque del algoritmo glotón, desarrollado por Jack Edmonds, y cuyo principio fundamental consiste en seleccionar los elementos de máximo peso del conjunto de la matroide para formar una base óptima. Esto hace que los algoritmos sean muy eficientes e implica que los problemas resueltos por este algoritmo sean "fáciles"

En este capítulo se analiza la forma en que el algoritmo glotón resuelve los problemas combinatorios que tienen estructura de una sola matroide. En la primera sección se introduce el concepto de orden lexicográfico, teoremas de unicidad de la base óptima y se mencionan sus aplicaciones más comunes. En la segunda se describe el algoritmo glotón para matroides de matrices se justifica, se discute su complejidad, se presenta en seudo código y se ejemplifica. En la tercera sección se describe el problema del árbol de expansión mínima y el algoritmo glotón, presentando dos algoritmos con seudocódigo, discusión de su complejidad y ejemplo para cada uno, y en la última sección una variación del algoritmo glotón para el problema de dualidad.

3.1 El algoritmo glotón para Matroides.

Considere una matroide $M = (E,F)$ y una función de ponderación positiva $w:E \to \mathbb{R}$, es decir, a cada elemento $e_i \varepsilon E$ se le asigna un peso no-negativo $w(e_i) \geq 0$. Se desea encontrar un conjunto independiente maximal para el que la suma de las ponderaciones de sus elementos sea máxima. Al ponderar los elementos de la matroide se induce un orden lexicográfico en los conjuntos independientes.

Una manera de definir el orden lexicográfico es como sigue: considere, $I_1 = \{a_1, \ldots, a_m\}$ e $I_2 = \{b_1, \ldots, b_n\}$, donde los elementos están listados decrecientemente según su peso, es decir $w(a_1) \geq w(a_2) \geq \ldots \geq w(a_m)$ y $w(b_1) \geq w(b_2) \geq \ldots \geq w(b_n)$. Se dice que I_1 es lexicográficamente mayor que I_2 si existe algún k tal que $w(a_i) = w(b_i)$ para $1 \leq i \leq k - 1$ y $w(a_k) > w(b_k)$ o si $w(a_i) = w(b_i)$ para $1 \leq i \leq n$ y $m > n$. Un conjunto que es lexicográficamente mayor que cualquier otro, es <u>lexicográficamente máximo</u>. Obviamente un conjunto lexicográficamente máximo debe ser una base y si todos los elementos ponderados son distintos esta base es única. Un conjunto lexicográficamente máximo tiene peso máximo.

Teorema 3.1 (Rado Edmonds). Sea F la familia de conjuntos independientes de una matroide, con una ponderación no negativa de los elementos en E entonces existe un conjunto lexicográficamente máximo en F que tiene peso máximo. Recíprocamente, si $M = (E,F)$ es un sistema de subconjuntos que satisface esta condición entonces M es una matroide.

Demostración. Sea F la familia de conjuntos independientes de una matroide cuyos elementos están ponderados. Sea $B = \{b_1, \ldots, b_n\}$ una base lexicográficamente máxima y sea $I = \{a_1, a_2, \ldots, a_m\}$ algún otro conjunto independiente donde $w(b_1) \geq w(b_2) \geq \ldots \geq w(b_n)$ y $w(a_1) \geq w(a_2) \geq \ldots \geq w(a_m)$.

Suponga $w(b_k) < w(a_k)$ para algún k, considere $B_{k-1} = \{b_1, b_2, \ldots, b_{k-1}\}$, $I_k = \{a_1, a_2, \ldots, a_k\}$. Entonces existe $a_i \in I_k - B_{k-1}$ tal que $B_k = \{b_1, b_2, \ldots, b_{k-1}, a_i\}$ es independiente y lexicográficamente mayor que B puesto que $w(a_i) \geq w(a_k) > w(b_k)$ para $1 \leq i \leq k$, lo cual contradice que B es lexicográficamente máximo y por tanto $w(b_p) \geq w(a_p)$ para todo p, y B es un conjunto independiente de peso máximo. Recíprocamente. Suponga que M no es una matroide, entonces puede existir un conjunto $A \subseteq E$ y dos conjuntos maximales I e I' de A en F donde $|I| < |I'|$. Suponga

$$w(e_i) = \begin{cases} 1+\varepsilon & \text{si } e_i \in I \\ 1 & \text{si } e_i \in I' - I \\ 0 & \text{en otro caso} \end{cases}$$

Entonces I está contenido en un conjunto lexicográficamente máximo cuyo peso es menor que I' y la condición del teorema no se cumple, por lo tanto M es matroide.∎

Como se demostró en el teorema una base lexicográficamente máxima no sólo tiene peso máximo, sino que tiene elemento a elemento mayor ponderación que cualquier otro conjunto independiente.

Se dice que un conjunto B en F es óptimo en el sentido de Gale si para algún otro conjunto $I \varepsilon F$ existe una función h uno a uno, $h: I \rightarrow B$ tal que $w(e) \leq w(h(e))$ para todo I. Así, solamente las bases pueden ser óptimas en este sentido, pues como se vió en el capítulo anterior para todo conjunto independiente, siempre habrá una base que lo contenga, entonces además de que $|B| \geq |I|$, los elementos de B tiene mayor ponderación elemento a elemento que los de I.

Teorema 3.2 (Gale). Sea F la familia de conjuntos independientes de una matroide entonces dada una ponderación de los elementos en E, existe un conjunto B que es óptimo en el sentido de Gale en F. Recíprocamente, si M = (E,F) es un sistema de subconjuntos que satisface esta condición entonces M es una matroide.

La prueba es semejante al del teorema 3.1 y no se reproduce. Los teoremas anteriores son un vehículo para demostrar que una base lexicográficamente, maximal es de peso máximo y es óptimo en el sentido de Gale.

Lo importante en la discusión anterior es que una base lexicográficamente maximal puede ser encontrada por el algoritmo glotón que se describe a continuación, cuya idea básica es: elegir los elementos de la matroide según su peso; primero el elemento más grande, rechazando un elemento solamente si su elección podría destruir la independencia del conjunto de elementos elegidos. El aspecto fundamental de este algoritmo es la prueba de la independencia para cada caso de aplicación. En el caso de que algunos elementos tengan peso negativo, el algoritmo los rechaza, es decir se aplica a una matroide obtenida por el aniquilamiento de los elementos negativos.

<u>Algoritmo</u>: Glotón

<u>Propósito</u>: Dado un conjunto $E = \{e_1, \ldots, e_n\}$ y una ponderación $W: E \to IR$ tal que $w(e_i) \geq 0$ se desea encontrar una base óptima de la matroide $M = (E, F, \beta)$

<u>DESCRIPCION</u>

Empezar

$I: = \phi$

Mientras $E \neq \phi$

Sea $e \in E$ que tenga mayor peso;

$E: = E - e$

Si $I + e \in F$. Entonces $I: = I + e$

FIN (Mientras)

FIN.

Aqui la pregunta de pertenecer a F es constructiva y determina la diferencia del algoritmo para cada aplicación concreta. La propiedad algorítmica de matroides es su íntima relación con el algoritmo Glotón, pues si primero logramos determinar que un problema de optimalidad combinatoria tiene una estructura de una matroide, y una función de ponderación entonces la aplicación del algoritmo glotón optimiza el problema de la manera más eficiente lo cual, por un lado hace interesante el estudio de la teoría matroidal con el Algoritmo Glotón, y por el otro da una equivalencia entre el algoritmo y las matroides. A continuación se establece un teorema de equivalencias.

<u>Teorema 3.3.</u> Sea $M = (E,F)$ un sistema de subconjuntos, entonces los siguientes postulados son equivalentes:

a. M es matroide

b. Si I_p, $I_{p+1} \in F$ con $|I_p| = p$ y $|I_{p+1}| = p + 1$ entonces existe $e \in I_{p+1} - I_p$ tal que $I_p \cup \{e\} \in F$ (propiedad de reemplazo).

c. Si A es un subconjunto de E, I e I' son subconjuntos maximales independientes de A entonces $|I| = |I'|$.

d. El algoritmo glotón resuelve cualquier caso de problemas de optimización combinatoria que se pueda representar por el sistema $M = (E,F)$.

Entre las aplicaciones del algoritmo glotón podemos mencionar:

Matroide de matriz.

En una matroide de matriz se desea encontrar un conjunto lexicográficamente máximo de columnas linealmente independientes, la prueba de independencia lineal, se puede llevar a cabo usando eliminación Gaussiana.

El árbol de expansión máxima.

En el problema de cableado (*), los elementos del cableado son elementos de una matroide gráfica, la prueba de independencia es equivalente a probar la existencia de un ciclo en un subconjunto de arcos.

El problema de semiacoplamiento.

Del ejemplo (*), los elementos de la matriz w son elementos de una matroide partición, donde los valores numérico son sus respectivas ponderaciones. Aquí los conjuntos independientes contienen a lo más un elemento de cada renglón.

Lo mismo ocurre en el problema de la digráfica, sus arcos son los elementos de una matroide partición y los conjuntos independientes son los conjuntos de arcos donde no hay dos que terminen en el mismo nodo.

*Problema del capítulo I.

El problema de secuenciación.

En el ejemplo (*) los trabajos para ser procesados son los elementos de una matroide transversal y sus ponderaciones son los valores de penalización. Esta matroide transversal tiene una estructura simple y la prueba de independencia es particularmente fácil.

* Un problema de secuenciación del capítulo 1.

3.2 Algoritmo glotón para matroides de matrices.

En esta sección se aplica el algoritmo glotón al problema de encontrar el conjunto maximal de columnas linealmente independientes y de peso máximo de una matriz A de orden mxn. Se describe con detalle el procedimiento y el seudocódigo que lo resuelve, un ejemplo ilustrativo y la discusión de su complejidad computacional.

El problema se describe en la siguiente forma:

Considere la matriz A de orden mxn donde A_i representa la i-ésima columna, y $w(a_i)$ denota su ponderación. Se desea obtener el conjunto maximal de columnas linealmente independientes cuya suma de ponderaciones tenga un valor máximo.

Algoritmo: Glotón para matrices.

Propósito: Este algoritmo va a ir formando para una matriz, con columnas ordenadas en forma decreciente según sus pesos, subconjuntos de columnas linealmente independientes en cada iteración por medio de eliminación gaussiana, hasta llegar al subconjunto maximal de columnas linealmente independientes y además de máxima ponderación.

Descripción

Paso 1. Eliminación

1.1 Si la columna k es cero ir a 1.2, de otro modo se escoge cualquier entrada diferente de cero en la columna, digamos a_{ik} y se usa para eliminar los elementos diferentes de cero que están a la derecha, es decir se le resta $\frac{a_{ij}}{a_{ik}}$, veces la columna k de cada columna j, j >k.

1.2 Si k < n hacer k = k+1 y regresar a 1.1 de otro modo parar.

FIN.

Justificación del algoritmo.

Sean $[\bar{a}_1, \bar{a}_2, \ldots, \bar{a}_n]$ las columnas de la matriz A, suponga que la columna $\bar{a}_1$ es diferente de cero, entonces se elige cualquier entrada diferente de cero para pivotear; por ejemplo a_{i1}. Ahora se desea hacer ceros a la derecha de a_{i1} es decir necesitamos que $a_{ij} - \beta_j \, a_{i1} = 0$ para $j \neq 1$ de donde $\beta_j = \dfrac{a_{ij}}{a_{i1}}$.

Entonces la primera iteración es:

$$\bar{a}_j^{(1)} = \bar{a}_j - \beta_j \; (\bar{a}_1) \qquad\qquad j=2,\ldots,n.$$

generando la siguiente matriz:

$$[\bar{a}_1, \bar{a}_2^{(1)}, \; \bar{a}_3^{(1)}, \ldots, \bar{a}_n^{(1)}]$$

con esto se ha construido los subconjuntos de cardinalidad dos de columnas linealmente independientes es decir $\{\bar{a}_1, a_j^{(1)}\}$ es linealmente independiente si $\bar{a}_j^{(1)} \neq 0$ y $a_{i2} \neq 0$ entonces

$$\bar{a}_j^{(2)} = \bar{a}_j^{(1)} - \beta_j (a_2^{(1)}); \; j=3,\ldots,n; \beta_j = \frac{a_{ij}}{a_{i2}}$$

se tiene

$$[\bar{a}_1, \; \bar{a}_2^{(1)}, \; \bar{a}_3^{(2)}, \ldots, \bar{a}_n^{(2)}]$$

donde $\{a_1, \; a_2^{(1)}, \; a_j^{(2)}\}$ son conjuntos linealmente independientes de cardinalidad 3 si $a_j^{(2)} \neq 0$.

Después de r pasos (r es el rango de la matriz) se tendrá excepto por el orden la matriz

$$\left[\bar{a}_1,\ \bar{a}_2^{(1)},\ \bar{a}_3^{(2)}, \ldots, \bar{a}_r^{(r-1)}\ \underbrace{\bar{0}, \bar{0}, \ldots, \bar{0},}_{n\,-\,r}\right]$$

cuyas columnas son el máximo conjunto linealmente independiente.

Es interesante observar que la forma en que se hace la eliminación no permite que se elija en una iteración p, el mismo nivel de la iteración p-1. Además aquí no se construye la factorización de una matriz en LU ni UL, pues no interesa resolver el sistema de ecuaciones asociado a la matriz, sino formar subconjuntos de columnas linealmente independientes que guarden su orden para obtener la base óptima en cuanto a sus pesos, Complejidad Algorítmica.

Este algoritmo converge a la solución en $O(mn^2)$.

Dem. En efecto, dado que como hemos visto sólo se hacen r iteraciones donde r es el rango de la matriz; y en cada iteración se efectuan m (n-k); K=1,...,r, flops (asignaciones, sumas y productos) entonces en total se tiene $3\sum_{k=1}^{r}(m(n-k))$ $= 3\left[mr(n-\frac{r+1}{2}\right]$.

Ahora bien si la matriz es de rango completo ie $r = n$ con $n \leq m$ entonces las operaciones a efectuar son :

$$3\left[m\,n\,(n-\frac{n+1}{2}\right] \leq mn^2.$$

<u>Algoritmo glotón en Seudo-código</u>

```
Empezar

Leer M.N.A(I,J)
K = 1
MIENTRAS K < N .  Hacer I = 1
   MIENTRAS I ≤ M.  Hacer
      SI A[IK] = 0, entonces
                  Empezar
                    AIFA [K] = I
                    PARA J = K+1,...,N  hacer
                         BETA [J] = A[ALFA[K],J]
                                    ------------
                                    A[ALFA[K] K]

                         PARA L = 1,...,M  Hacer
                            A[L,J] = A[L,J]-BETA [J]* A[L K]

                    K = K + 1
                 FIN
                    I = I + 1
      FIN (SI Y MIENTRAS I)
   K = K + 1
   FIN (MIENTRAS K)
   IMPRIMIR

FIN
```

Ejemplo 3.2.1 Encontrar un subconjunto maximal de peso máximo para la siguiente matriz:

$$A = \begin{bmatrix} 1 & 0 & 2 & 0 & 1 \\ 0 & -1 & -1 & 1 & 1 \\ 3 & 2 & 8 & 1 & 4 \\ 2 & 1 & 5 & 0 & 2 \end{bmatrix}$$

$$\begin{array}{llllll} \text{pesos} & 10 & 9 & 8 & 4 & 1 \end{array}$$

Iteración 1.

$K = 1$

$i = 1$

$a_{11}=1$, alfa $\boxed{1} = 1$

$j = 2$, Beta $\boxed{2} = \dfrac{a_{12}}{a_{11}} = 0$

$\ell = 1 \quad a_{12} = 0-0(1) = 0$

$\ell = 2 \quad a_{22} = -1-0(0) = -1$

$\ell = 3 \quad a_{32} = 2-0(3) = 2$

$\ell = 4 \quad a_{42} = 1-0(2) = 1$

$j = 3$, Beta $\boxed{3} = \dfrac{a_{13}}{a_{11}} = 2$

$\ell = 1 \quad a_{13} = 2-2(1) = 0$

$\ell = 2 \quad a_{23} = -1-2(0) = -1$

$\ell = 3 \quad a_{33} = 8-2(3) = 2$

$\ell = 4 \quad a_{43} = 5-2(2) = 1$

$J = 4$, Beta $\boxed{4} = \dfrac{a_{14}}{a_{11}} = 1$

$i = 1 \quad a_{14} = 0$

$i = 2 \quad a_{24} = 1$

$i = 3 \quad a_{34} = 1$

$i = 4 \quad a_{44} = 0$

$J = 5 \quad \text{Beta } \boxed{5} = \frac{a_{15}}{a_{i1}} = 1$

$i = 1 \quad a_{15} = 1-1(1) = 0$

$i = 2 \quad a_{25} = 1-1(0) = 1$

$i = 3 \quad a_{35} = 4-1(3) = 1$

$i = 4 \quad a_{45} = 2-1(2) = 0$

Al salir de la primera iteración se tiene la nueva matriz

$$A' = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & -1 & -1 & 1 & 1 \\ 3 & 2 & 2 & 1 & 1 \\ 2 & 1 & 1 & 0 & 0 \end{bmatrix}$$

donde cualquier columna $J(j>1)$ con la columna 1 es lin ind.

<u>Iteración 2</u>

$K = 2$

$i = 1 \qquad i = 2$

$a_{12} = 0; \qquad a_{22} \neq 0; \; a_{22} = -1 \text{ Alfa } \boxed{2} = 2$

$i = 2 \qquad J = 3 \quad \text{Beta } \boxed{3} = \frac{a_{21}}{a_{22}} = 1$

$i = 1 \quad a_{13} = 0 - (1)(\ 0) = 0$

$i = 2 \quad a_{23} = -1 - (1)(-1) = 0$

$i = 3 \quad a_{33} = 2 - (1)(\ 2) = 0$

$i = 4 \quad a_{43} = 1 - (1)\ (1) = 0$

$J = 4 \quad \text{Beta} \ \boxed{4} = \frac{a_{24}}{a_{22}} = -1$

$i = 1 \quad a_{14} = 0 - (-1)(\ 0) = 0$

$i = 2 \quad a_{24} = 1 - (-1)(-1) = 0$

$i = 3 \quad a_{34} = 1 - (-1)(\ 2) = 3$

$i = 4 \quad a_{44} = 0 - (-1)(\ 1) = 1$

$J = 5 \quad \text{Beta} \ \boxed{5} = \frac{a_{25}}{a_{22}} = -1$

$i = 1 \quad a_{15} = 0 - (-1)(\ 0) = 0$

$i = 2 \quad a_{25} = 1 - (-1)(-1) = 0$

$i = 3 \quad a_{35} = 1 - (-1)(-2) = 3$

$i = 4 \quad a_{45} = 0 - (-1)(\ 1) = 1$

Al salir de la segunda iteración se tiene la nueva matriz:

$$A^{(2)} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 \\ 3 & 2 & 0 & 3 & 3 \\ 2 & 1 & 0 & 1 & 1 \end{bmatrix}$$

donde las columnas $[\bar{a}_1, \bar{a}_2^{(1)}$ y $\bar{a}_J^{(2)}]$ para J = 4, 5 son linealmente independientes.

Iteración 3.

$K = 3$

$i = 2 \rightarrow i = 3 \rightarrow i = 4 \rightarrow i = 5$

$a_{23} \neq 0$?no $\quad a_{33} \neq 0$?no $\quad a_{43} \neq 0$?no $\quad$ Aquí el algoritmo no hizo ninguna operación.

$K = 4$

$i = 1 \rightarrow i = 2 \rightarrow i = 3$

$a_{14} \neq 0$?no $\quad a_{24} \neq 0$?no $\quad a_{34} = 3 \quad$ Alfa $\boxed{4} = 3$

$J = 5 \quad$ Beta $\boxed{5} = \frac{a_{35}}{a_{34}} = 1$

$$i = 1 \quad a_{15} = 0 - (1)(0) = 0$$

$$i = 2 \quad a_{25} = 0 - (1)(0) = 0$$

$$i = 3 \quad a_{35} = 3 - (1)(3) = 0$$

$$i = 4 \quad a_{45} = 1 - (1)(1) = 0$$

al salir de la tercera iteración se tiene:

$$A^{(3)} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 \\ 3 & 2 & 0 & 3 & 0 \\ 2 & 1 & 0 & 1 & 0 \end{bmatrix}$$

el algoritmo terminaría así:

$K = 5 \quad i = 3 \qquad \rightarrow i = 4 \qquad \rightarrow i = 5$

$\qquad a_{32} \neq 0 ? \text{no} \qquad a_{45} \neq 0 ? \text{no}$

$K = 6, \; K > 5 \quad$ FIN

Así las columnas $[\bar{a}_1, \bar{a}_2^{(1)}, \bar{a}_4^{(2)}]$ forman el conjunto maximal de columnas linealmente independientes cuyo peso máximo es:

$$W(a_1) + W(\bar{a}_2^{(1)}) + W(\bar{a}_4^{(2)}) = 10 + 9 + 4 = 23.$$

El programa en turbo Pascal del presente algoritmo se encuentra en el apéndice C.

3.3 El algoritmo glotón y el árbol de expansión mínima.

Uno de los problemas más frecuentes en Optimización Combinatoria es el de encontrar la red que comunique a una serie de nodos de manera óptima, tal es el caso del cableado telefónico, tubería de agua, de gas, etc. En esta sección se plantea el problema del árbol de expansión mínima que representa tal situación, su caracterización matroidal y los algoritmos glotones de solución.

Considere $G:(V,E)$ una gráfica simple y conectada, y una función de ponderación sobre los elementos de E; $\omega: E \to \mathbb{R}$ tal que $w(e_i) \geq 0$ el problema de encontrar el árbol de expansión de peso mínimo (AEMIN) consiste en determinar aquél árbol $T = (V,E')$ cuya $f(t)$ sea mínima o máxima, donde

$$f(t) = \sum_{e_i \in T} w(e_i)$$

La estructura matroidal que está detrás de la presente cuestión es la siguiente: el encontrar un árbol en una gráfica equivale a encontrar una base en una matroide gráfica con el mismo conjunto E de aristas. Debido a esto se sugiere la aplicación de algoritmos glotones apropiados, los cuales hacen uso del teorema 3.4 donde a partir de un nodo arbitrario (v_i) seleccionar el arco de menor ponderación que salga, por ejemplo el arco (v_i, v_j) para inicializar el árbol con $T = \{(v_i, v_j)\}$, y después ir añadiendo el arco de menor ponderación que salga de v_i o v_j pero no de los dos y así sucesivamente se forma el árbol.

Teorema 3.4 Sea $\{(U_1T_1), (U_2T_2), \ldots, (U_kT_k)\}$ un bosque de expansión de V y, sea $[v, u]$ el arco de menor ponderación de todos aquellos que sólo tienen un extremo final en U_1. Entonces entre todos los árboles de expansión que contienen todos los arcos en $T = \bigcup_{j=1}^{k} T_j$ hay uno que es óptimo, el que contiene a $[v,u]$.

Demostración. Suponga que existe un árbol de expansión (U,F) con $T \subseteq F$ y $[v,u] \notin F$ el cual tiene menor ponderación que todos los árboles de expansión que contienen a T y $[v,u]$. Si se añade el arco $[v,u]$ a F, se forma un único ciclo, que no contiene solamente nodos en U_1, porque $v \in U_1$ entonces existe un arco $[v',u'] \neq [v,u]$ en este ciclo con $u' \in U_1$ y $v' \in V-U_1$. Por hipótesis $[v',u'] \geq [v,u]$ y no pertenece a T. por tanto si quitamos a $[v',u']$, obtendremos un nuevo árbol de expansión $(V,F') = F' \cup F \cup \{[v,u]\} - \{[v',u']\}$, conteniendo a T y $[v,u]$ con menor ponderación o igual a (V,F), lo cual contradice la suposición de que (V,F) es el de menor ponderación que cualquier árbol que contenga a $[v,u]$ y T.

Algoritmo: Prim (glotón) AEMIN

Propósito: Dada una gráfica $G = (V, E)$ y las ponderaciones de los arcos, encontrar el árbol de expansión con peso mínimo.

Descripción

Paso 1. Iniciar la construcción del árbol de expansión mínima T incluyendo un nodo arbitrario para formar el conjunto de nodos de T en U.

Paso 2. Seleccionar el arco (v_i, v_j) con menor ponderación de entre los arcos (v_i, v_k) con v_i en T y v_k fuera de T y añadir (v_i, v_j) a T y v_j a U.

Paso 3. Cuando $U = V$ se tiene el árbol de expansión de peso mínimo.

Algoritmo glotón AEMIN en Seudocódigo.

Entrada: V: Un conjunto de vértices

d_{ij}: ponderaciones de los arcos $[v_i, v_j]$ de E;

si $[v_i, v_j] \notin E$, $d_{ij} = \infty$.

Salida: T: El conjunto de aristas que forman el árbol de peso mínimo.

```
Empezar:
    U: = {V1}; T: = ∅;
    Para toda v∈V - {V1} hacer: CERCA [V]: = V1;
    Mientras U ≠ V hacer:
       MIN: = ∞;
       Para toda v∈V - U hacer
            Si d(V1CERCA [V]) < MIN.  Entonces
                                        MIN: = d(V1 CERCA [V]);
                                       PROX: = V
            U: = U + [PROX];
            T: = T + {[PROX, CERCA [PROX]};
            Para toda v∈V - U hacer
                 Si d(V, CERCA [V]) > d[V,PROX].  Entonces
                 CERCA [V] = PROX;
    FIN MIENTRAS
FIN
```

Complejidad Algorítmica.

Este algoritmo resuelve correctamente el problema de árbol de expansión mínima en $O(n^2)$ tiempo donde $n = |V|$.

Demostración.

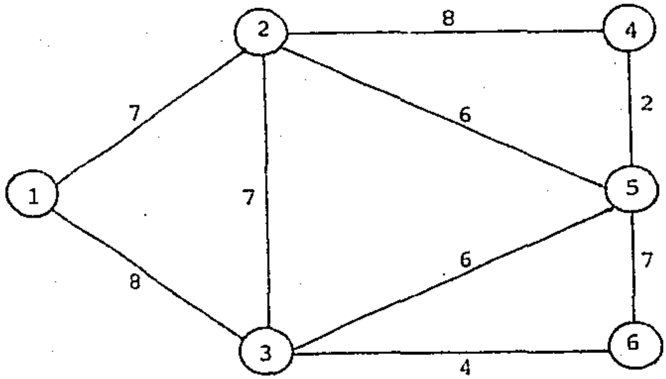
Debido a que al final del algoritmo $|U| = |V| = n$, y, a que siempre se le añaden a T arcos que salen de U, entonces la gráfica resultante (U,T) es un árbol de expansión de G. Para demostrar que éste es el óptimo, se prueba por inducción sobre la cardinalidad del conjunto U que existe siempre un árbol de expansión óptimo de G conteniendo el correspondiente conjunto T.

Para $U = \{V_1\}$ y $T = \phi$ es cierto. Se supone cierto para $j \varepsilon |U|$, $1 \leq j < |V|$ se puede observar que el árbol de expansión parcial (U,T) es parte de un bosque $\{(U_1 T_1),\ldots,(U_k,T_k)\}$ con $U_1 = U$, $K = |V| - |U| + 1$ y $T_2 = \ldots = T_k = \phi$. Por el teorema (3.4.), entre todos los árboles que contienen a T, existe uno que es de menor ponderación y que contiene a T y a la arista que sale de U, sin embargo por la hipótesis de inducción. Existe un árbol globalmente óptimo que contiene a T. Por tanto existe también un árbol óptimo que contiene a T y a la arista que sale de U, el cual es exactamente T en la siguiente iteración cuando $|U| = j + 1$.

Para el tiempo límite de operaciones notamos que este algoritmo hace n-1 iteraciones puesto que una arista es añadida en cada iteración. La inicialización hace n-1 asignaciones por tanto toma O(n) tiempo y, encontrar el próximo también requiere n-k asignaciones para la k-ésima iteración, asímismo las asignaciones para el arreglo CERCA son n-k para la k-ésima iteración, lo cual nos dan $\frac{n^2-n}{2}$ posibles asignaciones con lo que la cota del tiempo que se toma el algoritmo es de $O(n^2)$.

Ejemplo 3.3.1

Sea la siguiente gráfica conectada



$V = \{1, 2, 3, 4, 5, 6\}$

$d_{12}=7$, $d_{13}=8$, $d_{23}=7$, $d_{24}=8$, $d_{25}=6$, $d_{35}=6$, $d_{36}=4$, $d_{45}=2$, $d_{56}=7$.

$U = \{1\}$, $T: = \phi$ $V - \{1\} = \{2,3,4,5,6\}$

CERCA $\boxed{2}$ = 1, CERCA $\boxed{3}$ = 1, CERCA $\boxed{4}$ = 1, CERCA $\boxed{5}$ = 1

CERCA $\boxed{6}$ = 1,

MIN = ∞

D$\boxed{2,1}$ = 7, $7 < \infty \Rightarrow$ MIN: = 7; PROX: = 2

D$\boxed{3,1}$ = 8 $\quad 8 \nless 7$

D$\boxed{4,1}$ = ∞ $\quad \infty \nless 7$

D$\boxed{5,1}$ = ∞ $\quad \infty \nless 7$

D$\boxed{6,1}$ = ∞ $\quad \infty \nless 7$

$U:= \{1\} + \{2\} = \{1,2\}$

$T:= \phi + \{\boxed{2,1}\}$

¿D[3,1] > D[3,2]? 8 > 7 => CERCA [3] = 2

¿D[4,1] > D[4,2]? ∞ > 8 => CERCA [4] = 2

¿D[5,1] > D[5,2]? ∞ > 2 => CERCA [5] = 2

¿D[6,1] > D[6,2]? $\infty \not> \infty$.

2a. Iteración

MIN = ∞, V - U = {3, 4, 5, 6}

D[3,2] = 7, 7 < ∞ => MIN = 7, PROX: = 3

D[4,2] = 8 8 $\not<$ 7

D[5,2] = 6 6 < 7 => MIN = 6, PROX: = 5

D[6,2] = ∞ $\infty \not<$ 6

U = {1,2} + {5} = {1,2,5}

T = {[2,1], [5,2]}

¿D[3,2] > D[3,5]? 7 > 6 CERCA [3] = 5

¿D[4,2] > D[4,5]? 8 > 5 CERCA [4] = 5

¿D[5,2] > D[5,5]? 6 $\not>$ ∞

¿D[6,2] > D[6,5]? ∞ > 5 CERCA [6] = 5.

3a. Iteración.

MIN = ∞ V - U = {3,4,6}

D[3,5] = 6 6 < ∞ => MIN = 6; PROX: = 3

D[4,5] = 2 2 < 7 => MIN = 2 PROX: = 4

D[6,5] = 7 7 $\not<$ 2

U = {1,2,5} + {4} = {1,2,5,4}

T = {[2,1], [5,2]} + {[4,5]} = {[2,1], [5,2], [4,5]}

¿D[3,5] > D[3,4]? $6 \not> \infty$

¿D[4,5] > D[4,4]? $2 \not> \infty$

¿D[6,5] > D[6,4]? $7 \not> \infty$

4a. Iteración

$V - U = \{3,6\}$, $MIN = \infty$

D[3,5] = 6, $6 < \infty$ => MIN: = 6; PROX: = 3

D[6,5] = 7, $7 \not< 6$

$U = \{1,2,5,4\} + \{3\} = \{1,2,3,4,5\}$

T = { [2,1], [4,5], [3,5] }

¿D[3,5] > D[3,3]? $6 \not> \infty$

¿D[6,5] > D[6,3]? 7 > 4 => CERCA [6] = 3.

5a. Iteración

$V - U = \{6\}$, $MIN = \infty$

D[6,3] = 4; $4 < \infty$ => MIN = 4, PROX: = 6

$U = \{1,2,3,4,5,6\}$

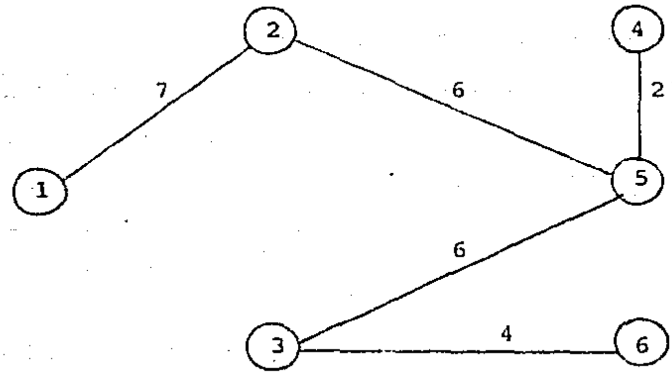
T = { [2,1], [5,2], [4,5], [3,5], [6,3] }

D[6,3] > D[6,6]

U = V

FIN.

El árbol de expansión mínima es:



cuyo costo es 25.

Un programa de computadora con el nombre de AEMIN asociado al presente algoritmo se encuentra en el apéndice C.

Un algoritmo para encontrar el árbol de expansión mínima en: $O(|E| \log |V|)$.

Algunos problemas plantean gráficas huecas, esto es que tienen cuando mucho $\binom{|V|}{2}$ aristas, esto genera matrices huecas de orden (nxn) de distancias, entonces el algoritmo anterior que tiene que revisar por lo menos una vez cada distancia para asegurar la entrada o exclusión de aristas al árbol, no es el mejor.

A continuación se presenta un algoritmo que resuelve este problema en $O(|E| \log |v|)$ tiempo, el cual es asintóticamente mejor que el anterior para gráficas huecas y, tamién se basa en el teorema 34 sólo que aquí en cada iteración se generan y crecen simultáneamente los componentes que formarán el árbol.

Algoritmo AEMIN 2 en seudocódigo.

Entrada: Una gráfica conectada G = (V,E) y para cada arista $[v_i, v_j]$ cE una distancia d_{ij}, $d_{ij}=\infty$ si $[v_i, v_j] \notin E$.

Salida: (V,T) Arbol de expansión de peso mínimo de G.

Empezar:

```
   T: = φ  C: = {{V1},...,{Un}}.
   (* C:se inicializa a C como el conjunto de componentes conec
    tados de (V,T)*)
   Mientras |C| ≠ 1 hacer:
      Para todo SiεC hacer MIN[i]: = ∞
      Para todo [u,v)εE hacer:
         Sea Si, Sj los conjuntos que contienen a v y u respec-
                                                     tivamente.
         Si i ≠ j.  Entonces:

         Si D[u,v] < MIN[i] Entonces   MIN[i] = D[u,v];
                                     CHICO[i] =  [u,v];
         Si D[u,v] < MIN[j] Entonces   MIN[J] = D[v,u];
                                     CHICO[J] =  [u,v],
         (Chico [j]: es el arco más chico que sale de Si).
      Para todo SjεC hacer T = TU{chico [j]}

      Encontrar el conjunto C de componentes conectados de
      (V,T).

      FIN Mientras
FIN.
```

Ejemplo. Dada la misma gráfica del ejemplo anterior y la matriz de distancias entre cada par de nodos:

$$\begin{array}{c c}
 & \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\
\begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} &
\left[\begin{array}{cccccc}
\infty & 7 & 8 & \infty & \infty & \infty \\
7 & \infty & 7 & 8 & 6 & \infty \\
8 & 7 & \infty & \infty & 2 & \infty \\
\infty & 8 & \infty & \infty & 2 & \infty \\
\infty & 6 & 6 & \infty & \infty & 7 \\
\infty & \infty & 4 & \infty & 7 & \infty
\end{array}\right]
\end{array}$$

E = {[1,2], [1,3], [2,3], [2,4], [2,5], [3,5], [3,6], [4,5], [5,6]}.

T: =ϕ C = {[1], [2,], [3], [3], [5], [6]}

S_1 = [1], S_2 = [2] ,..., S_6 = [6].

Iteración 1.

S_i εC; MIN [i] = ∞ i = 1, 2,...,6.

Para todo [u,v]εE| 1εS_1, 2εS_2.

¿D[1,2] < MIN [2]? 7 < ∞ => MIN [2] = 7 chico [2] = [1,2]

¿D[1,2] < MIN [1]? 7 < ∞ => MIN [1] = 7 chico [1] = [1,2]

¿D[1,3] < MIN [3]? 8 < ∞ => MIN [3] = 8 chico [3] = [1,3]

¿D[1,3] < MIN [1]? 8 $\nless$ 7

¿D[2,3] < MIN [3]? 7 < 8 => MIN [3] = 7 chico [3] = [2,3]

¿D[2,3] < MIN [2]? 7 $\nless$ 7

¿D[2,4] < MIN [4]? 8 < ∞ => MIN [4] = 8 chico [4] = [2,4]

¿D[2,4] < MIN [2]? 8 $\nless$ ∞

¿D[2,5] < MIN [5]? 6 < ∞ => MIN [5] = 6 chico [5] = [2,5]

¿D[2,5] < MIN [2]? 6 < 7 => MIN [2] = 6 chico [2] = [2,5]

¿D[3,5] < MIN [5]? 6 $\nless$ 6

¿D[3,5] < MIN [3]? 6 < 8 => MIN [3] = 6 chico [3] = [3,5]

¿D[3,6] < MIN [6]? 4 < ∞ => MIN [6] = 4 chico [6] = [3,6]

¿D[3,6] < MIN [3]? 4 < 6 => MIN [3] = 4 chico [3] = [3,6]

¿D[4,5] < MIN [5]? 2 < 6 => MIN [5] = 2 Chico [5] = [4,5]

¿D[4,5] < MIN [4]? 2 < 8 => MIN [4] = 2 chico [4] = [4,5]

¿D[5,6] < MIN [6]? 7 < 4

¿D[5,6] < MIN [5]? 7 < 2

[1] = S_1eC T = TU {chico [1]} = ϕ + {[1,2]}

[2] = S_2eC T = TU {chico [2]} = {[1,2]} + {[2,5]}

[3] = S_3eC T = TU {chico [3]} = {[1,2], [2,5]} + {3,6]}

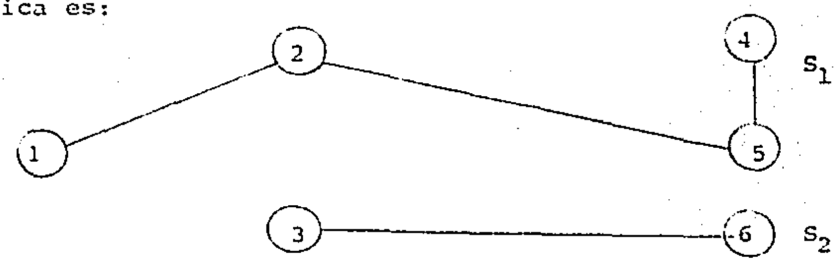
[4] = S_4eC T = TU {chico [4]} = {[1,2], [2,5], [3,6]} + {[4,5]}

Encontrar el conjunto C de componentes conectadas

$$C = \left\{\{1,\ 2,\ 4,\ 5\},\ \{3,6\}\right\}$$

$$S_1 = \{1,2,4,5\},\ S_2 = \{3,6\}$$

y la gráfica es:



Iteración 2.

$|C| \neq 1$

Para $S_1 \in C$ MIN [1] = ∞

$S_2 \in C$ MIN [2] = ∞

[1,2] $\in E$ $1 \in S_1$, $2 \in S_1$

[1,3] $\in E$ $1 \in S_1$ $3 \in S_2$

D[1,3] < MIN [1] 8 < ∞ => MIN [1] = 8 chico [1] = [1,3]

D[1,3] < MIN [2] 8 < ∞ => MIN [2] = 8 chico [2] = [1,3]

[2,3] $\in E$, $2 \in S_1$, $3 \in S_2$

D[2,3] < MIN [1]; 7 < 8 => MIN [1] = 7 chico [1] = [2,3]

D[2,3] < MIN [2]; 7 < 8 => MIN [2] = 7 chico [2] = [2,3]

[2,4], $2 \in S_1$, $4 \in S_1$

[2,5] $2 \in S_1$ $5 \in S_1$

[3,5] $3 \in S_2$ $5 \in S_1$

D[3,5] < MIN [1] 6 < 7 => MIN [1] = 6 chico [1] = [3,5]

D[3,5] < MIN [2] 6 < 7 > MIN [2] = 6 chico [2] = [3,5]

[3,6] $\in E$ $3 \in S_2$, $6 \in S_2$

[4,5] $\in E$ $4 \in S_1$, $5 \in S_1$

[5,6] $\in E$ $5 \in S_1$, $6 \in S_2$

D[5,6] < MIN [1]; 7 $\nless$ 6

D[5,6] < MIN [2]; 7 $\nless$ 6

$S_2 \in C$ T = {[1,2], [2,5], [3,6], [4,5]} + {[3,5]}

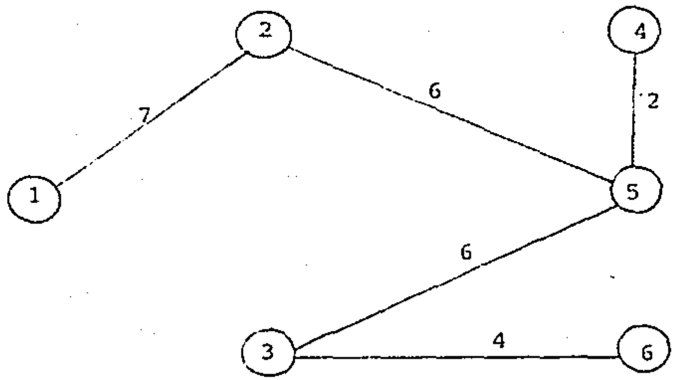
T = {[1,2], [2,5], [3,6], [4,5], [3,5]}

$$C = \left\{\{1,\ 2,\ 3,\ 4,\ 5,\ 6\}\right\}$$

$|C| = 1$

FIN

El árbol de expansión mínima que fué obtenido en 2 iteraciones es:



Teorema

Este árbol encuentra correctamente el árbol de expansión mínima para una gráfica (V,E) y una función de distancias en $O(|E| \log |V|)$ tiempo). Su demostración se puede ver en [8].

3.4 Variación del algoritmo glotón.

3.4.1 Dualidad en el algoritmo glotón.

En esta sección se verá una variante del glotón que permite obtener la solución de la matroide primal, y al mismo tiempo la de la matroide dual, es decir obtiene un conjunto maximal de peso máximo para el primal y otro de peso mínimo para el dual, basándose en las siguientes ideas: Suponga que los elementos de E tienen una ponderación y que A es una base lexicográficamente máxima. Entonces E - A es una cobase lexicográficamente mínima, es claro que al resolver un problema de maximización para la matroide primal también se resuelve el problema de minimización para su dual, así un algoritmo glotón para el primal corresponde a uno de abstención para el dual y viceversa.

Cada elemento que es descartado por el glotón es el elemento más pequeño de al menos un circuito de la matroide.

Suponga que $A = e_1, e_2, \ldots, e_k$ han sido elegidos por el glotón, pero $A + e_{k+1}$ es dependiente, entonces e_{k+1} es rechazado, dado que A es independiente y $A+e_{k+1}$ es dependiente se sigue que e_{k+1} forma un circuito con algún subconjunto de A cuyos elementos ya se sabe que son mayores que e_{k+1}.

Similarmente cada elemento que es descartado por el algoritmo de abstensión aplicado a la matroide dual; será el elemento más pesado de al menos un cocircuito de la matroide primal. Estas relaciones duales han hecho que autores como Rosenstiehl hagan las siguientes observaciones acerca de los cálculos del árbol de expansión máxima:

Todos los algoritmos para calcular el árbol de expansión máxima de una gráfica cumplen las siguientes condiciones:

1. Ningún arco del árbol de expansión máxima es el arco más pequeño de algún ciclo de la gráfica.

2. Cada arco del árbol de expansión máxima es el más grande de al menos un cociclo de la gráfica.

Basados en estos principios se fórmula una variante del procedimiento glotón para matroides en general, que requiere solamente de la construcción de circuitos y cocircuitos de la matroide.

A continuación se da dos versiones del algoritmo.

3.4.2 Variante del algoritmo glotón versión 1.

Paso (0) Empezar:

Sea A = conjunto de elementos del conjunto óptimo

A^1 = complemento de A

B = conjunto de elementos que no han sido elegidos para ninguno de los dos conjuntos.

Paso 1. Selección de elementos

1.0 Mientras $B \neq \phi$

1.1 Trate de encontrar un circuito de elementos en $A \cup B$. Si un circuito C existe, mueva el elemento más pequeño de C de B a A'.

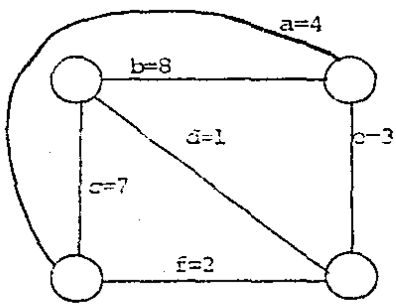
Si no existe un circuito, mueva los elementos que queden en B a A y pare.

1.2 Trate de encontrar un circuito de elementos $A' \cup B$. Si un cocircuito D existe, mueva el elemento más grande de D de B a A.

Si no existe un cocircuito mueva los elementos que quedan en B a A' y pare.

Antes de ver la otra versión, se ilustrará el algoritmo con un ejemplo sencillo.

Ejemplo. Considere la gráfica



cuya matroide asociada es: $M = (E, C, D)$, donde:

$E = \{a, b, c, d, e, f\}$

$C = \{(a,bc), (afe), (bde)(dfc)(bcef)\}$

$D = \{(def), (bcd), (acf), (ade)\}$.

Ahora siguiendo el algoritmo se encontrará el conjunto óptimo tanto para la matroide primal como para su dual.

$A = A' = \phi$; $B = \{a, b, c, d, e, f\}$

1.1 Sea $C = \{a\ b\ c\}$ en $A \cup B = \{a\ b\ c\ d\ e\ f\}$

$a \varepsilon C$ es el más pequeño $\Rightarrow A' = \{a\}$ $B = \{b\ c\ d\ e\ f\}$

1.2 Sea $D = \{c\ d\ b\}$ en $A' \cup B = \{a\ b\ c\ d\ e\ f\}$

$c \varepsilon D$ es el más grande $\Rightarrow A = \{c\}$ $B = \{b\ d\ e\ f\}$

1.1 Sea $A \cup B = \{b\ c\ d\ e\ f\}$ y $C = \{b\ d\ e\}$;

$d \varepsilon C$ es el más pequeño $\Rightarrow A' = \{a, d\}$, $B = \{b\ e\ f\}$.

1.2 Sea $A' \cup B = \{a\ b\ d\ e\ f\}$; $D = \{f\ e\ d\}$

$e \varepsilon D$ es el más grande $\Rightarrow A = \{c\ e\}$, $B = \{b\ f\}$

1.1 $A \cup B = \{b\ c\ e\ f\}$; $C = \{b\ c\ e\ f\}$;

$f \varepsilon C$ es el más pequeño $\Rightarrow A' = \{a\ d\ f\} \Rightarrow B = \{b\}$

1.2 $A' \cup B = \{a\ b\ d\ f\} = D$;

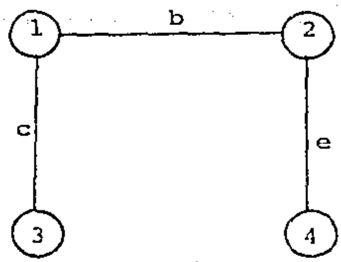
$b \varepsilon D$ es el mayor $\Rightarrow A = \{c\ b\ e\} \Rightarrow B = \phi$

1.1 $A \cup B = \{c\ b\ e\}$ no existe circuito y

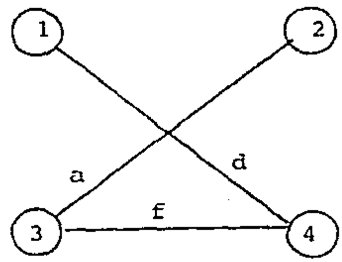
$A = \{c, b\ e\} \cup \{\phi\} = \{c\ b\ e\}$.

FIN

Y la solución óptima para la matroide primal es: $A = \{b\ c\ e\}$ y para la matroide dual es: $A' = \{a\ d\ f\}$, cuyos pesos máximo y mínimo respectivamente son 20 y 7. Y cuyas gráficas son:



A = Conjunto independiente de peso máximo.



A' Conjunto independiente de peso mínimo.

Este algoritmo se puede presentar con las operaciones de borrado y contracción de matroides para los pasos 1.1 y 1.2, a continuación se da la versión 2 de la variante del algoritmo glotón y se ilustra con el mismo ejemplo para mostrar su equivalencia.

3.4.3 Variante del algoritmo glotón versión 2.

0. Empezar

   A = Conjunto de elementos del conjunto óptimo

   A' = Complemento de A

   B = Conjunto de elementos que no han sido elegidos para ninguno de los conjuntos A ó A'.

1. Selección de elementos

   1.1 Trate de encontrar un circuito de elementos de B. Si existe un circuito en C, mueva el elementos más pequeño de C de B a A' y borre el elemento de la matroide. Sin no existe, mueva los elementos que queden de B a A y pare.

   1.2 Trate de encontrar un cocircuito de elementos de B. Si existe un cocircuito D mueva el elemento más grande de D a B a A y contraiga los elementos de la matroide. Si no existe un cocircuito, mueva los elementos que queden de B a A' y pare.

Con el mismo ejemplo anterior.

$M = (E,C;D)$; $E = \{a,b,c,d,e,f\}$; $C = \{(a,b,c), (a,f,e), (b,d,e), (b,c,e,f)\}$; $D = \{(d,e,f), (b,c,d), (a,c,f), (a,d,e)\}$

1.1 Sea $C = (a,b,c)$; a es el menor $\Rightarrow B = \{b,c,d,e,f\}$; $A'=\{a\}$
$M^{(1)} = M - \{a\}$: $E\{b,c,d,e,f,\}$, $C = \{(b,d,e), (c,d,f)\}$;
$D = \{(b,d,c), (e,d,f)\}$

1.2 Sea $D = (b,d,c)$, b es el mayor $\Rightarrow B = \{c,d,e,f\}$; $A = \{b\}$
$M^{(2)} = M^{(1)} . \{b\}$: $E = \{c,d,e,f\}$ $C = \{(c,d,f)\ (c,d,e)\}$

1.1 Sea $C = (c,d,f)$ d es el menor $\Rightarrow B = \{c,e,f\}$; $A' = \{a,d\}$
$M^{(3)} = M^{(2)} - \{d\}$: $E = \{c,e,f\}$, $C = \{c,d,f\}$ $D = \{(c,e), (e,f)\}$.

1.2 Sea $D = (e,f)$, e es el mayor $\Rightarrow B = \{c,f\}$, $A = \{b,e\}$
$M^{(4)} = M^{(3)} . \{e\}$: $E = B$; $C = (c,f)$.

1.1 Sea $C = (c,f)$ f es el menor $\Rightarrow B = \{c\}$, $A' = \{a,d,f\}$
$M^{(5)} = M^{(4)} - \{f\}$; $E = \{c\}$; $C = \phi$ y $D = \{c\}$.

1.2 Sea $D = \{c\} \Rightarrow B = \phi$ y $A = \{b,e,d\}$.

Como $B = \phi$ todos los elementos han sido elegidos y el proceso termina con el resultado:

$$A = \{b,e,d\} \quad y \quad A' = \{a,d,f\}. \blacksquare$$

CAPITULO 4

ACOPLAMIENTO Y TEORÍA TRANSVERSAL

Un problema básico de la Optimización Combinatoria es el problema de acoplamiento que consiste en encontrar el número de parejas o acoplamientos que existen en un par de conjuntos ajenos y cuya aplicación inmediata se encuentra en problemas de asignación de recursos. Esta problemática se enfoca desde dos puntos de vista equivalente; con teoría de gráficas y con teoría transversal, las cuales encuentran su generalización en teoría matroidal. En general este tipo de problemas plantean la intersección de matroides y en consecuencia el algoritmo glotón no funciona. Aunque cabe mencionar que existe al menos un problema de transversal que contempla una sóla matroide, el de la asignación de trabajos en una sola máquina y por tanto es factible resolverlo por el algoritmo glotón.

Este capítulo se desarrolla de la siguiente manera: En la primera sección se plantea el problema de acoplamiento completo en una gráfica bipartita, el teorema de Philip Hall y se describen ejemplos. En la segunda sección se discute la teoría transversal y el teorema de Hall para transversales. En la tercera sección se discuten las transversales parciales. En la cuarta sección se presentan dos aplicaciones; cuadros latinos y matrices (0,1). En la quinta sección se presentan las matroides transversales y el glotón para una matroide transversal, y se ejemplifica con el problema de la secuenciación en la asignación de trabajo, en la última sección se definen las matroides y transversales comunes y la intersección de matroides.

4.1 El problema de acoplamiento completo en gráficas bipartitas.

Sea $G = (V_1, V_2)$ una gráfica bipartita donde V_1 y V_2 son conjuntos de vértices disjuntos, A es un conjunto de arista' que une a un nodo de V_1 con un nodo de V_2. Un subconjunto $X \subseteq A$ es un acoplamiento si no hay dos arcos en X que incidan en el mismo nodo. Con respecto a un acoplamiento X, un nodo j es acoplado o cubierto si existe un arco que incide en j, si un nodo no es acoplado, se dice no acoplado o expuesto.

Un acoplamiento completo o máximo entre V_1 y V_2 en una gráfica bipartita $G = (v_1, v_2)$ es una correspondencia biunívoca entre los vértices de V_1 y un subconjunto de los vértices de V_2 con la propiedad de que los vértices correspondientes queden unidos o bien, dicho de otro modo: dada una gráfica bipartita encontrar un acoplamiento que contenga un número máximo de arcos.

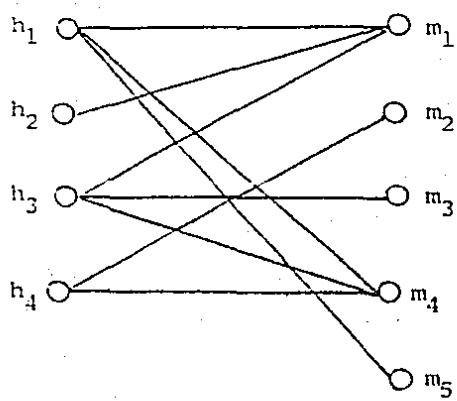
Uno de los problemas fundamentales que se plantean en una gráfica bipartita es: ¿En qué condiciones existe un acoplamiento máximo?, la respuesta a esto está dada por el teorema de P. Hall. Motivamos esta problemática a través del problema matrimonial.

Considere un conjunto finito de jóvenes, cada uno de los cuales conoce a varias señoritas. ¿En qué condiciones es posible casar a los jóvenes de forma que cada uno se case con una señorita que conoce?.

Considere el caso específico de cuatro jóvenes $\{h_1, h_2, h_3, h_4\}$ y cinco señoritas $\{m_1, m_2, m_3, m_4, m_5\}$ y que la relación entre ellos es como la que se muestra en la siguiente tabla, donde c_{ij} representa que el joven h_i conoce a la señorita m_j

	m_1	m_2	m_3	m_4	m_5
h_1	c_{11}	-	-	c_{14}	c_{15}
h_2	c_{21}				
h_3	c_{31}		c_{33}	c_{34}	
h_4		c_{42}		c_{44}	

La representación gráfica de la problemática es:



Una posible solución sería:

$$\{(h_1, m_5), (h_2, m_1), (h_3, m_3), (h_4, m_2)\}$$

es decir es un acoplamiento completo o máximo acoplamiento.

De aquí se desprende que una condición necesaria, para que el problema matrimonial tenga solución es que cada k jóvenes conozca colectivamente al menos k señoritas para todos los enteros k, $1 \leq k \leq n$ donde n es número de muchachos.

El que ésta sea una condición necesaria se desprende directamente del hecho que si no fuera cierta para un conjunto k de jóvenes, entonces no se podrían casar a todos los jóvenes de ese conjunto ni a los restantes.

Esta condición necesaria es suficiente, esto último queda demostrado por el teorema de Hall, demostrado en 1935.

Teorema 4.1.

(Philip Hall, 1935). Una condición necesaria y suficiente para la solución del problema matrimonial es que cada conjunto de k jóvenes conozca colectivamente al menos a k chicas $1 \leq k \leq n$.

Demostración. La condición es obviamente necesaria y para demostrar la suficiencia procederemos por inducción sobre n (el número de jóvenes).

El teorema es evidentemente cierto si $n = 1$. Suponga que hay n jovenes, entonces hay dos casos a considerar:

i) Supóngase que cada k jóvenes $(k < n)$ conoce colectivamente al menos a $k+1$ señoritas (de forma que la condición es siempre cierta con una señorita de sobra). Entonces tomando un joven cualquiera y casándolo con cualquiera de las señoritas que conozca, la condición original sigue siendo cierta para los otros m-1 jovenes. Estos m-1 jovenes pueden ser casados por inducción, comletándose así la demostración en este caso.

ii) Supóngase ahora que existe un conjunto de k jovenes $(k < m)$ que conocen colectivamente a k señoritas exactamente. Entonces estos k jovenes pueden ser casados por inducción quedando aún m-k jovenes. Pero cualquier colección de h de estos m-k jovenes $(h \leq m - k)$ debe conocer al menos h de las señoritas restantes, de otro modo estos h jovenes más los k jovenes conocerían colectivamente menos de $h + K$ señoritas en contra de lo supuesto. Por tanto la condición original se cumple para los $m - K$ jóvenes.

En términos de gráficas el enunciado del teorema se establece de la siguiente forma: Sea $G = (V_1, A, V_2)$ una gráfica bipartita y sea $\phi(X)$, $(X \subset V_1)$, el conjunto de aquellos vértices de V_2 que son adyacentes a un vértice al menos de X; entonces existe un acoplamiento completo entre V_1 y V_2 si y sólo si $|X| \leq |\phi(X)|$ para cada conjunto X de V.

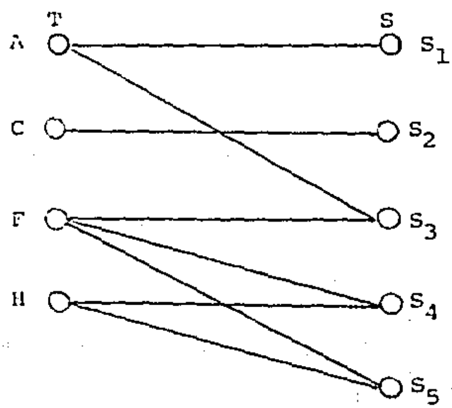
Corolario 4.1. Si cada joven tiene $(r \geq 1)$ novias, y cada señorita tiene r novios, entonces el problema matrimonial tiene una solución.

En términos de gráficas el corolario es como sigue:

Corolario 4.2. Si G es una gráfica bipartita regular (de grado r) con $r > 0$ entonces G tiene un acoplamiento perfecto.

Ejemplo 1. Un contratista publica un anuncio en el que ofrece cuatro puestos de trabajo: uno para albañil, otro para un carpintero, un tercero para un fontanero y finalmente otro para un herrero. Poco después recibe cinco solicitudes, una para el trabajo de carpintero, otra para el trabajo de albañil, una tercera para los trabajos de albañil y fontanero, y dos para los trabajos de fontanero y herrero. ¿Pueden quedar cubiertos los cuatro trabajos?.

Es posible representar la situación antes descrita a través de la siguiente gráfica bipartita:



donde:

$$T = \{\text{Albañil, carpintero, fontanero, herrero}\}$$

$$S = \{s_1, s_2, s_3, s_4, s_5\}$$

Evidentemente los cuatro trabajos quedan cubiertos pues es cumplen las condiciones de Hall.

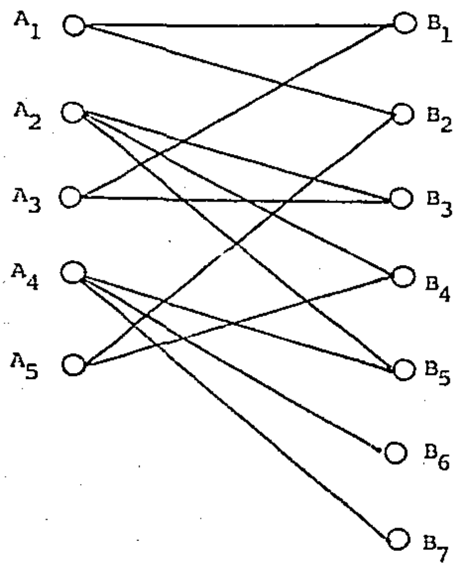
Ejemplo 2. El departamento de Economía de una universidad cuenta con cinco profesores para cubrir siete cursos en un semestre dado. Los cursos requeridos y las preferencias de cada profesor se muestran en la siguiente tabla:

curso profesor	B_1	B_2	B_3	B_4	B_5	B_6	B_7
A_1	si	si	no	no	no	no	no
A_2	no	no	si	si	si	no	no
A_3	si	no	si	no	no	no	no
A_4	no	no	no	no	si	si	si
A_5	no	si	no	si	no	si	no

Donde los cursos son: B_1 = Historia; B_2 = Economia Política; B_3 = Microeconomía; B_4 = Macroeconomía; B_5 Econometría; B_6 = Matemáticas y B_7 = Computación.

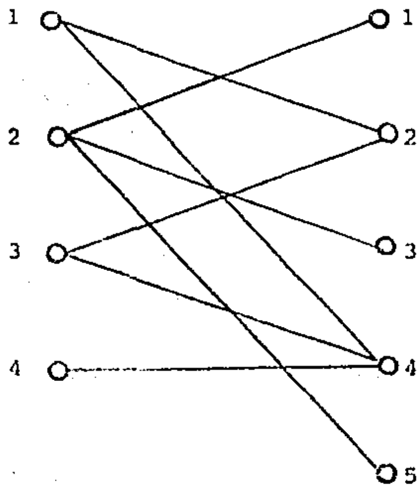
El problema consiste en determinar el número máximo de cursos que se pueden atender suponiendo que cada profesor sólo pueda impartir un curso.

La gráfica asociada es:



La respuesta es que dado que se cumplen las condiciones de Hall para que exista un acoplamiento completo se sigue que se pueden atender 5 cursos.

Ejemplo 3. En la siguiente gráfica bipartita se puede comprobar que no existe acoplamiento:



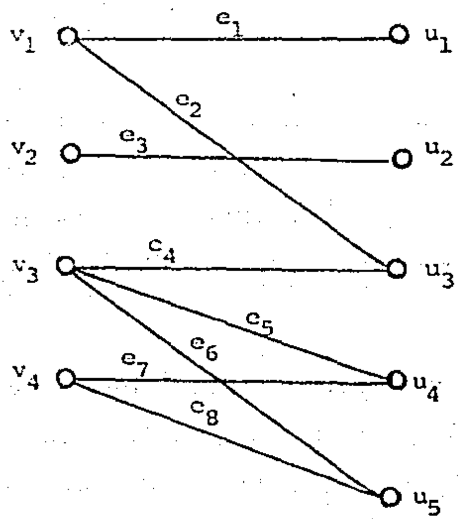
Obsérvese que cuando se toman los nodos 1, 3, y 4 de V_1, sólo se relacionan con los nodos 2, 4 de V_2 es decir: $X = \{1, 3, 4\}$, $\phi(X) = \{2,4\}$ y $|X| \not\leq |\phi(X)|$. Por tanto no se cumplen las condiciones del teorema de Hall implicando la ausencia de acoplamiento máximo.

Para finalizar esta sección diremos que el enfoque matroidal del problema de acoplamiento bipartita, involucra la intersección de dos matroides y el cual queda formulado de la siguiente manera:

Sea $G = (V_1, A, V_2)$ una gráfica bipartita. Sea Π_1 una partición de A donde dos aristas estaran en un bloque de tal partición si y sólo si ellas son incidentes en un mismo nodo de V_1. Análogamente sea Π_2 definida para los nodos de V_2. Sea $M_1 = (A_1\ F_1)$ y $M_2 = (A_2\ F_2)$ matroides partición determinados por la partición Π_1 y Π_2. Un subconjunto $I \subseteq A$ es un acoplamiento en G si y sólo si I es una intersección de M_1 y M_2 o bien $I \in F_1 \cap F_2$.

Ejemplo, el problema del contratista toma su caracter matroidal de la siguiente forma:

Sea $G(V_1, A, V_2)$ la gráfica bipartita:



donde:

$$V_1 = \{v_1\ v_2\ v_3\ v_4\},\quad V_2 = \{u_1, u_2, u_3, u_4, u_5\}$$

$$A = \{e_1, e_2, e_3\ e_4, e_5, e_6, e_7, e_8\}$$

considere las siguientes particiones

$$\Pi_1 = \{\{e_1, e_2\}, \{e_3\}, \{e_4, e_5, e_6\}, \{e_7, e_8\}\}$$

$$\Pi_2 = \{\{e_1\}, \{e_3\}, \{e_2, e_4\}, \{e_5, e_7\}, \{e_6, e_8\}\}$$

Entonces $M_1 = (A, F_1)$ y $M_2 = (A, F_2)$ son las matroides particiones determinados por Π_1 y Π_2 respectivamente, un acoplamiento de G será:

$$I = \{e_1, e_3, e_4, e_7\} \text{ pues } I \subset F_1 \cap F_2.$$

4.2 Teoria transversal.

Considere una familia de subconjuntos no necesariamente diferentes $U = (A_1, \ldots, A_n)$ de un conjunto dado E. Un subconjunto $X = \{x_1, \ldots, x_n\}$ $(x_i \neq x_j)$ de E tal que $x_j \varepsilon A_j$ para cada j $(j = 1, \ldots, n)$ es llamado un transversal de U.

Un transversal consiste de elementos distintos donde cada uno representa a un subconjunto de la familia U, por lo que se le denomina "sistema de representantes distintos" "SRD".

Observe que el problema de la determinación de un acoplamiento completo en gráficas bipartitas se puede plantear como un problema de determinar un transversal de un conjunto dado. Específicamente en el ejemplo del problema del matrimonio de la sección anterior tenemos que si hacemos:

$$E = \{m_1, m_2, m_3, m_4, m_5\}; \quad U = \{h_1, h_2, h_3, h_4\}$$

$$h_1 = \{m_1, m_4, m_5\},\ h_2 = \{m_1\},\ h_3 = \{m_2, m_3, m_4\},\ h_4\{m_2, m_4\}$$

Los posibles transversales son:

$$T_1 = \{m_5, m_1, m_2, m_4\};\ m_5\varepsilon h_1,\ m_1\varepsilon h_2,\ m_2\varepsilon h_3,\ m_4\varepsilon h_4$$

$$T_2 = \{m_4, m_1, m_3, m_2\};\ m_4\varepsilon h_1,\ m_1\varepsilon h_2,\ m_2\varepsilon h_3,\ m_2\varepsilon h_4$$

$$T_3 = \{m_5, m_1, m_3, m_4\};\ m_5\varepsilon h_1,\ m_1\varepsilon h_2,\ m_3\varepsilon h_3,\ m_4\varepsilon h_4$$

que corresponde a tres sistemas de representantes distintos o tres acoplamientos completos distintos.

El determinar los transversales para $U = (A_1, \ldots, A_n)$; corresponde a encontrar acoplamientos completos o máximos de la gráfica bipartita $G = (J, \hat{\Phi}, E)$ donde
$J = \{1, \ldots, n\}$, $E \cap J = \phi$ y $\hat{\Phi}$ es definida como

$$\begin{aligned} \hat{\Phi} &= \{je: e \in A_j\} \\ \hat{\Phi}(J') &= \{e: je \in \hat{\Phi},\ j \in J\} \\ &= \{e, e \in A_j \text{ para algun } j \in J\} \\ &= \bigcup_{j \in J'} A_j \quad J' \subseteq J. \end{aligned}$$

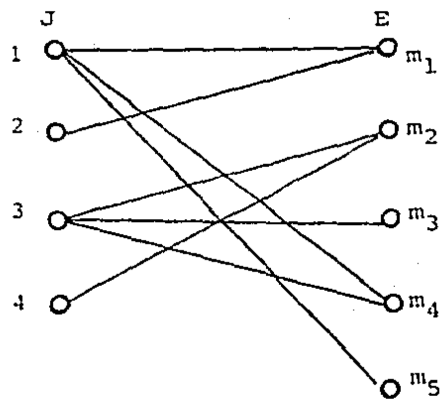
En el ejemplo anterior:

$U = \{h_1, h_2, h_3, h_4\}$; $e = \{m_1, m_2, m_3, m_4, m_5\}$ $J = \{1,2,3,4\}$

$h_1 = \{m_1, m_4, m_5\}$, $h_2 = \{m_1\}$, $h_3 = \{m_2, m_3, m_4\}$, $h_4 = \{m_2, m_4\}$

$\hat{\Phi} = \{1m_1, 1m_4, 1m_5, 2m_1, 3m_2, 3m_3, 3m_4, 4m_2, 4m_4\}$

cuya gráfica bipartita $G = (J, \hat{\Phi}, E)$



y un transversal $T_1 = \{5m_1, 1m_2, 3m_3, 4m_4\}$, note que con esta notación es más explícito los acoplamientos que se forman.

La condición necesaria y suficiente para que $\mathcal{U}$ = tenga una transversal $X = \{x_1, \ldots, x_n\}$ $(x_i \neq x_j)$ con $x_j \in A_j$ para cada j es:

$$\left| \bigcup_{j \in J'} A_j \right| \geq |J'| \text{ para toda } J' \subseteq \{1, \ldots, n\}$$

o equivalentemente

$$|\S(J)| \geq |J'| \text{ para toda } J' \subseteq J.$$

Lo cual queda demostrado en el siguiente teorema.

<u>Teorema 4.2.</u> (Philip, Hall para transversales). La familia $U = (A_1,\ldots,A_n)$ de subconjuntos de E posee una transversal si y sólo si:

$$\left|\bigcup_{j\in J'} A_j\right| \geq |J'| \text{ para toda } J' \subseteq \{1,\ldots,n\}$$

<u>Demostración</u>. La condición de necesidad es trivial. Para la suficiencia, suponga que $|\hat{\wp}(J')| \geq (J')$ para cada $J' \subseteq \{1,\ldots,n\}$. En particular $|A_1| = |\hat{\wp}(1)| \geq |\{1\}| = 1$ y $A_1 \neq 0$. Por lo tanto el subconjunto A_1 de U tiene una transversal, haciéndolo por inducción sobre m se demuestra que $(A_1,\ldots,A_m)$ tiene una transversal para $1 \leq m \leq n$. Suponga que para $1 \leq m \leq n$, el resultado es cierto y sin pérdida de generalidad, escribimos: $E = \{x_1,\ldots,x_m\}$ $(x_i \neq x_j)$ donde $x_1 \varepsilon A_1,\ldots, x_m \varepsilon A_m$, ahora queremos encontrar un transversal de $(A_1,\ldots, A_{m+1})$.

Si $A_{m+1} \not\subseteq \{x_1,\ldots,x_m\}$ entonces es claro que existe un elemento de A_{m+1} distinto de $x_1,\ldots,x_m$ el cual completará una transversal de $(A_1, A_2,\ldots,A_{m+1})$ pero si $A_{m+1} \subseteq \{x_1,\ldots,x_m\}$, entonces hacemos el siguiente proceso:

1. Sea $E_0 = A_{m+1}$

2. Dado $E_t \subseteq \{x_1,\ldots,x_m\}$ denote por $J_t (\subseteq J)$ el conjunto de suficiencia de las x's en E_t y forme el conjunto $E_{t+1} = \hat{\wp}(J_t)$.

3. O bien $E_{t+1} \subseteq \{x_1,\ldots,x_m\}$ en cuyo caso aplicamos 2 con $t + 1$ en lugar de t, o $E_{t+1} \nsubseteq \{x_1,\ldots,x_m\}$ en cuyo caso se detiene el proceso.

Observe que, dado $x_i \varepsilon A_i$ para cada $j = \{1,\ldots,m\}$, se sigue que $E_t \subseteq \cup_{j \varepsilon J_t} A_j = \hat{\phi}(J_t) = E_{t+1}$ para cada t hasta que el proceso se detiene. Resumiendo, dado $A_{m+1} = E_0 \subseteq E_1 \subseteq, \ldots, \subseteq E_{t+1}$ se sigue que

$$|E_{t+1}| = |\hat{\phi}\ (J_t) \cup A_{m+1}|$$

$$= |\hat{\phi}\ (J_t \cup \{m+1\}| \geq |J_t \cup \{m+1\}| > |J_t| = |E_t|.$$

Así $E_t \subset E_{t+1}$, por tanto $E_0 \subset E_1 \subset E_2, \ldots, \subseteq \{x_1,\ldots,x_m\}$ y se sigue que el proceso debe terminar en: $E_0,\ldots,E_{t+1}$ donde $E_t = \hat{\phi}(J_{t-1}) \subseteq \{x_1,\ldots,x_m\}$ y $E_{t+1} = \hat{\phi}\ (J_t) \nsubseteq \{x_1,\ldots,x_n\}$.

Dado $b \varepsilon \hat{\phi}(J_t) - \{x_1,\ldots,x_m\}$ y $b\ \varepsilon A_{jt}$, para algún $b \varepsilon J_t - J_{t-1}$. Use a b para representar al conjunto A_{jt} en la transversal que se va a construir.

El siguiente $x_{jt} \varepsilon E_t - E_{t-1} = \hat{\phi}(J_{t-1}) - \hat{\phi}(J_{t-2})$ y $x_{jt} \varepsilon A_{jt-1}$ para algún $jt-1 \varepsilon J_{t-1} - J_{t-2}$. Use a: x_{jt} para representar a A_{it-1}, x_{jt-1} para representar a A_{it-2}. etc.

El siguiente $x_{j2} \varepsilon E_2 - E_1 = \hat{\phi}(J_1) - \hat{\phi}(J_0)$ y $x_{j2} \varepsilon A_{j1}$ para algun $j_1 \varepsilon J_1 - J_0$; use a x_{j2} para representar a A_{j1}. En la última iteración $x_{j1} \varepsilon E_1 - E_0 = \hat{\phi}(J_0) - E_0$ y $x_{j1}\ \varepsilon\ A_{i0}$ para algún $j_0 \varepsilon J_0$ use x_{j1} para representar a A_{j0} y x_{j0} para representar a

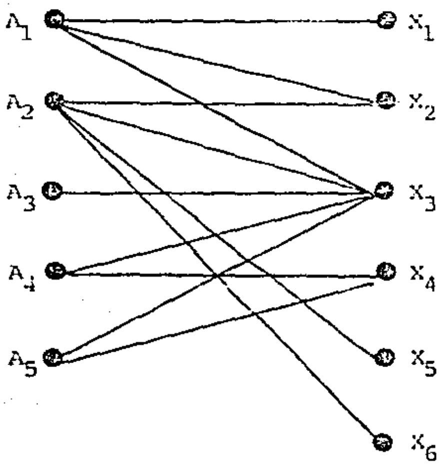
$A_{m+1} = E_o$.

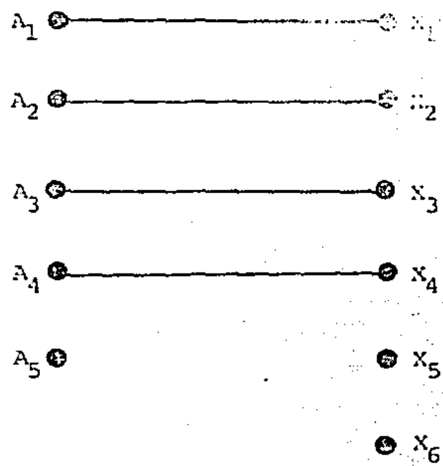
Finalmente observemos que dado $x_j \varepsilon A_j$ para $1 \leq j \leq m$ el conjunto x_j: $j\varepsilon(\{1,\ldots,m\} - \{j_0,\ldots,j_t\})$ junto con $x_{j0}\ \varepsilon A_{m+1}$, $x_{j1}\ \varepsilon A_{i0},\ldots,x_{jt}\ \varepsilon A_{jt-1}$ y b εA_{jt} nos da una transversal $\{x_1, x_2,\ldots,x_m,b\}$ de elementos diferentes de $(A_1,\ldots,A_{m+1})$. Por tanto se ha exibido una transversal y se sigue por inducción que $(A_1,\ldots,A_n)$ tiene una transversal.

<u>Ejemplo 4.1</u>: Sea $U = (A_1, A_2, A_3, A_4, A_5)$ una familia de subconjuntos de $\{x_1, x_2, x_3, x_4, x_5, x_6\}$ $(x_i \neq x_j)$ donde $A_1 = \{x_1, x_2, x_3\}$ $A_2 = \{x_2, x_3, x_5, x_6\}$, $A_3 = \{x_1, x_3\}$ $A_4 = A_5 = \{x_3, x_4\}$. Suponga que:

x_1 representa a A_1
x_2 representa a A_2
x_3 representa a A_3
x_4 representa a A_4

gráficamente tenemos la siguiente situación:





El problema consiste en encontrarle un representante al conjunto A_5, pero $A_5 \subseteq \{x_1, x_2, x_3, x_4\}$, entonces utilizando el procedimiento de la demostración del teorema 4.2.

1. Sea $E_0 = A_5$; $E_0 = \{x_3, x_4\} \subseteq \{x_1, x_2, x_3, x_4\}$; $T = 0$

2. $J_0 = \{3,4\}$; $E_1 = \hat{S}(J_0) = \hat{S}(\{3,4\}) = A_3 \cup A_4 = \{x_1, x_3, x_4\}$

3. $\{x_1, x_3, x_4\} \subseteq \{x_1, x_2, x_3, x_4\}$ implica ir a 2; $T = 1$

2. $J_1 = \{1,3,4\}$; $E_2 = \hat{S}(J_1) = \hat{S}(\{1,3,4\}) = A_1 \cup A_3 \cup A_4 = \{x_1, x_2, x_3 x_4\}$

3. $\{x_1, x_2, x_3, x_4\} \subseteq \{x_1, x_2, x_3, x_4\}$ implica ir a 2; $T = 2$

2. $J_2 = \{1,2,3,4\}$; $E_3 = \hat{S}(J_2) = \hat{S}(\{1,2,3,4\}) = A_1 \cup A_2 \cup A_3 \cup A_4$
   $= \{x_1, x_2, x_3, x_4, x_5, x_6\}$

3. $\{x_1, x_2, x_3, x_4, x_5, x_6\} \not\subseteq \{x_1, x_2, x_3, x_4\}$ parar en $T=2$.
   $b \in \hat{S}(J_2) - \{x_1, x_2, x_3, x_4\}$
   $b \in \{x_1, x_2, x_3, x_4, x_5, x_6\} - \{x_1, x_2, x_3, x_4\} = \{x_5, x_6\}$

   Sea $b = x_5$, $x_5 \in A_{i2}$; $j_2 \in J_2 - J_1 = \{1,2,3,4\} - \{1,3,4\} = 2$
   $\Rightarrow x_5 \in A_2$

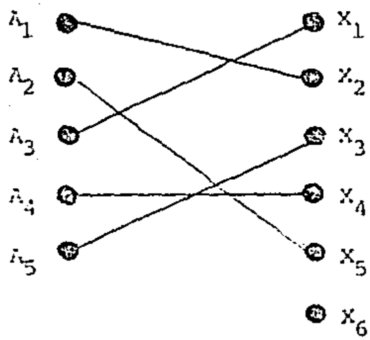
   $x_2 \in E_2 - E_1$; $x_2 \in A_{i1}$; $i_1 \in J_1 - J_0 = \{1,3,4\} - \{3,4\} = 1$
   $x_2 \in A_1$.
   $x_1 \in E_1 - E_0$; $x_1 \in A_{i0}$ $j_0 \in J_0 = \{3,4\}$ sea $j_0 \in 3$
   $x_1 \in A_3$
   $x_3 \in E_0 = A_5$.

lo cual produce

$$\begin{array}{ccccc} x_1 & x_2 & x_3 & (x_4)^* & x_5 \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\ A_3 & A_1 & A_5 & A_4 & A_2 \end{array}$$

---

* no se movió



Un problema común que se presenta en la literatura de optimización combinatoria y gráfica es el del Harem, el cual consiste en: Considere un conjunto de hombres denotados por A_1, $A_2,\ldots,A_n$ y suponga que cada hombre A_j conoce m_j mujeres, donde $j=1,\ldots,n$; Se desea determinar el máximo de "parejas" compatibles suponiendo que un mismo hombre se puede casar con varias mujeres y que lo recíproco no es cierto.

Las condiciones para resolver este problema se encuentra en el teorema 4.3.

Ejemplo: Considere $U = \{A_1, A_2\}$ una familia donde A_i es el hombre i; $A_1 = \{m_1, m_2, m_5\}$, $A_2 = \{m_1, m_3, m_4, m_5\}$; A_1 desea tener 2 esposas y A_2 tres. Como se cumple

$$\left| \bigcup_{j \in J'} A_j \right| \geq \sum_{j \in J'} r_j \quad \forall\, J' \subseteq \{1,2\}$$

Entonces existe $X = X_1 \cup X_2$ con $|X_1| = 2$ y $|X_2| = 3$ y $X_1 = \{m_2, m_5\}$, $X_2 = \{m_3, m_4, m_5\}$ para A_1 y A_2 respectivamente, es decir X_1 son las esposas del señor A_1 y X_2 las del señor A_2.

<u>Teorema 4.3</u>. Sea $A_1,\ldots,A_n$ subconjuntos de un conjunto E_1 y sea $r_1,\ldots,r_n$ enteros positivos. Entonces existe un conjunto X el cual puede ser particionado en $X_1 \cup \ldots \cup X_n$ con $|X_j| = r_j$ y $X_j \subseteq A_j$ para cada j sí y sólo si

$$\left|\bigcup_{j\in J'} A_j\right| \geq \sum_{j\in J'} r_j \text{ para todo } J' \subseteq \{1,\ldots,n\}$$

<u>Demostración</u>. Sea $U = (\underbrace{A_1,\ldots,A_1}_{r_1}\ \underbrace{A_2,\ldots,A_2}_{r_2},\ldots,\underbrace{A_n,\ldots,A_n}_{r_n})$

una familia que consiste de r_j copias de A_j para $1 \leq j \leq n$. Existe un conjunto $X = X_1 \cup,\ldots,\cup X_n$ con $|X_j| = r_j$ y $X_j \subseteq A_j$ si y sólo si U tiene una transversal.

Por el teorema de Hall esto sucede si y sólo si para cada $S \leq \sum_{r\leq 1}^{n} r_j$ la unión de los conjuntos en cualquier subfamilia de s miembros de U tiene cardinalidad al menos S.

Considere una típica subfamilia de S miembros de U, la cual consistirá de digamos S_j copias de A_j, donde $0 \leq S_j \leq r_j$ para $1 \leq j \leq n$ y $\sum_{i=1}^{n} S_j = S$. Si escribimos $j' = \{j : S_j > 0\}$, entonces $\sum_{j\in J'} S_j = S$; y un conjunto X existe como se había requerido si y sólo si:

$\left|\bigcup_{j\in J'} A_j\right| \geq \sum_{j\in J'} S_j$ para todo $J' \subset \{1,\ldots,n\}$ y S_j entero con $0 < S_i < r_j$

y si $S_i = r_i$ se tiene que X existe si y sólo si

$$\left|\bigcup_{j\in J'} A_j\right| \geq \sum_{j\in J'} r_j \text{ para todo } J' \subseteq \{1,\ldots,n\} \blacksquare$$

4.3 Transversales parciales.

Considere una familia de subconjuntos no necesariamente diferentes de $U = (A_1, \ldots, A_n)$. Entonces $X_\ell = \{x_1, x_2, \ldots, x_\ell\}$ es una transversal parcial de tamaño ℓ de U si X_ℓ es una transversal de $(A_1, \ldots, A_\ell)$, $1 \le \ell \le n$.

Ejemplo. Considere la familia $U = \{\{a,b\}, \{a,b\}, \{a\}\}$ entonces $x = \{a,b\}$ es una parcial transversal de tamaño 2 de U, pues x es transversal de $\{\{a,b\}, \{a,b\}\}$.

La familia $U = (A_1, \ldots, A_n)$ de subconjuntos de E tiene una transversal parcial de tamaño ℓ $(\ell > 0)$ si y sólo si

$$\left|\bigcup_{j \in J'} A_j\right| \ge |J'| - n + \ell \text{ para todo } J' \subseteq \{1, \ldots, n\}$$

dicho resultado se prueba en el teorema 4.4.

Ejemplo 4.3. Considere $E = \{a, b, c, d, e\}$; $U = \{\{a,c,e\}, \{b,d\}, \{b,d\}, \{b,d\}\}$, se desea ver si U tiene transversales parciales de tamaño $\ell = 1, 2, 3, 4$. Para resolver este problema se determinan todas las uniones UA_j, $j' \in J'$ para cada cardinalidad de J' y se compara $|UA_{j'}|$ con $|J'| - n + \ell$, los resultados se encuentran en los cuadros 1 y 2:

CUADRO 1

$\lvert J' \rvert$	J'	$U A_j,\ j' \varepsilon J'$
1	{1}	{a c e}
	{2} = {3} = {4}	{b d}
2	{1,2} = {1,3} = {1,4}	{a b c d e}
	{2,3} = {2,4} = {2,5}	{b d}
3	{1,2,3} = {1,2,4}, {1,3,4}	{a b c d e}
	{2,3,4}	{b d}
4	{1,2,3,4}	{a b c d e}

CUADRO 2

1	2	3	4	5
i	$\lvert J' \rvert$	$\lvert J' \rvert - n + i$	$\lvert UA_j \rvert\ j' \varepsilon J'$	$\lvert U A_j \rvert \geq \lvert j' \rvert - n + i$
1	1	− 2	3 6 2	3 > − 2, 2 > − 2
	2	− 1	5 6 2	5 > − 1, 2 > − 1
	3	0	5 6 2	5 > 0 2 > 0
	4	1	5	5 > − 2
2	1	− 1	3 6 2	3 > − 1, 2 > − 1
	2	0	5 6 2	5 > 0, 2 > 0
	3	1	5 6 2	5 > 1, 2 > 1
	4	3	5	5 > 2
3	1	0	3 6 2	3 > 0, 2 > 0
	2	1	5 6 2	5 > 1, 2 > 1
	3	2	5 6 2	5 > 2, 2 = 2
	4	3	5	5 > 3
4	1	1	3 6 2	3 > 1, 2 > 1
	2	2	5 6 2	5 > 2, 2 = 2
	3	3	5 6 2	5 > 3, 2 $\not\geq$ 3
	4	4	5	

De la columna 5 del cuadro 2 se deduce que U solo tiene transversales parciales de tamaño 1, 2 y 3, sin embargo no tiene transversales de tamaño 4, pues para J' = {2, 3, 4}, $|\bigcup_{j \in J'} A_j| = 2$, mientras que $|J'| - n + t = 3$ y no se cumple la condición del teorema 4.3.

Teorema 4.4

La familia $U = (A_1, \ldots, A_n)$ de subconjuntos de E tiene una transversal parcial de tamaño ℓ ($\ell > 0$) si y sólo si

$$|\bigcup_{j\in J'} A_j| \geq |J'| - n + \ell \quad \forall\ J' \subseteq \{1,\ldots,n\}$$

Dem: Si U tiene tal transversal parcial, o si las condiciones dadas se cumplen, ciertamente $\ell < n$. Entonces sea D un conjunto de $n - \ell$ elementos disjuntos de E y sea $A_j' = A_j \cup D$ para cada $j \in \{1,\ldots,n\}$. Entonces U tiene una transversal parcial de longitud ℓ si y sólo si la familia $(A_1', \ldots, A_n')$ tiene una transversal, lo cual (por el teorema de Hall) sucede sí y solo si

$$|\bigcup_{j\in J'} A_j'| \geq |J'| \quad \forall\ J' \subseteq \{1,\ldots,n\}$$

o dado que estas condiciones son automáticamente satisfechas si $I' = \emptyset$, si y sólo si

$$|\bigcup_{j\in J'} A_j'| \geq |J'| \quad \forall\ J' \neq \emptyset,\ J' \subseteq \{1,\ldots,n\}$$

En ocaciones lo que interesa es saber si un conjunto dado $M \subseteq E$ es una transversal parcial de la familia $U = (A, \ldots, A_n)$. Esto sucede cuando $|(\bigcup_{j\in J'} A_j) \cap M| \geq |J'| + |M| - n$ para todo $J' \subseteq \{1,\ldots,n\}$, lo cual se demuestra en el teorema 4.5.

Ejemplo Sea $E = \{a,b,c,d,e\}$; $U = (\{ace\}, \{bd\}, \{bd\}, \{bd\})$ y $M = \{a, b, c\}$. ¿Es M una transversal parcial de U?.

Obteniendo todos los conjuntos $\{(\bigcup_{j\in J'} A_j) \cap M\}$ y comparando la cardinalidad de estos con $|J'| + |M| - n$ como se observa en el cuadro 1, se concluye que M no es transversal parcial de U, pues $|\bigcup_{j\in J'} A_j \cap M| \ngeq |J'| + |M| - n$ cuando $J = \{1,2,3\}$.

C U A D R O 1

A $\|J'\|$	B $\|J'\|+\|M\|-n$	C J'	D $(\cup A_j) \cap \{abc\}$	$\|D\| \geq B$
1	0	$\{1\}$	$\{a,c\}$	$2 > 0$
		$\{2\} = \{3\} = \{4\}$	$\{b\}$	$1 > 0$
2	1	$\{1,2\}=\{1,3\}=\{1,4\}$	$\{a,b,c\}$	$3 > 1$
		$\{2,3\}=\{2,4\}=\{2,5\}$	$\{b\}$	$1 = 1$
3	2	$\{1,2,3\}=\{1,2,4\}=\{1,3,4\}$	$\{a,b,c\}$	$3 \ngeq 2$
		$\{2,3,4\}$	$\{b\}$	$1 \ngeq 2$

4.4 Aplicaciones de transversales.

Cuadros latinos.

Un rectángulo latino mxn es una matriz $M = (m_{ij})$ de orden mxn, cuyos elementos son números enteros que satisfacen:

i) $1 \leq m_{ij} \leq n$.

ii) ningún par de elementos de cualquier fila o columna son iguales.

Si m = n entonces es un cuadro latino, donde sólo se cumple ii).

El problema que se plantea es: ¿Todo rectángulo latino de orden mxn, m < n se puede convertir en un cuadrado latino?, dicho problema tiene solución con base en el siguiente resultado:

Sea M un rectángulo latino de mxn con m < n entonces M puede ser convertido en un cuadrado latino añadiendo n-m filas nuevas.

Ejemplo. Considere el siguiente rectángulo latino:

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 3 & 4 & 2 & 5 & 1 \end{pmatrix}$$

$$E = \{1, 2, 3, 4, 5\}$$

construya la familia $U = (A_1, \ldots, A_n)$ donde A_i denota el conjunto de elementos de E que no aparecen en la i-ésima columna de M.

$$A_1 = (2,4,5),\ A_2 = (1,3,5),\ A_3 = (1,4,5),\ A_4 = (1,2,3)$$

y

$$A_5 = (2,3,4).$$

Obtenga una transversal T de U y añádala como tercer renglón al rectángulo

$$T_1 = (4,1,5,2,3) \rightarrow \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 3 & 4 & 2 & 5 & 1 \\ 4 & 1 & 5 & 2 & 3 \end{pmatrix}$$

Repitiendo el proceso hasta obtener el cuadro

$$\begin{matrix} T_2 = (5,3,4,1,2) \\ \\ T_3 = (2,5,1,3,4) \end{matrix} \rightarrow \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 3 & 4 & 2 & 5 & 1 \\ 4 & 1 & 5 & 2 & 3 \\ 5 & 3 & 4 & 1 & 2 \\ 2 & 5 & 1 & 3 & 4 \end{pmatrix} \blacksquare$$

Matrices (0.1).

Las matrices (0,1) son muy importantes pues estas son las de adyacencia de una gráfica, también podemos obtener una matriz de adyacencia A de orden $m \times n$ de la familia $U = (A_1, \ldots, A_n)$ de $E = \{e_1, \ldots, e_n\}$; donde $A = (a_{ij})$; $a_{ij} = 1$ si $e_j \in A_j$ $a_{ij} = 0$ en otro caso. El rango de término de A se define como el mayor número de unos de A, tal que ningún par de uno están en la misma fila o columna, entonces U tiene una transversal si y sólo sí el rango de términos de A es r, y este es el número de elementos de una transversal parcial de U.

Ejemplo. Sea $E = \{\overset{1}{a}, \overset{2}{b}, \overset{3}{c}, \overset{4}{d}, \overset{5}{e}\}$; $u = \{\{\overset{1\,3\,5}{ace}\}, \{\overset{1\,2}{bd}\}, \{\overset{1\,2}{bd}\}, \{\overset{1\,2}{bd}\}\}$

$$A = \begin{array}{c} \\ A_1 \\ A_2 \\ A_3 \\ A_4 \end{array} \begin{array}{c} \begin{array}{ccccc} a & b & c & d & e \end{array} \\ \begin{pmatrix} 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \end{pmatrix} \end{array}$$

Aquí el rango de término de A es 3 y por tanto U tiene una transversal parcial de tamaño 3.

El encontrar el rango de términos de una matriz (0,1) asociada a una gráfica bipartita es equivalente a encontrar el número máximo de acoplamientos en la gráfica.

4.5 Matroides transversales.

Una vez que se ha introducido el concepto de transversales y transversales parciales, resulta natural introducir el concepto de matroides transversales el cual se puede mirar desde dos enfoques equivalentes.

Si E es un conjunto finito no vacio y si $U = (A_1, \ldots, A_m)$ es una familia de subconjuntos no vacios de E, entonces todas las transversales parciales de U pueden ser considerados como los conjuntos independientes de una matroide $M = (E, F(U))$ sobre E. El rango de un subconjunto X de E es igual al tamaño del transversal parcial mayor contenido en X.

Ejemplo. Considere $E = \{1,2,3\}$, $U = \{\{1\}, \{2,3\}\}$.
Las transversales parciales de tamaño 1 son:

$$T_1^1 = \{1\},\ T_2^1 = \{2\},\ T_3^1 = \{3\}$$

Las transversales parciales de tamaño 2 son:

$$T_1^2 = \{1,2\},\ T_2^2 = \{1,3\}$$

que corresponde a la familia F de conjuntos independientes de E.

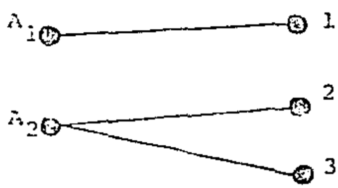
$$r(x) = \begin{cases} 1 & \text{si} \quad x = \{1\} \text{ o } \{2\} \text{ o } \{3\} \text{ o } \{2,3\} \\ 2 & \text{si} \quad x = \{1,2\} \text{ o } \{1,3\} \end{cases}$$

Otra forma de definir una matroide transversal es a partir de una gráfica bipartita.

Considere una gráfica bipartita $G = (U, \hat{\Phi} E)$ donde $U = \{A_1, \ldots, A_n\}$ en esta familia de subconjuntos de E; $E = \{e_1, e_2, \ldots, e_m\}$ y el arco $(i,j) \in \hat{\Phi}$ si y sólo si $e_j \in A_i$.

Entonces, $M = (E,F)$ es una matroide transversal donde F es el conjunto de transversales parciales de U.

<u>Ejemplo</u>. Considere $E = \{1,2,3\}$, $U = \{A_1, A_2\}$ donde $A_1 = \{1\}$, $A_2 = \{2,3\}$ y $G = (U, \hat{\Phi}, E)$ la siguiente gráfica bipartita, donde $\hat{\Phi} = \{(1,1), (2,2), (2,3)\}$



La matroide transversal asociada a esta gráfica es: $M=(E,F)$ donde

$$F = \{\phi, \{1\}, \{2\}, \{3\}, \{1,2\}, \{1,3\}\}$$

o bien

$$F = \{T \text{ tal que } T \text{ es transversal parcial de } U\}$$

$$r(x) = \begin{cases} 1 & \text{si } x = \{1\} \text{ ó } \{2\} \text{ ó } \{3\} \text{ ó } \{2,3\} \\ 2 & \text{si } x = \{1,2\} \text{ ó } \{1,3\} \end{cases}$$

Hasta aquí se ha visto como un transversal se puede ver como un sistema de representantes distintos, como un acoplamiento de una gráfica bipartita o como un conjunto independiente de una matroide. En los dos primeros casos se discutieron las condiciones necesarias y suficientes para la existencia de transversales y acoplamiento máximo a través del teorema de Hall. Resulta natural plantearse la siguiente pregunta. ¿Cuáles son las condiciones necesarias y suficientes para que una matroide transversal tenga un conjunto independiente maximal?. La respuesta se encuentra en el teorema de Rado el cual es una extensión del teorema de Philip Hall.

Teorema 4.5. De Rado (1942). Sea $M = (E, F)$ una matroide con función rango r. Entonces la familia $U = (A_1, \ldots, A_n)$ de subconjunto de E tiene una transversal sí y sólo si

$$r\left(\bigcup_{j \in J'} A_j\right) \geq |J'| \text{ para toda } J' \subseteq \{1, \ldots, n\}$$

Con el teorema de Rado tenemos la condición necesaria y suficiente para que una familia U de subconjuntos de E tengan una transversal que corresponde a un conjunto independiente maximal de M. Si ahora el problema consiste en encontrar una transversal óptima, entonces para los casos particulares donde se plantea una sola matroide transversal se puede utilizar alguna versión apropiada del algoritmo glotón, a continuación damos esta versión.

Algoritmo Glotón para una transversal.

Propósito.- Encontrar una transversal parcial o total de una familia $U = (A_1, \ldots A_n)$ de máximo peso, representada por una matriz (0,1), cuyas columnas estan ordenados en forma decreciente según las ponderaciones de sus columnas, y con la condición $A_i \cap A_j = \phi$.

Descripción

```
Empezar
Leer  M,N, A[I,J],  W(J)
W(0) = 0  P = W(0)   I = 1,   J = 1;
MIENTRAS I ≤ N  hacer J = 1
MIENTRAS J ≤ M  hacer
Si A[I,J] = 1  entonces
               Empezar
               Escribir [I,J], W(J).
               P = P + W(J); K = I+1; L=J + 1;
               Mientras  L ≤ N hacer
               A[I,L] = 0 ;  L = L + 1



               FIN
J = J + 1
(FIN  SI y MIENTRAS J)
I = I + 1
FIN  (MIENTRAS I)
     Escribir P.
FIN
```

Ejemplo. Considere el conjunto de trabajos $E = \{1,2,3,4,5,6\}$; La familia $U = \{A_i \mid A_i$ son los trabajos que se tienen que entregar a la hora $i\}$ es decir $A_1 = (1,2)$, $A_2 = (4)$, $A_3 = \{3,5\}$, $A_4 = A_5 = \phi$, $A_6 = (6)$ y los pesos $w(1) = 10$, $w(2) = 9$, $w(3)=7$, $w(4) = 6$, $w(5) = 4$, $w(6) = 2$.

La matriz (0,1) asociada es la siguiente:

$$\begin{array}{rc} A = & \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \\ \text{pesos} & \begin{matrix} 10 & \;9 & \;7 & \;6 & \;4 & \;2 \end{matrix} \end{array}$$

Siguiendo el algoritmo

M = 6, N = 6

P = 0 I = 1, J = 1

A[1,1] = 1 [1,1], 10

P= 10, K = 2, L = 2

A[1,2] = 0, A[1,3] = 0, A[1,4] = 0, A[1,5] = 0, A[1,6] = 0,

J = 2

J = 1, J = 2 J = 3 J = 4

A[2,1]≠1, A[2,2]≠1, A[2,3]≠1, A[2,4]=1 [2,4]

P = 16, K = 3, J = 5

A[2,5]=0, A[2,6]=0

J = 5

I = 3, J = 1 J = 2 J = 3

A[3,1]≠1, A[3,1]≠1, A[3,3].....[3,3]

P = 23, K = 4, J = 4

A[3,4]=0, A[3,5]=0, A[3,6]=0

J = 4

I = 4 J = 1 J = 2 J = 3 J = 4 J = 5 J = 6

A[4,1]≠1, A[4,2]≠1, A[4,3]≠1, A[4,4]≠1, A[4,5]≠1, A[4,6]≠1

I = 5 J = 1 J = 2 J = 3 J = 4 J = 5 J = 6

A[5,1]≠1, A[5,2]≠1, A[5,3]≠1, A[5,4]≠1, A[5,5]≠1, A[5,6]≠1

I = 6 J = 1 J = 2 J = 3 J = 4 J = 5 J = 6

A[6,1]≠1, A[6,2]≠1, A[6,3]≠1, A[6,4]≠1, A[6,5]≠1, A[6,6]=1

P = 25 [6,6].

La secuencia óptima de trabajos es:

[1,1] a la hora 1 el trabajo 1 con ahorro de 10

[2,4] a la hora 2 el trabajo 4 con ahorro de 6

[3,3] a la hora 3 el trabajo 3 con ahorro de 7

[6,6] a la hora 6 el trabajo 6 con ahorro de 2

En total se tiene un ahorro de 25.

Observe que con este procedimiento se obtienen todas las posibles transversales o bien parejas de dos subconjuntos, de peso máximo. Pero es importante hacer notar que sólo funciona cuando $A_i \cap A_j = \phi$, pues es el caso de una sola matroide.

4.6 Matroides y transversales Comunes.

Un transversal común se define como el conjunto de m elementos que pertenecen a las familias $U = (A_1, \ldots, A_m)$ y $W = (B_1, \ldots, B_m)$ de sunconjuntos de un conjunto finito E.

Las condiciones para que un par de familias de subconjunto de un conjunto tengan un transversal común se establecen en la siguiente generalización del teorema de Hall.

Teorema 4. Sea E un conjunto finito no vacio y sean $U = (A_1,$ $\ldots, A_n)$ y $W = (B_1, \ldots, B_m)$ dos familias de subconjuntos no vacios de E. Entonces U y W tienen un transversal comun si y sólo si, para todos los subconjuntos J y K de $\{1, 2, \ldots, m\}$

$$\left| \left(\bigcup_{j \in J} A_j \right) \cap \left(\bigcup_{k \in K} B_k \right) \right| \geq |J| + |K| - m$$

La demostración se omite.

No se sabe en que condiciones existe un transversal común para tres familias de subconjuntos no vacios de un conjunto.

Una aplicación de transversales comunes se encuentra en problemas de horarios, donde E puede denotar el conjunto de horas en que deben celebrarse las clases, los conjuntos A_i denotan las horas disponibles por m profesores dados y las conjuntos B_i denotan las horas en las que hay n aulas. Entonces el descubrimiento de una transversal común de U y W, permite asignar a cada profesor en aula disponible y a una hora conveniente para el.

Específicamente considere el siguiente:

Ejemplo: En una escuela que tiene un horario de 7 a 11 AM. diario, se imparten clases con duración de una hora. En tal escuela laboran tres profesores (P_1, P_2 y P_3); los cuales tienen las siguientes horas disponibles; P_1 a las 7, 9, 11 hrs. P_2 a las 7 y 8 hrs. P_3 a las 9 a 10 hrs. cada profesor sólo debe dar un curso. Por otro lado se cuenta con tres aulas A_1, A_2 y A_3 a las 7, 8 y 10 se tiene disponible el aula A_1, a las 7 y 9 el aula A_2 y a las 11 hrs. el aula A_3.

¿Como asignar a cada profesor un aula a la hora que el pueda dar su curso?.

Sea $E = \{7, 8, 9, 10, 11\}$, $U = \{P_1, P_2, P_3\}$ y $W = \{A_1, A_2, A_3\}$ donde $P_1 = (7, 9, 11)$, $P_2 = (7, 8)$, $P_3 = (9, 11)$ y $A_1 = (7, 8, 10)$, $A_2 = (7, 9)$ y $A_3 = (11)$.

El problema es encontrar una transversal común entre U y W.

Una transversal común es

$$T = \{7, 9, 10\} \text{ pues } 7 \in P_2 \cap A_1,\ 9 \in P_1 \cap A_2$$

$$11 \in P_3 \cap A_3.$$

La respuesta es:

El profesor P_2 en el aula A_1 a las 7 hrs.
El profesor P_1 en el aula A_2 a las 9 hrs.
El profesor P_3 en el aula A_3 a las 11 hrs.

Matroides de Transversales comunes.

Así como a un transversal se le asocia una matroide también a un transversal común se le asocia una matroide.

Considere un conjunto finito no vacio E y, sea $U = (A_1,\ldots,A_n)$ y $W = (B_1,\ldots,B_n)$ dos familias de subconjuntos no vacios de E, entonces U y W tienen una matroide transversal común si y sólo si para todo los subconjuntos J y K de $\{1,\ldots,n\}$ se cumple:

$$r\left(\left(\bigcup_{j\in J} A_j\right) \cap \left(\bigcup_{k\in K} B_k\right)\right) \geq |J| + |K| - n$$

Demostración:

Sea $M = (E,r)$ la matroide cuyos conjuntos independientes son los transversales parciales de la familia U, entonces U y W tienen una transversal común si y sólo si W tiene una transversal independiente, pero por el teorema de Rado para matroides (4.5), esto es así sí y sólo si $r(\cup B_j) \geq |K|$ $\forall$ $K(\{1\ldots n\}$ es decir la unión de k cualesquiera B_j contiene un transversal parcial de U de tamaño k.

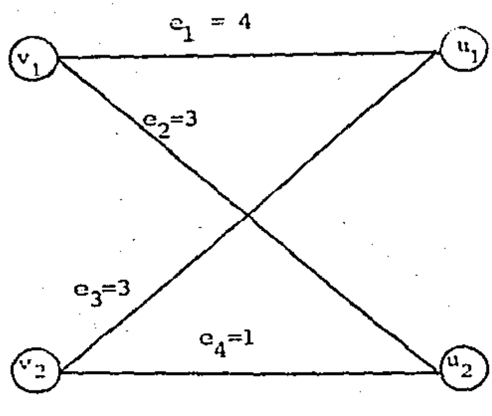
Intersección de Matroides.

Una transversal común y el problema de acoplamiento en gráficas bipartitas resultan ser problemas que involucran la intersección de matroides:

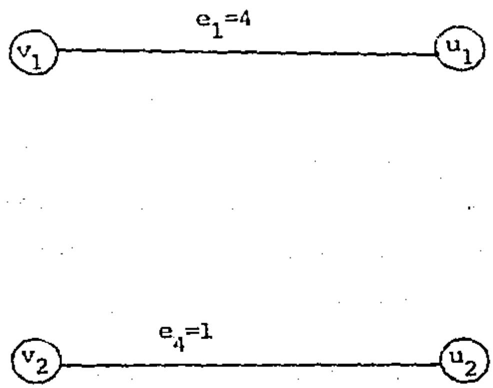
Dados $M_1 = (E, F_1)$ y $M_2 = (E, F_2)$ dos matroides sobre un mismo conjunto E, entonces $M_1 \cap M_2 = (E, F_1 \cap F_2)$ es una intersección de matroides, cuando se desea encontrar una transversal común entre los conjuntos F_1 y ϵF_2, se plantea la cuestión de encontrar un conjunto $I \epsilon F_1 \cap F_2$.

Si los elementos del conjunto E tienen asignado una ponderación, entonces se puede pedir que el conjunto $I \epsilon F_1 \cap F_2$ sea óptimo es decir maximal y aquí aparece la interrogante; ¿Se puede usar el algoritmo glotón?. La respuesta desafortunadamente es negativa.

Ejemplo. Considere la gráfica $G = (V, U, E)$ donde $V = \{v_1, v_2\}$, $U = \{u_1, u_2\}$ y $E = \{e_1, e_2, e_3, e_4\}$ con $w(e_1) = 4$, $w(e_2) = 3$, $w(e_3) = 3$, $w(e_4) = 1$.

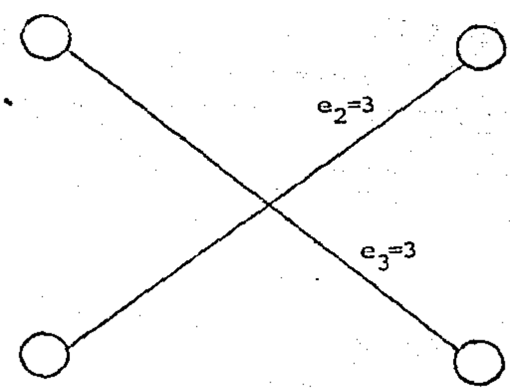


Si se utiliza el algoritmo glotón para encontrar el acoplamiento de máximo peso el resultado sería:



Sin embargo esto no es el máximo.

El máximo es:



CAPITULO 5

APLICACIONES DEL ALGORITMO GLOTÓN

En este capítulo se describen dos aplicaciones del algorítmo glotón relacionados con el problema del árbol de expansión ya sea de peso mínimo o máximo. Específicamente el de Síntesis de Redes de flujo y el de la red de Steiner.

En la primera sección se plantea el problema de síntesis de redes de flujo, que consiste en la construcción de una red factible que minimice el costo de proveer una unidad de capacidad de un arco que conecta dos nodos. Asimismo se presenta un algoritmo que como primer paso calcula el árbol de paso máximo; entonces si para este paso se usa el algoritmo glotón el algoritmo resulta eficiente.

El problema de la Red de Steiner que se discute en la segunda sección, y que consiste en encontrar un árbol de expansión de peso mínimo sobre n nodos, usando s puntos auxiliares (los puntos de Steiner). No ha sido posible, pese a la aplicación del algoritmo glotón encontrar un procedimiento eficaz para resolverlo, pues las posibles elecciones de los puntos de Steiner conducen a que este problema sea de la clase NP duro.

Aplicaciones del glotón.

5.1 Síntesis de Redes: una aplicación del árbol de expansión máxima.

El problema de encontrar el árbol de expansión máxima es decir el árbol de expansión de peso máximo de una gráfica $G = (V,E)$ conectada con pesos $w_{ij} \geq 0$ asignados a cada arco $\{v_i, v_j\}$ es esencialmente el mismo que el problema de encontrar el árbol de expansión de peso mínimo descrito en el capítulo 3. Si definimos a la distancia $d_{ij} = W - w_{ij}$ donde $W = \max_{ij} \{w_{ij}\}$.

Entonces la suma de las distancias $d(t)$ de cualquier árbol de expansión T estará relacionado con el peso total $w(t)$ de t por

$$w(t) = \sum_{ij \varepsilon T} w_{ij} = \sum_{ij \varepsilon T} (w - d_{ij}) = (|V| - 1)W - d(T).$$

Así es inmediato que el árbol de expansión mínima bajo d es el mismo que al máximo de G bajo w.

Si G fuera desconectada, el bosque de peso máximo de $G = (V,E)$ bajo w es la unión de las componentes conectadas bajo d.

Síntesis de Redes de Flujo.

En el problema de síntesis de flujo en una red se ha encontrado una interesante aplicación del cálculo del árbol de expansión con peso máximo.

Suponga que tenemos una matriz Rnxn $R = [r_{ij}]$ de requerimientos de flujo. Se llamará una red factible si es posible inducir un flujo de valor w_{ij} entre los nodos i y j donde $w_{ij} \geq r_{ij}$. Aquí se plantea el problema de la construcción de una red factible que minimice alguna función de las capacidades de los arcos c_{ij} por ejemplo $\sum_{i,j} a_{ij} c_{ij}$ donde a_{ij} puede ser el costo de proveer una unidad de capacidad de un arco entre v_i, v_j.

Este es un problema de programación lineal y puede ser resuelto aplicando el método dual simplex para un sistema de $2^n - 1$ desigualdades lineales de la forma:

$$\sum_{\substack{i\varepsilon S\\ j\varepsilon T}} C_{ij} \geq \max_{\substack{i\varepsilon S\\ j\varepsilon T}} \{r_{ij}\}$$

para cada corte (S,T) de la red.

Sin embargo existe un procedimiento propuesto por Gomory and Hu que no requiere una enumeración explícita de esas constantes. A continuación se describe el proceso para la versión simplificada donde los a_{ij} son iguales.

Algoritmo: Gomory y Hu

Propósito: Determinar una red de síntesis de flujo.

Descripción

Entrada:

R: Una matriz de requerimientos

V: Un conjunto de nodos

Salida:

S = La red de síntesis de flujos con costos constantes.

Paso 1. (Arbol de requerimiento dominante). Sea r_{ij} el peso representante del arco (v_i, v_j) en una gráfica de n nodos. Encontrar el árbol de expansión máxima el cual es el árbol de requerimientos dominantes.

Paso 2. (Descomposición del árbol de requerimiento dominante). Descomponer el árbol de requerimiento dominante en una suma de un árbol de requerimiento uniforme más un pedazo, por medio de la resta del árbol interior más pequeño al árbol dominante.
Descomponer cada árbol no uniforme que quede, de la misma manera hasta que el árbol de requerimiento dominante sea expresado como una suma de árboles uniformes de requerimientos.

Paso 3. (Síntesis del ciclo). Sintetizar cada árbol uniforme como un ciclo a través de sus nodos y algún otro. Cada arco del ciclo tiene capacidad igual a la mitad del requerimiento uniforme. Suponer los ciclos resultantes para formar la red final sumando las capacidades en los arcos.

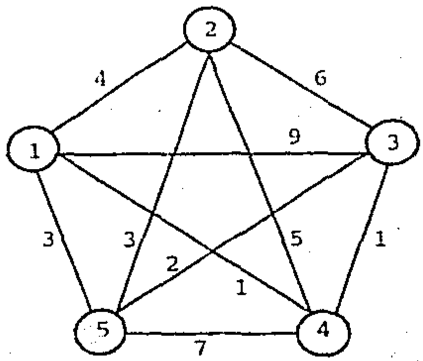
Cada arco de la red final corresponde a un arco de la capacidad requerida en cada dirección.

Ejemplo 5.1

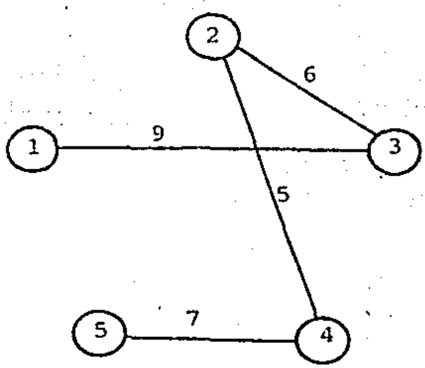
Considere la siguiente matriz de requerimientos:

$$R = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{bmatrix} 0 & 4 & 9 & 1 & 3 \\ 4 & 0 & 6 & 5 & 3 \\ 9 & 6 & 0 & 1 & 2 \\ 1 & 5 & 1 & 0 & 7 \\ 3 & 3 & 2 & 7 & 0 \end{bmatrix} \end{array}$$

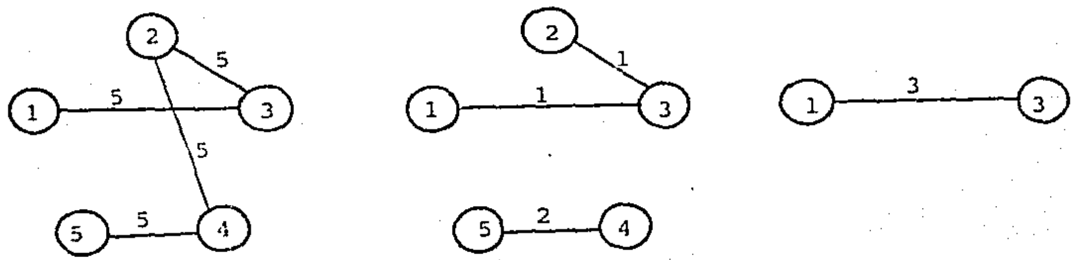
Cuya red asociada es la siguiente:



Paso 1. Arbol de requerimiento dominante

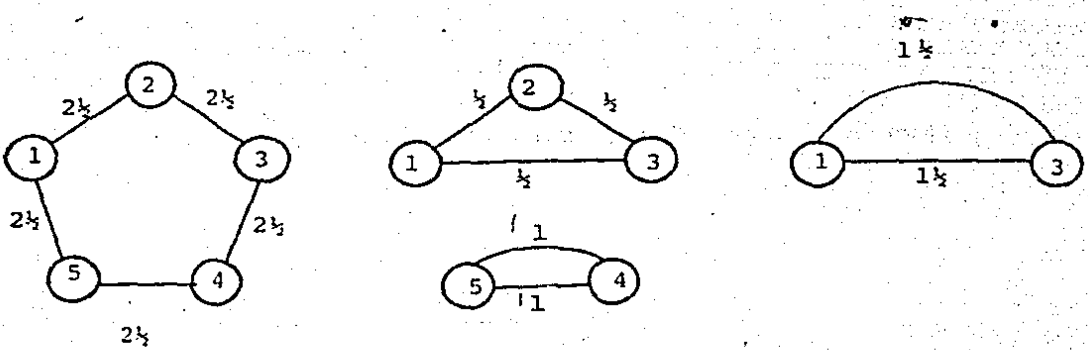


Paso 2.   Descomposición del árbol de requerimiento dominante en la suma de uniformes

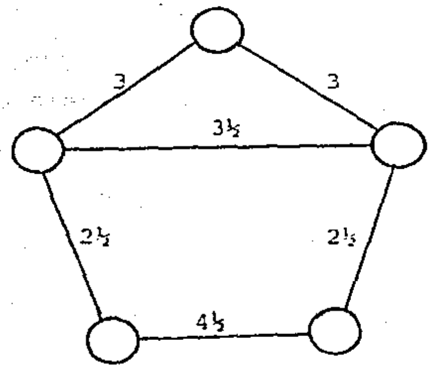


Paso 3.   (Síntesis de la red)

3.1 Síntesis en ciclos de cada árbol

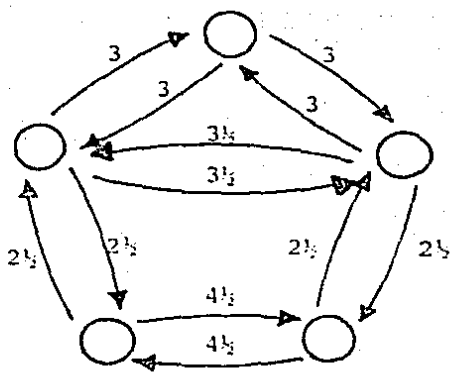


Superposición de los ciclos resultantes



Red final.

Como cada arco final corresponde a un arco de la capacidad requerida en cada dirección se tiene la red final.



Justificación del algoritmo.

Para justificar el algoritmo, primero probaremos que la red es factible y luego que es óptima.

La red resultante es factible, pues como en cada nodo se tiene que flujo que entra es igual al que sale entonces se puede inducir un flujo w_{ij} mayor o igual al que requiere el nodo (j) $\therefore$ $w_{ij} \geq r_{ij}$ siempre y cuando las capacidades de los arcos lo permitan i.e., $\sum C_{ij} \geq \max \{r_{ij}\}$.

Ahora bien la red es óptima pues para alguna red factible se cumple que:

$$\sum_{j} C_{ij} \geq \max \{r_{ij}\} \quad i=1,2,\ldots,n$$

Esto es la suma de las capacidades de los arcos incidentes que entran en algún nodo (i) debe ser al menos tan grande como el máximo del flujo de requerimientos de i.

Aun sí la desigualdad se satisface con igualdad para la red sintetizada por el algoritmo $\therefore$ la red es óptima.

5.2 El problema de la red Steiner*.

Los puntos de Steiner.

Una aplicación interesante del árbol de expansión mínima se encuentra en el problema de Steiner el cual consiste en encontrar un árbol de longitud mínima que una n puntos en el plano, con la ayuda de otros S puntos (los puntos de Steiner). Este problema se planteó y resolvió para 3 puntos de la siguiente manera:

Considere que tres aldeas A, B y C han de ser unidas por un sistema de carreteras de longitud total mínima.

En términos matemáticos el problema equivale a que dados tres puntos en el plano, A, B y C se pide un cuarto punto P tal que la suma a + b + c sea mínima, donde a, b y c son las distancias del punto P a A, B y C respectivamente.

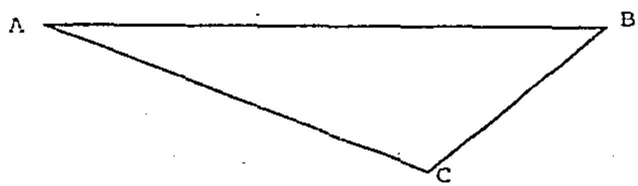
La solución del problema es la siguiente: Si en el triángulo A B C todos los angulos son menores que 120°, P es el punto desde el cual cada uno de los tres lados AB, BC y CA subtiende un ángulo de ABC. Ahora bien si un ángulo de ABC, por ejemplo C es igual o mayor que 120° el punto P coincide con el vértice C.

---

* Mascheroni Jacob Steiner (1796-1863) prof. de geometría en la Universidad de Berlin.

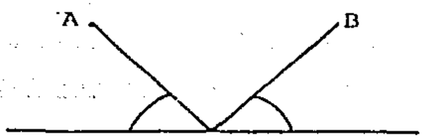
Demostración:

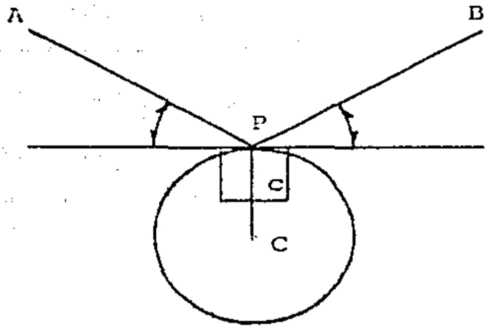
Existen dos posibilidades: P coincide con uno de los vértices o P es distinto de ellos. En el primer caso es evidente que P debe ser el vértice de mayor ángulo de ABC, sea el C, puesto que la suma CA + CB es menor que cualesquiera otra suma de los otros dos lados del triangulo ABC (ver figura)



$$CA + CB < CA + AB \quad y \quad CA + CB < CB + AB$$

En el segundo caso sea K una circunferencia de radio c y centro C entonces si P es un punto de K, tal que PA + PB es mínimo y si A y B son exteriores a K PA y PB deben formar ángulos iuales con el radio PC, puesto que los ángulos de PA y PB con respecto a la tangente son iguales, ya que la distancia más corta de A a B pasando por P (P está en la tangente al círculo) es aquélla donde PA y PB con respecto a la tangente forma ángulos iguales (de incidencia y reflexión) observe las figuras:





Dado que el mismo razonamiento se aplica a la posición de P y al círculo de radio a y centro A, entonces los ángulos formados por PA, PB y PC son iguales, es decir de 120°. Aquí se ha supuesto que A y B son exteriores a K, si al menos A se encontrara en K o fuera interior, y dado que P no coincide con A o B tendríamos $a + b \geq AB$, pero $AC \leq c$ ya que A no es exterior a K de donde resulta:

$$a + b + c \geq AB + AC$$

lo que significa que sólo obtendríamos la misma suma de distancias si P coincide con A y por tanto A y B son exteriores.

Resulta muy dificil la generalización de estas ideas para más de tres puntos.

5.3 La red de Steiner. Un problema matroidal.

Sin embargo un problema un poco más fácil y que basado en los puntos de Steiner es el de la red de Steiner, el cual consiste en encontrar el árbol de peso mínimo que una a n nodos, en una gráfica de (n+s) nodos, dicho árbol puede o no incluir algunos de los s nodos. El problema de la red de Steiner tiene interpretación matroidal:

Sea T un árbol de expansión para los n nodos específicos, entonces el problema se puede formular como sigue:

En la matroide gráfica de la red encontrar un conjunto independiente I de peso mínimo tal que $S_p(T) \subseteq S_p(I)$, es decir; dada una gráfica con n nodos y un árbol T de ella de peso mínimo, podemos encontrar un árbol T' con nodos y puntos de Steiner, el cual siempre será menor.

Antes de ver como se puede resolver este problema se verá un resultado que dice cual es el número máximo de puntos de Steiner de una gráfica de n nodos.

Lema. Suponga que las longitudes de los arcos a_{ij} de una red satisfacen una métrica, es decir son no negativos y cumplen con

$$a_{ij} \leq a_{ik} + a_{kj} \qquad \forall\ i,j,k.$$

Demostración.

Sea p el número de puntos de Steiner en un árbol mínimo. Sea x la media del número de arcos del árbol incidentes en los puntos de Steiner y sea Y la media del número de arcos del árbol incidente en los n puntos. El número de arcos en el árbol es:

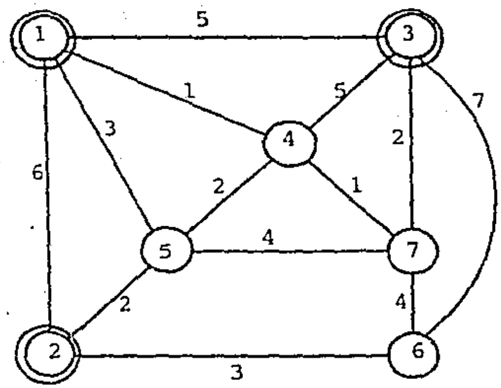
$$n+p-1 = \frac{px + ny}{2}$$

pero debido a la condición de métrica, $x \geq 3$, pues si $x = 0$ el punto de Steiner no está en el árbol, si $x = 1$ no conectaría ni siquiera dos nodos, y si $x = 2$ por la condición de métrica no tendría sentido ese punto pues $a_{ij} \leq a_{ij} + a_{kj}$ siendo k el punto de Steiner; además, $y \geq 1$, esto es obvio pues los nodos tienen que estar conectados, por tanto $n+p-1 \geq \frac{3p+n}{2}$ de donde $p \leq n - 2$.

La resolución de la red de Steiner hace uso de algoritmos que desafortunadamente son NP. En el presente trabajo se presenta un algoritmo que resulta polinomial en n y exponencial en S. Su descripción se hace paralelamente con su ilustración sobre un ejemplo.

Algoritmo para la Red de Steiner.

Considere la gráfica de la figura donde se desea unir los puntos 1, 2 y 3 por un árbol de expansión mínima que puede usar los puntos de Steiner 4, 5, 6 y 7.



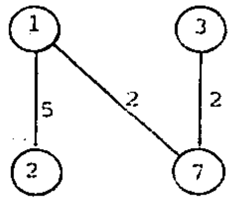
Algoritmo para la red de Steiner.

Paso 1. Cálculo de la ruta más corta.

Si los arcos pesados no satisfacen las condiciones métricas, calcular la ruta más corta entre todos los pares de nodos, y replantee los arcos pesados con las longitudes de las rutas más cortas añadiendo los arcos en la red donde sea necesario:

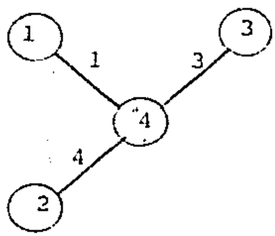
Para este paso construimos la matriz de distancias

$$
\begin{array}{c}
\\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7
\end{array}
\begin{array}{c}
\begin{array}{ccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{array} \\
\begin{bmatrix}
\infty & 6 & 5 & 1 & 3 & \infty & \infty \\
6 & \infty & \infty & \infty & 2 & 3 & \infty \\
5 & \infty & \infty & 5 & \infty & \infty & 2 \\
1 & \infty & 5 & \infty & 2 & \infty & 1 \\
3 & 2 & \infty & 2 & \infty & \infty & 4 \\
\infty & 3 & \infty & \infty & \infty & \infty & 4 \\
\infty & \infty & 2 & 1 & 4 & 4 & \infty
\end{bmatrix}
\end{array}
$$

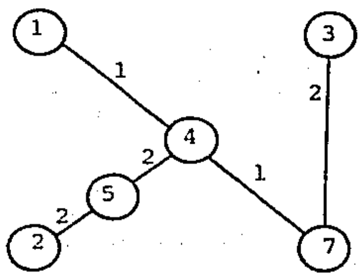
1	(1,2,3,7)	(1,2) = 5 (1,3) = 4 (1,7) = 2 (2,3) = 7 (2,7) = 5 (3,7) = 2		9

Paso 3. Seleccionar el árbol de mínimo peso, y transfórmelo dentro del árbol de la red original es decir reemplazar cada arco del árbol de expansión con los arcos de la ruta más corta entre los nodos en cuestión.

El árbol de peso mínimo fue el que contiene los nodos (1, 2, 3, 4):



transformándolo en la red original y tomando las rutas más cortas se tiene:

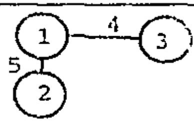
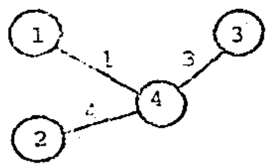
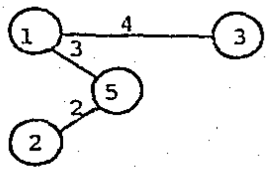
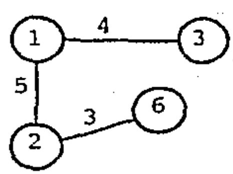


y con el algoritmo de la cascada por ejemplo encontramos la matriz de rutas más cortas antre par de nodos.

<u>Paso 2.</u> Cálculo del árbol de expansión mínima.

Para cada posible subconjunto de n-2 o menos puntos de Steiner resuelva el árbol de expansión mínima.

En nuestro ejemplo habrá $\binom{4}{0} + \binom{4}{1}$ árboles de expansión mínima.

Cantidad de puntos de Steiner	Nodos en el árbol	distancias entre los nodos	árbol	Peso
0	(1,2,3)	(1,2) = 5 (1,3) = 4 (3,2) = 7		9
1	(1,2,3,4)	(1,2) = 5 (1,3) = 4 (1,4) = 1 (2,3) = 7 (2,4) = 4 (3,4) = 3		8
1	(1,2,3,5)	(1,2) = 5 (1,3) = 4 (1,5) = 3 (2,3) = 7 (2,5) = 2 (3,5) = 5		9
1	(1,2,3,6)	(1,2) = 5 (1,3) = 4 (1,6) = 6 (2,3) = 7 (2,6) = 3 (3,6) = 6		12

Complejidad.

Este algoritmo requiere la solución de un problema de árbol de expansión mínima sobre no más que 2n - 2 nodos para cada σ cambio de puntos de Steiner, donde:

$$\sigma = \sum_{i=0}^{n-2} \binom{2}{i} \leq 2^{s}$$

y de aqui se sigue que la complejidad computacional está acotada por $0(n^2, 2^s)$ el cual es polinomial en n, pero exponencial en s y esto sin contar con que el cálculo de la ruta más corta del paso uno es de $0(n+s)^3$.

Por tanto este problema está dentro de la clase NP.

CAPITULO 6

C O N C L U S I O N E S

El problema de optimización combinatoria ha sido analizado tanto en su estructura como su factibilidad de aplicación a problemas típicos de la Investigación de Operaciones. Como consecuencia de este análisis se han revisado conceptos tan importantes como la complejidad algorítmica y se ha identificado una clase de problemas de optimización combinatoria que aceptan un análisis teórico por medio del concepto de matroide introducido por Whitney y popularizado en la última década.

El concepto de matroide y sus implicaciones en la optimización combinatoria están resumidos en el denominado algoritmo glotón cuya simplicidad de aplicación a una variedad de problemas prácticos es cada vez más conocida. Esta simplicidad evita el fantasma de la complejidad computacional y cada día alienta la identificación de problemas combinatorios prácticos que pueden ser resueltos bajo este esquema.

Un aspecto importante del enfoque matroidal seguido en este trabajo respecto a la optimización combinatoria es la unificación de conceptos de campos tales como el álgebra lineal y la teoría de gráficas y su correspondiente aplicación en problemas de Investigación de Operaciones.

Los problemas "difíciles" que corresponden a aquéllos que tienen algoritmo NP como son, El agente viajero, satisfacción mochila y todos los equivalentes en el sentido computacional, tienen una estructura que corresponde a la intersección de tres matroides, la cual no ha sido resuelta. Este es un problema que queda abierto, y en la medida en que la teoría matroidal como computación avancen se tendrán mayores posibilidades de canalizarlo y resolverlo.

Además de las aplicaciones que se presentaron en este trabajo existen otros campos donde la teoría matroidal es aplicable, como por ejemplo en redes eléctricas con el lema de coloreamiento de arcos y las relaciones entre circuitos y cocircuitos de una matroide y su dual.

También existen estructuras combinatorias y geométricas que resultan ser variaciones de matroides, entre los que podemos mencionar a las matroides orientadas cuyas aplicaciones se tienen en Programación Lineal ver [11].

Finalmente, me parece que estas estructuras son muy ricas y es importante seguir incursionando en esta teoría, pues sus aplicaciones en la matemática aplicada son relevantes.

BIBLIOGRAFIA

[1] Aho, Hopcroft, Ullman, "THE DESIGNAND ANALYSIS OF COMPUTER ALGORITHMS", Adisson Wesley., 1975.

[2] Bondy and Murt, "BRAPH THEORY WITH APLICATIONS", The McMillan Press L.T.D., 1976.

[3] Bryant Victor and Perfect Hazel, "INDEPENDENCE THEORY IN COMBINATORICS", Chapman and Hall, 1980.

[4] Cristofides, Mingozzi, Toth, Sandi, "COMBINATORIAL OPTIMIZATION", Wiley, 1979.

[5] Courant and H. Robbins., "GAHT IS MATHEMATICS"?, Oxford University Press., 1941.

[6] Kung., Joseph P.H., "A SOURCE BOOK IN MATROID THEORY", Birkhäuser., 1986.

[7] Lawler, Eugene L., "COMBINATORIAL OPTIMIZATION NETWORKS AND MATROIDS", Hoit, Rinehart and Winston, 1976.

[8] Papadimitriou C. and Steiglyts. K., "COMBINATORIAL OPTIMIZATION ALGORITHMIC AND COMPLEXITY", Prentice-Hall, 1982.

[9] Reingold, Nievergelt, Deo, "COMBINATORIAL ALGORITHMS THEORY AND PRACTICE", Prentice Hall, 1977

[10] Sakarovitch, Michel, "OPTIMIZATION COMBINATOIRE METHODES MATHEMATIQUES ET ALGORITHMIQUES", Hermann Enseignement des Sciences, 1984.

[11] Sánchez Arroyo Abdon. "PROGRAMACION LINEAL EN EL CONTEXTO DE MATROIDES ORIENTADOS" Tesis, 1984.

[12] Welsh D.J.A., "MATROID THEORY", Academic Press, 1976.

[13] Wilson, Robin J., "INTRODUCCION A LA TEORIA DE GRAFOS" Alianza Universidad, 1982.

A. CONCEPTOS BÁSICOS DE GRÁFICAS.

Una gráfica G es un par ordenado de conjuntos (V.E), donde $V = (v_1, \ldots, v_n)$ es un conjunto no vacio de elementos llamados vértices o nodos y E es una familia finita de pares no ordenados de elementos de V, llamados aristas que se denota (v_i, v_j). Una gráfica G se dice simple si E es solo un subconjunto de pares no ordenados de v.

Una digráfica (ó gráfica orientada o red) es un par ordenado de conjuntos (V,A), donde V es un conjunto no vacio de elementos llamados vértices o nodos y A es una familia finita de pares ordenados de elementos de V, llamados arcos o aristas orientadas o di-aristas, donde $(v_i, v_j) \neq (v_j, v_i)$. Se dice que dos vértices V y w de una gráfica son adyacentes si la gráfica contiene una arista que los une, y se dicè que dicha arista es incidente en los vértices.

Se dice que dos aristas diferentes de G son adyacentes si tienen, al menos, un vértice en común.

El grado de un vértice V de G es igual al número de aristas que inciden en v. Un vértice aislado es un vértice de grado cero. Un vértice extremo o terminal tiene grado uno.

La matriz de adyacencia asociada a una gráfica G con el conjunto de vértices: $\{v_1, \ldots, v_n\}$ es la matriz $A = [a_{ij}]$ de orden nxn donde a_{ij} es el número de aristas que unen a un par de vértices adyacentes. La suma de los elementos de cada fila o columna es igual al grado del vértice correspondiente.

La matriz de incidencia asociada a una gráfica G con el conjunto de vértices: $\{v_1, \ldots, v_n\}$ y el conjunto o familia de aristas $\{e_1, \ldots, e_n\}$ es la matriz $B = [b_{ij}]$ de orden mxn donde $b_{ij} = 1$ si el vértice v_j y la arista e_i son incidentes y $b_{ij} = 0$ en otro caso.

Una gráfica completa es una gráfica simple donde cualquier par de vértices son adyacentes.

Una gráfica regular es una gráfica donde todos sus vértices tienen el mismo grado.

Una gráfica bipartita es aquella donde el conjunto de vértices puede ser particionado en dos subconjuntos V_1 y V_2 tal que cada arista tiene un extremo en V_1 y otro en V_2, tal partición (V_1, V_2) es llamada una bipartición de la gráfica.

La trayectoria de una gráfica G es una sucesión de aristas adyacentes sin repetición que une un par de vértices (v_i, v_j).

El circuito de una gráfica es una trayectoria donde el vértice o nodo inicial coincide con el nodo final y tiene una sola arista.

Trayectoria Euleriana de una gráfica G es una trayectoria cerrada que incluye a todas las aristas de G.

Circuito Hamiltoneano de una gráfica, es una trayectoria cerrada que pasa exactamente una vez a través de cada vértice de G.

Un bosque es una gráfica que no contiene circuito. Un árbol es un bosque conexo.

Teorema. Sea T una gráfica con n vértices. Los siguientes enunciados referentes a T son equivalentes.

a) T es un árbol
b) T no contiene circuitos, y posee n-1 aristas
c) T es conexo, y tiene n-1 aristas
d) Cada par de vértices de T estan conectados por una única trayectoria.
e) T no contiene ningún circuito, pero la adición de cualquier nueva arista crea exactamente uno.

B. EL PROBLEMA DE SATISFACCION.

Dado un conjunto $X = \{x_1,\ldots,x_n\}$ de variables booleanas y una expresión boolena de la forma

$$E = C_1 \wedge C_2, \ldots, \wedge C_m$$

donde cada 'cláusula' C_i $(i=1,\ldots,m)$ es una expresión de la forma:

$$C_i = u_{j_1} V u_{j_2} V, \ldots, V u_{j_{k(i)}}$$

y donde cada u_j es una de las variables de X. El problema consiste en determinar si existen valores de las variables x_k $k=1,\ldots,n$ tales que $E = 1$.

Ejemplos. Sea $(x_1 \; V \; x_2 \; V \; \bar{x}_3) \wedge (\bar{x}_1 \; V \; x_2) \wedge (x_3)$. La respuesta es verdadera cuando $x_1 = 0$ ó 1 y $x_2 = 1$, $x_3 = 1$ pues, $E = 1$. Mientras que para $E = (x_1 \; V \; x_2 \; V \; \bar{x}_3) \wedge (\bar{x}_1 \; V \; x_2) \wedge (\bar{x}_2 \; V \; \bar{x}_3)$. La respuesta es falsa pues $E = 0$.

P R O G R A M A S

En estos apéndices se muestran tres programas de computadora, correspondientes al algoritmo glotón que resuelven algunos problemas específicos de optimización combinatoria tratados en el presente trabajo. Los programas estan escritos en Pascal, y se corrieron en la versión turbo Pascal-versión 3, de una microcomputadora Printaform.

Los programas fueron escritos por Irene Sánchez Guevara en la Sección de Investigación de Operaciones de la División de Estudios de Posgrado de la Facultad de Ingeniería, UNAM, en diciembre de 1987.

```
{*                                                                          *}
{*          NOMBRE: GLOMAT                                                  *}
{*                                                                          *}
{*          NOMBRE DEL ARCHIVO: GLOMAT                                      *}
{*                                                                          *}
{*          PROPOSITO: DADA UNA MATRIZ CON COLUMNAS ORDENADAS EN FORMA     *}
{*                     DECRECIENTE SEGUN SU PESO, ESTE PROGRAMA ENCUEN     *}
{*                     TRA EL SUBCONJUNTO DE COLUMNAS LINEALMENTE INDE     *}
{*                     PENDIENTES Y DE PESO MAXIMO, POR MEDIO DE ELIMI     *}
{*                     NACION GAUSSIANA.                                    *}
{**************************************************************************}
{*                                                                          *}
{* DESCRIPCION: ESTE PROGRAMA SOLO CONSTA DEL PROGRAMA PRINCIPAL.          *}
{* LAS VARIABLES QUE INTERVIENEN SON LAS SIGUIENTES:                        *}
{* M y N DE TIPO ENTERO, SON LAS VARIABLES QUE CORRESPONDEN AL NU-          *}
{*        MERO DE RENGLONES Y COLUMNAS DE LA MATRIZ                         *}
{*     A: DE TIPO ARREGLO DE DOS DIMENSIONES, ES LA MATRIZ DE DATOS         *}
{*        QUE SE LEE Y DONDE SE EFECTUA LA ELIMINACION GAUSSIANA.           *}
{*     W: DE TIPO ARREGLO UNIDIMENSIONAL, ES EL VECTOR DE PESOS DE          *}
{*        LAS COLUMNAS DE A Y EL CUAL SE LEE.                               *}
{*     P: DE TIPO REAL, ES LA VARIABLE DONDE SE CALCULA EL MAXIMO PESO      *}
{*        DE LAS COLUMNAS LINEALMENTE INDEPENDIENTES.                       *}
{*                                                                          *}
{**************************************************************************}

program Gloton;
{*          D E C L A R A C I O N E S                                       *}

const
     Renglones = 10;
     Columnas  = 10;
var
   N,M,I,J,K,L:integer;
   A:array[1..Renglones,1..Columnas] of real;
   W,Alfa,Beta:array[1..Columnas] of real;
   P:real;

{*  LECTURA DE  DATOS          *}

begin
     CLRSCR;
     WRITELN('*****************************************************************');
     WRITELN('*                                                               *');
     WriteLN('*          Dame el numero de columnas                           *    ');
     Readln(N);
     WriteLN('*          Dame el numero de renglones                          *    ');
     Readln(M);
     WRITELN('*                                                               *');
     WRITELN('*****************************************************************');
     WRITELN; READLN;CLRSCR;
     for I:=1 to M do
         for J:=1 to N do
         begin
             WriteLN('*          Dame el valor de A(',I,',',J,')                    *

             Readln(A[I,J])
         end;
         WRITELN('*********************************************');
         WRITELN;READLN;CLRSCR;
         for J:=1 to N do
         begin

             WriteLN('*          Dame el valor de la comuna  ',J,'                  *  ');
         Readln(W[J])
     end; READLN; CLRSCR;
```

```
(* EMPIEZA EL PROGRAMA *)

      P:=0.0;
      K := 1;
      While K < N do
      begin
            I:=1;
            while I <= M do
            begin
                  if A[I,K] <> 0 then
                  begin
                     Alfa[K] := I;
                     P:=P + W[K];
                     for J:=K+1 to N do
                     begin
                        Beta[J] := A[trunc(Alfa[K]),J] / A[trunc(Alfa[K]),K];
                        for L:=1 to M do
                            A[L,J]:=A[L,J]-Beta[J]*A[L,K]
                      end;
                     K := K + 1
                  end;
                  I := I + 1
            end;
            K := K + 1
       end;
(*          ESCRITURA DE LOS RESULTADOS                                                    *)
WRITELN;WRITELN;
WRITELN('*******************************************************************');
WRITELN('*          LAS COLUMNAS LINEALMENTE INDEPENDIENTES                  *');
WRITELN('*           ESTAN EN ESTA MATRIZ:                                   *');
WRITELN('*******************************************************************');
WRITELN;WRITELN;

          for I:=1 to M do begin

           for J:=1 to N do
               Write(A[I,J]:5:0);

           Writeln;
      end;
WRITELN;
WRITELN;
WRITELN('***********************************************************');
WRITELN('*                                                         *');
WRITELN('*             EL PESO MAXIMO ES ', P:10:2, '              *');
WRITELN('*                                                         *');
WRITELN('***********************************************************')
end.
```

```
**************************************************************
*                                                            *
*        Dame el numero de columnas                          *    5
*        Dame el numero de renglones                         *    4
*                                                            *
**************************************************************


*       Dame el valor de A(1,1)                   *  1
*       Dame el valor de A(1,2)                   *  0
*       Dame el valor de A(1,3)                   *  2
*       Dame el valor de A(1,4)                   *  0
*       Dame el valor de A(1,5)                   *  1
*       Dame el valor de A(2,1)                   *  0
*       Dame el valor de A(2,2)                   *  -1
*       Dame el valor de A(2,3)                   *  -1
*       Dame el valor de A(2,4)                   *  1
*       Dame el valor de A(2,5)                   *  1
*       Dame el valor de A(3,1)                   *  3
*       Dame el valor de A(3,2)                   *  2
*       Dame el valor de A(3,3)                   *  8
*       Dame el valor de A(3,4)                   *  1
*       Dame el valor de A(3,5)                   *  4
*       Dame el valor de A(4,1)                   *  2
*       Dame el valor de A(4,2)                   *  1
*       Dame el valor de A(4,3)                   *  5
*       Dame el valor de A(4,4)                   *  0
*       Dame el valor de A(4,5)                   *  2
**************************************************


*       Dame el valor de la columna  1             *  10
*       Dame el valor de la columna  2             *  9
*       Dame el valor de la columna  3             *  8
*       Dame el valor de la columna  4             *  4
*       Dame el valor de la columna  5             *  1
**************************************************************
*       LAS COLUMNAS LINEALMENTE INDEPENDIENTES              *
*        ESTAN EN ESTA MATRIZ:                               *
**************************************************************


    1    0    0    0    0
    0   -1    0    0    0
    3    2    0    3    0
    2    1    0    1    0


******************************************************
*                                                    *
*          EL PESO MAXIMO ES        23.00            *
*                                                    *
******************************************************

 >
```

```
 *                                                                  *
 *   NOMBRE: AEMIN                                                  *
 *                                                                  *
 *   NOMBRE DEL ARCHIVO: AEMIN                                      *
 *                                                                  *
 *   PROPOSITO: DADA UNA GRAFICA (V,E) Y LAS PONDERACIONES DE LOS AR-  *
 *              COS, ENCONTRAR EL ARBOL DE EXPANSION CON PESO MINIMO   *
 *                                                                  *
 ********************************************************************

ESTE PROGRAMA CONSTA DE UN SOLO MODULO QUE ES EL PROGRAMA PRINCIPAL,Y CO-
RRESPONDE A UN ALGORITMO DEL TIPO GLOTON.
LAS VARIABLES QUE INTERVIENEN SON:
V DE TIPO CONJUNTO: ES EL CONJUNTO DE NODOS DE LA GRAFICA, QUE SE GE-
                    NERA AL LEER EL NUMERO DE NODOS.
U DE TIPO CONJUNTO: ES EL CONJUNTO DE NODOS QUE SE VA FORMANDO EN CADA
                    ITERACION.


D DE TIPO ARREGLO: ES LA MATRIZ DE DISTANCIAS ENTRE CADA PAR DE NODOS
                   ADYACENTES, CUANDO LOS NODOS NO LO SON SE PIDE QUE
                   LAS DISTANCIAS SEAN DE ORDEN MUY GRANDE.(100 O 10000).
T ES UN ARREGLO DE REGISTRO: ES LA LISTA DONDE SE ENCUENTRAN LAS ARISTAS
                             QUE CONFORMAN EL ARBOL.


 *                                                                  *
 *                       D E C L A R A C I O N E S                  *
 *                                                                  *}
CONST
   MAXINT=32000;
   M = 20;

TYPE
   ARCO=RECORD
        INI: INTEGER;
        FIN: INTEGER;
      END;
var
   X: ARRAY[1..M] OF INTEGER;
   D: ARRAY[1..M,1..M] OF REAL;
   U,V: SET OF 1..M;
   MIN:REAL;
   CONT,IND,PROX, I, J, N ,K: INTEGER;
   T:ARRAY [1..M] OF ARCO;
   SUM:REAL;
{********************************************************************
*               P R O G R A M A   P R I N C I P A L                 *
*********************************************************************}
BEGIN
  K:=0;
  {******************************************************
  *  LECTURA DE DATOS                                    *
  ******************************************************}
  WRITELN ('##################################################');
  WRITELN ('#                                                #');
  WRITELN ('# DAME EL NUMERO DE NODOS, ( MAXIMO 20) SI QUIERES MAS   #');
  WRITELN ('# CAMBIA EL VALOR DE LA CONSTANTE M DEL PROGRAMA          #');
  WRITELN ('#                                                #');
  WRITELN ('##################################################');
  WRITELN;WRITELN;
  READLN (N);
  V:=[];
  WRITE ('EL CONJUNTO DE NODOS ES: {');
```

```
writeln;
WRITELN('###########################################################');
WRITELN('#                                                         #');
WRITELN('#  LOS ARCOS QUE FORMAN EL ARBOL DE PESO MINIMO SON:      #');
WRITELN('#                                                         #');
   CONT:=0;
    FOR I:= 1 TO N DO
     FOR J:= 1 TO N  DO
       BEGIN
         FOR IND:=1 TO K DO
          BEGIN
            IF (T[IND].INI=I) AND (T[IND].FIN=J) THEN
             BEGIN
               WRITE ('    (',I,',',J,')');
               CONT:=CONT + 1;
               IF CONT>=N THEN
                          BEGIN
                          WRITELN;
                          CONT :=0
                          END
             END
          END;
       END;
 WRITELN;
 WRITELN('#                                                         #');
 WRITELN('###########################################################');
 WRITELN;
 WRITELN;
 WRITELN('###########################################################');
 WRITELN('#                                                         #');
 WRITELN('#  EL VALOR, PESO O LONGITUD DEL ARBOL ES: ',SUM:10:2, ' #');
 WRITELN('#                                                         #');
 WRITELN('###########################################################')
END.
```

```
##################################################################
#                                                                #
# DAME EL NUMERO DE NODOS, ( MAXIMO 20) SI QUIERES MAS           #
# CAMBIA EL VALOR DE LA CONSTANTE M DEL PROGRAMA                 #
#                                                                #
##################################################################

6
EL CONJUNTO DE NODOS ES: {1 2 3 4 5 6 }


######################################################
#                                                    #
# DAME LAS DISTANCIAS ENTRE CADA PAR DE NODOS        #
# DIRECTAMENTE CONECTADOS, EN CASO CONTRARIO         #
# DALES DISTANCIAS DE 100, 1000 O 10 000             #
# SEGUN EL ORDEN DE LOS DATOS.                       #
#                                                    #
######################################################


#        Dame la distancia de 1 A 2        #    7
#        Dame la distancia de 1 A 3        #    8
#        Dame la distancia de 1 A 4        #    100
#        Dame la distancia de 1 A 5        #    100
#        Dame la distancia de 1 A 6        #    100
#        Dame la distancia de 2 A 3        #    7
#        Dame la distancia de 2 A 4        #    8
#        Dame la distancia de 2 A 5        #    6
#        Dame la distancia de 2 A 6        #    100
#        Dame la distancia de 3 A 4        #    100
#        Dame la distancia de 3 A 5        #    6
#        Dame la distancia de 3 A 6        #    4
#        Dame la distancia de 4 A 5        #    2
#        Dame la distancia de 4 A 6        #    100
#        Dame la distancia de 5 A 6        #    7


iteracion  1  arista entrante  21
iteracion  2  arista entrante  52
iteracion  3  arista entrante  45
iteracion  4  arista entrante  35
iteracion  5  arista entrante  63

##################################################################
#                                                                #
#  LOS ARCOS QUE FORMAN EL ARBOL DE PESO MINIMO SON:             #
#                                                                #
   (2,1)   (3,5)   (4,5)   (5,2)   (6,3)
#                                                                #
##################################################################


##############################################################
#                                                            #
#  EL VALOR, PESO O LONGITUD DEL ARBOL ES:          25.00 #
#                                                            #
##############################################################

 >
```

```
program GLOTON_PARA_ENCONTRAR_UNA_TRANSVERSAL;
{**********************************************************************}
{*                                                                    *}
{*      NOMBRE: GLOTON PARA ENCONTRAR UNA TRANSVERSAL                 *}
{*                                                                    *}
{*      NOMBRE DEL ARCHIVO: GLOTRANS                                  *}
{*                                                                    *}
{*      PROPOSITO: ESTE PROGRAMA CALCULA UN SUBCONJUNTO TRANSVERSAL   *}
{*                 DE UNA FAMILIA DE SUBCONJUNTOS. PARA TAL OBJETIVO  *}
{*                 ES NECESARIO DARLE LA INFORMACION A TRAVES DE UNA  *}
{*                 MATRIZ CERO UNO; CON LAS COLUMNAS ORDENADAS DECRE  *}
{*                 CIENTEMENTE SEGUN SUS PESOS.UNA APLICACION IMPOR-  *}
{*                 TANTE DEL CALCULO DE ESTA TRANSVERSAL ES LA OBTE   *}
{*                 NER UNA SECUENCIA DE N TRABAJOS EN UNA MAQUINA     *}
{*                                                                    *}
{*      DESCRIPCION: ESTE ES UN PROGRAMA QUE CORRESPONDE A UN ALGORIT *}
{*                   MO DEL TIPO GLOTON, Y SOLO CONSTA DEL PROGRAMA   *}
{*                   PRINCIPAL. LAS VARIABLES QUE INTERVIENEN SON:    *}
{*      A: DE TIPO ARREGLO, ESTA ES LA MATRIZ CERO UNO, LA CUAL DEBE  *}
{*                        SER LEIDA CON LAS COLUMNAS EN FORMA DECRE   *}
{*                        CIENTE, CON RESPECTO A SU PESO.             *}
{*      W: DE TIPO ARREGLO, ESTE ES EL VECTOR DE PESOS DE CADA COLUM- *}
{*                        NA                                          *}
{*      P; DE TIPO REAL, ESTA ES LA VBARIABLE DONDE SE CALCULA EL PE  *}
{*                      SO MAXIMO DE LA TRANSVERSAL OBTENIDA          *}
{*                                                                    *}
{**********************************************************************}
{*                                                                    *}
{*           D E C L A R A C I O N E S                                *}

 var
A: ARRAY [1..50,1..50] OF INTEGER;
W: ARRAY [1..50] OF REAL;
I,J,K,L,M,N : INTEGER;
P : REAL;
{*                                                                    *}
{*           LECTURA  DE DATOS                                        *}
{*                                                                    *}

BEGIN        CLRSCR;
WRITELN('****************************************************************');
WRITELN('*                                                              *');
WRITELN('*     DAME EL NUMERO DE RENGLONES DE LA MATRIZ                 *');
WRITE('*                                                                *');
READLN (M);
WRITELN('*     DAME EL NUMERO DE COLUMNAS DE LA MATRIZ                  *');
WRITE('*                                                                *');
READLN (N);
WRITELN('*     DAME LA MATRIZ POR RENGLONES                             *');
WRITELN('*                                                              *');
WRITELN('****************************************************************');R
FOR I:= 1 TO M DO
 FOR J:= 1 TO N DO
    BEGIN
    WRITE ('*.         DAME EL VALOR DE A[',I,',',J,']=          *      ');
    READLN (A[I,J])
    END;
 WRITELN;WRITELN;READLN;CLRSCR;
 WRITELN('*                                                             *');
 WRITELN('*           LA MATRIZ (0,1)  ES:                              *');
 WRITELN('*                                                             *');
 WRITELN;
  FOR I:=1 TO M DO
```

```
    FOR J:=1 TO N DO
    WRITE(A[I,J]:5);
    WRITELN;
 END;WRITELN; READLN;CLRSCR;

 FOR J:=1 TO N DO
     BEGIN
     WRITE ('*          DAME EL VALOR DE LA COLUMNA ',J,'           *       ');
     READLN (W[J])
     END;READLN;CLRSCR;
WRITELN;WRITELN;
WRITELN('*******************************************************************');
WRITELN('*                                                                 *');
WRITELN('*          LA SECUENCIA OPTIMA DE TRABAJOS ES:                    *');
WRITELN('*                                                                 *');
WRITELN('*******************************************************************');
WRITELN;
P:=0.0;
I:=1; J:=1;
WHILE I<= M DO
     BEGIN
     J:=1;
     WHILE J<= N DO
     BEGIN
     IF A[I,J] = 1 THEN
                    BEGIN
                    WRITE ('***  (',I,',',J,'); A LA HORA  ',I);
                    WRITELN(' EL TRABAJO ', J,' CON AHORRO DE ', W[J]:10:2);
                    WRITELN;
                    P:=P+W[J];
                    FOR L:= J+1 TO N DO
                    A[I,L]:= 0;
                    FOR K:= I+1 TO M DO
                    A[K,J]:=0
                    END;
      J:=J+1
      END;
      I:=I+1
      END;
  WRITELN('*******************************************************************');
  WRITELN('*                                                                 *');
  WRITELN('*  EL PESO O AHORRO MAXIMO DE ESTA SECUENCIA ES:',P:10:2,'        *');
  WRITELN('*                                                                 *');
  WRITELN('*******************************************************************')
  END.
```

```
*****************************************************************
*                                                               *
*     DAME EL NUMERO DE RENGLONES DE LA MATRIZ                  *
*                                                               *6
*     DAME EL NUMERO DE COLUMNAS DE LA MATRIZ                   *
*                                                               *6
*     DAME LA MATRIZ POR RENGLONES                              *
*                                                               *
*****************************************************************

*                                                             *
*        LA MATRIZ (0,1)  ES:                                 *
*                                                             *

    1    1    0    0    0    0
    0    0    0    1    0    0
    0    0    1    0    1    0
    0    0    0    0    0    0
    0    0    0    0    0    0
    0    0    0    0    0    1

*        DAME EL VALOR DE LA COLUMNA 1         *     10
*        DAME EL VALOR DE LA COLUMNA 2         *     9
*        DAME EL VALOR DE LA COLUMNA 3         *     7
*        DAME EL VALOR DE LA COLUMNA 4         *     6
*        DAME EL VALOR DE LA COLUMNA 5         *     4
*        DAME EL VALOR DE LA COLUMNA 6         *     2


*****************************************************************
*                                                               *
*        LA SECUENCIA OPTIMA DE TRABAJOS ES:                    *
*                                                               *
*****************************************************************

***  (1,1); A LA HORA  1 EL TRABAJO 1 CON AHORRO DE      10.00

***  (2,4); A LA HORA  2 EL TRABAJO 4 CON AHORRO DE       6.00

***  (3,3); A LA HORA  3 EL TRABAJO 3 CON AHORRO DE       7.00

***  (6,6); A LA HORA  6 EL TRABAJO 6 CON AHORRO DE       2.00

*****************************************************************
*                                                               *
*  EL PESO O AHORRO MAXIMO DE ESTA SECUENCIA ES:      25.00     *
*                                                               *
*****************************************************************

>
```

APENDICE D

MATROIDES

Antes de dar la definición formal de matroide observemos ciertas propiedades que poseen las matrices.

Considere la matriz:

$$A = \begin{bmatrix} 1 & 3 & -6 \\ 2 & 6 & -10 \end{bmatrix}$$

Cualquier subconjunto de columnas de A es linealmente independiente o linealmente dependiente, y estas clases de conjuntos no son arbitrarios, para verlo consideremos a E como el conjunto de columnas de A. Es decir:

$$E = \left\{ \left(\begin{bmatrix} 1 \\ 2 \end{bmatrix}, \begin{bmatrix} 3 \\ 5 \end{bmatrix}, \begin{bmatrix} -6 \\ -10 \end{bmatrix} \right) \right\}$$

Observemos que cada columna por si sola es linealmente independiente pues

$$K \begin{bmatrix} 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$K \begin{bmatrix} 3 \\ 5 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$K \begin{bmatrix} -.6 \\ -10 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Entonces de la matriz A se ha formado la pareja (E,F) E = conjunto de columna de A y F = {subconjunto de columnas linealmente independientes}. Además F tiene las siguientes propiedades:

1) Cualquier subconjunto de un conjunto de columnas linealmente independiente también lo es

Por ejemplo si $\left\{ \begin{bmatrix} 1 \\ 2 \end{bmatrix}, \begin{bmatrix} 3 \\ 5 \end{bmatrix} \right\}$ es l.i.

entonces $\begin{bmatrix} 3 \\ 5 \end{bmatrix}$ también es l.i.

a) Si Ipe I_{p+1} son conjuntos de columnas l.i de p y p+1 columnas respectivamente, entonces I_p junto con alguna columna de I_{p+1} formará un conjunto de columnas l.i. de p+1 columnas.

Haremos esta prueba exhaustiva para $I_1 = \begin{bmatrix} 3 \\ 5 \end{bmatrix}$

I_1,	I_2	$e \in I_2 - I_1$	¿$I_2 \Theta I_1 \cup \{e\}$ está en F?
$\begin{bmatrix} 3 \\ 5 \end{bmatrix}$	$\begin{bmatrix} 1 \\ 2 \end{bmatrix}\begin{bmatrix} 3 \\ 5 \end{bmatrix}$	$\begin{bmatrix} 1 \\ 2 \end{bmatrix}$	$\begin{bmatrix} 1 \\ 2 \end{bmatrix}\begin{bmatrix} 3 \\ 5 \end{bmatrix}$ esta en F.
$\begin{bmatrix} 3 \\ 5 \end{bmatrix}$	$\begin{bmatrix} 1 \\ 2 \end{bmatrix}\begin{bmatrix} -6 \\ -10 \end{bmatrix}$	$\begin{bmatrix} 1 \\ 2 \end{bmatrix}$	$\begin{bmatrix} 1 \\ 2 \end{bmatrix}\begin{bmatrix} 3 \\ 5 \end{bmatrix}$ esta en F.
$\begin{bmatrix} 3 \\ 5 \end{bmatrix}$	$\begin{bmatrix} 1 \\ 2 \end{bmatrix}\begin{bmatrix} -6 \\ -10 \end{bmatrix}$	$\begin{bmatrix} -6 \\ -10 \end{bmatrix}$	$\begin{bmatrix} 1 \\ 2 \end{bmatrix}\begin{bmatrix} -6 \\ -10 \end{bmatrix}$ no está en F.

En todos los casos siempre existió al menos una columna de I_2 que junto con I formó un conjunto de columnas l.i de cardinalidad 2.

que son las combinaciones lineales de cada una, su única solución posible es K = 0.

Las columnas $\begin{bmatrix}1\\2\end{bmatrix}$ y $\begin{bmatrix}3\\5\end{bmatrix}$; $\begin{bmatrix}1\\2\end{bmatrix}$ y $\begin{bmatrix}-6\\-10\end{bmatrix}$ son dos conjuntos de columnas linealmente independientes pues para:

$$K_1\begin{bmatrix}1\\2\end{bmatrix} + K_2\begin{bmatrix}3\\5\end{bmatrix} = \begin{bmatrix}0\\0\end{bmatrix}$$

y

$$K_1\begin{bmatrix}1\\2\end{bmatrix} + K_2\begin{bmatrix}-6\\-10\end{bmatrix} = \begin{bmatrix}0\\0\end{bmatrix}$$

La única solución posible para ambas combinaciones son $K_1=K_2=0$.

Mientras que para las columnas $\begin{bmatrix}3\\5\end{bmatrix}$ y $\begin{bmatrix}-6\\-10\end{bmatrix}$. Son linealmente dependientes. La combinación:

$$K_1\begin{bmatrix}3\\5\end{bmatrix} + K_2\begin{bmatrix}-6\\-10\end{bmatrix} = \begin{bmatrix}0\\0\end{bmatrix}$$

Tiene el siguiente conjunto de soluciones

$$K_1 = -2t, \ K_2 = t \qquad t \in \mathbb{R}$$

y obviamente las tres columnas son l.d.

Con todos los conjuntos de columnas l.i. podemos formar la familia F que la llamaremos familia de conjuntos independientes.

$$F = \left\{\phi, \ \left\{\begin{bmatrix}1\\2\end{bmatrix}\right\}, \ \left\{\begin{bmatrix}3\\5\end{bmatrix}\right\}, \ \left\{\begin{bmatrix}-6\\-10\end{bmatrix}\right\}, \ \left\{\begin{bmatrix}1\\2\end{bmatrix}, \begin{bmatrix}3\\5\end{bmatrix}\right\}, \ \left\{\begin{bmatrix}1\\2\end{bmatrix}, \begin{bmatrix}-6\\-10\end{bmatrix}\right\}\right\}$$

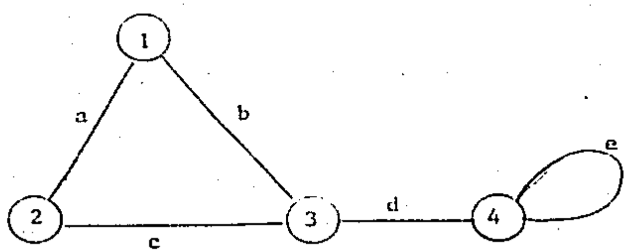
Esto siempre se cumple pues de lo contrario sucedería que un conjunto de p+1 columnas ℓ.i estaría generado por el conjunto de p columnas ℓ.i y se contradice que I_{p+1} es ℓ.i.

Estas propiedades son generalizables para todas las matrices y podemos decir que para toda matriz, si tomamos a E como el conjunto de vectores columna de A y F la familia de subconjuntos de E linealmente independientes entonces la pareja (E,F) satisface las propiedades 1 y 2 y se le denominará matroide de A.

Además de los vectores columna de las matrices existen otros conjuntos que también tienen estas propiedades tales como los conjuntos de aristas de una gráfica donde la cualidad de independencia se define como los subconjuntos de aristas de la gráfica que no forman ciclo.

Ejemplo.

Considere $G = (V,E)$; $V = \{1,2,3,4\}$ $E = \{a,b,c,d,e\}$



Los conjuntos de aristas de G que no forman ciclos estan en la familia F

$F = \{\phi, \{a\}, \{b\}, \{c\}, \{d\}, \{ab\}, \{ad\}, \{a,c\}, \{bc\}, \{b,d\}$

$\{c,d\}, \{a,b,d\}, \{a,c,d\}, \{b,c,d\}\}$.

1) De nuevo: cualquier subconjunto de uno que no contiene ciclos tampoco contiene ciclos.

Por ejemplo si se toma $\{a,c,d\}$ entonces $\{a,c\}, \{a,d\}, \{c,d\}, \{a\}, \{c\}, \{d\}$ son independientes pues tampoco forma ciclo. Y también se cumple que:

2) Dados dos subconjuntos I_p e I_{p+1} de aristas de cardinalidad p y p+1 siempre habrá una arista en la diferencia tal que unida al conjunto de cardinalidad p genere otro conjunto independiente de cardinalidad p+1.

Por ejemplo tomemos a $I_2 = \{a,c\}$

I_2	I_3	$e \varepsilon I_3 - I_2$	$I_2 U\{e\}$
$\{a,c\}$	$\{a,b,d\}$	$\{b\}$	$\{a,b,c\}$ no es independiente
	$\{a,b,d\}$	$\{d\}$	$\{a,c,d\}$ si es independiente
$\{a,c\}$	$\{a,c,d\}$	$\{d\}$	$\{a,c,d\}$ si es independiente
$\{a,c\}$	$\{a,c,d\}$	$\{b\}$	$\{a,b,c\}$ no es independiente
$\{a,c\}$	$\{b,c,d\}$	$\{d\}$	$\{a,c,d\}$ si es independiente

y de nuevo de una gráfica $G = (V,E)$ se formó una pareja (E,F) donde $F = \{I \subseteq E \mid I$ no contiene ciclo$\}$ que cumple con dos propiedades.

Así pues, podemos ahora dar la definición de matroide en forma general.

<u>Una Matroide</u> $M = (E,F)$ es una pareja de subconjuntos donde E es un conjunto finito de elementos y F es una familia de subconjuntos de E que satisface las siguientes propiedades.

1) $\phi \in F$

2) Si $A \in F$ y $B \subset A$ entonces $B \in F$

3) Si I_p e $I_{p+1} \in F$ con cardinalidad p y p+1 respectivamente entonces existe $x \in I_{p+1} - I_p$ tal que $I_p \cup \{x\} \in F$.

A los elementos de F se les denomina conjuntos independientes.