DIVISION DE ESTUDIOS DE POSGRADO
FACULTAD DE INGENIERIA

PROBLEMAS DE APAREAMIENTO

ERIC MORENO QUINTERO

TESIS

PRESENTADA A LA DIVISION DE ESTUDIOS DE

POSGRADO DE LA

FACULTAD DE INGENIERIA

DE LA

UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

COMO REQUISITO PARA OBTENER

EL GRADO DE

MAESTRO EN INGENIERIA

(INVESTIGACION DE OPERACIONES)

CIUDAD UNIVERSITARIA

MEXICO, D.F., JULIO DE 1987.

I N D I C E

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

Una variedad importante de problemas que surgen en la toma de decisiones está caracterizada por tener soluciones factibles de tipo discreto. La decisión de una compañía de comprar o no maquinaria, la asignación de recursos o el empaque de artículos en cajas de tamaños predeterminados, son ejemplos de este tipo de problemas

Desde el punto de vista conceptual, manejar una región factible de tipo discreto da la ventaja de establecer fácilmente soluciones concretas al problema y con frecuencia, el uso de gráficas elementales permite asignar significado intuitivo a una solución particular. Este hecho, sin embargo, no garantiza que se pueda resolver trivialmente cualquier problema de tipo discreto, pues si el número de objetos involucrados es grande, resolver satisfactoriamente el problema puede representar la manipulación o evaluación de una cantidad explosiva de casos, posibilidades y combinaciones.

Si se tiene el problema de asignar tres trabajadores a tres trabajos distintos, conociendo la eficiencia de cada trabajador en cada uno de ellos, encontrar la asignación de mayor eficiencia total puede efectuarse examinando las 3!=6 asignaciones posibles y escogiendo la mejor de ellas. Resolver el mismo problema para veinte trabajadores y veinte trabajos, con el método anterior, obligaría a examinar las 20! posibles asignaciones. Si se usara una computadora extraordinariamente veloz que, digamos, escribiera en papel una asignación posible cada microsegundo (exagerada para la década de los 80), el listado final terminaría de imprimirse en algo así como 77094 años, dejando obsoleto cualquier intento posible de espera.

Esta curiosa paradoja de problemas de tipo discreto, de mostrar una estructura conceptual sencilla, pero un procedimiento difícil para determinar una solución adecuada, aunada al hecho de que este tipo de problemas son comunes en muchas áreas de investigación de operaciones e ingeniería, ha llamado la atención de investigadores y científicos desde el siglo pasado.

Un caso específico de análisis de problemas discretos se encuentra en el Análisis Combinatorio, que se considera como el estudio matemático del arreglo, agrupamiento, ordenamiento o selección de objetos discretos (por lo general finitos en número). Tradicionalmente, esta disciplina ha distinguido sus problemas según se trate de: 1) existencia de arreglos especificos, 2) exhibición o evaluación de arreglos requeridos, y 3) enumeración o cuenta de posibles arreglos.

Como ejemplos de esta clase de problemas se pueden citar: calcular el número de combinaciones de n objetos tomados de r en r, encontrar el espacio muestral del experimento que consiste en extraer sin reemplazo dos bolas de una urna conteniendo m bolas rojas y n bolas negras, o exhibir las permutaciones distinguibles de n objetos, de los cuales hay k1,k2,...,kj de ellos iguales.

En la segunda mitad del siglo 20, el surgimiento de la investigación de operaciones, y el enorme influjo de la computadora electrónica en el modo de resolver problemas ha motivado la aparición de una línea diferente en los problemas combinatorios. El interés ha cambiado de confirmar la existencia de cierto arreglo, o de contar el número de arreglos que cumplan cierta condición hacia encontrar el mejor de los arreglos posibles, de acuerdo a un criterio predeterminado de selección. Este enfoque de encontrar un arreglo óptimo, se conoce como Optimización Combinatoria.

Un ejemplo típico de Optimización Combinatoria lo constituye el llamado 'problema del matrimonio', que se formula como sigue: "Dado un conjunto de m hombres y un conjunto de n mujeres, donde cada hombre tiene una lista de las mujeres a las que propondría matrimonio, ¿cuál es el máximo número de parejas que pueden formarse?".

El problema del matrimonio se formula de modo natural usando gráficas. En el ejemplo anterior no es difícil ver que la situación se puede modelar con una gráfica en la que existan m nodos representando a los hombres, n nodos representando a las mujeres, y un arco que una al nodo "i" con el nodo "j", siempre que la mujer "j" sea una de las elegidas por el hombre "i".

El ejemplo del matrimonio es un caso particular de un problema básico de optimización combinatoria, conocido como "problema de apareamiento". En un problema de apareamiento se tiene una gráfica con m nodos y n arcos, y se busca una colección de arcos (parejas de nodos) que satisfagan ciertos criterios de optimalidad.

Los problemas que se enuncian enseguida son ejemplos estrechamente relacionados con problemas de apareamiento.

PROBLEMA 1. En una empresa se tiene un conjunto E de empleados y un conjunto T de trabajos por asignar. El sindicato propone una asignación que ocupe a tantos empleados como sea posible, sujetándose a un sistema de antigüedad laboral, mientras que la empresa propone una asignación que designe los trabajos sujetándose a un sistema de prioridades en los mismos. ¿Existe alguna asignación de empleados a los trabajos que satisfaga tanto a la empresa como al sindicato?.

PROBLEMA 2. Un periódico sabe que en la Reunión Internacional de Legisladores se tienen programadas conferencias a ciertas horas y de duración conocida. Cuando una serie de conferencias C1,C2,...,Ck ocurren sin traslaparse, se tiene una cadena (y por tanto pueden ser cubiertas por un solo reportero). ¿Cuál es el número mínimo de reporteros que cubrirán todas las conferencias de la reunión?.

PROBLEMA 3. En una línea de producción que trabaja en serie hay n obreros que trabajan en n estaciones de la línea. Si se llama Tij a la velocidad con la que el obrero "i" ejecuta su tarea en la estación "j", es claro que la velocidad de producción de la línea está limitada por la velocidad del trabajador más lento. ¿Qué asignación de los obreros a las estaciones maximiza la velocidad de la línea de producción?.

En este trabajo se presenta la formulación y análisis de los modelos básicos de apareamiento. Se muestran los resultados teóricos más significativos y los algoritmos de solución.

La presentación del tema del apareamiento se hace de manera progresiva a lo largo de los próximos capítulos. En el Capítulo 1 se presentan los conceptos y definiciones de Teoría de Gráficas usados en el desarrollo del tema, así como una breve descripción de los modelos básicos de apareamiento. En el Capítulo 2 se tratan los modelos en los cuales la gráfica involucrada en la representación del problema es bipartita. El Capítulo 3 prosigue con problemas cuya representación gráfica no es bipartita. En el Capítulo 4 se muestran algunos programas de computadora escritos en Pascal que manejan ciertos problemas presentados en el texto. Al final se incluye un Apéndice con las demostraciones detalladas de algunos de los teoremas mencionados en los capítulos.

C A P I T U L O 1 .

PROBLEMAS BASICOS DE APAREAMIENTO

Los problemas de apareamiento que se resuelven en la práctica invariablemente son representados por una gráfica con nodos y arcos, donde los nodos simbolizan los elementos involucrados en el apareamiento y los arcos la posibilidad de formar pareja

El tema del apareamiento frecuentemente hace referencia a resultados propios de la Teoría de Gráficas. Esta última especialidad, relativamente reciente entre las ramas de las matemáticas aplicadas, no ha llegado todavía a presentar una notación y terminología única entre todos los autores. Debido a esto, en la primera sección de este capítulo se resumen los conceptos y definiciones de Teoría de Gráficas que se adoptarán en los capítulos próximos.

En la segunda sección se describen brevemente los modelos básicos en la teoría del apareamiento y se dan ejemplos de cada caso. Estos modelos resuelven problemas de apareamiento atendiendo a distintos criterios de optimización. Los modelos que se presentan son los siguientes:

1) Máximo apareamiento, 2) apareamiento Max-min, llamado también 'de cuello de botella', 3) apareamiento ponderado y 4) apareamiento de Gale-Shapley.

(1.1) PRELIMINARES.

Una gráfica G es una estructura formada por un conjunto finito N cuyos elementos son llamados nodos, y otro conjunto finito A de parejas de nodos, llamados arcos. La notación usada es: G=(N,A). Cuando los arcos de una gráfica G son parejas ordenadas, G es una gráfica dirigida, llamada también digráfica; cuando los arcos no son parejas ordenadas, se dice que G es una gráfica no-dirigida. En este trabajo se consideran modelos que usan solamente gráficas no-dirigidas.

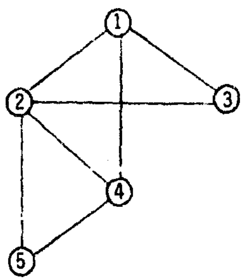
Si en la gráfica G existe el arco s=(k,m), se dice que los nodos "k" y "m" son adyacentes, mientras que el arco "s" es incidente tanto al nodo "k" como al "m". Del mismo modo se dice que ambos nodos - son incidentes al arco "s".

Las gráficas se representan frecuentemente por dibujos donde los nodos corresponden a puntos o pequeños círculos, y los arcos a líneas que unen los nodos. En la Figura 1.1, G es una gráfica donde se tiene N=(1,2,3,4) y A=((1,3), (1,4), (2,3), (2,4), (3,4)) mientras que H es una digráfica con N=(1,2,3) y A=((2,1), (2,3), (3,1)).

Figura 1.1

En una gráfica G, la CONTRACCION DE UN ARCO (i,j), denotada como G ctr ((i,j)), consiste en el reemplazo de los nodos 'i' y 'j' por un nuevo nodo 'k', añadiendo en la gráfica resultante un arco (k,m) por cada arco (i,m) o (j,m) de la gráfica original. La contracción de una gráfica puede resultar en gráficas con arcos múltiples entre sus nodos. A tal tipo de gráfica se le conoce como MULTIGRAFICA. En la Figura 1.2, la gráfica G' representa la contracción de G, luego de eliminar el arco (1,4).

G



G'= G ctr (1,4)

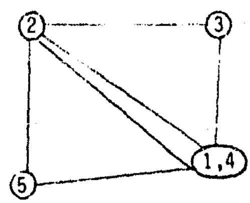


Figura 1.2

Cuando se trabajan gráficas con un número considerable de nodos y arcos deja de ser práctica la representación esquemática, siendo la representación en computadora el medio más común.

Algunas de las representaciones más usadas para almacenar una gráfica en una computadora son: a) MATRIZ DE INCIDENCIA, b) MATRIZ DE ADYACENCIA.

Dada una gráfica G con 'm' nodos y 'n' arcos, su MATRIZ DE INCIDENCIA A (llamada a veces matriz nodos-arcos) tiene 'm' filas (identificadas con los nodos) y 'n' columnas (identificadas con los arcos) y se encuentra definida a continuación.

$$A = \begin{bmatrix} a11 & a12 & . & . & . & a1k \\ a21 & a22 & . & . & . & a2k \\ . & & . & . & . & . \\ . & & . & . & . & . \\ am1 & am2 & . & . & . & amk \end{bmatrix}$$

$$\text{donde } aij = \begin{cases} 1 & \text{si el arco 'j' incide en el nodo 'i'} \\ 0 & \text{en otro caso.} \end{cases}$$

La matriz de incidencia de G en la Figura 1.1 es:

$$\begin{array}{c|ccccc} & (1,3) & (1,4) & (2,3) & (2,4) & (3,4) \\ \hline \end{array}$$

$$\begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

Para una gráfica G con 'n' nodos, la MATRIZ DE ADYACENCIA A (a veces llamada matriz nodos-nodos) tiene 'n' filas y 'n' columnas (ambas identificadas con los nodos) y se define como:

$$A = \begin{bmatrix} a11 & a12 & . \; . \; . & a1n \\ a21 & a22 & . \; . \; . & a2n \\ . & . & . \; . \; . & . \\ . & & . \; . \; . & . \\ an1 & an2 & . \; . \; . & ann \end{bmatrix}$$

$$\text{donde } aij = \begin{cases} 1 & \text{si los nodos 'j' e 'i' son adyacentes.} \\ 0 & \text{en otro caso.} \end{cases}$$

La matriz de adyacencia de G en la Figura 1.1 es:

$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix} \end{array}$$

Fácilmente se observa que por construcción, la matriz de adyacencia de una gráfica es siempre simétrica.

En una gráfica cualquiera G, al número de arcos que inciden en un nodo 'i' se le llama GRADO DE 'i', y se denota por d(i). De la construcción de la matriz de incidencia, resulta claro que la suma de los elementos a lo largo de la fila 'i' (que representa al nodo i) es igual al valor d(i). Es decir:

$$d(i) = \sum_{k} M_{ik}$$

Si se simboliza por: |A| al número de arcos de la gráfica G, y tomando en cuenta que cada arco une a dos nodos de la misma, no es difícil ver que la suma de los grados de todos los nodos de G viene a igualar al doble de |A|. O sea:

$$\sum_{i \text{ en } N} d(i) = 2|A|$$

La proposición siguiente muestra una conocida propiedad de los nodos de grado impar de una gráfica:

PROPOSICION 1.1a. En una gráfica finita G=(N,A) el número de vértices de grado impar es par.

El concepto de conexidad en una gráfica G está estrechamente relacionado con la idea intuitiva de poder 'unir' dos nodos cualesquiera de ella por medio de una trayectoria formada con arcos. La formalización de la idea es como sigue: dados dos nodos 'p' y 'q' de G=(N,A), una TRAYECTORIA entre 'p' y 'q' es una sucesión de arcos de la forma: $(p,j1)$, $(j1,j2)$, ..,(jk,q) con $js < jt$ para $s < t$. La trayectoria se llama ABIERTA si 'p' es distinto a 'q', y CERRADA en caso contrario. A una trayectoria que tiene al menos un arco y solamente un nodo repetido se le llama CICLO. Una gráfica sin ciclos es llamada ACICLICA.

Dos nodos de G se llaman CONECTADOS si existe una trayectoria que los una. Cuando en una gráfica cualquier par de nodos es conectado, se dice que la gráfica es CONEXA.

Si una gráfica es conexa y carece de ciclos, forma lo que se conoce como ARBOL. La simplicidad de la estructura de los árboles ha permitido una gran variedad de aplicaciones en matemáticas discretas y en las ciencias de la computación. Una caracterización de lo que es un árbol se muestra en la siguiente proposición:

PROPOSICION 1.1b. Si G es una gráfica con 'n' nodos, las condiciones siguientes son equivalentes:

a) G es un árbol.
b) Cualquier par de nodos de G está unido por un trayectoria única.
c) G no tiene ciclos, pero exactamente uno se forma al agregar un arco.
d) G es conexa, pero deja de serlo si se suprime algún arco.

e) G no tiene ciclos y tiene 'n-1' arcos.
f) G es conexa y tiene 'n-1' arcos.

Una gráfica G=(N,A) es llamada BIPARTITA si se puede separar al conjunto de nodos N en dos subconjuntos ajenos S y T de manera que cualquier arco de A necesariamente conecte un nodo en S con uno de T. La notación G=(N,A) para este caso suele escribirse como: G=(S,T,A).

Si en una gráfica bipartita tomamos una trayectoria y la recorremos es fácil ver que al avanzar por los arcos se tocan nodos que alternativamente pertenecen a S y a T. De tal observación se puede concluir que si una gráfica G es bipartita, entonces cualquier ciclo en ella necesariamente tiene un número par de arcos y un número par de nodos (longitud par del ciclo). La idea recíproca también es cierta, y el resultado se resume en la siguiente proposición, que se demuestra en el apéndice.

PROPOSICION 1.1c. Una gráfica G=(N,A) es bipartita si y sólo si carece de ciclos de longitud impar.

Supongamos ahora que G=(S,T,A) es una gráfica bipartita, donde S={1,3,6,7} y T={2,4,5,8,9}. Si numeramos los nodos de G como: n1=1, n2=3, n3=6, n4=7, n5=2, n6=4, n7=5, n8=8 y n9=9, la matriz de adyacencia de G se ve como sigue:

$$A = \begin{array}{c|ccccccccc|} & n1 & n2 & n3 & n4 & n5 & n6 & n7 & n8 & n9 \\ \hline n1 & 0 & 0 & 0 & 0 & a15 & a16 & a17 & a18 & a19 \\ n2 & 0 & 0 & 0 & 0 & a25 & a26 & a27 & a28 & a29 \\ n3 & 0 & 0 & 0 & 0 & a35 & a36 & a37 & a38 & a39 \\ n4 & 0 & 0 & 0 & 0 & a45 & a46 & a47 & a48 & a49 \\ n5 & a51 & a52 & a53 & a54 & 0 & 0 & 0 & 0 & 0 \\ n6 & a61 & a62 & a63 & a64 & 0 & 0 & 0 & 0 & 0 \\ n7 & a71 & a72 & a73 & a74 & 0 & 0 & 0 & 0 & 0 \\ n8 & a81 & a82 & a83 & a84 & 0 & 0 & 0 & 0 & 0 \\ n9 & a91 & a92 & a93 & a94 & 0 & 0 & 0 & 0 & 0 \end{array}$$

La forma especial que adopta esta matriz de adyacencia y el hecho de que sea una matriz simétrica se resumen en la siguiente:

PROPOSICION 1.1d. G es una gráfica bipartita si y sólo si sus nodos pueden enumerarse de manera que la matriz de adyacencia asociada sea de la forma:

$$A = \left[\begin{array}{c|c} 0 & B \\ \hline B^t & 0 \end{array}\right]$$

Donde B es una submatriz de A, en general rectangular, cuyos elementos pueden ser cero o uno.

(1.2) DESCRIPCION Y EJEMPLOS.

(1.2.1) Modelos Básicos.

La palabra apareamiento significa literalmente formar parejas. Si tenemos un conjunto finito de elementos con los que se pueden formar parejas, se puede representar la situación con una gráfica que tenga un nodo por cada elemento, y donde un arco entre dos nodos signifique que los elementos representados por tales nodos pueden formar pareja.

De modo formal, si se tiene una gráfica $G=(N,A)$, un APAREAMIENTO en G es un subconjunto $S \subset A$ de arcos en donde no hay dos de ellos que incidan en un nodo común.

En la Figura 1.3 los arcos con línea ondulada representan un apareamiento. En G, el apareamiento no es el más grande posible, en H sí lo es.

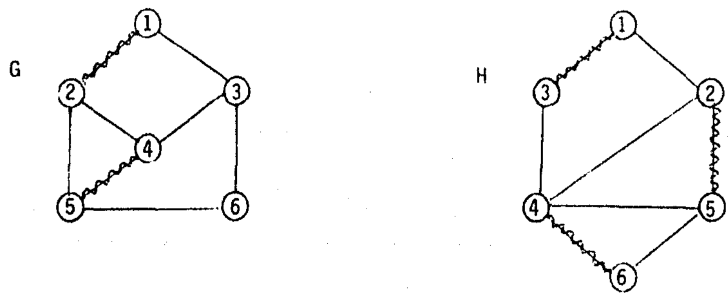


Figura 1.3

En los problemas de apareamiento en gráficas, la idea más general consiste en encontrar subconjuntos de arcos que cumplen ciertos criterios de optimización, donde los parámetros involucrados pueden ser: el número de arcos en la solución, el número de arcos que pueden incidir en un nodo o el peso asociado a los arcos, si es que lo tienen.

En esta sección se describen los problemas tipo fundamentales en el tema del apareamiento, mostrándose algunos ejemplos. Ellos son:

1) MAXIMO APAREAMIENTO.- En este problema, dada una gráfica G con un número finito de nodos y arcos, se desea determinar el apareamiento de mayor cardinalidad (número de arcos) posible.

Ejemplo 1.1 El centro de lenguas de una universidad tiene una lista de 7 profesores para cubrir 10 trabajos de traducción. Los idiomas requeridos y las habilidades de los profesores se muestran en la siguiente tabla:

Idiomas: I) inglés, II) francés, III) alemán, IV) polaco, V) ruso, VI) chino, VII) árabe, VIII) turco, IX) griego, X) búlgaro.

Prof.: \ Idioma:	I	II	III	IV	V	VI	VII	VIII	IX	X
1	sí	sí	no	no	no	no	no	no	no	sí
2	no	no	sí	sí	sí	no	no	no	no	no
3	sí	no	no	no	no	sí	sí	no	no	no
4	no	no	no	no	no	no	no	sí	no	no
5	sí	no	no	sí	no	no	sí	no	sí	no
6	no	sí	no	no	sí	sí	no	no	no	no
7	sí	no	sí	sí	no	no	no	no	sí	sí

¿Cúal es el máximo número de traducciones de que se puede encargar el centro de lenguas?.

2) APAREAMIENTO MAX-MIN o 'DE CUELLO DE BOTELLA'.- El ejemplo de motivación más común para este problema está en el siguiente planteamiento: En un taller industrial se tienen 'n' trabajadores para ser asignados a 'n' estaciones de trabajo de una línea de producción en serie. Puesto que los obreros tienen distintas velocidades de trabajo en las diferentes estaciones de la línea, es claro que para cada asignación habrá al menos una estación que sea la más lenta de todas (el 'cuello de botella'), y que determinará la velocidad de producción de toda la línea. El problema a resolver entonces es encontrar la asignación (apareamiento obreros-estaciones) donde la más lenta de las estaciones sea lo más rápida que se pueda.

En general, dada una gráfica bipartita G=(S,T,A) con número finito de nodos y arcos, donde cada arco de G tiene asociado un peso (positivo, negativo o cero), el problema es encontrar el apareamiento de máxima cardinalidad cuyo peso mínimo sea lo mayor posible.

Ejemplo 1.2 El laboratorio fotográfico de una empresa periodística procesa los rollos de película que recibe en 6 etapas como sigue:

1) Registro del rollo y control de trabajo.
2) Revelado del negativo.
3) Secado y selección de tomas.
4) Impresión de fotografías
5) Secado de impresiones y ensobretado.
6) Registro contable y distribución.

Para ello cuenta con 6 empleados que pueden ejecutar cualquier etapa indistintamente, siendo su velocidad de trabajo (en rollos/hora) la indicada en la siguiente tabla:

Empleado: \ Etapa--->	1	2	3	4	5	6
A	12	6	10	5	6	15
B	15	4	10	3	5	10
C	15	6	6	4	6	15
D	20	5	5	3	3.75	6
E	10	6	10	4	4	7.5
F	12	4	6	2.5	6	10

¿Cuál asignación de los trabajadores a las etapas de proceso maximiza la velocidad de producción del laboratorio?.

3) APAREAMIENTO PONDERADO.- En este problema se tiene una gráfica G=(N,A) donde cada arco está ponderado por un peso que puede ser positivo, negativo o cero. La intención es hallar un apareamiento en donde la suma de los pesos de sus arcos sea máxima.

En el caso de que G sea bipartita (G=(S,T,A)), el problema del apareamiento ponderado es fácilmente reconocido como un problema de asignación, pues se aplica de modo natural a la búsqueda de la asignación óptima de trabajadores (elementos de S) a tareas (elementos de T) cuando se conoce la eficiencia del obrero 'i' en la tarea 'j', que se denota por el peso wij.

Ejemplo 1.3 Un servicio de proceso de datos tiene 4 impresoras de líneas que pueden conectarse a cualquiera de 5 microcomputadoras para emitir reportes a usuarios. Las velocidades en líneas por minuto de cada impresora según la microcomputadora con la que se use están dadas en la siguiente tabla:

Micro--->		1	2	3	4	5
Impresora:						
	A	300	350	320	300	280
	B	315	350	300	290	300
	C	350	340	340	300	290
	D	275	300	350	300	300

¿Cómo deben asignarse las impresoras a las microcomputadoras para maximizar el número total de líneas impresas por minuto?.

4) APAREAMIENTO DE GALE-SHAPLEY.- Este modelo permite manejar la característica de estabilidad en las parejas formadas que aparece en algunos problemas de apareamiento, donde los elementos involucrados tienen una preferencia en la formación de las parejas. Por ejemplo, supongamos que H es un conjunto de 'n' hombres y M un conjunto de 'n' mujeres, y que los hombres tienen una relación de preferencia hacia las mujeres para formar matrimonio. Entonces, para cualquier apareamiento dado, siempre se podrá preguntar si habrá un hombre y una mujer que no forman pareja, pero que se prefieren más el uno al otro que al cónyuge asignado por el apareamiento. Cuando se da este caso, el conjunto de parejas propuesto no es estable, pues al menos habrá un hombre y una mujer dispuestos a dejar a sus respectivos cónyuges para formar pareja entre sí.

Cuando un apareamiento es estable (nadie deseoso de sustituír a su pareja encuentra respuesta) y además ocurre que cualquier hombre de H está por lo menos tan satisfecho en él como con cualquier otro apareamiento estable, se tiene un apareamiento óptimo (para los hombres). Un teorema de Gale y Shapley muestra una prueba constructiva de tal óptimo, independiente de que se tengan pesos asociados a los arcos.

La asignación de solicitantes de empleo a puestos, de médicos pasantes a hospitales o de profesores a grupos universitarios son algunos ejemplos donde los elementos involucrados en el apareamiento manifiestan una preferencia en las posibles asignaciones, y que se pueden tratar adecuadamente con el modelo de Gale-Shapley.

(1.2.2) Otros problemas equivalentes.

Existen diversos problemas de Optimización Combinatoria que pueden ser planteados de manera equivalente a alguno de los problemas que se han mencionado anteriormente. Un par de ejemplos se citan enseguida.

5) SISTEMAS DE REPRESENTANTES DISTINTOS.- Dado un conjunto finito T, y una colección de subconjuntos de T: (T1,T2,...,Tk), se llama SISTEMA DE REPRESENTANTES DISTINTOS de T a un subconjunto de elemen-

tos de T:(t1,t2,...,tk) distintos dos a dos, y que cumplen la condición $t_i \in T_i$, para $i=1,2,...,k$.

El problema de los representantes distintos es que, conociendo T y la colección de subconjuntos T1,T2,...,Tk se construya un sistema de representantes distintos para T.

Ejemplo 1.4 Un despacho de nueve asesores industriales atiende a 7 empresas distintas. Enumerando a los asesores del 1 al 9, la forma en que están asignados a las empresas es como sigue:

Empresa # 1: (2,3)
Empresa # 2: (2,5)
Empresa # 3: (1,6)
Empresa # 4: (6)
Empresa # 5: (3,4,7)
Empresa # 6: (1,6)
Empresa # 7: (5,8,9)

Para la reunión anual de evaluación del despacho, se desea formar un grupo de 7 asesores, representando c/u de ellos a una empresa diferente. ¿Cómo se integra el grupo deseado?.

La gráfica que se muestra enseguida modela el problema. Los nodos del lado izquierdo simbolizan a las empresas y los del lado derecho a los asesores. Un arco (i,j) indica que la empresa 'i' tiene asignado al asesor 'j'. A partir de esta gráfica, el problema se puede resolver encontrando el máximo apareamiento en ella. Si el máximo apareamiento contiene menos de 7 arcos, el problema de los representantes no tiene solución, y entonces alguno de los asesores deberá representar a más de una empresa en la reunión de evaluación.

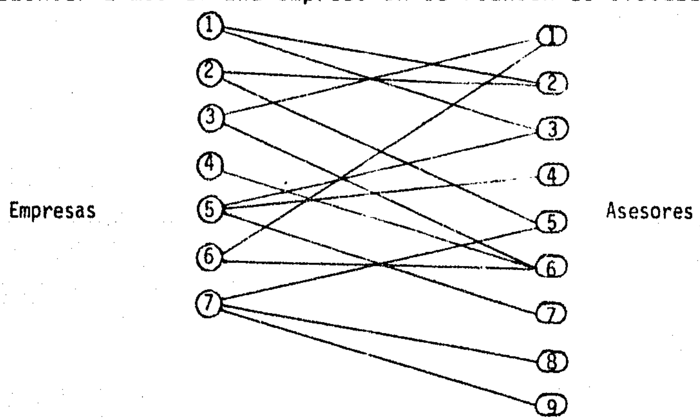


Figura 1.4

6) DESCOMPOSICION EN CADENAS. En algunos problemas que se manejan con gráficas, los elementos representados por los nodos pueden ser ordenados de modo estricto (a<b) de manera que se refleje un 'orden de precedencia' entre ellos. Por supuesto, existe la posibilidad de que dos elementos en particular no puedan compararse.

La formalización de la idea anterior es como sigue: para un conjunto finito N, un ORDEN PARCIAL ESTRICTO es una relación definida en el producto cartesiano NxN que es transitiva y antisimétrica. Un ejemplo típico de esto es la relación '<' en los números reales.

Dos elementos i,j de N se llaman COMPARABLES si ocurre i<j o j<i, en caso contrario se llaman NO-COMPARABLES. Una CADENA en N es un subconjunto de elementos de N comparables dos a dos, y claramente tal subconjunto puede escribirse en forma de orden ascendente como: n1 < n2 < ... < nk. Una descomposición en cadenas de N es una colección de cadenas disjuntas cuya unión forma N.

El problema de la DESCOMPOSICION EN CADENAS de un conjunto N, consiste en encontrar aquella descomposición de N cuya cardinalidad (número de cadenas) sea mínima.

Ejemplo 1.5 Un periódico planea cubrir las conferencias de la Reunión Internacional de Legisladores a celebrar en cierta fecha. Para tal día se ofrecen 10 conferencias de duración y hora de inicio como se muestra a continuación:

Conferencia:	1	2	3	4	5	6	7	8	9	10
Inicio:	9:00	8:00	8:00	9:00	9:30	10:30	8:30	8:30	9:00	8:30
Duración (horas)	1	1	1.5	1.5	1.5	1	1	1.5	1.5	1

¿Cuál es el mínimo número de reporteros que pueden cubrir el total de las conferencias?

No es difícil ver que las conferencias se pueden ordenar de manera que 'a < b' signifique que la conferencia 'a' termina antes de que la conferencia 'b' comience. La gráfica a continuación modela el orden parcial estricto que existe entre las conferencias.
Cada nodo representa la conferencia con la que está numerado, y si se puede alcanzar el nodo 'b' bajando desde el nodo 'a', se tiene la condición a < b.

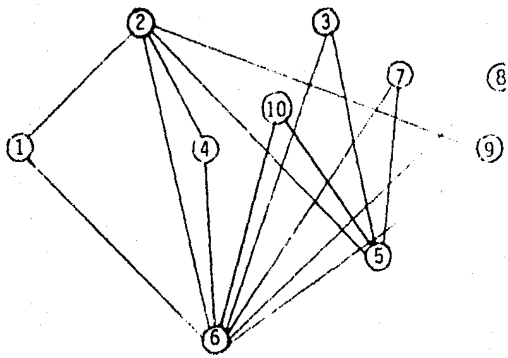


Figura 1.5

Cada cadena de la gráfica anterior representa una secuencia de conferencias sucesivas que no se traslapan, y que pueden ser cubiertas por un reportero.

El problema se puede reducir a un caso de máximo apareamiento como sigue: sea S={1,2,3,...,10} el conjunto de las conferencias, y G=(S,T,A) una gráfica bipartita donde T=S, y donde un arco (i,j) represente la condición (conferencia i) < (conferencia j). La gráfica G para el ejemplo se muestra enseguida:

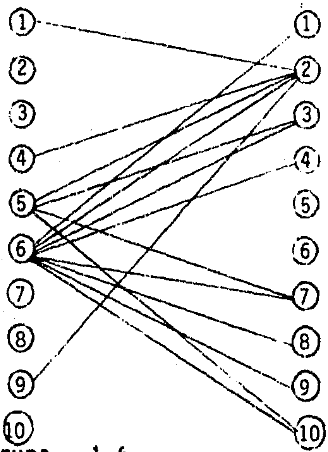


Figura 1.6

Encontrar un apareamiento máximo en la gráfica de la figura anterior, equivale a encontrar un subconjunto de pares de elementos de S que son comparables y de modo que nigún elemento aparece más de una vez como primer elemento de un par ni más de una vez como segundo elemento de un par. Identificando los elementos comunes en los pares encontrados, se 'unen' los pares y se forman cadenas disjuntas.
Por ejemplo, si en el apareamiento maximal para la Figura 1.6 aparecen los arcos (6,4) y (4,2) se puede construir en la Figura 1.7 la cadena: 6 < 4 < 2. Los elementos que no sean cubiertos por los arcos de el apareamiento maximal se consideran como cadenas de longitud unitaria, y así se tiene la descomposición en cadenas de S que se buscaba.

C A P I T U L O 2.

APAREAMIENTO EN GRAFICAS BIPARTITAS.

Las primeras investigaciones hechas sobre problemas de apareamiento trataron sólo con gráficas bipartitas. Algoritmos eficientes para encontrar un apareamiento máximo en este tipo de gráficas ya estaban elaborados en la segunda mitad de la década de los 50, por autores como Ford y Fulkerson.

El tratamiento adecuado del apareamiento en gráficas generales tuvo su fundamento en los resultados de las investigaciones hechas para el caso bipartita, y algoritmos eficaces en gráficas generales se conocieron en la segunda mitad de la década de los 60, de autores como Edmonds y Balinski.

Estos antecedentes, aunados a la relativa facilidad con que se manejan los algoritmos del caso bipartita motivan a tratar primeramente éste caso, preparando terreno para abordar el apareamiento en gráficas generales con mayor naturalidad.

Los dos primeros modelos expuestos en el capitulo son: Máximo apareamiento y Apareamiento ponderado. Después se muestran dos modelos que necesariamente usan gráficas bipartitas: Apareamiento maxmin o 'cuello de botella' y el Apareamiento de Gale-Shapley. Se dan además un par de aplicaciones del máximo apareamiento a dos problemas de tipo combinatorio: el problema de construír Sistemas de representantes distintos y el problema de la Descomposición en cadenas de un conjunto parcialmente ordenado.

Cada modelo se presenta acompañado con un algoritmo de solución y un breve ejemplo aclaratorio.

(2.1) DEFINICIONES Y RESULTADOS BASICOS.

Dada una gráfica $G=(S,T,A)$ un APAREAMIENTO M en G es un subconjunto de arcos $M \subset A$ con la propiedad de que no existen dos arcos en M que incidan en el mismo nodo. Dado un apareamiento M en $G=(S,T,A)$ si el nodo 'k' $\in S \cup T$ es tocado por un arco de M, se dice que 'k' es un NODO SATURADO por M; en caso contrario, a 'k' se le llama NODO EXPUESTO. Una trayectoria en G es llamada TRAYECTORIA ALTERNANTE relativa a M, si se encuentra formada por arcos que alternativamente están en M y en su complemento: $A \setminus M$. En el ejemplo de la Figura 2.1 un apareamiento M está indicado por líneas onduladas. Los nodos cubiertos son: 1, 2, 3 y 6; los expuestos son: 4, 5, y 7. La trayectoria ((5,2),(2,1),(1,3),(3,6)) es alternante relativa a M, mientras que ((5,7),(7,6),(6,3),(3,5)) no lo es.

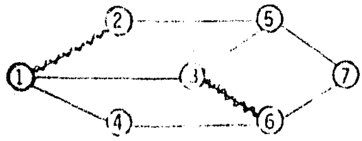


Figura 2.1

Cuando un apareamiento M en una gráfica no deja nodos expuestos en la gráfica, el apareamiento se llama COMPLETO o PERFECTO. Por supuesto, si una gráfica posee un apareamiento de este tipo, debe tener un número par de nodos.

Una condición suficiente para que exista un apareamiento completo en una gráfica se establece en la siguiente proposición, cuya demostración aparece en el Apéndice.

PROPOSICION 2.1a. Si G es una gráfica con un número 'n' par de nodos, y cada uno de ellos tiene grado mayor o igual a: n/2, entonces existe un apareamiento completo para G.

(2.1.1) TEOREMA DE MENDELSOHN-DULMAGE.

El siguiente teorema establece la manera de 'combinar' dos apareamientos distintos en una gráfica bipartita para producir un nuevo apareamiento que cumple con ciertas condiciones.

TEOREMA 2.1b. (Mendelsohn-Dulmage). Sean G=(S,T,A) una gráfica bipartita, y X1, X2 dos apareamientos en G. Entonces, existe un apareamiento $X \subseteq X1 \cup X2$ tal que X cubre todos los nodos de S cubiertos por X1, y todos los nodos de T cubiertos por X2.

Demostración.

Tomando la diferencia simétrica $X1 \nabla X2 = (X1 \cup X2) - (X1 \cap X2)$, y observando que al ser G bipartita los ciclos posibles son de longitud par, y además que cualquier trayectoria en $X1 \nabla X2$ sólo puede: 1) empezar y terminar con arcos de X1, o 2) empezar y terminar con arcos de X2, o 3) empezar con arco de X1 y terminar con arco de X2, o 4) empezar con arco de X2 y terminar con arco de X1, no es difícil ver que los tipos de trayectorias y ciclos que pueden aparecer en $X1 \nabla X2$ son los mostrados en la Figura 2.2.

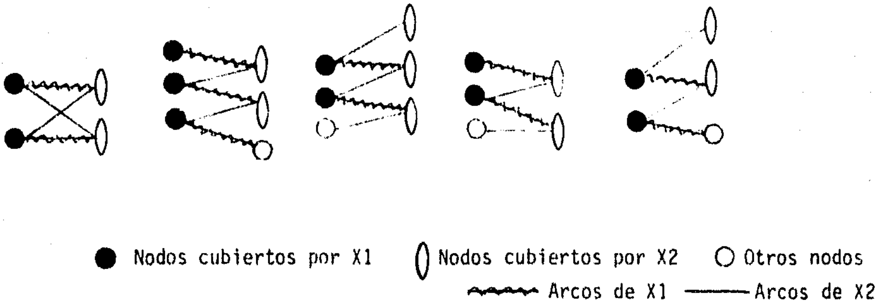


Figura 2.2

En cualquiera de los casos que aparecen en la Figura 2.2 se puede encontrar un apareamiento $Y \subset X1 \nabla X2$ tal que Y cubra todo nodo de S cubierto por $(X1-X2)$ y todo nodo de T cubierto por $(X2-X1)$. Así, $X = Y \cup (X1 \cap X2)$ es el apareamiento que se busca.

Una aplicación de este resultado se tiene en el Problema 1 que aparece en la Introducción. Sea X1 la asignación (apareamiento) propuesta por el sindicato, y X2 la que propone la empresa, entonces es posible encontrar una asignación X que satisfaga a ambas partes.

Una consecuencia inmediata del teorema anterior son los siguientes resultados:

TEOREMA 2.1c. Si $G=(S,T,A)$ es gráfica bipartita, y X es un apareamiento en G, entonces existe un apareamiento de máxima cardinalidad, Y, que cubre todos los nodos de G cubiertos por X.

Demostración.

Si M es un apareamiento de máxima cardinalidad en G, el teorema 2.1b garantiza la existencia de un apareamiento Y en G que cumple:

a) Y cubre todos los nodos de S cubiertos por X.
b) Y cubre todos los nodos de T cubiertos por M.

En particular, de la condición b) se concluye que $|M| = |Y|$, por lo que Y también es de máxima cardinalidad. Recurriendo otra vez al teorema 2.1b, se puede afirmar que existe un apareamiento Z en G que cumple:

c) Z cubre todos los nodos de S cubiertos por Y (y por tanto cubre todo nodo de S que cubra X).
d) Z cubre todos los nodos de T cubiertos por X.

de c) resulta claro que: $|Z| = |Y| = |M|$, con lo que Z es un apareamiento de máxima cardinalidad que cubre todos los nodos de G que haya cubierto X.

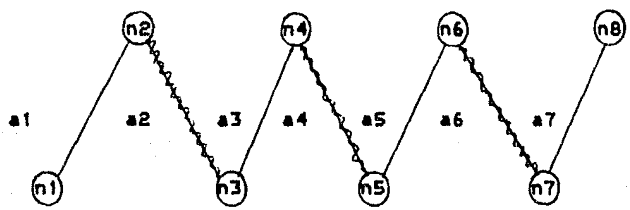
Si en una gráfica bipartita G=(S,T,A) se elige un nodo 'k' cualquiera que tenga grado d(k)$\geqslant$1, junto con cualquiera de sus arcos incidentes, se tiene un apareamiento de un solo arco. Del teorema 2.1c es posible entonces concluir el siguiente:

COROLARIO 2.1d. Si G=(S,T,A) es una gráfica bipartita, entonces para cualquier nodo 'k' con grado d(k)$\geqslant$1 en G existe un apareamiento de máxima cardinalidad que cubre a 'k'.

Siguiendo con el ejemplo del Problema 1 de la Introducción, si suponemos que el gerente de la empresa ha encontrado una asignación de empleados a trabajos que sea factible, del teorema 2.1c se puede afirmar que existe una asignación (apareamiento) de máxima cardinali- que corresponde, por supuesto, a un plan de uso total de la capacidad productiva, y en la cual todos los empleados y trabajos asignados originalmente por el gerente permanecen sin cambio.

(2.2) MAXIMO APAREAMIENTO.

El problema del máximo apareamiento en una gráfica G consiste en encontrar un apareamiento M cuya cardinalidad sea lo mayor posible.
El procedimiento para encontrar un apareamiento máximo se basa en la siguiente idea, que es sencilla y eficiente: -
Sean G una gráfica bipartita y X un apareamiento en G que no es de máxima cardinalidad. Si P es una trayectoria alternante relativa a X que une dos nodos expuestos, P debe ser del tipo de trayectoria mostrada en la Figura 2.3, donde los arcos ondulados son arcos del apareamiento X.



P=(a1,a2,a3,a4,a5,a6,a7)   n1 y n8 son nodos expuestos de X.
a2,a4,a6 son arcos de X.

Figura 2.3

Escribiendo '1' cuando un arco pertenezca al apareamiento X y '0' en caso contrario, la sucesión de arcos que forman la trayectoria P de la Figura 2.3 sería: 0101010. Escribiendo el 'complemento a uno' (0 cambia por 1 y viceversa) de tal sucesión resulta: 1010101 , lo cual sugiere que si los arcos a1, a3, a5 y a7 se incluyen en el apareamiento X sustituyendo a los arcos originales a2, a4 y a6, se obtiene otro apareamiento que contiene un arco más que el apareamiento original X.

Para formalizar la idea, se define una TRAYECTORIA AUMENTANTE relativa a un apareamiento X, como una trayectoria alternante que tiene por nodos inicial y final a nodos expuestos por X, y se establece la siguiente:

PROPOSICION 2.2a. Si G es una gráfica bipartita, X es un apareamiento en G, y P es una trayectoria aumentante relativa a X, entonces $Y = X \nabla P = (X - P) \cup (P - X)$ es un apareamiento de cardinalidad: $|Y| = |X| + 1$.

Demostración.

Lo primero que se mostrará es que no hay dos arcos de $X \nabla P$ compartiendo un nodo común, de modo que $X \nabla P$ es un apareamiento en G.

Suponiendo que a,b son 2 arcos de $X \nabla P$ que comparten un mismo nodo, hay tres casos:

1) $a,b \in X-P$, lo cual no ocurre, pues equivale a tener dos arcos del apareamiento X incidentes en un mismo nodo.

2) $a,b \in P-X$, que significa tener en la trayectoria P dos arcos que no pertenecen a X incidentes en el mismo nodo, lo cual contradice que P sea trayectoria alternante.

3) $a \in X-P$, $b \in P-X$. Sea 'n' el nodo común en que inciden los arcos a y b. Entonces, como el arco a (apareado y fuera de P) incide en 'n', dicho nodo no puede ser expuesto relativo a X.
Por otra parte, al incidir el arco b (en P y no apareado) en el nodo 'n' no expuesto, debe existir otro arco $c \in (X-P)$ apareado, que forme parte de P y que incida en 'n'. De esta manera, se tendrían los arcos $a,c \in X$ incidiendo en el nodo común 'n', lo que resulta contradictorio.

De la consideración de los casos anteriores, resulta que $X \nabla P$ es un apareamiento.

Finalmente, ya que P es trayectoria aumentante, su cardinalidad debe ser un número impar: $|P| = 2k-1$, para algún k=1,2,3,..., teniendo k-1 arcos pertenecientes a X y k arcos en P-X. Entonces:

$|Y| = |(X-P) \cup (P-X)|$ y como $(X-P) \cap (P-X) = \emptyset$, se tiene:

$$|Y| = |X-P| + |P-X| = |X|-(k-1) + k = |X| + 1.$$

La Figura 2.4 ilustra el contenido de la Proposición 2.2a. Los arcos ondulados indican el apareamiento.

Si en una gráfica G se tiene un apareamiento X, y se puede encontrar una trayectoria aumentante, de acuerdo a la proposición anterior

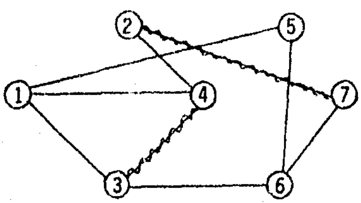
X no es de máxima cardinalidad. La idea recíproca también es cierta, y este resultado debido a Claude Berge se resume en la siguiente proposición:

PROPOSICION 2.2b. (C.Berge, 1957) Un apareamiento X en una gráfica G es de máxima cardinalidad si y sólo si no existen trayectorias aumentantes en G respecto a X.

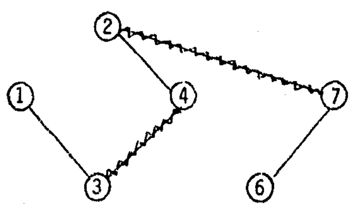
Demostración.

En la dirección 'sólo si': suponiendo que X es de máxima cardinalidad, no puede existir en G ninguna trayectoria aumentante P, pues de ser así, $Y = X \nabla P$ sería un apareamiento de cardinalidad mayor a X.

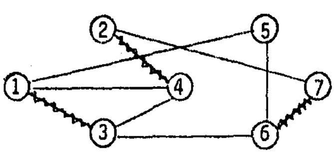
En la dirección 'si': suponiendo que no hay trayectorias aumentantes en G respecto a X, pero que existe un apareamiento Y tal que: - $|Y| > |X|$. Si se considera al conjunto de arcos $Y \nabla X$, dado que Y y X son apareamientos, se tendrá una subgráfica de G donde los nodos tienen a lo más grado 2. Si un nodo tiene grado dos, uno de sus arcos incidentes pertenece a X y el otro a Y. En este caso, los ciclos de la subgráfica $X \nabla Y$ son de longitud par (pues G bipartita) y tienen igual número de arcos de X que de Y; mientras que las trayectorias tienen arcos que alternadamente pertenecen a Y y a X, siendo por tanto trayectorias alternantes respecto a X. Finalmente, como $|Y| > |X|$ debe haber al menos una trayectoria 'P' alternante respecto a X, que tenga más arcos de Y que de X, y que inicie y termine con arcos de Y. Esta trayectoria 'P' es aumentante respecto a X, lo que contradice la hipótesis original. Por tal razón no puede existir un apareamiento Y con cardinalidad mayor a X.



apareamiento X.



trayectoria aumentante P.



apareamiento $Y = X \nabla P$

Figura 2.4

(2.3) METODOS DE SOLUCION Y DUALIDAD

Para resolver el problema de máximo apareamiento en una gráfica G, existen por lo menos tres métodos:

1) Formular el problema como programa lineal.
2) Formular el problema como un caso de flujo máximo.
3) Usar la búsqueda repetida de trayectorias aumentantes que permitan hacer crecer un apareamiento M ya dado en G, hasta llegar al punto en que no existan más cadenas aumentantes.

El primer método, expuesto brevemente más adelante en la sección, obliga a usar la matriz de incidencia de G, lo que puede significar requerimientos considerables de memoria en una computadora. Por otro lado, este enfoque plantea rápidamente la existencia de un problema dual del máximo apareamiento: el PROBLEMA DEL RECUBRIMIENTO MINIMO DE NODOS de G, que consiste en encontrar el más pequeño subconjunto L de nodos con la propiedad de que cualquier arco de G incide en al menos un nodo de L. Este problema dual facilita el tratamiento del Teorema de König-Egerváry al final de la sección.

El segundo método usa el eficiente algoritmo de solución para el flujo máximo, pero requiere de agregar un nodo fuente y otro sumidero a la gráfica, así como arcos desde estos nuevos nodos hacia los nodos originales.

El tercer método es consecuencia directa de las proposiciones 2.2a y 2.2b, y es la esencia de los algoritmos de solución que se muestran en la sección.

El algoritmo que se describe enseguida usa un proceso de etiquetado en los nodos para encontrar trayectorias alternantes. Antes de comenzar su descripción conviene establecer algunas definiciones.

Si G=(S,T,A) es una gráfica bipartita y M es un apareamiento en G, se llama ARBOL ALTERNANTE relativo a M a un árbol que cumple:
1) El árbol contiene exactamente un nodo expuesto de S, al que se llama nodo raíz. 2) Toda trayectoria entre la raíz y cualquier otro nodo en el árbol es trayectoria alternante respecto a M.

El algoritmo inicia con un apareamiento factible, que puede ser vacío. Cada nodo expuesto en S se hace la raíz de un árbol alternante, y arcos y nodos se añaden a los árboles usando un proceso de etiquetado. A medida que el algoritmo avanza, uno de dos eventos ocurre: O se añade un nodo expuesto en T a alguno de los árboles, o no se puede agregar más nodos o arcos a ninguno de los árboles. En el primer caso, el apareamiento es aumentado en un arco más, y el proceso de construcción de árboles alternantes se repite respecto del nuevo apareamiento; en el segundo caso se dice que los árboles son 'húngaros', o que forman un 'bosque húngaro', y se pueden usar para construír la solución dual del problema de máximo apareamiento, con lo cual el algoritmo se detiene.

ALGORITMO 2.3.1. MAXIMO APAREAMIENTO BIPARTITA

Paso 0). INICIO. Se tiene una gráfica bipartita G=(S,T,A) donde $|S| \leq |T|$, y M es un apareamiento que puede ser vacío. No hay nodos con etiqueta.

Paso 1) ETIQUETADO.
(1.0) Si todos los nodos de S son saturados, el apareamiento M es de máxima cardinalidad, y representan una solución para el problema dual. Alto.
En caso contrario, coloque la etiqueta '0' a cada nodo expuesto en S.
(1.1) Si todas las etiquetas han sido examinadas ir al Paso 3. En caso contrario, encuentre un nodo 'k' con etiqueta sin examinar.
Si $k \in S$ ir al Paso 1.2; si $k \in T$ ir al Paso 1.3.
(1.2) Examine la etiqueta en el nodo 'k' (k S) como sigue Para cada arco $(k,j) \notin M$ incidente al nodo k, ponga al nodo j la etiqueta "k", a menos que el nodo j ya esté etiquetado. Regrese a 1.1.
(1.3) Examine la etiqueta en el nodo 'k' ($k \in T$) como sigue: Si el nodo k es expuesto, ir al Paso 2. En caso contrario, identifique al único arco $(k,j) \in M$ incidente al nodo k y ponga al nodo j la etiqueta "k". Regrese a 1.1.

Paso 2). AUMENTO Se ha encontrado una trayectoria aumentante que termina en el nodo k (identificado en Paso 1.3). Los nodos que preceden al k en la trayectoria se encuentran por un rastreo regresivo sobre las etiquetas Así, si la etiqueta en el nodo k es "j", el penúltimo nodo de la trayectoria es j. Si la etiqueta en el nodo j es "m", el antepenúltimo nodo de la trayectoria es m, y así sucesivamente.
El nodo inicial en la trayectoria tiene etiqueta "0". Aumente el apareamiento M agregándole todos los arcos en la trayectoria aumentante que no pertenezcan a M y eliminando de M los que sí estén en la trayectoria. Borre todas las etiquetas en todos los nodos. Regrese al Paso 1.0.

Paso 3). ETIQUETADO HUNGARO. El etiquetado en los árboles es Húngaro, no hay trayectorias aumentantes y el apareamiento M es de máxima cardinalidad. Sea $L \subsetneq S \cup T$ el conjunto de los nodos con etiqueta. Entonces $C=(S-L) \cup (T \cap L)$ es la solución al problema dual del máximo apareamiento. Alto.

En el ejemplo que aparece en la Figura 2.5 se tiene una gráfica G con el apareamiento inicial M=((1,9), (2,6), (3,8)). Los nodos expuestos de S: 4 y 5 son las raíces de los árboles alternantes que aparecen en la Figura 2.6. Luego de localizar la trayectoria aumentante P=(5,9,1,10) el apareamiento se actualiza quedando como:
M'=((5,9), (1,10), (2,6), (3,8)), como se ve en la Figura 2.7.
Repitiendo el proceso de etiquetado del algoritmo resulta el árbol

alternante A=(4,8,3,6,2), con la raíz en el nodo 4 de S. Este etiquetado es húngaro, así que el apareamiento M' es de máxima cardinalidad. La solución al problema dual está formada por los nodos del siguiente conjunto: (S-A)∪ (T∩A) = (1,5,6,8).

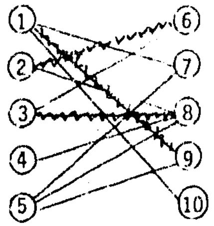


Figura 2.5

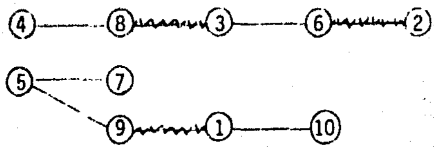


Figura 2.6

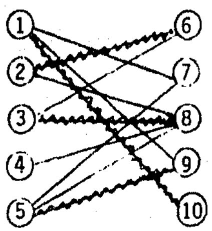
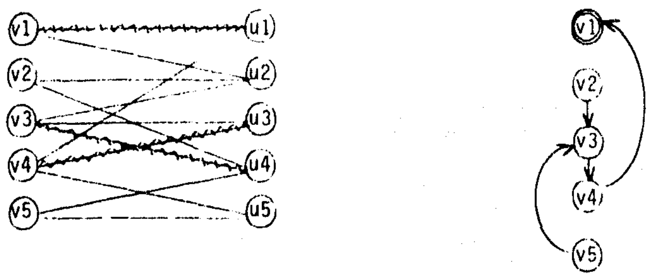


Figura 2.7

Para una gráfica bipartita G=(V,U,E) Papadimitrou [3] propone un algoritmo que busca trayectorias aumentantes 'P' respecto a un apareamiento M (puede ser vacío para iniciar) en G, y actualiza el apareamiento sustituyendo M por $M \nabla P$. Apoyándose en el hecho de que toda trayectoria aumentante tiene un número impar de arcos, y por ser G bipartita, tales trayectorias deben tener un nodo terminal en V y el otro en U, este algoritmo encuentra trayectorias aumentantes construyendo trayectorias alternantes que parten de nodos expuestos en V hasta encontrar nodos expuestos en U.

Para aclarar el modo de operar del algoritmo, consideremos la Figura 2.8 (a), donde los arcos ondulados señalan un apareamiento en G.

Nótese que cualquier trayectoria aumentante que parta de un nodo en V, inicia en un arco libre y regresa a otro nodo de V por medio de un arco apareado. Las posibilidades de pasar de un nodo a otro de V por medio trayectorias aumentantes se ve en la Figura 2.8 (b). En esta gráfica auxiliar A (que es una gráfica dirigida) el nodo v1 es adyacente al nodo expuesto u2. Esto sugiere que la trayectoria: v2,v3, v4,v1 en A puede formar la trayectoria aumentante P=(v2,u4,v3,u3,v4, u1,v1,u2) respecto a M en G.



(a) (b)

Figura 2.8

El algoritmo de Papadimitrou usa además los arreglos MATE, LABEL y EXPOSED. El arreglo MATE tiene |V|+|U| entradas, y representa el apareamiento en turno de la gráfica (al comienzo es vacío, al final es el máximo). Para cada nodo 'j' en G, MATE(j) es la pareja de 'j' en el apareamiento óptimo. La asignación MATE(j)=0 significa que el nodo 'j' es expuesto en la solución óptima.

El arreglo LABEL es usado en la búsqueda de los nodos que forman una trayectoria aumentante. Para cada $v \in V$, LABEL(v) es el nodo precedente en una trayectoria aumentante de la gráfica auxiliar A. Cuando LABEL(v)=0, se llega a un nodo expuesto de V donde inicia dicha trayectoria.

La subrutina AUGMENT es un procedimiento recursivo que se invoca al localizar una trayectoria aumentante 'P', y sirve para actualizar el apareamiento en turno 'M' sustituyéndolo por $M \nabla P$ (que es almacenado en el arreglo MATE nuevamente)

Para cada $v \in V$, EXPOSED[v] es un nodo de U que es expuesto y además adyacente a v; si no existe un nodo así, EXPOSED[v]=0 Cuando en la búsqueda se localiza un nodo $v \in V$ con EXPOSED[v] diferente de cero, se tiene una trayectoria aumentante. La Figura 2.9 exhibe el algoritmo escrito en pseudocódigo, que se puede traducir a algún lenguaje de computadora.

```
ALGORITMO 2.3.2. MAXIMO APAREAMIENTO BIPARTITA
Input:  Una gráfica bipartita B=(V,U,E).
Output: Un máximo apareamiento de B, representado en el
        arreglo MATE.
begin
  for all  v ∈ V ∪ U do mate[v]:=0;(comment: inicializa)
stage: begin
      for all  v ∈ V do  exposed[v]:=0;
      A:=∅;(comment. inicia construcción de gráfica auxiliar (V,A))
      for all [v,u] ∈ E do
      if mate[u]=0 then exposed[v]:=u else
         if mate[u] ≠ v  then A:=A ∪ (v,mate[u]);

         Q:=∅;
         for all  v ∈ V do  if mate[v]=0 then Q:=Q  (v),label[v] =0;

         while Q ≠ ∅ do
          begin
          sea v un nodo en Q;
          elimine v de Q;

           if exposed[v] ≠ 0 then augment(v), goto stage;
           else
             for all v' sin etiquetar tales que (v,v')  A do
             label[v']:=v, Q:=Q ∪ (v');
          end
      end
  end

  procedure augment(v)
    if label[v]=0 then mate[v] = exposed[v],
                       mate[exposed[v]]:=v;
    else begin
         exposed[label[v]]:=mate[v];
         mate[v]:=exposed[v],
         mate[exposed[v]]:=v;
         augment(label[v])
         end
```

Figura 2.9

Como se dijo al principio de esta sección, el problema de máximo apareamiento también se puede escribir como programa lineal

Para facilitar la exposición consideremos la gráfica bipartita de la Figura 2.4, con S=(1,3,5,7) y T=(2,4,6). Su matriz de incidencia A es:

$$
A = \begin{array}{c|ccccccc}
\text{arcos} & (1,4) & (1,6) & (3,2) & (5,2) & (5,6) & (7,2) & (7,4) \\
\text{nodos} & & & & & & & \\
1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
2 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\
3 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
4 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\
5 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
6 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\
7 & 0 & 0 & 0 & 0 & 0 & 1 & 1
\end{array}
$$

Si reescribimos los nodos de modo que primero aparezcan los que están en S y luego los que están en T, la matriz A queda así:

$$
A = \begin{array}{c|ccccccc}
\text{arcos} & (1,4) & (1,6) & (3,2) & (5,2) & (5,6) & (7,2) & (7,4) \\
\text{nodos} & & & & & & & \\
1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
3 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
5 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
7 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
\hline
2 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\
4 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\
6 & 0 & 1 & 0 & 0 & 1 & 0 & 0
\end{array}
$$

De este ejemplo resulta fácil ver que si en una gráfica bipartita G=(S,T,E) con |S|=s, |T|=t, |E|=n enumeramos primero los nodos en S y luego los nodos en T, la matriz de inicidencia A es de la forma:

$$
A = \begin{bmatrix} A_s \\ \hline A_t \end{bmatrix}
$$

donde As (matriz de sxn) y At (matriz de txn) son las filas de A representando a los nodos en S y T, respectivamente, y cualquier columna de As o de At es un vector unitario.

Debido a esta forma especial de la matriz de incidencia y del Teorema de Heller-Tompkins-Gale que aparece en el apéndice final, se puede concluír que la matriz de incidencia A de una gráfica bipartita es totalmente unimodular (cualquier submatriz cuadrada tiene determinante 0,1 o -1).

Entonces, si tenemos una gráfica bipartita G=(S,T,E) con m nodos y n arcos, asignando un peso de '1' a cada arco en E, el programa lineal que resuelve el máximo apareamiento es:

$$\max Z = X1 + X2 + \ldots + Xn$$

sujeta a:

$$(P) \qquad \begin{bmatrix} a11 & a12 & a13 & \ldots & a1n \\ a21 & a22 & a23 & \ldots & a2n \\ \vdots & & & & \vdots \\ am1 & am2 & am3 & \ldots & amn \end{bmatrix} \begin{bmatrix} X1 \\ X2 \\ \vdots \\ Xn \end{bmatrix} \leqslant \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}$$

$$Xk \geqslant 0, \ Xk \text{ entero}, \ k=1,2,\ldots n$$

donde $A=(aij)$ es la matriz de incidencia de G. Escribiendo el programa (P) en forma standard resulta:

$$\max Z = X1 + X2 + \ldots + Xn$$

sujeta a:

$$(P') \qquad \begin{bmatrix} A & : & I \end{bmatrix} \begin{bmatrix} X1 \\ \vdots \\ Xn \\ h1 \\ \vdots \\ hm \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ \vdots \\ 1 \\ 1 \end{bmatrix}$$

$$Xk \geqslant 0, \quad k=1,2,\ldots n \quad hi \geqslant 0, \quad i=1,2,\ldots,m.$$

donde I es la matriz identidad de orden 'm' que representa a las variables de holgura 'hi'. Recurriendo nuevamente al Teorema de Heller-Tompkins-Gale mencionado anteriormente, resulta que la matriz de restricciones del programa (P') es totalmente unimodular, lo que permite garantizar que la base óptima B (que es una submatriz cuadrada de [A:I]) tenga determinante unitario, y por tanto la solución óptima:

$$X = B^{-1} \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}$$

sea entera.

Por otra parte, ya que los elementos de [A:I] son uno o cero, de la restricción general de (P): $ai1X1 + \ldots + ainXn \leqslant 1$ y de las restricciones de no-negatividad se concluye que los únicos valores que pueden tomar los X_i componentes de X^* son uno o cero.

De este modo, X_j representa la 'participación' del arco j en la solución óptima, siendo 1 cuando el arco está en la solución y 0 en caso contrario. La restricción genérica del programa (P): $a_{i1}X_1 + a_{i2}X_2 + \ldots + a_{in}X_n \leq 1$ representa la condición de que el nodo 'i' tenga a lo más grado 1 respecto de los arcos del apareamiento óptimo.

Escribiendo el programa dual de (P) resulta:

$$\min W = Y_1 + Y_2 + \ldots + Y_m$$

sujeta a:

(D)
$$\begin{bmatrix} A^T \end{bmatrix} \begin{bmatrix} Y_1 \\ \cdot \\ \cdot \\ \cdot \\ Y_m \end{bmatrix} \geqslant \begin{bmatrix} 1 \\ 1 \\ 1 \\ \cdot \\ 1 \end{bmatrix}$$

$$Y_j \geqslant 0, \ Y_j \text{ entero}, \ j=1,2,\ldots m$$

donde A^T es la transpuesta de la matriz de incidencia de G, que también es totalmente unimodular, pues si B es submatriz cuadrada de A^T, entonces B es submatriz cuadrada de A y $\det(B)=\det(B^T)=$ 0,-1 o 1. Así, por un razonamiento similar al usado en la solución óptima primal, resulta que la solución dual Y^* también es entera.

Identificando las Y_j $j=1,2,\ldots,m$ con los nodos de G, la restricción genérica: $a_{1j}Y_1 + a_{2j}Y_2 + \ldots + a_{mj}Y_m \geqslant 1$ representa la condición de que el arco 'j' incida al menos una vez en el conjunto de nodos que representa la solución óptima.

Las observaciones anteriores sugieren la siguiente definición:

Un RECUBRIMIENTO DE NODOS de una gráfica $G=(V,E)$ es un subconjunto de nodos $L \subset V$ tal que cada arco de E incide en algún $v \in L$.

Claramente, el problema dual (D) mencionado anteriormente equivale a buscar un recubrimiento de nodos de G de cardinalidad mínima.

Puesto que el problema primal (P) tiene la solución factible $X=0$ (apareamiento nulo), y su función objetivo está acotada superiormente por la parte entera de $V/2$, es claro que (P) posee solución objetivo óptima finita, y por el Teorema Fundamental de Dualidad de programación lineal, se tiene la misma condición para problema dual (D). En consecuencia se puede establecer el siguiente teorema de dualidad para el problema de máximo apareamiento:

PROPOSICION 2.3a. (Teo. de König-Egerváry) Si $G=(S,T,E)$ es una gráfica bipartita, entonces:

$$\max \{ |M| : M \text{ es apareamiento en } G \} =$$
$$\min \{ |L| : L \text{ es recubrimiento de nodos de } G \}.$$

en la figura 2.10 se ilustra un ejemplo de un apareamiento máximo y un recubrimiento de nodos mínimo en una gráfica G. Los arcos del apareamiento se indican con linea ondulada, y los nodos del recubrimiento con doble círculo

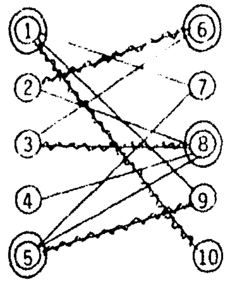


Figura 2.10

Si se resuelve el problema de máximo apareamiento en una gráfica bipartita usando el Algoritmo 2.3.1, dado en esta sección, los nodos que forman el conjunto $C=(S-L)\cup(T\cap L)$ del Paso 3, representan un recubrimiento de nodos de G de cardinalidad mínima.

(2.4) APLICACIONES DEL MAXIMO APAREAMIENTO.

En esta sección se muestran algunos problemas de tipo combinatorio oue pueden ser resueltos mediante una adecuada conversión del problema original a uno de máximo apareamiento, el cual se puede resolver usando alguno de los algoritmos descritos en la sección 2.3.

(2.4.1) Sistemas de Representantes Distintos.

Supongamos que tenemos un conjunto finito T, y una colección de subconjuntos de T: $(T1,T2,\ldots,Tk)$, ¿es siempre posible elegir un elemento de cada subconjunto Tj, de modo que los 'k' elementos escogidos sean distintos entre sí?. En el ejemplo 5) de la sección 1.2.2 se ilustra un problema de este tipo.

Para responder a la pregunta es conveniente describir el problema de apareamiento de la sección 2.2 haciendo consideraciones de 'alcanzabilidad' como sigue:

Si $G=(S,T,E)$ es una gráfica bipartita, para cualquier subconjunto $Q\subset S$ se define el conjunto:

$$N(Q)=\{\ j\in T\ :\ (i,j)\text{ es un arco de }E,\text{ con }i\in Q\ \}.$$

N(Q) representa al conjunto de nodos en T, 'alcanzables' desde Q.

Un primer resultado que relaciona la definición anterior con la idea de apareamiento se da a continuación:

PROPOSICION 2.4a. Si $G=(S,T,E)$ es gráfica bipartita con $|S| \leq |T|$ entonces: G posee un apareamiento que cubre cada uno de los nodos en S si y sólo si

$$N(Q) \geq |Q| \quad \text{para todo } Q \subset S \qquad (*)$$

Demostración.

En la dirección 'sólo si': Suponiendo que M es un apareamiento que cubre cada nodo de S, entonces para cualquier $Q \subset S$, cada nodo de Q se encuentra apareado con un nodo distinto de $N(Q)$, así que en $N(Q)$ hay al menos tantos nodos como en Q: $|N(Q)| \geq |Q|$.

En la dirección 'si': Supóngase que la condición (*) se cumple, y no existe ningún apareamiento de G que cubra todos los nodos de S. En particular, si M es un apareamiento de cardinalidad máxima, hay por lo menos un nodo 'v' en S no cubierto por M.

Considérese ahora la colección 'P' de trayectorias alternantes que parten del nodo 'v'. Ya que M es máximo, no hay ninguna trayectoria en P que termine con nodo expuesto respecto a M (no existen trayectorias aumentantes). Entonces, si se define el subconjunto $Q \subset S$ por la condición: $q \in Q$ siempre que 'q' forme parte de alguna trayectoria en P, resulta que todos los nodos de Q (salvo 'v') son nodos cubiertos por M, de modo que en $N(Q)$ hay tantos nodos como en Q, menos uno: $|N(Q)| = |Q| - 1$, lo cual contradice la hipótesis (*). Consecuentemente, debe existir un apareamiento que cubra todos los nodos de S.

La respuesta a la pregunta formulada al principio de la sección se encuentra en el Teorema de Hall, que es un resultado muy conocido en análisis combinatorio. Recordando que para un conjunto finito T y una colección de sus subconjuntos: $(T1, T2, \ldots, Tk)$, un Sistema de representantes distintos es una colección de elementos de T: $(t1, t2, \ldots, tk)$ tales que $ti \in Ti$ para $i=1,2,\ldots,k$, el teorema de Hall se establece como sigue:

PROPOSICION 2.4b. (Teorema de Hall) La colección de subconjuntos $T1, T2, \ldots, Tk$ del conjunto T tiene un sistema de representantes distintos si y sólo si:

$$\left| \bigcup_{i \in Q} Ti \right| \geq |Q| \quad \text{para todo } Q \subset (1,2,\ldots,k).$$

La construcción de un sistema de representantes distintos para T (si es que existe) se puede lograr resolviendo un problema de máximo apareamiento en una gráfica bipartita $G=(U,V,E)$ donde los nodos en U representan a la familia de subconjuntos $Ti \subset T$, y los nodos en V representan a los elementos de T; la existencia de un arco entre 2 nodos significa que el nodo (elemento) en V pertenece al nodo (subconjunto) en U. La Figura 2.11 muestra la construcción de la gráfica.

G=(U,T,E) gráfica bipartita.

U=(1,2,3,...,k)   T=(t1,t2,...,tn)

E=((i,j): i∈U, con j∈Ti)

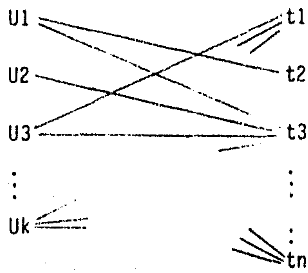


Figura 2.11

(2.4.2) Descomposición en Cadenas.

Supóngase que T es un conjunto finito en el cual está definido un orden parcial estricto, simbolizado por '<'. Recordando el ejemplo (6) de la sección 1.2.2, dos elementos a,b,∈ T se llaman COMPARABLES si ocurre a<b o b<a, y en caso contrario se llaman NO-COMPARABLES, mientras que una CADENA es un subconjunto de elementos de T comparables dos a dos, de modo que es posible escribirlos en orden ascendente: t1<t2<...<tn. La LONGITUD de una cadena es igual al número de nodos que contiene. Una descomposición en cadenas de T es una colección de cadenas disjuntas tal que al unirse forman T, y el problema de la descomposición en cadenas consiste en encontrar la descomposición de cardinalidad mínima (con el menor número de cadenas posible).

El ejemplo (6) de la sección 1.2.2 muestra un problema tipo, y otro caso se da a continuación: sea T=(t1,t2,...,tn) un conjunto de tazones de tamaños variados, donde la relación 'ti < tj' significa que se puede acomodar el tazón ti dentro del tj. Esta relación resulta ser un orden parcial estricto en T, y una cadena es un subconjunto de tazones que pueden 'anidarse'. El problema es entonces acomodar los tazones unos dentro de otros formando el menor número posible de 'nidos'

Una vez que se tiene un conjunto T con un orden parcial estricto, se puede reducir el problema de la descomposición en cadenas a uno de máximo apareamiento de la siguiente manera:

Sean U y V dos copias extra del conjunto T (U=V=T). A la copia en U de un elemento $t \in T$ se le denotará como t', y a la copia en V como t''. Se forma entonces la gráfica bipartita G=(U,V,E) de modo que un arco (i,j) pertenezca a E siempre y cuando se cumpla $ti < tj$. En la Figura 2.12 (a) se muestra un conjunto T parcialmente ordenado; la relación $ti < tj$ se cumple si se puede llegar a ti desde tj por una trayectoria hacia abajo en T. En la Figura 2.12 (b) se puede ver la gráfica bipartita G asociada, donde los arcos ondulados indican un apareamiento máximo.

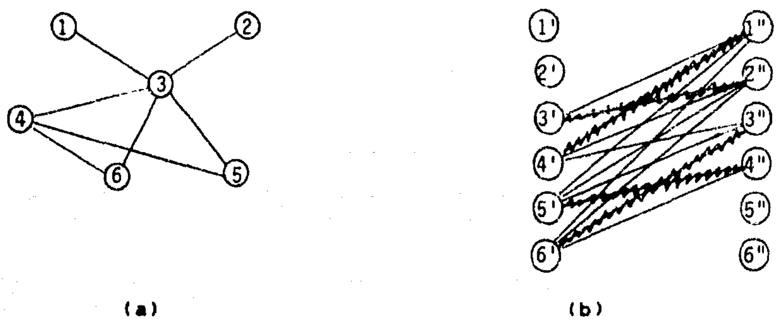


Figura 2.12

Encontrar un máximo apareamiento en la gráfica G así contruída es lo mismo que hallar el más grande subconjunto de parejas de elementos comparables de T, donde no hay dos elementos que se repitan como primer elemento o como segundo elemento en las parejas.

Las parejas del máximo apareamiento se pueden 'encadenar' identificando los elementos comunes en ellas. Los elementos restantes de T se pueden considerar como cadenas de longitud uno, y así se obtiene la descomposición en cadenas de T que se busca. Para aclarar la formación de las cadenas, consideremos la Figura 2.12 (b) donde un apareamiento máximo lo forman: (6',3''),(3',2''),(5',4'') y (4',1''). De este modo, una descomposición en cadenas mínima para T es:
D={(6,3,2) , (5,4,1) }.

Como puede verse de la construcción anterior, a cada descomposición en cadenas D del conjunto T corresponde un único apareamiento M en la gráfica asociada G y viceversa. Aún más, bajo esta correspondencia, cada cadena de longitud m en D se asocia con m-1 parejas en M. Si las cadenas en D son: C1,C2,...,Ck se tiene:

$$|M| = (|C1|-1) + \dots + (|Ck|-1)$$

$$|M| = |C1|+\dots+|Ck| - k$$

$$|M| = |T| - |D|$$

De esta última ecuación salta a la vista que el valor $|M|$ es máximo a costa de hacer al valor $|D|$ mínimo para lograr la suma constante $|T|$; esto justifica la reducción del problema de descomposición en cadenas a uno de máximo apareamiento.

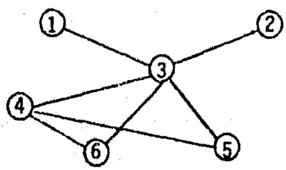
Si se define una ANTICADENA o CONJUNTO NO-COMPARABLE en T como un subconjunto de elementos de T no-comparables 2 a 2, se pueden expresar algunos resultados duales, como son las siguientes proposiciones, cuyas demostraciones se incluyen en el apéndice al final.

PROPOSICION 2.4c. (Teorema de Dillworth) Si T es un conjunto finito con un orden parcial estricto, entonces:

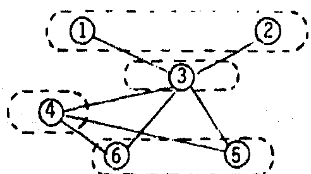
$$\min \{ |D| : D \text{ es descomposición en cadenas de } T \} = \max \{ |N| : N \text{ es una anticadena en } T \}.$$

PROPOSICION 2.4d. Sea T un conjunto finito con un orden parcial estricto, y supóngase que la cadena más larga en T es de longitud n. Entonces se puede hacer una partición de T en n anticadenas disjuntas.

La Figura 2.13 (a) muestra un conjunto T con un orden parcial semejante al de la Figura 2.12 (a), donde la cadena de mayor longitud C={1,3,4,6} tiene largo 4; en la Figura 2.13 (b) las líneas punteadas indican la partición en anticadenas disjuntas.



(a)



(b)

Figura 2.13

(2.4.3) Apareamiento en Gráficas Convexas.

El problema del apareamiento máximo se resuelve fácilmente para un tipo especial de gráficas que F. Glover llama 'convexas'.

Una gráfica bipartita $G=(S,T,A)$ se llama CONVEXA siempre que para dos arcos (i,j) y (k,j) cualesquiera en A, con $i < j$ se cumpla que los arcos: $(i+1,j)$, $(i+2,j)$, ... ,$(k-1,j)$ también están en A. Por ejemplo, supongamos que en un servicio de cómputo se tienen disponibles 4 terminales que son solicitadas por 5 usuarios como se indica a continuación:

Terminal#	1.	2.	3.	4.
Horario:	14:00 a 16:00	15:00 a 18:00	16:00 a 19:00	16:00 a 18:00

Usuario #	1.	2.	3.	4.	5.
Hora de inicio pedida:	14:00	15:00	16:00	17:00	18:00
Tiempo pedido:	1hr.	1hr.	1hr.	1hr.	1hr.

La gráfica de la Figura 2.14 modela el problema, y se trata de una gráfica convexa; los nodos tipo S representan a los usuarios, y los nodos tipo T representan las terminales. Un arco (i,j) uniendo dos nodos significa que el usuario 'i' puede utilizar la terminal 'j'.

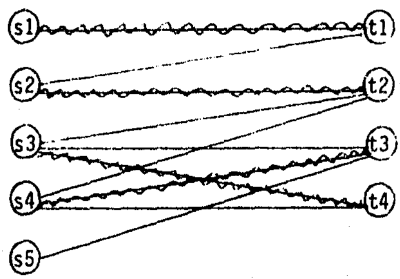


Figura 2.14

El máximo apareamiento en una gráfica convexa se resuelve mediante el procedimiento que sigue: para cada nodo $j \in T$, se define

$$P_j = \max(i : (i,j) \in A)$$

Se inicia con el apareamiento vacío, y se itera sobre el índice $i=1,2,\ldots,m$ (para nodos en S). Si se encuentran arcos (i,j) para los cuales 'j' es nodo expuesto, se agrega al apareamiento el arco que tenga el valor P_j lo más pequeño posible.

Aplicando este proceso al ejemplo de la Figura 2.14, se obtiene el apareamiento máximo indicado por los arcos ondulados.

(2.5) APAREAMIENTO PONDERADO.

En este problema de apareamiento se tiene una gráfica $G=(S,T,E)$ donde cada arco (i,j) en E tiene asociado un peso w_{ij}. El problema consiste en encontrar un apareamiento M en G tal que la suma de los arcos en M sea lo más grande posible.

El problema del máximo apareamiento de la sección 2.2 puede verse como un caso particular de apareamiento ponderado donde cada arco de la gráfica tiene asociado un peso unitario $w_{ij}=1$.

En el clásico problema de asignación se trata de maximizar el peso total al aparear elementos de dos conjuntos finitos de igual cardinalidad. Usando terminología de apareamiento, el problema de asignación también es un caso particular de apareamiento ponderado, donde la gráfica subyacente es bipartita y el apareamiento buscado es además completo. Recíprocamente, todo problema de apareamiento ponderado en una gráfica bipartita puede convertirse en uno de asignación simplemente añadiendo a la gráfica original $G=(S,T,E)$ nodos y arcos suficientes para que los conjuntos 'S' y 'T' sean de igual cardinalidad, y asignando peso cero a los nuevos arcos agregados.

El problema del apareamiento ponderado es resuelto por un método primal-dual, llamado 'Método Húngaro' por H.W. Kuhn.

Para dar idea del funcionamiento del método se puede suponer, sin pérdida de generalidad, una gráfica bipartita completa $G=(S,T,S\times T)$ donde $|S|=m$, $|T|=n$, $m \leq n$. El programa lineal equivalente al problema de apareamiento ponderado es:

$$\max Z = \underset{i,j}{\quad} w_{ij} X_{ij}$$

sujeta a:

$$\sum_{j} X_{ij} \leq 1$$

$$\sum_{i} X_{ij} \leq 1$$

$$X_{ij} \geq 0$$

No es difícil observar que las restricciones de este programa corresponden a la matriz de incidencia de la gráfica (bipartita) asociada, que es totalmente unimodular. Usando un argumento similar al del caso de apareamiento máximo, resulta que los únicos valores posibles para las X_{ij} óptimas son 0 o 1, según que el arco representado pertenezca al apareamiento óptimo o no.

Escribiendo el programa dual resulta:

$$\min W = \sum_i U_i + \sum_j V_j$$

sujeta a:

$$U_i + V_j \quad w_{ij}$$

$$U_i \geqslant 0 \;, \quad V_j \geqslant 0$$

Para las soluciones óptimas primal y dual las condiciones de holgura complementaria necesarias y suficientes son:

$$X_{ij} > 0 \implies U_i + V_j = w_{ij} \qquad \text{(h1)}$$

$$U_i > 0 \implies \sum_j X_{ij} = 1 \qquad \text{(h2)}$$

$$V_j > 0 \implies \sum_i X_{ij} = 1 \qquad \text{(h3)}$$

El método húngaro mantiene factibilidad primal y dual en todo momento, y además satisface todas las condiciones de holgura complementaria, excepto las condiciones (h2). El número de condiciones que no se satisfacen decrece en forma monótona a lo largo del proceso.

Lawler [1] presenta un algoritmo que arranca con el apareamiento vacío y usando una técnica de etiquetado localiza trayectorias aumentantes que hacen crecer el apareamiento original. A continuación se describe el algoritmo:

ALGORITMO 2.5.1. APAREAMIENTO PONDERADO BIPARTITA.

Paso 0). INICIO. Se tiene una gráfica bipartita $G=(S,T,A)$ con un peso w_{ij} para cada arco (i,j) en A, y que cumple $|S| \leqslant |T|$ El apareamiento inicial es $M = \phi$. Se asignan valores iniciales: $V_j=0$ y $p_j = \infty$ para cada nodo j T, $U_i = \max (w_{ij})$ para cada nodo $i \in S$, y ningún nodo está etiquetado.

Paso 1). ETIQUETADO.

(1.0) Si todos los nodos de S son saturados, el apareamiento M es óptimo y los valores de U_i, V_j corresponden a la solución dual óptima. Alto.

En caso contrario, coloque la etiqueta '0' a cada nodo expuesto en S.

(1.1) Si todas las etiquetas se han examinado, o si hay etiquetas sin examinar pero cada una de ellas está en un nodo 'i' de T para el cual $p_i > 0$, ir al Paso 3.

(1.2) Encuentre un nodo 'i' con etiqueta sin examinar, ya sea que

$i \in S$ o que $i \in T$ con la condición $pi=0$. Si $i \in S$, ir al Paso 1.3; si $i \in T$, ir al Paso 1.4.
(1.3) Examine el nodo 'i' ($i \in S$) en la siguiente forma: para cada arco $(i,j) \notin M$ incidente al nodo i, si $Ui + Vj - wij < pj$, entonces ponga al nodo j la etiqueta "i" (reemplazando cualquier etiqueta existente) y haga $pj = Ui + Vj - wij$. Regrese al Paso 1.1.
(1.4) Examine la etiqueta en el nodo 'i' ($i \in T$) como sigue: Si el nodo es expuesto, ir al Paso 2. En caso contrario, identifique al único arco (i,j) en M incidente al nodo i, y ponga al nodo j la etiqueta "i". Regrese al Paso 1.1.

Paso 2). AUMENTO. Se ha encontrado una trayectoria aumentante que termina en el nodo i (identificado en Paso 1.4). Los nodos que preceden al nodo i en la trayectoria se identifican por un rastreo regresivo sobre las etiquetas. Aumente el apareamiento M añadiéndole todos los arcos en la trayectoria aumentante que no pertenezcan a M y eliminando de M los que sí estén en la trayectoria. Haga $pj= \infty$ para cada nodo j en T. Borre las etiquetas en todos los nodos y regrese al Paso 1.0.

Paso 3). CAMBIOS EN LAS VARIABLES DUALES. Calcule los siguientes valores:

$$d1 = \min(Ui : i \text{ en } S)$$

$$d2 = \min(pj : pj > 0, j \text{ en } T)$$

$$d = \min(d1, d2)$$

Haga la resta $Ui-d$ para cada nodo $i \in S$ con etiqueta. Haga la suma $Vj + d$ para cada nodo $j \in T$ con $pj=0$. Calcule $pj-d$ para cada nodo $j \in T$ con $pj > 0$. Si $d < d1$ ir al Paso 1.1. En caso contrario, M es un apareamiento ponderado óptimo, y las variables Ui, Vj corresponden a la solución óptima dual. Alto.

El ejemplo de la Figura 2.15 muestra una gráfica G con los pesos de sus arcos y el apareamiento M = { (1,7), (2,5), (3,6) } que aún no es óptimo.

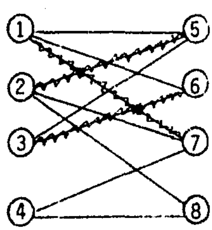


Figura 2.15

Pesos de los arcos.

	5	6	7	8
1	9	10	15	-
2	4	-	-1	3
3	5	7	-	-
4	-	-	12	1

Los valores calculados en el Paso 3 son: d1=4, d2=1, d=1.
Las etiquetas en los nodos son:

Nodo--->	1	2	3	4	5	6	7	8
Etiqueta:	7	5	6	0	1	1	4	2

Los valores de 'pj' para los nodos j=5,6,7,8 son todos cero luego de ejecutar el Paso 3. Así, se localiza fácilmente la trayectoria aumentante: (4,7), (7,1) (1,5), (5,2), (2,8), y el apareamiento M se actualiza como: M = { (1,5),(2,8),(3,6),(4,7) }. Este apareamiento satura los nodos del conjunto S={1,2,3,4} y es por tanto óptimo. En la Figura 2.16 se ve este apareamiento óptimo cuyo peso total es 31.

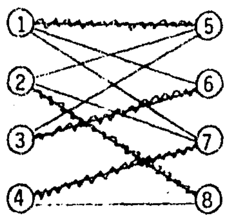


Figura 2.16

Los valores de las variables duales son:

Variable:	U1	U2	U3	U4	V5	V6	V7	V8
	8	3	5	5	1	2	7	0

La suma de variables duales es 31, que coincide con el peso total del apareamiento óptimo.

(2.6) OTROS TIPOS DE APAREAMIENTO BIPARTITA.

En esta sección se tratan dos modelos de apareamiento que necesariamente utilizan gráficas bipartitas: el problema del apareamiento max-min o 'cuello de botella' y el apareamiento tipo Gale-Shapley.

El hecho de que las gráficas en las que se usan estos modelos sean exclusivamente bipartitas no les resta utilidad a los mismos, ya que los problemas que resuelven se presentan comúnmente en la realidad, y los algoritmos de solución son bastante eficientes en ambos casos

(2.6.1) APAREAMIENTO MAX-MIN o 'CUELLO DE BOTELLA'.

Dada una gráfica $G=(S,T,E)$ donde cada arco (i,j) en E tiene asociado un peso w_{ij}, el problema del apareamiento max-min o "de cuello de botella" consiste en encontrar el apareamiento de máxima cardinalidad para el cual el peso mínimo de los arcos es lo mayor posible.

Un procedimiento conocido como 'algoritmo del umbral' resuelve el problema; su funcionamiento es como sigue:

Dada la gráfica $G=(S,T,E)$ se comienza con el apareamiento vacío y un valor de umbral W suficientemente grande. En la iteración genérica del algoritmo, un apareamiento de cardinalidad k es obtenido. Entonces se busca una trayectoria aumentante en la subgráfica de G formada por todos los arcos (i,j) que cumplen: $w_{ij} \geqslant W$.

Si existe una trayectoria aumentante, se consigue un apareamiento max-min de cardinalidad $k+1$; en caso contrario se reduce el valor W de umbral lo suficiente como para permitir la existencia de una trayectoria aumentante.

Claramente, el número de valores de umbral que se deben considerar en el algoritmo no excede al número máximo de distintos pesos en los arcos: $(m)x(n)$, donde $|S|=m$, $|T|=n$, así que el algoritmo se detiene en un número finito de pasos con el apareamiento buscado.

En el algoritmo mostrado a continuación, a cada nodo 'j' de T se le asocia un número 'p_j', que mide el nivel al cual debe reducirse el umbral para que el nodo j pueda ser añadido a un árbol alternante. Los nodos son etiquetados, pero ningún nodo j es marcado a menos que se cumpla $p_j \geqslant W$. Cuando ya no hay más nodos elegibles para ser marcados, el umbral W es reducido al máximo valor de p_j estrictamente menor que W. Esto permite la adición de al menos un nodo más en algún árbol. A medida que se desarrolla el algoritmo, se encuentran trayectorias aumentantes o los árboles alternantes se vuelven húngaros.

ALGORITMO 2.6.1. APAREAMIENTO MAX-MIN.

Paso 0). INICIO. Se tiene una gráfica bipartita $G=(S,T,A)$ con un peso w_{ij} asociado a cada arco $(i,j) \in A$. Se comienza con el apareamiento vacío $M=\phi$, con el umbral inicial $W=\infty$, y se asigna $p_j=-\infty$ para cada nodo $j \in T$. Nigún nodo está etiquetado.

Paso 1). ETIQUETADO.
(1.0) Ponga la etiqueta '0' a cada nodo expuesto de S.
(1.1) Si todas las etiquetas están ya marcadas, vaya al Paso 3.
Si existen etiquetas sin marcar, pero cada una de ellas está en un nodo 'i' de T para el cual $p_i < W$, entonces se actualiza el valor de umbral como:

$$W = \max\{ p_i : p_i < W \}$$

(1.2) Encuentre algún nodo 'i' con etiqueta sin marcar, que cumpla: 'i' está en S o 'i' está en T con un valor $p_i \geqslant W$. Si el

nodo 'i' está en S, ir al Paso 1.3; si 'i' está en T ir a 1.4.
(1.3) Marque el nodo 'i' (i S) de la siguiente forma: Para cada arco $(i,j) \notin M$ incidente al nodo i, si $p_j < w_{ij}$ y $p_j < W$, entonces ponga al nodo j la etiqueta "i" (reemplazando cualquier etiqueta que hubiera) y haga $p_j = w_{ij}$. Regrese al Paso 1.1
(1.4) Marque el nodo 'i' ($i \in T$) como sigue: Si el nodo i es expuesto, ir al Paso 2. En caso contrario identifique al único arco (i,j) M incidente al nodo i, y ponga al nodo j la etiqueta "i". Regrese al Paso 1.1

Paso 2). AUMENTO. Se ha encontrado una trayectoria aumentante que termina en el nodo i (identificado en Paso 1.4). Los nodos que preceden al i en la trayectoria se identifican por un rastreo regresivo. Así, si la etiqueta en el nodo i es "j", el penúltimo nodo en la trayectoria es j. Si la etiqueta en el nodo j es "k", el antepenúltimo nodo de la trayectoria es k, y así sucesivamente. El nodo inicial en la trayectoria tiene etiqueta "0". Aumente el apareamiento M agregándole todos los arcos en la trayectoria aumentante que no pertenezcan a M y eliminando de M los que sí estén en la trayectoria. Borre todas las etiquetas en todos los nodos. Haga $p_j =$ para cada nodo j en T. Regrese al Paso 1.0.

Paso 3). ETIQUETADO HUNGARO. No existen trayectorias aumentantes, y el apareamiento M es apareamiento max-min de cardinalidad máxima. Sea L el conjunto de nodos etiquetados. Sea (a,b) el arco de M que tiene el peso mínimo. La subgráfica de G obtenida al eliminar los nodos en el conjunto: $(S - L) \cup (T \cap L) - \{ (a,b) \}$ es una solución min-max dual a M. Alto.

En la Figura 2.17 aparece una gráfica G con pesos indicados en sus arcos. El apareamiento óptimo es M = { (2,8),(3,7),(4,5) } que tiene un peso mínimo de -4 (que es el último valor de umbral calculado en el algoritmo) y que se muestra con arcos ondulados en la gráfica.

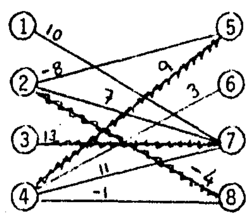


Figura 2.17

(2.6.2) APAREAMIENTO DE GALE-SHAPLEY

En este tipo de apareamiento se tiene una gráfica bipartita y un 'valor de preferencia' asociado a cada arco de la gráfica. Para cada arco (i,j) este valor de preferencia mide la predilección que tiene el nodo 'i' a formar pareja con el nodo 'j'.

Esta situación, en la cual los elementos a formar pareja manifiestan preferencias se presenta en muchos problemas reales. La formación de parejas de hombres y mujeres para formar matrimonio, la selección de estudiantes becados a diversas universidades, o de solicitantes de empleo a puestos de trabajo son algunos ejemplos concretos.

A fin de facilitar la exposición, consideremos el ejemplo donde se tienen n hombres y n mujeres para formar parejas, y cada hombre tiene una lista de las mujeres ordenadas según su preferencia. Esta situación tiene semejanza con el manejo de funciones de utilidad que se hace en Microeconomía; sin embargo conviene hacer hincapié en que los hombres asignan valores de preferencia y no de utilidad a las mujeres. La razón de esto se debe a que manejar utilidad puede ser complicado, ya que es difícil comparar utilidades de diferentes hombres. Así, por ejemplo, si Juan asigna un valor de utilidad de 100 para María, quizá Pedro asigne tan sólo 85. En este caso, tratar de maximizar la utilidad total puede llevar al caso donde haya un hombre y una mujer que no formen pareja, pero que se prefieran más entre sí que a sus respectivas parejas.

Para completar el esquema, se supondrá también que cada mujer tiene una lista de preferencias por los hombres, y se llamará una asignación de hombres a mujeres INESTABLE si bajo ella, existen un hombre y una mujer que no forman pareja, pero que se prefieren más entre sí que a sus respectivos compañeros asignados.

Puesto que tanto hombres como mujeres manifiestan preferencia, la intención será entonces encontrar una asignación de hombres a mujeres que sea estable respecto de alguna de las partes (hombres o mujeres).

Esto constituye el criterio de optimalidad, y motiva la siguiente definición:

Un apareamiento estable de hombres y mujeres es óptimo (para los hombres) si cada hombre se siente bajo él por lo menos tan satisfecho como con cualquier otro apareamiento estable.

Dicho de otro modo, en el apareamiento óptimo (para los hombres) ningún hombre podrá intercambiar mujer con otro hombre a menos que alguno pierda preferencia.

El teorema que se muestra enseguida da una prueba constructiva de la existencia de un apareamiento óptimo como el mencionado en la definición anterior.

TEOREMA 2.6.1 (Gale-Shapley) Para cualquier conjunto de preferencias de hombres y mujeres, existe un apareamiento que es óptimo para los hombres.

Demostración.

Siguiendo literalmente a Gale y Shapley se describirá el algoritmo que conduce al apareamiento óptimo:

"Para iniciar, cada hombre le hace una propuesta a su chica favorita. Cada chica que recibe más de una propuesta se queda solamente con la que más prefiere de entre ellas. Sin embargo, no acepta enseguida tal propuesta, sino que la pone en fila de espera aguardando la llegada de una propuesta mejor.

En la segunda etapa, los hombres rechazados al inicio, hacen su segunda propuesta. Cada chica que recibe propuestas, elige la que más prefiere de entre ellas y la que tiene en lista de espera (si es que hay), y nuevamente deja la preferida en espera.

Los pasos siguientes son semejantes. Los hombres rechazados hacen su siguiente propuesta a las chicas, y ellas seleccionan de entre sus opciones la predilecta, dejándola en espera de mejorar.

A la larga (de hecho en a lo más $n^2 - 2n + 2$ pasos) cada chica ha recibido una propuesta, puesto que en cada paso se generan rechazados y nuevas propuestas, además de que un hombre no puede insistir otra vez con una chica que lo ha rechazado. En cuanto la última chica sin pareja recibe una propuesta, el proceso de asignación termina, y cada chica acepta al hombre que está en su lista de espera.".

No es difícil ver que el algoritmo produce matrimonios estables "Por ejemplo, si Juan y María no forman pareja, pero Juan prefiere a María más que a su propia esposa, entonces en algún paso del algoritmo Juan hizo una propuesta a María, y ésta lo rechazó. Claramente María prefiere más a su esposo que a Juan, y no hay inestabilidad.".

Se verá ahora que el conjunto de matrimonios logrado es óptimo para los hombres.

"Llamaremos a una mujer 'posible' para un hombre si existe un apareamiento estable en el cual ambos estén casados. Se prueba por inducción. Supóngase que hasta cierto paso del algoritmo ningún hombre ha sido rechazado por una mujer que es posible para él. En ese punto supóngase que una mujer A al recibir una propuesta de un hombre X, al que ella prefiere, rechaza al hombre Y. Se probará entonces que A es imposible para Y. Es claro que X prefiere más a la mujer A que a las demás, excepto por las que lo han rechazado y por lo tanto son imposibles para él. Considerando un apareamiento hipotético en el cual Y está casado con A, y X está casado con una mujer que es posible para él, resulta que la esposa de X es menos preferida para él que la misma A. Pero eso significa que tal apareamiento es inestable, pues A y X se prefieren más entre sí que a sus respectivos cónyuges. La conclusión es que el algoritmo rechaza hombres solamente de mujeres con las que no podrían estar casados bajo un apareamiento estable. El apareamiento final es por tanto, óptimo.".

Nótese que por simetría, un apareamiento óptimo para las mujeres se puede consguir si las mujeres son las que hacen propuesta a los hombres.

En el capítulo 4 se muestra un programa de computadora que implementa el algoritmo. En la Figura 2.18 a continuación se muestran las tablas de preferencias de 5 hombres (V,W,X,Y,Z) y de 5 mujeres (A,B, C,D,E). La solución dada por el algoritmo también aparece.

TABLAS DE PREFERENCIAS.

Hombre	Preferencias. (mayor a menor)
V	B, E, A, C, D
W	A, B, C, D, E
X	B, C, E, D, A
Y	A, C, B, D, E
Z	E, C, B, A, D

Mujer	Preferencias. (mayor a menor).
A	Z, V, Y, W, X
B	Y, Z, W, V, X
C	V, Y, W, X, Z
D	X, W, Y, V, Z
E	Y, W, X, Z, V

ASIGNACION OPTIMA (para hombres)
(V,A) , (W,D), (X,E), (Y,C), (Z,B)

Figura 2.18

C A P I T U L O 3.

APAREAMIENTO EN GRAFICAS NO-BIPARTITAS.

Los problemas de apareamiento en gráficas generales son la continuación natural de los planteamientos del caso bipartita. La diferencia esencial entre ambos casos es aparente en el contenido de la Proposición 1.1c del Capítulo 1: Las gráficas no-bipartitas poseen ciclos de longitud impar. Este sencillo hecho basta para que los algoritmos del caso bipartita dejen de ser confiables en gráficas generales. El desarrollo de nuevas ideas para tratar ciclos impares es la clave para encontrar nuevos algoritmos, y es de lo que trata este capítulo.

Los dos modelos presentados en este capítulo son: Máximo apareamiento y Apareamiento ponderado. En cada caso se muestra el algoritmo de solución y se da un ejemplo resuelto.

(3.1) ANTECEDENTES.

En el capítulo dos se observa que la estructura de las gráficas bipartitas facilita la construcción de algoritmos de solución relativamente sencillos en los problemas de apareamiento.

En gráficas generales (no-bipartitas) los procesos de solución se complican, y se requiere introducir conceptos nuevos para encontrar algoritmos eficientes. Esto, sin embargo, no quiere decir que el tratamiento del caso general carezca de rasgos comunes con el caso bipartita. Las ideas básicas de trayectoria alternante y trayectoria aumentante respecto de un apareamiento dado siguen siendo útiles.

Los dos teoremas que se muestran enseguida son generalización de los teoremas 2.2b y 2.1c del caso bipartita, al caso no-bipartita.

TEOREMA 3.1a. (Berge,Norman,Rabin) Un apareamiento M en una gráfica G no-bipartita contiene un número máximo de arcos si y sólo si no admite trayectorias aumentantes.

Demostración.

Se probará la proposición equivalente: "un apareamiento M en una gráfica G no-bipartita no es máximo si y sólo si existe una trayectoria aumentante".

En el sentido 'si' es inmediato: si existe una trayectoria aumentante, se puede obtener un nuevo apareamiento de cardinalidad $|M|+1$, y por tanto M no es máximo.

En el sentido 'sólo si'. Supóngase que M y N son apareamientos tales que $|M| < |N|$. La diferencia simétrica $M \nabla N$ forma una subgráfica de G con cierto número de componentes conexas. Cada componente puede ser: o trayectoria alternante o ciclo alternante, como se ilustra en la Figura 3.1.

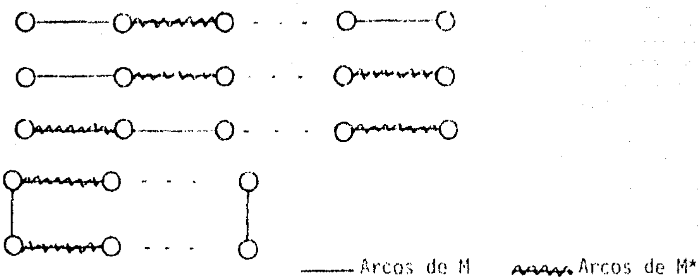


Figura 3.1

Cada ciclo contiene tantos arcos de M como de N, y como |M| < |N|, debe existir al menos una trayectoria alternante que contiene más arcos de M que de N. Semejante trayectoria tiene que extenderse entre dos nodos expuestos de M, de modo que es trayectoria aumentante respecto de M.

Cuando un apareamiento M se hace crecer por medio de una trayectoria aumentante, el nuevo apareamiento M' no deja expuesto a ninguno de los nodos originalmente cubiertos por M. De esto es sencillo concluir que sucesivos crecimientos de M resultarán en un apareamiento M* de máxima cardinalidad que cubre todos los nodos cubiertos por M. El teorema siguiente formaliza esta idea:

TEOREMA 3.1b. Si M es apareamiento en la gráfica no-bipartita G, entonces existe un apareamiento M* de máxima cardinalidad que cubre todos los nodos de G cubiertos por M.

Hasta el teorema anterior, el tratamiento del caso general no ha variado respecto del caso bipartita. El hecho fundamental que distingue al caso general del bipartita es la presencia de ciclos impares en las gráficas no-bipartitas (recuérdese que esto no ocurre para el caso bipartita).

El más simple ciclo impar en una gráfica no permite usar todos los resultados del apareamiento bipartita.

Consideremos la gráfica G de la Figura 3.2, donde el arco ondulado indica apareamiento.

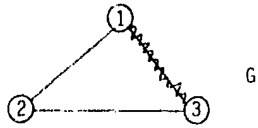


Figura 3.2

Claramente, el apareamiento mostrado en G es máximo y de cardinalidad 1. Si para este caso recurrimos al teorema de König-Egerváry del caso bipartita (proposición 2.3a), sería de esperar que el recubrimiento de nodos mínimo para G constara de un solo nodo, lo que evidentemente es falso.

Si no reparamos en esta falla teórica y tratamos de resolver el problema de máximo apareamiento con el algoritmo de etiquetado del caso bipartita, también habrá dificultades. Ensayemos con la gráfica de la Figura 3.3, donde los arcos ondulados indican el apareamiento inicial M = { (2,3) (4,5) } y se puede distinguir la trayectoria aumentante P: 1-2-3-5-4-6, que une los nodos expuestos 1 y 6.

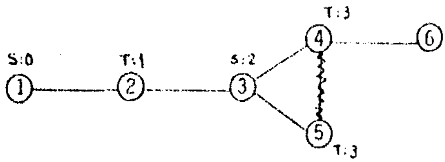


Figura 3.3

Comenzando con el nodo 1 como raíz del árbol, cada nodo sucesivo es etiquetado con "S" o "T" para indicar (como en el caso bipartita) a cuál conjunto de nodos pertenece, y con el número de nodo que le precede, a fin de permitir el "rastreo regresivo" de las trayectorias aumentantes. Así, por ejemplo, "S:2" en el nodo 3 significa que dicho nodo pertenece al conjunto S, y su predecesor es el nodo 2.

Cuando se construye el árbol alternante a partir de la raíz "S:0", la intención del algoritmo bipartita es hallar un nodo expuesto con etiqueta "T" para formar una trayectoria aumentante por rastreo regresivo sobre las etiquetas. Para ello, cada vez que un nodo "i" es etiquetado con "S", para cada arco (i,j) fuera del apareamiento inicial M que incida en "i", la etiqueta "T:i" es asignada al nodo "j", a menos que éste ya posea una etiqueta "T"; mientras que si un nodo "i" es etiquetado con "T", se identifica al único arco (i,j) en M que incide en "i", y se asigna la etiqueta "S:i" al nodo "j".

Siguiendo el procedimiento indicado, resultan las etiquetas que se muestran en la Figura 3.3. La dificultad surge al intentar etiquetar el siguiente nodo a partir del nodo 4 o del nodo 5. En cualquier caso uno de los nodos 'nos lleva' al otro por el arco (4,5) de M, obligándonos a cambiar una etiqueta "T" por una "S" (cosa no prevista en el algoritmo). Así, por ejemplo, si elegimos el nodo 4 para continuar, el nodo 5 resultaría con etiqueta "S:4", y en ese punto, dado que cualquier nodo accesible desde el nodo 5 ya ha sido explorado, el algoritmo bipartita juzgaría que el árbol es húngaro y que el apareamiento dado es máximo, conclusión visiblemente errónea.

Los inconvenientes que se han expuesto resaltan la necesidad de buscar un trato adecuado para los ciclos impares de las gráficas. La respuesta a tal necesidad está en el concepto de FLORACION, desarrollado por J. Edmonds.

A grandes rasgos, el enfoque de Edmonds implica la construcción de árboles alternantes (como se hizo en el caso bipartita), la detección de ciertos ciclos impares llamados 'floraciones' y el 'encogimiento' de tales floraciones por contracción de la gráfica. A continuación se formaliza la idea.

DEFINICION 3.1c. Sea M apareamiento en la gráfica no-bipartita $G=(N,A)$. Sea $N_B \subset N$ un conjunto de $2k+1$ nodos, con $k \geq 1$, y sea $B \subset A$ el subconjunto de arcos de G con ambos extremos incidiendo en nodos de N_B.

Se dice que B es FLORACION respecto a M si cumple:

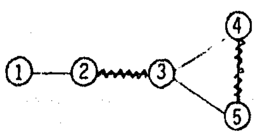
a) $|M \cap B| = k$

es decir, el apareamiento M es maximal dentro de B. Al único nodo 'b' en N_B expuesto por $M \cap B$ se le llama BASE de la floración.

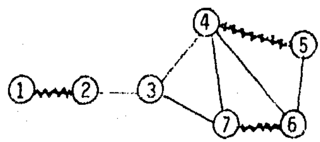
b) Existe una trayectoria alternante S, llamada TALLO de la floración, con $|S|$ par, $S \cap B = \emptyset$, que une la base de la floración con un nodo expuesto de M, llamado RAIZ del tallo. Cuando $S = \emptyset$, se dice que la floración está enraizada.

c) Para cada nodo $i \in N_B$, existe una trayectoria alternante $S_{b,i}$ B, con $|S_{b,i}|$ par, entre el nodo 'i' y la base de la floración. De esto se concluye que hay una trayectoria alternante de la forma: $S, S_{b,i}$ entre la raíz del tallo y el nodo 'i'.

En la Figura 3.4, los arcos ondulados señalan apareamiento. En la Figura 3.4a) se puede ver la floración $B=\{(3,4),(3,5),(4,5)\}$, con $N_B = \{3,4,5,6,7\}$, base en el nodo 3, raíz en el nodo 1 y tallo $S = \{(1,2),(2,3)\}$. En la Figura 3.4b) la floración es $B= \{(3,4),(3,7),(4,5),(4,6),(4,7),(5,6)\}$, con $N_B = \{3,4,5\}$, raíz y base en el nodo 3 y tallo $S = \emptyset$; en este caso la floración está enraizada.



(a)



(b)

Figura 3.4

La trayectoria alternante que une al nodo 4 de la Figura 3.4a) con su raíz es P1: 4-5-3-2-1. La trayectoria alterntante que une al nodo 7 de la Figura 3.4b) con su raíz es P2: 7-6-5-4-3.

(3.2) MAXIMO APAREAMIENTO NO-BIPARTITA.

El problema del máximo apareamiento en gráficas generales es idéntico al del caso bipartita: encontrar el apareamiento con el mayor número de arcos posible en la gráfica.

Supóngase que se usa un proceso de etiquetado para generar árboles alternantes como en el ejemplo de la Figura 3.3, iniciando con un apareamiento M dado. Si en cierto paso del algoritmo se descubre un arco fuera de M uniendo dos nodos "S", o un arco de M uniendo dos nodos "T", en el siguiente paso de etiquetado siempre uno de los nodos nos conducirá al otro, y viceversa. No es difícil ver que entonces se habrá detectado un ciclo impar en la gráfica, y por tanto una floración. Puesto que entre dos nodos "S" o entre dos nodos "T" hay un número par de arcos, al considerar el arco que une los dos nodos "S" o los dos nodos "T", se forma un ciclo alternante de longitud impar, es decir, la floración mencionada.

Para que el algoritmo continúe sin los tropiezos comentados en la sección anterior, siempre que una floración sea descubierta, se le 'encogerá' reemplazando la gráfica original G por G ctr B. El nodo correspondiente a B en G ctr B se llamará PSEUDONODO, y se le asignará etiqueta "S" a fin de continuar la construcción del árbol alternante.

Este proceso de construcción de árboles alternantes puede implicar varias operaciones de encogimiento. Aún más, floraciones podrían ser contraidas dentro de otras floraciones en varios niveles de profundidad. Sin embargo, si se localiza una trayectoria aumentante en los árboles alternantes que al final resultasen (libres de floraciones), entonces existirá una trayectoria aumentante en la gráfica original G. La existencia de tal trayectoria se garantiza por sucesivas aplicaciones del siguiente teorema, que se demuestra en el apéndice.

TEOREMA 3.1d. Sea B floración con respecto a M en la gráfica G. Existe una trayectoria aumentante en G ctr B con respecto a M\B, si y sólo si existe una trayectoria aumentante en G con respecto a M.

El algoritmo que resuelve el apareamiento no-bipartita se fundamenta en el teorema anterior.

A continuación se da un ejemplo que ilustra el funcionamiento del algoritmo. Consideremos la gráfica de la Figura 3.5, donde los arcos ondulados indican apareamiento y se puede distinguir la trayectoria aumentante P: 1-2-5-4-7-6-8-9-10-11. El ejemplo construirá de forma sistemática tal trayectoria aumentante.

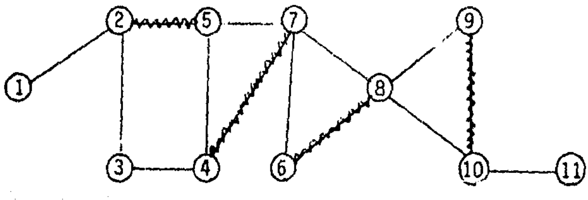


Figura 3.5

Comenzaremos con el nodo 1 como raíz de un árbol alternante, con la etiqueta "S:0". Al nodo 2 se le asigna la etiqueta "T:1", y continuando por el arco (2,5) se le asigna al nodo 5 la etiqueta "S:2". A partir del nodo 5 se etiquetan los nodos 4 y 7 con "T:5" y se descubre el arco apareado (4,7) uniendo tales nodos. Entonces se forma la floración B1 y es reemplazada por el pseudonodo B1, como se ve en el segundo diagrama. Al pseudonodo B1 se le asigna la misma etiqueta "S" que tenía su nodo base, y esta etiqueta es explorada. Al continuar el etiquetado se detectan y "encogen" las floraciones B2 y B3, como se aprecia en el tercer diagrama. Finalmente, una trayectoria aumentante se encuentra en G ctr B1 ctr B2 ctr B3, como se muestra en el cuarto diagrama.

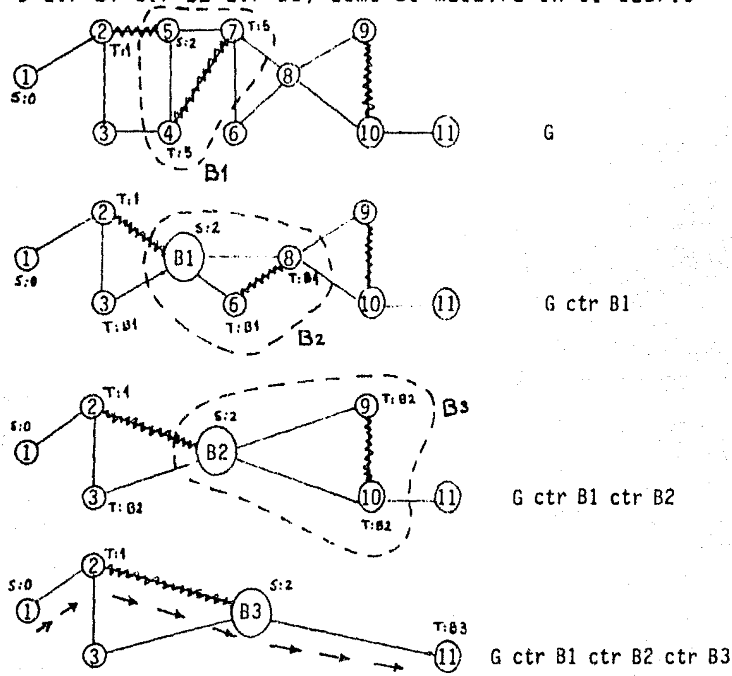


Figura 3.6

La construcción de una trayectoria aumentante en la gráfica original G a partir de la correspondiente en G ctr B1 ctr B2 ctr B3 es como sigue. Primero, el rastreo regresivo en la última gráfica produce la secuencia de nodos: 11-B3-2-1. Es necesario entonces hallar una trayectoria alternante en G ctr B1 ctr B2, que atraviese la floración B3. La trayectoria apropiada pasa por los nodos: 10-9-B2. De manera análoga, la trayectoria a través de B2 en la gráfica G ctr B1 es. 8-6-B1, y la que atraviesa por B1 en G es: 7-4-5. Al unir todas esas partes se logra la trayectoria aumentante deseada en G: 1-2-5-4-7-6-8-9-10-11.

Como puede verse del ejemplo, en el algoritmo hay dos tareas principales a ejecutar. Primero, es necesario detectar y "encoger" las floraciones. Segundo, deben poderse encontrar trayectorias alternantes adecuadas a través de las floraciones encogidas, de modo que una trayectoria aumentante en la gráfica original pueda reconstruirse.

Estas tareas no son demasiado difíciles de efectuar. Sin embargo, no es trivial el desarrollar estas operaciones de la forma más eficiente posible.

(3.3) ALGORITMO DE SOLUCION.

El algoritmo que se presenta en esta sección es una generalización del algoritmo 2.3.1 del caso bipartita. El tratamiento que se le da a las floraciones es la parte central del algoritmo. En realidad, el procedimiento de etiquetado que se usa no necesita de una auténtica contracción de las floraciones; más bien éstas se manejan 'como si se hubieran encogido'. Para lograr esto es suficiente llevar tan sólo un registro de las floraciones más externas, mismas que son identificadas por sus nodos-base. Así, a cada nodo 'i' de la gráfica se le asocia un índice 'b(i)' que indica el nodo-base de la floración más exterior en la que está contenido. Cuando un nodo 'i' no está dentro de ninguna floración el índice es: b(i)=i; y cuando una nueva floración es encontrada, su respectivo nodo-base 'n' es identificado, y la asignación b(i)=n se hace para todos los nodos dentro de la nueva floración. De este modo, una floración determinada queda identificada por su nodo-base.

En cada iteración, dado el apareamiento inicial (que puede ser vacío), el algoritmo trata de encontrar: o una trayectoria aumentante o una floración respecto a tal apareamiento. En el primer caso, el resultado es aumentar el apareamiento, y en el segundo, identificar la base de la floración y asignar etiquetas a los nodos que la forman, a fin de poder cruzar por la floración cuando ésta forme parte de alguna trayectoria aumentante.

Al comienzo de cada iteración, se asigna la etiqueta "S:0" (indicando nodo-raíz) a cada nodo expuesto de la gráfica. Después de eso, etiquetas "S" y "T" son aplicadas a los demás nodos. Una etiqueta "S" en un nodo 'j' indica que existe una trayectoria alternante de longitud par entre 'j' y el nodo-raiz; mientras que una etiqueta "T" indica el caso de trayectoria alternante de longitud impar. Las trayectorias aumentantes surgen entre nodos-raíz de árboles distintos, como sugiere la Figura 3.7.

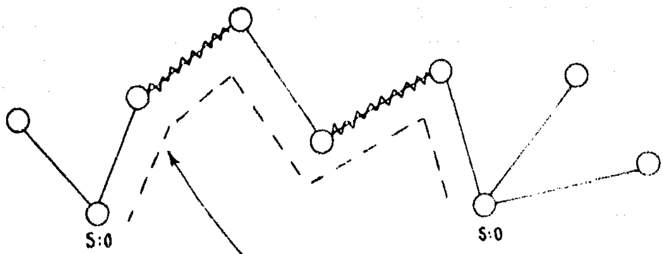


Figura 3.7

Continuando con el etiquetado, supongamos que se descubre un arco (i,j) fuera del apareamiento inicial con los nodos 'i', 'j' teniendo etiquetas "S", o un arco (i,j) del apareamiento inicial con los nodos 'i', 'j' teniendo etiquetas "T". Supóngase además que 'i', 'j' no están dentro de la misma floración, es decir, que $b(i) \neq b(j)$. Entonces, si 'i' y 'j' pertenecen a diferentes árboles alternantes, se ha encontrado una trayectoria aumentante, y en caso contrario, una nueva floración más exterior se ha formado. Para saber cuál de las dos posibilidades se tiene, basta con efectuar rastreo regresivo sobre las etiquetas a partir de los nodos 'i' y 'j'. Si se alcanzan nodos-raíz distintos, se obtiene una trayectoria aumentante; en otro caso se ha encontrado una nueva floración más exterior.

(3.3.1) CONSTRUCCION Y ETIQUETADO DE FLORACIONES.

Ya que se ha encontrado una floración en la gráfica, se debe determinar el conjunto de nodos que la forman, así como el nodo-base de la misma. Esto se hace como sigue.

Haciendo el rastreo regresivo desde los nodos 'i', 'j' se obtienen dos sucesiones de nodos:

$$i1,\ i2,\ i3,\ \ldots\ ,ip$$

$$j1,\ j2,\ j3,\ \ldots\ ,jq$$

donde $i1=j1$ (el nodo-raíz del árbol alternante), $ip=i$ y $jq=j$ (donde inició el rastreo). Puesto que $i1=j1$ y $ip \neq jq$, debe existir un índice 'm', tal que $i1=j1,\ i2=j2,\ \ldots,\ im=jm$, y ocurra que: $im=i$ o $jm=j$ o $im+1 \neq jm+1$. La base de la nueva floración es ik, para $k \leq m$, donde $ik=b(im)$, y el tallo de la floración pasa por los nodos i1, i2, i3,...,ik.

La nueva floración contiene a todos los nodos 'n' que cumplen la siguiente condición:

$$b(n) \in \{\ b(im), b(im+1), \ldots, b(ip), b(jm+1), b(jm+2), \ldots, b(jq)\ \} \quad (*)$$

Por consiguiente, para todos los nodos 'n' que forman la floración se hace la asignación $b(n)=ik$. Un ejemplo de esta identificación se muestra en la Figura 3.8: se tienen etiquetas "S" en los nodos 7 y 8, con el arco (7,8) fuera del apareamiento inicial. Haciendo rastreo regresivo desde los nodos mencionados se logran las siguientes sucesiones:

1,2,3,4,5,6,7

y

1,2,3,4,5,9,8.

En este caso, im=jm=5 y b(5)=3, puesto que el nodo 5 ya formaba parte de una floración, con el nodo 3 como base. Los nodos 3,4,5,6,7, 8,9,10 y 11 están en la nueva floración, con base en el nodo 3 y tallo formado por 1,2 y 3.

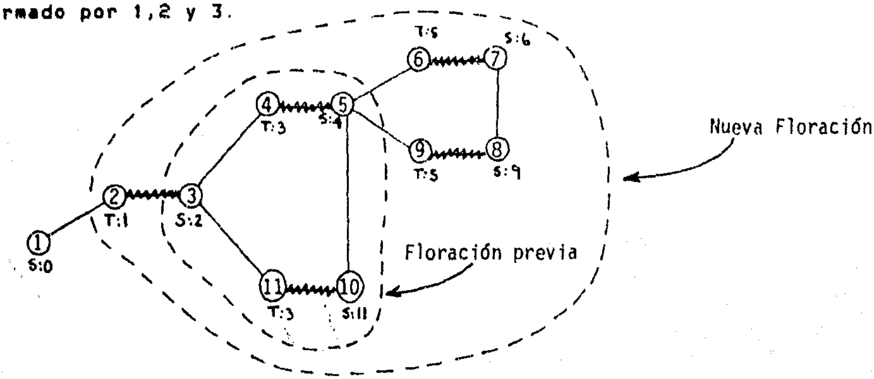


Figura 3.8

Finalmente, observamos que entre cualquier nodo no-base de la floración y el nodo raíz del árbol existen tanto una trayectoria alternante de longitud par como otra de longitud impar. De este modo, los nodos no-base de la floración deberían poseer tanto una etiqueta "S" como una etiqueta "T". La asignación de 'etiquetas faltantes' en los nodos no-base de la floración se explica enseguida.

Supóngase que la floración se halló por rastreo regresivo a partir de los nodos i=ip y j=jq, donde 'i', 'j' tienen etiqueta "S", y el arco (i,j) no pertenece al apareamiento inicial.(Reglas para el caso donde 'i','j' tienen etiqueta "T", con (i,j) en el apareamiento son análogas.)

Las etiquetas faltantes se aplicarán sólo en los nodos: im+1,im+2, im+3,...,ip,jm+1,jm+2,...,jq. Para facilitar la exposición se discutirá sólo el manejo de los nodos im+1,im+2,...,ip.(Para los nodos jm+1,...,jq es semejante) Las etiquetas "S" en los nodos: ip,ip-2, ...,im+2, y las etiquetas "T" en los nodos ip-1,ip-3,...,im+1 fueron usadas en el rastreo regresivo. Por tanto, cualquier etiqueta faltante debe ser etiqueta "T" en ip, ip-2, ...,im+2, o etiqueta "S" en ip-1, ip-3,...,im+1. La etiqueta perdida que se asigne a un nodo 'ik' debe ser tal que al efectuar rastreo desde ella produzca la sucesión de nodos: ik, ik+1, ...,ip, jq, jq-1, ...,j1.

Para aclarar el modo de proceder, asignemos etiquetas faltantes a los nodos im+1, im+2, ...,ip en orden. Supóngase que al nodo 'ik' le falta etiqueta "S". Entonces necesariamente el arco (ik,ik+1) pertenece al apareamiento original, y al nodo ik+1 debe faltarle etiqueta "T". Asignaremos a 'ik' la etiqueta "S:ik+1", y la etiqueta "T" que se ponga al nodo 'ik+1' producirá el rastreo regresivo correcto.

De la misma forma, si al nodo 'ik' le falta una etiqueta "T", entonces el arco (ik,ik+1) no está dentro del apareamiento original. Si además al nodo 'ik+1' le falta etiqueta "S", asignaremos a 'ik' la etiqueta "T:ik+1". La etiqueta "S" que se ponga en 'ik+1' nos dará el rastreo regresivo adecuado.

Pero ahora supóngase que 'ik' carece de etiqueta "T", y que 'ik+1' ya tiene etiqueta "S". Entonces 'ik' debe ser la base de una floración que existía previamente conteniendo a 'ik+1', de modo que en el rastreo regresivo desde la etiqueta "S" de 'ik+1' volveremos de nueva cuenta a 'ik'. Consecuentemente no debe asignarse a 'ik' la etiqueta "T:ik+1".

Para resolver este problema encontraremos el último nodo 'iz' en la sucesión: ik+1, ik+2, . . . , ip que esté contenido en esta floración previamente existente con 'ik' como base. Asignaremos entonces a 'ik' la etiqueta especial "T:iz+1,iz". Esta etiqueta significa que hay una trayectoria alternante impar entre 'ik' y el nodo raíz del árbol. Para hallar tal trayectoria se hace rastreo regresivo partiendo de las etiquetas "S": desde el nodo 'iz+1' hasta la raíz y desde 'iz' hasta 'ik'. Los arcos así encontrados, junto con el arco (iz,iz+1), ordenados adecuadamente, forman la trayectoria deseada. Un ejemplo de la aplicación de etiquetas dentro de una floración se ve en la Figura 3.9 a continuación.

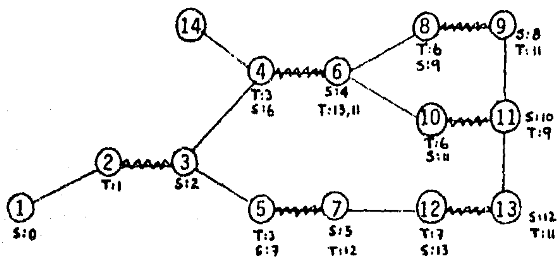


Figura 3.9

El algoritmo completo de apareamiento no-bipartita puede entonces resumirse como sigue.

ALGORITMO 3.3.1. MAXIMO APAREAMIENTO NO-BIPARTITA.

Paso 0). INICIO. Se tiene una gráfica general G=(N,A) y un apareamiento inicial M que puede ser vacío. Asígnese b(i)=i para todos los nodos de G. No hay nodos con etiqueta.

Paso 1). ETIQUETADO.
(1.0) Coloque la etiqueta "S:0" a cada nodo expuesto.
(1.1) Si todas las etiquetas han sido examinadas ir al Paso 4. En caso contrario, encuentre un nodo 'i' con etiqueta sin examinar. Si la etiqueta es "S", ir al Paso 1.2; si la etiqueta es "T", ir al Paso 1.3.
(1.2) Examine la etiqueta "S" en el nodo 'i' haciendo el siguiente procedimiento para cada arco (i,j) fuera de M que incida en 'i'. Si b(i)=b(j) no hacer nada. En caso contrario, si 'j' tiene etiqueta "S", haga rastreo regresivo partiendo de las etiquetas "S" en los nodos 'i', 'j', y si se alcanzan distintos nodos-raíz, ir al Paso 2; si se alcanza el mismo nodo-raíz, ir al Paso 3. Si el nodo 'j' no tiene ni etiqueta "S" ni etiqueta "T", aplique la etiqueta "T:i" al nodo 'j'.
Una vez examinado el nodo 'i', regresar al Paso 1.1.
(1.3) Examine la etiqueta "T" en el nodo 'i' como sigue. Encuentre el único arco (i,j) de M que incide en 'i'. Si b(i)=b(j) no hacer nada. En caso contrario, si 'j' tiene etiqueta "T", haga rastreo regresivo partiendo de las etiquetas "T" en los nodos 'i', 'j', y si se alcanzan diferentes nodos-raíz, ir al Paso 2; si se alcanza el mismo nodo-raíz, ir al Paso 3. Si el nodo 'j' no tiene ni etiqueta "S" ni etiqueta "T", aplique la etiqueta "S:i" al nodo 'j'.
Regrese al Paso 1.1.

Paso 2). AUMENTO
Una trayectoria aumentante se ha encontrado en el Paso 1.2 o en el 1.3. Aumente el apareamiento M. Borre todas las etiquetas de los nodos y asigne b(i)=i para todos los nodos de G. Regrese a 1.0.

Paso 3). FLORACION.
Una floración se ha formado en el Paso 1.2 o en el Paso 1.3. Determine el nodo-base y los nodos que forman la floración como se explicó al inicio de esta sección. Añada etiquetas faltantes para los nodos no-base en la nueva floración. Corrija el valor b(i) para los nodos dentro de la floración. Regrese a 1.2 o 1.3, según el caso.

Paso 4). ETIQUETADO HUNGARO.
El etiquetado que se tiene es húngaro. No existen trayectorias aumentantes, y el apareamiento M es de máxima cardinalidad. Alto.

(3.3.2) EJEMPLO DE APAREAMIENTO NO-BIPARTITA.

Sea G una gráfica no-bipartita con el apareamiento inicial mostrado en la Figura 3.10a). El identificador de la floración más externa a que pertenece cada nodo 'j', b(j), es como sigue:

j	1	2	3	4	5	6	7	8	9	10	11	12	13	14
b(j)	1	2	3	4	5	6	7	8	9	10	11	12	13	14

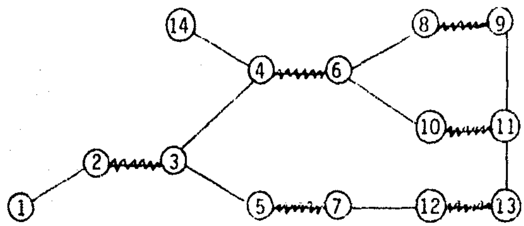


Figura 3.10a).

Etapa I: Tomando como raíz al nodo 1, se logra el etiquetado de la Figura 3.10b). Explorando el nodo 9 se detecta la etiqueta "S" en el nodo 11, y se descubre la floración B1 que forman los nodos: 6, 8, 9, 11 y 10, con base en el nodo 6. Los nuevos valores del identificador b(j) son ahora:

j	1	2	3	4	5	6	7	8	9	10	11	12	13	14
b(j)	1	2	3	4	5	6	7	6	6	6	6	12	13	14

Asignando etiquetas faltantes a los nodos no-base de B1 se obtiene el etiquetado de la Figura 3.10c).

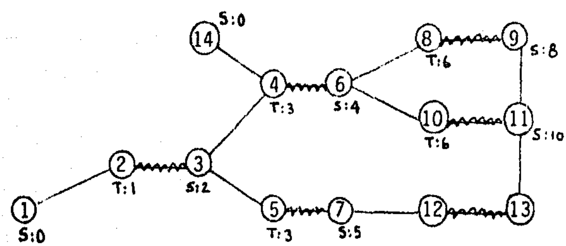


Figura 3.10b).

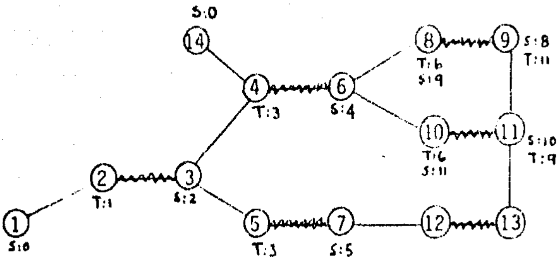


Figura 3.10c).

Etapa II: Continuando con la exploración en el nodo 11, el arco (9,11) no se considera para nada, puesto que b(9) = 6 = b(11); mientras que el arco (11,13) permite asignar la etiqueta "T:11" en el nodo 13. Prosiguiendo con el etiquetado, al explorar el nodo 12 se encuentra la etiqueta "S" en el nodo 7, con lo que se descubre una nueva floración B2, con base en el nodo 3. Usando la notación de la sección 3.3.1, ip=7 y jq=12 son dos nodos de un mismo árbol con etiqueta "S" que están unidos por un arco no-apareado, con im=3, y las sucesiones: im+1,...,ip = 4,6,10,11,13,12 y jm+1,...,jq = 5,7. En esta floración B2 cualquier nodo 'k' debe cumplir la condición (*) de la sección 3.3.1:

$$b(k) \in \{ b(3),b(4),b(6),b(10),b(11),b(13),b(12),b(5),b(7) \}.$$

o sea:

$$b(k) \in \{ 3,4,6,13,12,7,5 \}.$$

así, los nodos dentro de B2 son: 3,4,6,8,10,9,11,13,12,7 y 5. Los valores del identificador b(j) son ahora:

j	1	2	3	4	5	6	7	8	9	10	11	12	13	14
b(j)	1	2	3	3	3	3	3	3	3	3	3	3	3	14

Las etiquetas faltantes en nodos no-base de B2 se asignan con facilidad en los nodos 5,7 y 4 como se aprecia en la Figura 3.10d). Pero al considerar el nodo 6 (que carece de etiqueta "T"), se descubre una etiqueta "S" en el nodo 10. Esto ocurre porque 6 es nodo base de la floración B1 ya existente.

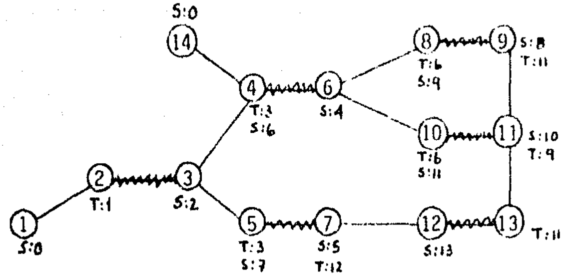


Figura 3.10d).

En la notación de la sección 3.3.1, ik=6 es un nodo que carece de etiqueta "T", mientras que ik+1=10 ya posee una etiqueta "S". En estos términos, el último nodo 'iz' de la sucesión ik+1,...,ip dentro de la floración B1 previa es iz=11. De esta manera, en el nodo 6 se asigna la etiqueta especial "T:13,11", indicando la existencia de una trayectoria alternante impar entre el nodo 6 y el nodo raíz 1. Las etiquetas faltantes que restan se asignan en los nodos 12 y 13, resultando el etiquetado de la Figura 3.10e).

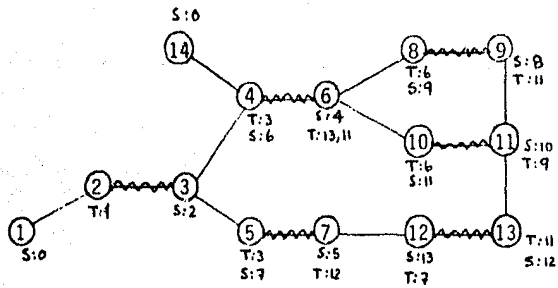


Figura 3.10e).

Etapa III: Al explorar el nodo 14 (que es raíz de un árbol) llegamos al nodo 4, y se inicia un rastreo regresivo a partir de la etiqueta "S" en tal nodo. Esto nos lleva enseguida al nodo 6 con la etiqueta "T:13,11". Es gracias a esta etiqueta especial que se encuentra una trayectoria alternante que atraviese la floración B2 y llegue al nodo raíz 1.

Como se explicó en la sección 3.3.1, tal trayectoria se construye por rastreo regresivo de la siguiente forma:

Primero, desde la etiqueta "S" en el nodo 13 hasta el nodo raíz 1, se logra la trayectoria P1: 1-2-3-5-7-12-13.

Segundo, desde la etiqueta "S" en el nodo 11 hasta el nodo 6, se tiene la trayectoria P2: 6-10-11.

Finalmente, agregando el arco (13,11) se logra la trayectoria completa P: 1-2-3-5-7-12-13-11-10-6.

Ya que la trayectoria final desde el nodo 14 al nodo 1 une dos nodos raíz, se ha conseguido una trayectoria aumentante. Actualizando el apareamiento original con esta trayectoria, obtenemos el apareamiento de la Figura 3.10f), que es solución óptima al máximo apareamiento en G (hay en total 7 arcos apareados).

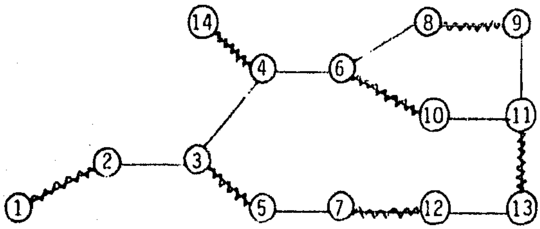


Figura 3.10f).

(3.4) DUALIDAD.

En esta sección se establece un teorema de dualidad para el problema de máximo apareamiento no-bipartita que es una generalización del Teorema de König-Egerváry (Proposición 2.3a) del caso bipartita

Como se recordará, en el caso bipartita el problema dual del apareamiento máximo es un problema de recubrimiento mínimo de nodos. En el caso de una gráfica no-bipartita es fácil ver que un recubrimiento mínimo de nodos no coincide necesariamente con un apareamiento máximo. Un ejemplo sencillo de esto se tiene al analizar la gráfica formada por un ciclo impar de 2k+1 nodos. En tal ciclo, el apareamiento máximo es de 'k' arcos, mientras que el recubrimiento mínimo debe incluir al menos 'k+1' nodos. La figura 3.11 muestra una gráfica G formando un ciclo de longitud 5, con un apareamiento máximo de 2 arcos señalado con línea ondulada, y donde no se puede conseguir un recubrimiento de nodos menor de 3.

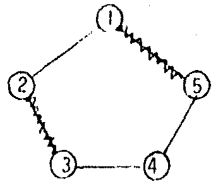


Figura 3.11.

El problema dual del apareamiento no-bipartita es también un problema de recubrimiento mínimo, aunque no propiamente de nodos, sino de una clase especial de conjuntos de nodos, llamados CONJUNTOS IMPARES DE NODOS, que se describen a continuación.

Sean $G=(N,A)$ una gráfica no-bipartita, y $Q=(N1,N2,\ldots,Np)$ una familia de subconjuntos de nodos de G, donde cada Ni contiene un número impar de elementos. Cuando $|Ni|=1$, se dice que Ni CUBRE todos los arcos que inciden en el único nodo de Ni, y que la CAPACIDAD de Ni, denotada $c(Ni)$, es 1. Cuando $|Ni|=2k+1$ para $k \geq 1$, se dice que Ni CUBRE todos los arcos con ambos extremos incidiendo en nodos pertenecientes a Ni, y que la CAPACIDAD de Ni es $c(Ni)=k$. La familia Q es llamada RECUBRIMIENTO DE CONJUNTOS IMPARES DE NODOS, siempre que cada arco en G sea cubierto por al menos un $Ni \in Q$. La CAPACIDAD DE Q, denotada $c(Q)$, es igual a la suma de las capacidades de todos los conjuntos impares contenidos en Q.

El teorema de dualidad para el problema de apareamiento máximo en gráficas no-bipartitas establece que en cualquier gráfica G, el máximo número de arcos en un apareamiento coincide con la mínima cardinalidad de un recubrimiento de conjuntos impares de nodos de G. Formalmente se expresa como sigue:

PROPOSICION 3.4a. (J.Edmonds) Para cualquier gráfica $G=(N,A)$:

$$\max\ (\ |M| : M \text{ es apareamiento en } G\ ) =$$

$$\min\ \Big(\sum_{Ni \in Q} c(Ni) : Q \text{ es recubrimiento de conjuntos impares de nodos de } G\ \Big).$$

Demostración.

Lo primero que se probará es que, dados un apareamiento M y un recubrimiento de nodos Q cualesquiera en G, se cumple:

$$|M| \leq c(Q) \qquad (**).$$

Por definición de Q, todo arco en M es cubierto por algún Ni Q, mientras que al ser M apareamiento no hay más de un arco apareado incidiendo en un nodo. Entonces, el número de arcos apareados que puede cubrir cada $Ni \in Q$ es, a lo más, igual a su propia capacidad $c(Ni)$.
De este modo, si sumamos las capacidades de todos los Ni para obtener $c(Q)$, estaremos contando todos los arcos de M y además las posibles repeticiones (arcos de M cubiertos por más de un Ni), lográndose un

número que excede o iguala a $|M|$

Para terminar, se mostrará la construcción de un recubrimiento mínimo $Q*$ a partir del apareamiento $M*$ resultante al finalizar el algoritmo 3.3.1 de la sección anterior, tal que: $|M*| = c(Q*)$. En virtud de la condición (**) verificada en el párrafo anterior, $M*$ y $Q*$ deben ser maximal y minimal, respectivamente.

Como se vió en la sección anterior, el algoritmo de máximo apareamiento no-bipartita avanza construyendo árboles alternantes, hallando y 'encogiendo' floraciones, localizando trayectorias aumentantes y actualizando el apareamiento en turno, hasta el punto en que los árboles presentes en la gráfica son húngaros: no se puede hacerlos crecer y tampoco hay trayectorias aumentantes. Sea $M*$ el apareamiento que queda cuando el algoritmo finaliza.

Cuando el algoritmo se detiene, para cualquier arco de G se cumple sólo una de las siguientes condiciones:

1) el arco toca nodos del bosque húngaro (dentro del cual no existen trayectorias aumentantes y por tanto no hay dos nodos "S" unidos por arco no-apareado), de modo que debe incidir en algún nodo "T".

2) el arco pertenece a una floración.

3) el arco toca con ambos extremos nodos sin etiqueta alguna (que forman la parte de la gráfica con 'actualización' por trayectorias aumentantes más reciente; Paso 2 del algoritmo).

De estas observaciones se construye un recubrimiento de conjuntos impares de nodos $Q*$ en G, como sigue: Cada nodo 'i' con etiqueta "T" forma un conjunto 'N_i' de capacidad 1 en $Q*$. Por supuesto, el número de nodos "T" en el bosque húngaro es igual al número de arcos de $M*$ en dicho bosque.

Los nodos de cada floración B_j, con $2k+1$ nodos, forman un conjunto 'N_j' de capacidad impar $c(N_j)=2k+1$, conteniendo 'k' arcos de $M*$.

Finalmente, si hay 'n' arcos con ambos extremos en nodos sin etiqueta, habrá en total '2n' de tales nodos, y 'n' arcos apareados entre ellos. En caso de que $n=0$, el recubrimiento $Q*$ está completo, y se cumple $c(Q*)=|M*|$. En caso de que $n=1$, se elige cualquiera de los extremos del arco (p,q) como conjunto 'N_p' de capacidad igual a uno, y $Q*$ está completo. Cuando $n \geqslant 2$, se elige alguno de los 2n nodos como conjunto 'N_x' de capacidad uno, y los restantes $2n-1$ nodos forman un conjunto 'N_z' de capacidad impar conteniendo '$n-1$' arcos de $M*$. Así, se completa el recubrimiento $Q*$ que tiene la misma capacidad que la cardinalidad de $M*$. De la condición (**) se concluye entonces, que $M*$ debe ser apareamiento maximal y $Q*$ recubrimiento de conjuntos impares de nodos en G. Esto completa la demostración.

(3.5) APAREAMIENTO MAXIMO Y RECUBRIMIENTO DE ARCOS.

Al analizar los aspectos de dualidad del problema de apareamiento máximo, la idea de RECUBRIMIENTO DE NODOS en una gráfica surgió de modo natural. La relación entre apareamiento máximo y recubrimiento mínimo de nodos está resumida en los teoremas de König-Egerváry y de J. Edmonds (Prop. 2.3a. y 3.4a) presentados anteriormente.

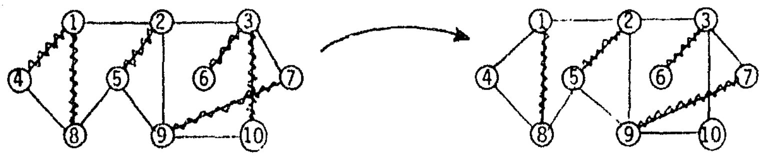
Prosiguiendo con la idea de recubrimiento de una gráfica, se puede pensar en recubrimientos de arcos con una definición como ésta:

Un RECUBRIMIENTO DE ARCOS en una gráfica G=(N,A) es un subconjunto C de arcos de A con la propiedad de que cada nodo de N es tocado por al menos un arco de C.

Para que la definición anterior tenga sentido se requiere que cada nodo de la gráfica tenga por lo menos grado 1 (al menos un arco incidente en él).

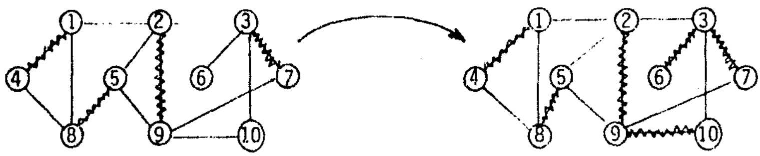
Cuando en una gráfica G existe un recubrimiento de arcos de cardinalidad mínima C*, es posible construir a partir de él un apareamiento máximo simplemente quitando de los nodos con arcos múltiples de C* todos excepto uno. Lo que quede de C* luego de esta eliminación será el apareamiento máximo mencionado.

La idea recíproca también se cumple. Si en una gráfica G existe un apareamiento máximo M*, se puede conseguir a partir de él un recubrimiento mínimo de arcos de G con sólo agregar a M* un arco en cada nodo expuesto; M* más los arcos añadidos forman un recubrimiento mínimo de arcos de G. En la Figura 3.12 se ilustra un ejemplo.



Recubrimiento mínimo C*

Apareamiento máximo M*c obtenido de C*



Apareamiento máximo M*

Recubrimiento mínimo C*m obtenido de M*

Figura 3.12

En el teorema que se presenta enseguida se formaliza la relación entre apareamiento máximo y recubrimiento de arcos mínimo.

PROPOSICION 3.5a Sean G=(N,A) una gráfica donde cada nodo tiene al menos grado uno, y C*,M* ⊂ A un recubrimiento de arcos mínimo y un apareamiento máximo de G, respectivamente.

Entonces, el apareamiento Mc obtenido de C* al eliminar todos excepto uno de los arcos incidentes en cada nodo, y el recubrimiento Cm obtenido al agregar a M* un arco en cada nodo expuesto, son maximal y minimal, respectivamente. Además, se cumple la relación:

$$|C^*| + |M^*| = |N| \qquad (***).$$

Demostración.

Consideremos primero un recubrimiento mínimo C*, y el apareamiento Mc obtenido al eliminar todos excepto uno de los arcos en nodos con incidencia múltiple. Fácilmente se verifica que no existe arco de C* que cubra dos nodos con incidencia múltiple, pues de ser así, la eliminación de tal arco podría disminuír la cardinalidad de C*, contradiciendo su carácter de mínimo. Dicho de otro modo, todo arco de C* tiene al menos un extremo que cubre tan sólo un nodo de G. Tomando en cuenta este hecho y la forma de construír Mc, resulta que en este apareamiento hay tantos nodos expuestos como arcos de C* se eliminaron en su construcción. Puesto que además cada arco de Mc satura (cubre) dos nodos de G, tenemos lo siguiente:

Núm. de nodos expuestos = Núm. de arcos eliminados de C* al construír Mc.

es decir:

$$|N| - 2|Mc| = |C^*| - |Mc|$$

de donde:

$$|C^*| + |Mc| = |N| \qquad (I).$$

Ahora, consideremos un apareamiento máximo M*, y el recubrimiento Cm obtenido al agregar a M* un arco en cada nodo expuesto. Los arcos añadidos a M* deben ser diferentes, pues en caso contrario, un arco repetido significaría un arco uniendo dos nodos expuestos, lo que podría usarse para aumentar el tamaño de M*, contradiciendo su carácter de máximo. Por tanto resulta lo siguiente:

Núm. de nodos expuestos = Núm. de arcos agregados a M* para construír Cm.

es decir:

$$|N| - 2|M^*| = |Cm| - |M^*|$$

de donde:

$$|Cm| + |M^*| = |N| \qquad (II).$$

Combinando (I) con (II) resulta:

$$|N| = |M^*|+|Cm| \geqslant |M^*|+|C^*| \geqslant |Mc|+|C^*| = |N|$$

donde las desigualdades son válidas por el carácter mínimo de C^* y máximo de M^*.

La conclusión de la ecuación (***) en el enunciado del teorema es inmediata, así como la condición de mínimo en Cm y de máximo en Mc.

(3.6) APAREAMIENTO PONDERADO NO-BIPARTITA

Dada una gráfica no-bipartita $G=(N,A)$ donde cada arco en A tiene asociado un peso (positivo, negativo o cero), el problema de apareamiento ponderado consiste en hallar un apareamiento tal que la suma de los arcos que lo forman sea máxima.

Como en el caso bipartita, el algoritmo de solución se desarrolla por un método primal-dual. El punto de partida está en el teorema de dualidad para máximo apareamiento (Prop. 3.4a), que motiva un conjunto de desigualdades lineales, relacionadas con los conjuntos impares de nodos de G, y que cualquier apareamiento satisface.

Para comenzar, consideremos un apareamiento M cualquiera en G, y la variable 'Xij' definida como:

$$Xij = \begin{cases} 1 & \text{si el arco } (i,j) \text{ está en } M. \\ 0 & \text{en caso contrario.} \end{cases}$$

Si Rp es conjunto impar de nodos en G, claramente $|Rp|=2rp+1$ para algún entero $rp \geq 1$, y las variables Xij cumplen:

$$\sum_{i \in Rp} \sum_{j \in Rp} Xij \leq rp \qquad (\#)$$

Puesto que la familia de subconjuntos impares de nodos en G es finita (por ser N finito), puede representarse la totalidad de restricciones del tipo (#) por la desigualdad matricial:

$$R\,X \leq r$$

donde R es una matriz con tantas filas como subconjuntos impares de nodos tenga N, y tantas columnas como arcos tenga G, llamada matriz de incidencia conjuntos impares de nodos - arcos, X es el vector con componentes Xij (representando a los arcos de G) y $r=(r1,r2,\ldots,rm)$ es un vector cuya i-ésima componente es la capacidad del i-ésimo conjunto impar de nodos, es decir: $|Ri|=2ri+1$.

Con estos antecedentes, el programa lineal que representa el problema de apareamiento ponderado, se plantea fácilmente como sigue:

$$\max\ W\ X$$

sujeta a:

$$\begin{bmatrix} A \\ \hline R \end{bmatrix} X \leqslant \begin{bmatrix} 1 \\ \hline r \end{bmatrix} \qquad \text{(P)}$$

$$X \geqslant 0$$

donde 'A' es la matriz de incidencia nodos-arcos de G, 'R' es la matriz de incidencia conjuntos impares de nodos-arcos de G, 'X' es un vector cuyas coordenadas son Xij (representando a los arcos de G), 'r' es el vector de capacidades de conjuntos impares de nodos, 'W' es el vector de pesos asociados a los arcos de G, y '1' es un vector con todas sus coordenadas iguales a uno.

En el caso especial en que wij=1 para todo arco de G, el problema (P) corresponde al máximo apareamiento estudiado con anterioridad, y del teorema de dualidad (Proposición 3.4a) se garantiza que existe una solución entera. Para el caso general, se verificará la existencia de solución entera al desarrollar un algoritmo de tipo primal-dual.

El problema dual que corresponde al programa (P) se plantea como sigue:

$$\min\ [U:Z] \begin{bmatrix} 1 \\ \hline r \end{bmatrix} = \sum_i U_i + \sum_p r_p Z_p$$

sujeta a:

$$[U:Z] \begin{bmatrix} A \\ \hline R \end{bmatrix} \geqslant W \qquad \text{(D)}$$

$$[U:Z] \geqslant 0$$

donde 'Ui' y 'Zp' se relacionan con el nodo 'i' y el conjunto impar Rp, respectivamente.

Para las soluciones óptimas dual y primal las condiciones de holgura complementaria necesarias y suficientes son:

$$X_{ij} > 0 \implies U_i + U_j + \sum_{((i,j))\ R_p} Z_p = w_{ij} \qquad \text{(H1)}$$

$$U_i > 0 \implies \sum_j X_{ij} = 1 \qquad \text{(H2)}$$

$$Z_p > 0 \implies [R_p][X] = r_p \qquad \text{(H3)}$$

Análogamente al caso bipartita, el algoritmo conserva factibilidad dual y primal en todo momento, y además satisface todas las condiciones de holgura complementaria, salvo las (H2). El número de condiciones (H2) insatisfechas decrece monótonamente a lo largo del proceso.

En la versión del algoritmo que presenta Lawler [1], el cálculo se inicia con el apareamiento factible vacío $X=\emptyset$ y con la solución dual factible:

$$U_i = W \quad \text{para toda } i$$

$$Z_p = 0 \quad \text{para toda } p$$

donde W es un valor suficientemente grande, como

$$W = 0.5 \max_{i,j}(w_{ij})$$

Las soluciones primal y dual dadas, satisfacen las condiciones de holgura (H1) y (H3), pero no las (H2).

En el paso genérico del algoritmo, X es solución primal factible, se cumplen las condiciones (H1) y (H3), pero algunas de las (H2) no. Lo que el algoritmo intenta entonces es encontrar una trayectoria aumentante en la subgráfica de G formada por nodos y arcos que satisfacen (H1) y (H3), es decir, la subgráfica formada al contraer todas las floraciones 'p' con $Z_p > 0$ y eliminar todos los arcos (i,j) para los cuales: $U_i + U_j + \sum Z_p > w_{ij}$ (que violan H1).

Si se encuentra una trayectoria aumentante, debe extenderse entre dos nodos 'i', 'j' con $U_i, U_j > 0$. De este modo, luego de la actualización del apareamiento, el número de condiciones (H2) insatisfechas habrá disminuído en dos. No es difícil ver que al actualizar el apareamiento, los cambios que se den en floraciones son tales que el número de arcos apareados dentro de cada floración sigue siendo igual a la capacidad de ésta (vista como conjunto impar), con lo que el apareamiento dentro de la floración es maximal, y las condiciones (H3) se mantienen satisfechas. Por otra parte, como los arcos (i,j) dentro de la trayectoria aumentante cumplen $U_i + U_j + \sum Z_p = w_{ij}$, las condiciones (H1) mantienen su validez.

En caso de que no se encuentren trayectorias aumentantes, sólo hay dos posibilidades: que los árboles formados sean húngaros, con lo que el apareamiento en turno es maximal, o que no se puedan agregar más arcos a los árboles debido a que todo arco (i,j) disponible para tal propósito cumple $U_i + U_j + \sum Z_p > w_{ij}$. En este último caso, el algoritmo intenta disminuír el valor del lado izquierdo de la desigualdad anterior, restándole un adecuado valor 'd' positivo, de modo que en al menos un arco (i,j), la desigualdad original se convierta en igualdad estricta, haciendo al arco (i,j) candidato a incluírse en los árboles alternantes. Se cumplen entonces (H1) y decrece en 2 el número de condiciones (H2) insatisfechas. La forma concreta de alterar las variables duales para lograr lo anterior es como sigue:

Para no hacer la descripción demasiado engorrosa, llamaremos a una floración 'externa' si no está contenida dentro de otra floración mayor. Entonces, para cada nodo 'i' con etiqueta "S" y cada nodo 'i' contenido en una floración externa cuyo nodo-base tenga etiqueta "S", el valor 'd' es restado a U_i Para cada nodo 'i' con etiqueta "T" y cada nodo 'i' contenido en una floración externa cuyo nodo-base tenga etiqueta "T", el valor 'd' es sumado a U_i. Para cada floración externa 'p' cuyo nodo-base tenga etiqueta "S", el valor 2d es sumado a Z_p, y para cada floración 'p' cuyo nodo-base tenga etiqueta "T", el valor 2d es restado de Z_p.

Si un arco (i,j) está dentro de una floración, el efecto total de los cambios de variables duales en $U_i + U_j + \sum Z_p$ es nulo. Pero, si 'i' tiene una etiqueta "S" o está contenido en una floración externa cuyo nodo-base tenga etiqueta "S", mientras que 'j' no tiene etiqueta alguna, el efecto neto en $U_i + U_j + \sum Z_p$ es igual a -d, con lo que disminuye el lado izquierdo de la desigualdad ya mencionada.

Si (i,j) es un arco tal que tanto 'i' como 'j' pertenecen a floraciones externas con nodos-base "S", el efecto total en $U_i+U_j+\sum Z_p$ será -2d, causando una disminución de valor, tal vez hasta igualar w_{ij}. En otros casos, cuando 'i' tiene etiqueta "T" pero 'j' carece de etiqueta, o cuando tanto 'i' como 'j' están en floraciones externas con nodos-base "T", el efecto neto en $U_i+ U_j+ \sum Z_p$ será d o 2d, lo que aumentará el valor original, y se excluirá la posibilidad de que dichos arcos se añadan a los árboles alternantes. Resumiendo, el valor máximo que puede tomar 'd' está restringido por las cuatro condiciones siguientes:

C1) Si 'i' es nodo con etiqueta "S" o pertenece a una floración externa con nodo-base "S", se debe cumplir: $U_i - d \geqslant 0$.

C2) Si (i,j) es un arco tal que tanto 'i' como 'j' tienen etiqueta "S" o pertencen a floraciones externas distintas con nodos-base "S", debe cumplirse: $(U_i-d) + (U_j-d) \geqslant w_{ij}$.

C3) Si el nodo-base de una floración 'p' tiene etiqueta "T", debe cumplirse: $Z_p -2d \geqslant 0$.

C4) Si (i,j) es un arco tal que 'i' tiene etiqueta "S" o pertenece a una floración externa con nodo-base "S", mientras que 'j' no tiene etiqueta o pertenece a una floración externa con nodo-base carente de etiqueta, se requiere: $(U_i-d) + U_j \geqslant w_{ij}$.

Si llamamos 'd1' al máximo valor permitido para d sujeto sólo a C1, 'd2' al máximo valor permitido a d sujeto sólo a C2, y análogamente 'd3' y 'd4' para los máximos que puede tomar d sujeto sólo a C3 y C4 respectivamente, es claro que el mayor valor que puede tener d sujeto a C1, C2, C3 y C4 será:

$$d = \min(d1,d2,d3,d4).$$

Una vez encontrado el máximo posible para d, pueden ocurrir cuatro casos:

Si d=d1, la nueva solución dual es tal que todas las condiciones H2 se cumplen, con lo que tanto la nueva solución dual como la primal asociada son óptimas, y el apareamiento resultante es maximal. (Conviene recordar que se eligió el mismo valor inicial en las variables Ui, de modo que luego de restar 'd', se tendrá el mismo valor mínimo de Ui en cada nodo expuesto)

Si d=d2, se habrá encontrado una trayectoria aumentante o se habrá formado una nueva floración.

Si d=d3, una floración externa se podrá 'extender'

Si d=d4, al menos un nuevo arco se podrá agregar a los árboles alternantes

Con estos antecedentes, se puede hacer una descripción a grandes rasgos del algoritmo:

Paso 0). INICIO.
Comience con el apareamiento nulo $X=\emptyset$, y $U_i=0\ 5 \max(w_{ij})$ como soluciones primal y dual, respectivamente.

Paso 1). ETIQUETADO.
Haga de cada nodo expuesto la raíz de un árbol alternante, y construya los árboles alternantes usando un proceso de etiquetado, utilizando solamente arcos (i,j) que cumplan la condición:

$$U_i + U_j + \sum Z_p = w_{ij}.$$

Si se encuentra una trayectoria aumentante, ir al Paso 2. Si se ha formado una floración, ir al Paso 3. Si los árboles son húngaros, ir al Paso 4.

Paso 2). AUMENTO.
Localice la trayectoria aumentante trazando la misma a través de floraciones contraídas. Actualice el apareamiento, borre todas las etiquetas de nodos y pseudonodos (floraciones contraídas) y regrese al Paso 1.

Paso 3). FLORACIONES.
Identifique la floración formada y contráigala en la gráfica. Al pseudonodo resultante se le asigna etiqueta "S" y su variable 'Z' se hace cero. Regrese al Paso 1.

Paso 4). CAMBIO EN VARIABLES DUALES.
Encuentre el máximo valor 'd' restringido por las condiciones C1 a C4, haciendo cambios adecuados en las variables duales. Si se cumplen las condiciones C1, alto; tanto el apareamiento como la solución dual son óptimos. En caso contrario, 'extienda' las floraciones con Zp=0 y regrese al Paso 1.

(3.7) ALGORITMO DE SOLUCION.

Como se ve de la sección anterior, el algoritmo para apareamiento ponderado no-bipartita resulta más complicado que el del caso bipartita. En esta sección se describen con cierta amplitud los detalles concretos que deben cuidarse para lograr una implementación del algoritmo de solución en un programa de computadora.

(3.7.1) TRATAMIENTO DE FLORACIONES.

Lo primero que se observa en el caso no-bipartita es que el tratamiento de las floraciones requiere más información. No basta con llevar solamente registro de las floraciones externas; cuando una floración externa es 'extendida' deben conocerse las floraciones inmediatas que tiene anidadas, para que tales floraciones puedan ser consideradas nuevamente como externas. Por otra parte, cuando hay aumento en el apareamiento, las floraciones con variables duales estrictamente positivas deben conservarse para su ulterior uso en la rutina de etiquetado. En suma, debe tenerse un registro completo de todas las floraciones, sus nodos-base y sus relaciones de anidamiento.

Como en el caso bipartita, el nodo base de la floración externa a la cual un nodo 'i' pertenezca se denotará por b(i). Las floraciones anidadas pueden tener el mismo nodo-base, (es decir, las floraciones no están únicamente determinadas por su nodo-base) pero no es posible tener dos floraciones externas distintas con la misma base. Para no hacer tediosa la redacción se llamará a un nodo 'i' nodo base cuando cumpla b(i)=i, aún cuando tal nodo no esté dentro de ninguna floración. Análogamente, la expresión "..nodos contenidos en la misma floración externa que i " tiene sentido a pesar de que 'i' no pertenezca a floración alguna.

En el caso bipartita, todas las floraciones son externas, y sus nodos-base siempre son "S"; en el caso no-bipartita, en cambio, se tienen las siguientes clases de floraciones:

(1) Floraciones sin etiqueta. Son las que corresponden a pseudonodos carentes de etiqueta. La floración es externa, y su variable dual es estrictamente positiva.

(2) Floraciones-S. Correspondientes a pseudonodos "S". El nodo base tiene etiqueta "S" pero no "T". La floración es externa, y su variable dual es mayor o igual a cero.

(2) Floraciones-T Correspondientes a pseudonodos "T". El nodo base tiene etiqueta "T" pero no "S". La floración es externa, y su variable dual estrictamente positiva.

(4) Floraciones internas. Corresponden a pseudonodos que han sido contraídos dentro de pseudonodos. El nodo base puede tener tanto etiqueta "S" como "T", y su variable dual es estrictamente positiva.

Finalmente, para etiquetar una floración "T" se debe proceder como sigue:

Supóngase que la etiqueta "S" en el nodo 'i' se explora, hallando un arco (i,j) fuera del apareamiento en turno que cumple:

$Ui+Uj-wij = 0$, $b(j) \neq b(i)$, $b(j) \neq j$, y b(j) sin etiqueta.

En este caso, el nodo 'j' pertenece a una floración sin etiqueta que recibirá la etiqueta "T" del nodo 'i'. En consecuencia, se aplicará la etiqueta especial "T:i,j" al nodo base b(j), con el mismo significado de etiqueta especial que se usó en el caso bipartita (Sección 3.3.1). Para mantener la notación, se colocará la etiqueta "T:i,j" en el nodo b(j) aún cuando b(j)=j. (en este caso, el segundo índice es ignorado durante el rastreo regresivo)

(3.7.2) CORRECCION DE ETIQUETAS LUEGO DE UN AUMENTO.

Supongamos que se encuentra una trayectoria aumentante que cruza por la floración B1. Es sencillo ver que al actualizar y aumentar el apareamiento en turno, los arcos apareados dentro de B1 ya no son los mismos que antes del aumento (aunque el número de ellos no varíe), y por tanto el nodo-base de B1 (nuevo nodo-base) también será distinto del nodo-base original (viejo nodo-base).

Esta observación sugiere que, luego de cada aumento en el apareamiento en turno, deben cuidarse los 3 aspectos siguientes:

1) Conservar todas las floraciones, para ulterior uso en el procedimiento de etiquetado.

2) Conservar todas las etiquetas en los nodos de las floraciones.

3) Corregir las etiquetas en los nodos que forman parte de la trayectoria aumentante.

Para llevar a cabo todo esto, se procede como sigue. Primero identificamos todas las floraciones (no solamente las externas) a través de las cuales pasa la trayectoria aumentante. A cada floración se le encuentra su nuevo nodo base. (La trayectoria aumentante conecta el viejo nodo base con el nuevo nodo base en cada floración por la que cruza.)

Para los nodos de la trayectoria aumentante que no sean ni viejos ni nuevos nodos base, solamente intercambiamos los índices de las etiquetas. Así, si uno de tales nodos tiene etiquetas "S:i" y "T:j", las nuevas etiquetas serán "S:j" y "T:i".

Cuando un nodo b es nuevo nodo base, buscamos la floración más interna en la cual está contenido, así como el viejo nodo base b' de tal floración. Se localizan los arcos (b,i) y (b',j) de la trayectoria aumentante, donde 'i','j' no pertenezcan a la floración mencionada, y las nuevas etiquetas para 'b' son "S:i" y " T:j,b' ".

Cuando un nodo b' es viejo nodo base, buscamos la floración más interna en el cual está contenido, y el nuevo nodo base b de tal floración. Cuando b=b', simplemente intercambiamos los índices de las etiquetas en b'. En caso contario, se hace rastreo regresivo a partir de la vieja etiqueta "T" en el nodo b, hasta encontrar un arco (b',j) fuera del apareamiento en turno, donde 'j' pertenece a la floración ya mencionada. Si (b',i) es un arco de la trayectoria aumentante donde 'i' pertenece a la floración, las nuevas etiquetas para b* son "S:i" y "T:j".

En la Figura 3.13 los arcos ondulados indican apareamiento, y se distingue una trayectoria aumentante entre los nodos 1 y 18. Las floraciones existentes son: B1, con nodo-base b'1=5; B2, con nodo-base b'2=14 y B3 con nodo-base b'3=3.

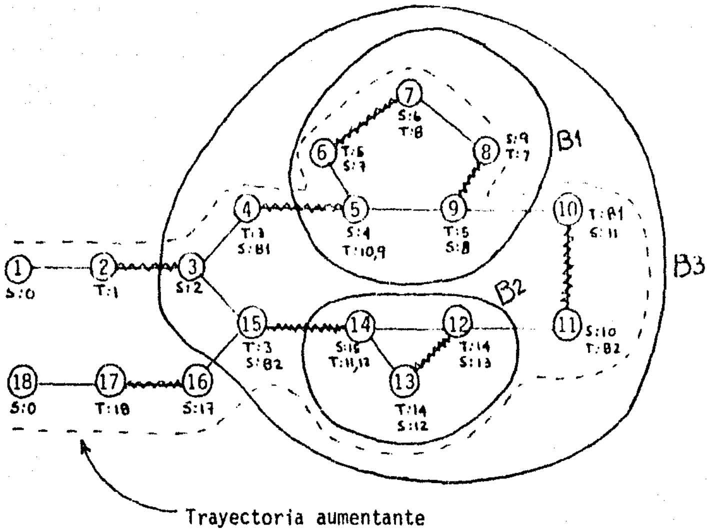


Figura 3.13

Una vez aumentado el apareamiento y corregidas las etiquetas, los nuevos nodos-base en las floraciones B1, B2 y B3 son, respectivamente: b1=9, b2=12 y b3=15. En la Figura 3.14 se aprecian las correcciones de etiquetas en los nodos que forman la trayectoria aumentante.
Así, en los nodos: 4,6,7,8,10 y 13 que no son ni nuevos ni viejos nodos-base, simplemente se intercambian índices en las etiquetas (por ejemplo, el nodo 7 aparece con S:6 , T:8 antes del aumento y con S:8 T:6 después). Para el nuevo nodo-base b1=9, la floración más interna que lo contiene es B1, y localizando los arcos: (b1,i) = (9,10) y (b'1,j)=(5,4) con 'i','j' fuera de B1 resultan las etiquetas S:10 y T:4,5 para b1.(análogamente se aplican etiquetas en b2 y b3) En el caso del viejo nodo-base b'3=3, la floración más interna que lo contiene es B3, y por rastreo regresivo desde la vieja etiqueta "T" en el nuevo nodo-base b3=15, se halla el arco no apareado (b'3,j)=(3,15) con el nodo 15 en B3; además, el arco (b'3,i) de la trayectoria aumentante con 'i' en B3 que incide en b'3 es (b'3,i) = (3,4), con lo que las etiquetas en b'3 son S:4 y T:15 (de modo similar se aplican las etiquetas en b'1 y b'2).

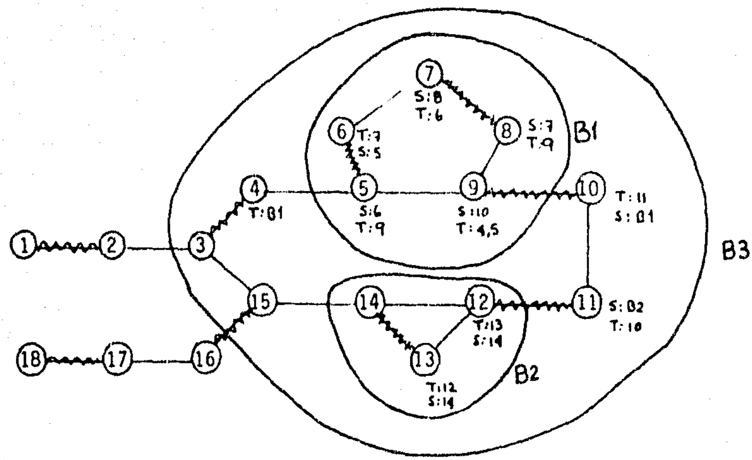


Figura 3.14

Para terminar, se introduce la variable 'D', que servirá para indicar el máximo valor que puede tomar la variable 'd' que afectará a las variables duales, consistentemente con las condiciones C2 y C4 de la sección anterior. Para comodidad en la descripción del algoritmo, se usará la notación w^ij para representar Ui + Uj - wij. De hecho, cada vez que los valores Ui, Uj y wij son encontrados, el valor de w^ij es calculado, ahorrando uso de memoria de computadora.

(3.7.3) DESCRIPCION DEL ALGORITMO Y EJEMPLO.

A continuación se describirá completo el algoritmo de apareamiento ponderado; enseguida se presenta un ejemplo.

ALGORITMO 3.3.2 APAREAMIENTO PONDERADO NO-BIPARTITA.

Paso 0). INICIO. Se tiene una gráfica no-bipartita G=(N,A) con un peso wij para cada arco (i,j). Asígnese Ui = 0.5max(wij) en cada nodo de G. Considérese D = ∞ , y el apareamiento inicial X = $\emptyset$. No existen nodos con etiqueta ni floraciones.

Paso 1). ETIQUETADO.
(1.0) Coloque la etiqueta "S:0" en cada nodo expuesto.
(1.1) Si todas las etiquetas han sido examinadas ir al Paso 4. En caso contrario, encuentre un nodo 'i' con etiqueta sin examinar. Si la etiqueta es "S", ir al Paso 1.2; si la etiqueta es "T", ir al Paso 1.3.

(1.2) Examine la etiqueta "S" en el nodo 'i' haciendo el siguiente procedimiento para cada arco (i,j) fuera del apareamiento en turno X, que incida en el nodo 'i':

Si b(i)=b(j) no hacer nada; en caso contrario continuar

Si el nodo b(j) tiene etiqueta "S" y w^ij=0, haga rastreo regresivo partiendo de las etiquetas "S" en los nodos 'i' y 'j'. Si se alcanzan distintos nodos raiz, ir al Paso 2; si se alcanza el mismo nodo raiz, ir al Paso 3.

Si el nodo b(j) tiene etiqueta "S" y w^ij > 0, corrija el valor de la variable 'D' por: D = min(D, 0.5w^ij).

Una vez que se ha examinado por completo el nodo 'i', regresar al Paso 1.1.

(1.3) Examine la etiqueta "T" en el nodo 'i' haciendo el siguiente proceso en el único arco (i,j) dentro del apareamiento en turno X:

Si b(i)=b(j) no hacer nada; en caso contrario contiunar.

Si el nodo 'j' tiene etiqueta "T" , haga rastreo regresivo desde las etiquetas "T" en los nodos 'i' y 'j'. Si se alcanzan diferentes nodos raiz, ir al Paso 2; si el mismo nodo es alcanzado, ir al Paso 3.

En otro caso, asigne al nodo 'j' la etiqueta "S:i". Todos los nodos "S" dentro de la misma floración externa con nodo base 'j', serán consideradas ahora sin examinar.

Regrese al Paso 1.1.

Paso 2). AUMENTO.

Una trayectoria aumentante se ha encontrado en el Paso 1.2 o 1.3. Aumente y actualice el apareamiento en turno X. Corrija las etiquetas en los nodos de la trayectoria aumentante, como se indicó en la sección anterior. Extienda las floraciones que tengan las variables duales igual a cero, reiniciando los números b(i) en nodos de la floración . Retire las etiquetas de todos los nodos base, y asigne la condición de "examinada" en el resto de las etiquetas. Asigne el valor $D = \infty$ y vaya al Paso 1.0.

Paso 3). FLORACIONES.

Una floración se ha formado en el Paso 1.2 o 1.3. Identifique los nodos de la nueva floración así como el nodo base, como se indicó en la sección anterior. Proporciones etiquetas faltantes en todos los nodos, salvo en el nodo base de la nueva floración. Reinicie los números b(i) y haga la variable dual Z=0 para la nueva floración .

Regrese al Paso 1.2 o al 1.3, según el caso.

Paso 4). REVISION DE LA SOLUCION DUAL.

Sea Ks el conjunto de floraciones-S y Kt el conjunto de floraciones-T.

Encuentre los valores:

$$d1 = \min(Ui),$$

$$d2 = 0.5\min(Zp \mid p \in Kt),$$

$$d = \min(d1, d2, D).$$

Asigne Ui=Ui-d, en cada nodo 'i' tal que b(i) tiene etiqueta "S".

Asigne Ui=Ui+d, en cada nodo 'i' tal que b(i) tiene etiqueta "T".
Asigne Zp=Zp-2d, para cada floración en Kt.
Asigne Zp=Zp+2d, para cada floración en Ks.
Si d=d1, alto; el apareamiento X es maximal, y los valores de Ui, Zp dan solución dual óptima.
Si d=d2, extienda cada floración-T para la cual Zp=0, determinando las floraciones inmediatamente anidadas dentro de ella, y reiniciando b(i) para todos los nodos dentro de la floración. Retire las etiquetas de todos los nuevos nodos base dentro de la floración extendida.
Todas las etiquetas en los nodos base, y las etiquetas "S" en nodos dentro de floraciones-S, se consideran ahora "sin examinar".
El resto de las etiquetas están en el estado "examinadas".
Asigne D = ∞ .
Regrese al Paso 1.1.

El ejemplo que se resuelve a continuación usa la descripción del algoritmo esbozada en la sección (3.6).

EJEMPLO: Consideremos la gráfica G de la Figura 3.15, con 10 nodos y 16 arcos, donde cada arco (i,j) tiene anotado su peso asociado wij.

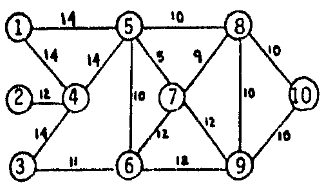


Figura 3.15

Al inicio del algoritmo se tiene el apareamiento nulo X=∅, y las variables duales toman los valores:

i-->	1	2	3	4	5	6	7	8	9	10
Ui-->	7	7	7	7	7	7	7	7	7	7

ya que 0.5max(wij) = 0.5(14) = 7.

Formando árboles solo con arcos (i,j) que cumplen Ui+Uj+ Zp=wij y aumentando el apareamiento en turno se consigue el apareamiento X1= ((1,5), (3,4)), y los árboles presentes en G se vuelven húngaros, como se ve en la Figura 3.16.

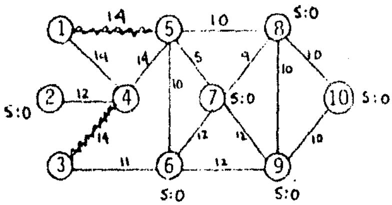


Figura 3.16

Se requiere entonces la primera modificación a variables duales. El máximo valor 'd' que se puede restar es d=1, ya que valores mayores violan las condiciones C2 en al menos el arco (7,6). Haciendo el cambio, resulta:

i-->	1	2	3	4	5	6	7	8	9	10
Ui-->	7	6	7	7	7	6	6	6	6	6

Prosiguiendo con el algoritmo se construyen más árboles con arcos (i,j) que cumplen Ui+Uj+ Zp=wij y se actualiza el apareamiento en turno, obteniéndose X2=((1,5), (3,4), (6,7)). Luego del aumento, y partiendo del nodo raiz 9 se detecta la floración B1 formada por los arcos (7,9), (6,9) y (6,7); una vez encogida la floración resulta la gráfica de la Figura 3.17, y se encuentra que los árboles presentes son húngaros.

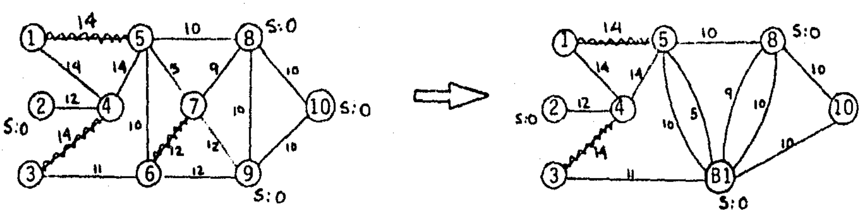


Figura 3.17

Se efectúa una segunda modificación a variables duales. El máximo valor 'd' que se puede restar es d=1, pues valores más grandes violan

C2 en por lo menos el arco (8,9) Cambiando las variables duales se tiene: (nótese que el nodo 4 tiene etiqueta "T")

i-->	1	2	3	4	5	6	7	8	9	10
Ui-->	7	5	7	8	7	6	6	5	5	5

Continuando con la construcción de árboles con arcos que cumplen la condición mencionada arriba, se actualiza de nuevo el apareamiento y se obtiene X3=((1,5), (3,4), (6,7), (8,B1)), como se aprecia en la Figura 3.18.

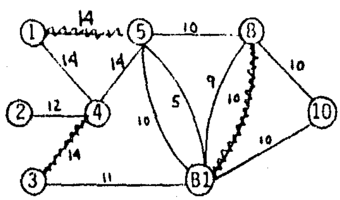


Figura 3.18

Luego del aumento y a partir del nodo 10, se detecta la floración B2, formada por los arcos (8,B1), (B1,10), (8,10); se encoge y resulta la gráfica de la Figura 3.19.

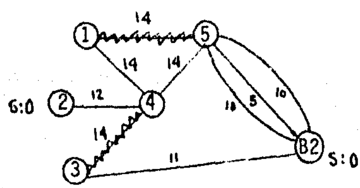


Figura 3.19

Al continuar el algoritmo se encuentra que el único árbol nuevo parte del pseudonodo B2 y no puede ir más allá del nodo 5. Corrigiendo nuevamente variables duales se ve que el máximo valor 'd' que es posible restar es d=5, que coincide con el máximo permitido a 'd' para que no viole C1. De este modo, el apareamiento en turno y las variables duales son soluciones óptimas primal y dual, respectivamente. La solución óptima aparece en la Figura 3.20, con un peso maximal de 50.

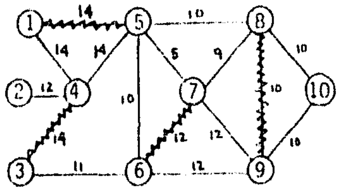


Figura 3.20

APENDICE

*** TEOREMAS ***

En este apéndice se presentan las demostraciones de algunos teoremas o proposiciones que se citan en los capítulos anteriores. Las fuentes consultadas aparecen en la bibiliografía al final del texto.

PROPOSICION 1.1a. En una gráfica finita $G=(N,A)$ el número de vértices de grado impar es par.

Demostración.

Sea $P \subset N$ el conjunto de nodos de grado par, y $Q \subset N$ el conjunto de nodos de grado impar. Entonces:

$$\sum_{i \in P} d(i) + \sum_{j \in Q} d(j) = 2|A|$$

La ecuación anterior se puede reescribir como

$$\sum_{i \in P} (2pi) + \sum_{j \in Q} (2qj+1) + 2|A|$$ con pi, qj enteros positivos.

De donde:

$$2\sum_{i \in P} pi + 2\sum_{j \in Q} qj + \sum_{j \in Q} 1 = 2|A|$$

con lo que:

$$|Q| = \sum_{j \in Q} 1 = 2|A| - 2\sum_{i \in P} pi - 2\sum_{j \in Q} qj$$

De la última igualdad es claro que la cardinalidad de Q es par.

PROPOSICION 1.1b. Si G es una gráfica con 'n' nodos, las condiciones siguientes son equivalentes:

a) G es un árbol.
b) Cualquier par de nodos de G está unido por un trayectoria única.
c) G no tiene ciclos, pero exactamente uno se forma al agregar un arco.
d) G es conexa, pero deja de serlo si se suprime algún arco.
e) G no tiene ciclos y tiene 'n-1' arcos.
f) G es conexa y tiene 'n-1' arcos.

Demostración.

Suponiendo que a) es cierta (G conexa y sin ciclos), sean X,Y dos nodos en G, entonces: Por ser G conexa existe trayectoria P que une

X con Y. Si existiera otra trayectoria P' distinta a P uniendo a los nodos X,Y la secuencia: X-P-Y-P'-X forma un ciclo en G, contradiciendo la hipótesis a). Por tanto la trayectoria P es única. Esto demuestra la implicación: a) $\Longrightarrow$ b).

Suponiendo b) cierta, sea (X,Y) el arco que se agrega a G. Entonces, si P es la trayectoria única ya existente en G que une X con Y, la secuencia Y-P-X-(X,Y) forma un ciclo único en G. Esto demuestra la implicación: b) $\Longrightarrow$ c).

Suponiendo que c) es cierta y que G=(V,A) no es conexa, entonces existen S,T subconjuntos ajenos de V tales que $S \cup T = V$, de modo que las subgráficas Gs=(S,As) y Gt=(T,At) son ajenas. De esta forma, al agregar un arco entre Gs y Gt no se forma ciclo alguno, contradiciendo la hipótesis c). Por lo tanto G debe ser conexa. Además, si se suprime un arco (X,Y) de G y G permanece conexa, entonces existe una trayectoria P uniendo X con Y, y la secuencia Y-P-X-(X,Y) sería un ciclo en G, contradiciendo de nuevo la hipótesis c). Esto demuestra la implicación: c) $\Longrightarrow$ d).

Suponiendo que d) es cierta, si G tiene un ciclo que une los nodos X, Y, digamos: X-N1-N2-...-Nk-X, eliminando cualquier arco del ciclo la gráfica permanece conexa, lo que contradice la hipótesis d). Por tanto, G no puede tener ciclos.

Para probar que hay n-1 arcos procedemos por inducción. Cuando n=1 el número de arcos es 0, y se cumple la condición. Supongamos que la condición vale para n=k (Hipót. de inducción). Entonces, si G tiene k+1 nodos y j arcos, al suprimir un arco la hipótesis d) garantiza que G se vuelve disconexa, con lo que se tendrán dos árboles: G1, G2 con N1, N2 nodos respectivamente. Entonces se cumplen las siguientes condiciones:

N1 + N2 = k+1 (el total de nodos no cambia).

Núm. de arcos en G1 = N1-1 (por hipót. de inducción).

Núm. de arcos en G2 = N2-1 (por hipót. de inducción).

Núm. de arcos en G = Arcos de G1 + Arcos de G2 + 1 (el suprimido).
Núm. de arcos en G = (N1 - 1) + (N2 - 1) + 1 = k.

Esto demuestra la implicación: d) $\Longrightarrow$ e).

Suponiendo que e) es cierta, si G no es conexa, entonces existen j componentes conexas: G1,G2, ...,Gj no conectadas entre si. Ya que G no tiene ciclos, cada subgráfica Gi resulta ser un árbol con Ni nodos y Ni-1 arcos. Entonces:

Núm. de nodos de G = n = N1+N2+...+Nj.
Núm. de arcos en G = Arcos de G1 + Arcos de G2 +...+Arcos de Gj.
Núm. de arcos en G = (N1-1) + (N2-1) + ... +(Nj-1)
Núm. de arcos en G = N1+N2+...+Nj - j = n - j = n - 1.

La última igualdad muestra que j=1, es decir G tiene una única componente conexa, con lo que G es conexa.

Esto demuestra la implicación: e)   f).

Suponiendo f) cierta, supóngase que se encuentra un ciclo que inicia y termina en el nodo X: X-N1-N2-...Nk-X. Si eliminamos cualquier arco del ciclo, G sigue siendo conexa y tiene n-2 arcos, contradiciendo la hipótesis f). Por tanto G no tiene ciclos.

Esto demuestra la implicación: f)   a), y con ello se completa la prueba del teorema.

PROPOSICION 1.1c. Una gráfica G=(N,A) es bipartita si y sólo si carece de ciclos de longitud impar.

Demostración.

Sea G=(V,A) la gráfica considerada.

En la dirección "sólo si": Si G es bipartita, existen S y T conjuntos disjuntos cuya unión es V, y tales que cada arco en A se extiende de un nodo en S a otro en T. Supongamos un ciclo cualquiera: n1,n2,...,nk=n1 en G, y sin perder generalidad supóngase que n1 está en S. Entonces, los nodos nj con 'j' impar están en S y los que tengan 'j' par están en T. Si la longitud del ciclo (el número de nodos que lo forman) es impar, k-1 es número impar, con lo que 'k' es par y nk=n1 debe pertencer a T, lo que es contradictorio. Por lo tanto, no hay ciclos impares.

En la dirección "si ": Supóngase que cada ciclo de G tiene longitud par. Elíjase cualquier nodo 'i', y para todo nodo 'j' con el que 'i' esté conectado, sea (Pj) el conjunto de arcos que forman trayectorias de longitud mínima entre ambos nodos. Entonces, defínanse los siguientes conjuntos:

S = (i)   (j: |Pj| es par)

T = (j: |Pj| es impar)

Se probará por contradicción. Supóngase que hay un arco a = (i,j) en A tal que i,j están ambos en S o ambos en T. Sea 'v' el último nodo común de las trayectorias Pi, Pj (tal vez v=i), y defínanse: - Qi y Qj como las partes de las trayectorias Pi, Pj respectivamente, que se extienden más allá de 'v'. Necesariamente, tales porciones de ambas trayectorias deben cumplir |Pi-Qi| = |Pj-Qj|, pues de no ser así, no serían de cardinalidad mínima. Pero, para i,j en S o i,j en T, |Pi| y |Pj| son ambas pares o ambas impares. Por tanto, la suma: |Qi| + |Qj| es par. De esto se concluye que Qi Qj (a) es un ciclo impar en G, lo que es una contradicción.

Por lo tanto, resulta que la componente conexa a la que el nodo 'i' pertence es bipartita. Repitiendo el proceso en cada componente de G se completa el teorema.

PROPOSICION 2.1a. Si G es una gráfica con un número 'n' par de nodos, y cada uno de ellos tiene grado mayor o igual a: n/2, entonces existe un apareamiento completo para G.

Demostración.

Supóngase que se ha comenzado a aparear nodos y que se han formado 'k' parejas, donde $k < (n/2)$. Se mostrará cómo aumentar el número de parejas hasta $k+1$:

Si existen en G dos nodos unidos por un arco, pero sin aparear, es inmediato que el $(k+1)$-ésimo par está representado por tal arco. En caso contrario, todos los nodos restantes de G tienen la propiedad de que ningún par de ellos está conectado por algún arco. Elíjase entonces un par cualquiera de tales nodos, digamos A,B. Se probará que debe existir un par de nodos U,V que ya han sido apareados, tales que hay un arco que une A con U y otro arco que une B con V. Entonces los apareamientos entre nodos se pueden reconstruír, de modo que A forme pareja con U, mientras que B forme pareja con V, como se ve en la Figura A.1 a continuación.

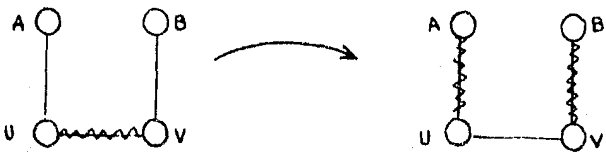


Figura A.1

Si el par de nodos U,V mencionados no existiera, entonces cada uno de los 'k' pares de nodos (X,Y) ya formados es tal que a lo más dos de los cuatro posibles arcos AX, AY, BX, BY pueden estar realmente presentes en G. De este modo el número total de arcos entre los 'k' pares ya formados y A,B es a lo más: $2k < n$, es decir: $k < n/2$. Pero como A,B tienen ambos grado mayor o igual que $(n/2)$ resulta una contradicción.

El algoritmo a seguir es por tanto así: Una vez que se han logrado 'k' parejas, examine los $(n-2k)$ nodos restantes para ver si dos de ellos están unidos por un arco. Si no lo están, elija 2 nodos cualesquiera A,B de ellos y busque entre las parejas (X,Y) ya formadas hasta encontrar una tal que haya un arco que una A con X y otro arco que una B con Y. Entonces reemplace la pareja (X,Y) por las dos parejas (A,X), (B,Y). Si $k+1 < (n/2)$, repita todo el proceso.

PROPOSICION 2.4c. (Teorema de Dillworth) Si T es un conjunto finito con un orden parcial estricto, entonces:

$$\min (|D| : D \text{ es descomposición en cadenas de } T) =$$

$$\max (|N| : N \text{ es una anticadena en } T).$$

Demostración.

Sean D una descomposición en cadenas y N una anticadena en T. Entonces se cumple $|D| \geq |N|$, puesto que ninguna cadena en D contiene más de un elemento en N.

Ahora, supóngase que D es una descomposición en cadenas de cardinalidad mínima, y sea M el apareamiento asociado tal como se describe en la sección 2.4.2, el cual cumple: $|M| = |T| - |D|$.

De acuerdo al teorema de König-Egerváry, existe un recubrimiento mínimo de nodos B que satisface: $|B|=|M|=|T| - |D|$. Si $t1, t2$ son elementos de T que cumplen $t1 < t2$, al ser B recubrimiento de nodos de T debe ocurrir (por la forma en que se construyó M) que B contiene: la primera copia $t1'$ de $t1$, o la segunda copia $t2''$ de $t2$.

Llamando N al conjunto de elementos t en T que no tienen ni la copia t' ni la t'' en B, no es difícil ver que N es una anticadena, y que se cumple:

$$|T| - |N| = \text{núm. de elementos } t \text{ en } T \text{ con } t' \text{ o } t'' \text{ en } B.$$

Para continuar, se mostrará que no hay elementos t en T que tengan simultáneamente t' y t'' en B, de modo que el valor $|T| - |N|$ es igual a $|B|$ exactamente.

Supóngase que hay un elemento t en T con t' y t'' en B. Dado que B es de cardinalidad mínima, no se puede eliminar de B ninguno de esos elementos sin alterar su condición de óptimo. Por tanto debe tenerse un elemento t en T tal que $t1<t$ (para tener el arco $(t1', t'')$) y $t1'$ no pertenezca a B. Análogamente, debe existir $t2$ en T tal que $t<t2$ y $t2''$ no esté en B. Entonces, por transitividad del orden en T debe ocurrir $t1<t2$, lo que implica tener el arco $(t1', t2'')$ sin incidir en ningún nodo del recubrimiento B, lo que es contradictorio.
De este modo, no existe un elemento t en T como el supuesto. Además $|T| - |N| = |B| = |T| - |D|$, con lo que N es una anticadena que cumple $|N| = |D|$. Esto completa la demostración.

PROPOSICION 2.4d. Sea T un conjunto finito con un orden parcial estricto, y supóngase que la cadena más larga en T es de longitud n. Entonces se puede hacer una partición de T en n anticadenas disjuntas.

Demostración.

La prueba es por inducción sobre la longitud 'n' de la cadena.

Cuando $n=1$, T contiene un solo nodo, y claramente se tiene una anticadena.

Supóngase ahora que la proposición vale para $n=k$ (hipótesis de inducción). Sea T conjunto con orden parcial estricto y con la longitud de sus cadenas más largas igual a $k+1$. Sea M el conjunto de elementos maximales en T, es decir, si m M, entonces para todo x T ocurre que x m. Es claro que M es una anticadena no vacía. Considérese entonces al conjunto parcialmente ordenado: T-M, con el orden heredado de T. Puesto que en T-M no existen cadenas de longitud $k+1$, la longitud de las cadenas más largas en T-M es a lo más k. Por otro lado, si la longitud de las cadenas más largas en T-M es menor de k, M debe contener dos o más elementos maximales pertenecientes a una misma cadena, lo que contradice la construcción de M. Por tanto, se concluye que las cadenas más largas de T-M son de longitud k, y por la hipótesis de inducción, T-M puede particionarse en k cadenas disjuntas. Si a estas k cadenas disjuntas se añade M, se obtiene una partición de T en k+1 cadenas disjuntas.

TEOREMA DE HELLER-TOMPKINS-GALE.

Sea A matriz mxn cuyos elementos son 0, 1 o -1 y tal que cada columna contiene a lo más dos elementos no nulos. Entonces A es totalmente unimodular (toda submatriz cuadrada tiene determinante 1, 0 o -1) siempre que A puedea ser particionada en dos conjuntos de filas: F1, F2 tales que cumplan:

a) Si los dos elementos no cero de una columna tienen igual signo, entonces uno de ellos está en las filas F1 y el otro en las F2.

b) Si los dos elementos no cero de una columna tienen signos contrarios, entonces ambos están en el conjunto de filas F1 o ambos están en el conjunto de filas F2.

Demostración.

Sea B una submatriz cuadrada de A. Se probará por inducción sobre el orden de B que el determinante de B debe ser 1, 0 o -1.

Si B es de orden 1x1 el resultado es obvio. Supóngase ahora que el resultado vale para orden nxn (Hipótesis inductiva), y que B es submatriz de orden (n+1)x(n+1). Entonces:

1) Si B tiene alguna columna de elementos iguales a cero, detB=0.

2) Si B tiene al menos una columna que posee exactamente un elemento (i,j) no nulo, desarrollando detB por la columna 'j' resulta:

$$\det B = \pm \det C$$

donde C es la submatriz de orden nxn que resulta al eliminar la fila 'i' y la columna 'j'. Por la hipótesis de inducción detC = 1, 0 o -1, con lo que detB = 1, 0, o -1.

3) Si cada columna de B tiene exactamente dos elementos no nulos, entonces la suma de filas F1 iguala a la suma de filas F2, ya que: Si en la columna 'j' los dos elementos no nulos son de igual signo, uno está en F1 y el otro en F2, con lo que la suma de filas F1 tendrá el mismo valor en la columna 'j' que la suma de filas F2; y en caso de que los elementos no nulos en la columna 'j' tengan signos opuestos ambos están en F2, con lo que la suma de filas F2 tendrá cero en la columna 'j', lo mismo que tiene la suma de filas F1.

De este modo, la condición suma de filas F1 = suma de filas F2, que equivale a: suma de Filas F1-suma de filas F2=0, representa una combinación lineal nula de las filas de B, con coeficientes no nulos, lo que implica que tales filas son linealmente dependientes y por lo tanto, detB=0.

B I B L I O G R A F I A.

[1] Eugene L. Lawler, COMBINATORIAL OPTIMIZATION: NETWORKS AND MATROIDS, Holt, Rinehart and Winston, 1976.

[2] R. Gary Parker, Ronald L. Rardin, DISCRETE OPTIMIZATION, 1983.

[3] L. Papadimitriou, COMBINATORIAL OPTIMIZATION: ALGORTIHMS AND COMPLEXITY, Prentice-Hall 1984.

[4] R.T. Rockafellar, NETWORK FLOWS AND MONOTROPIC PROGRAMMING, Wiley Interscience 1984.

[5] Ian Anderson, A FIRST COURSE IN COMBINATORIAL MATHEMATICS, Oxford University Press, 1979.

[6] Robert Sedgewick, ALGORITHMS, Addison-Wesley Publishing Co.,1984.

[7] C.L. Liu, ELEMENTS OF DISCRETE MATHEMATICS, Mc Graw-Hill, 1985.

* * * P R O G R A M A S * * *

En esta sección se muestran algunos programas de computadora que resuelven problemas relacionados con los modelos que se tratan en capítulos anteriores. Los programas están escritos en Pascal, y se corrieron en la versión TurboPascal de una microcomputadora Printaform. Se anexan copias de los programas-fuente.

ARCHIVO: MENU.PAS

```
program menu;
(**************************************************************************)
(*                    TURBO-PASCAL     Version 3.0                         *)
(*                                                                         *)
(*            - - -  PROBLEMAS  DE  APAREAMIENTO - - -                     *)
(*                                                                         *)
(* Este programa despliega el menu de opciones para resolver algu-        *)
(* nos problemas de apareamiento, lee la eleccion del usuario y se-       *)
(* gun el caso, 'encadena' el programa objeto que resuelve el pro-        *)
(* blema usando la instruccion "Chain". El programa objeto es             *)
(* MENU.COM y se invoca con la palabra "MENU". El programa fuente         *)
(* es MENU.PAS.                                                            *)
(*                                                                         *)
(* PROGRAMO: Mat. Eric Moreno Q.                FECHA: Abril/87            *)
(*                                                                         *)
(* SECCION: Investigacion de Operaciones,  D.E.P.F.I.     U N A M          *)
(**************************************************************************)
type
     letras=set of char;

const let:letras=['a'..'e','A'..'E','Z','z'];

var
   respu:char;
   filvar:file;
   k:integer;
   flag:boolean;

begin
flag:=true;
while flag do
   begin
   ClrScr;write(' ');
   for k:=1 to 77 do
        write('*');
   writeln;
   writeln('*','*':78);
   writeln('*','- - - PROBLEMAS  DE  APAREAMIENTO - - -':55,'*':23);
   writeln('*','*':78);write(' ');
   for k:=1 to 77 do
        write('*');
   writeln;writeln;
   writeln('OPCIONES:':30);writeln;writeln;
   writeln(' A)  Prueba para Graficas Bipartitas.':60);writeln;
   writeln(' B)  Maximo Apareamiento Bipartita.':58);writeln;
   writeln(' C)  Apareamiento Ponderado Bipartita.':61);writeln;
   writeln(' D)  Apareamiento Max-Min. (Cuello de Botella)':69);
   writeln;
   writeln(' E)  Apareamiento de Gale-Shapley.':57);writeln;
```

```
writeln(' Z)  --- Fin ---':39);
writeln;writeln;
repeat
GotoXY(41,23);ClrEol;
write('Opcion elegida: ':10);
read(Kbd,respu);
if not (respu in let) then
   write(^G);
until respu in let;
Case respu of
    'a','A':begin
            assign(filvar,'PROG01.CHN');Chain(filvar);
            end;
    'b','B':begin
            assign(filvar,'PROG02.CHN');Chain(filvar);
            end;
    'c','C':begin
            assign(filvar,'PROG03.CHN');Chain(filvar);
            end;
    'd','D':begin
            assign(filvar,'PROG04.CHN');Chain(filvar);
            end;
    'e','E':begin
            assign(filvar,'PROG05.CHN');Chain(filvar);
            end;
    'z','Z':flag:=false;
end;(* fin del CASE OF *)
end;
ClrScr;
GotoXY(30,12);write('* * * FIN * * *');
end.
```

ARCHIVO: ESPERA PAS

```
procedure espera;
var
   zz:char;
begin
writeln;
writeln('RETURN para continuar...':75);
repeat
read(Kbd,zz);
if(zz()^M) then
   write(^G);
until(zz=^M);
end;
```

ARCHIVO: PROG01.PAS

```
program bipart;
(************************************************************************)
(*                                                                      *)
(* Este programa detecta si una grafica leida es bipartita o no lo     *)
(* es. En caso de serlo, separa los nodos en dos conjuntos disjun-     *)
(* tos "S" y "T", mientras que en caso contrario despliega un aviso*)
(* informando al usuario que su grafica no es bipartita.              *)
(* El programa tambien trabaja con graficas disconexas. En caso de *)
(* que la grafica disconexa sea bipartita, se informa del numero de*)
(* componentes conexas que la conforman.                               *)
(* Este programa es invocado desde el menu "MENU.COM" y al terminar*)
(* regresa invocando al menu con la instruccion "EXECUTE".            *)
(* El programa objeto es PROG01.CHN, y el fuente es  PROG01.PAS.     *)
(*                                                                      *)
(************************************************************************)
(* PROGRAMA PARA DETECTAR GRAFICAS BIPARTITAS *)
type arco=record
      x,y,marca:integer;
      end;
      datos=array[1..400] of arco;

var
n,k,l:integer;
arch:datos;
nodos:array[1..100] of integer;
conexa,alto:boolean;
FILVAR:FILE;
(* la siguiente instruccion incluye el archivo ESPERA.PAS *)
(*$I ESPERA.PAS *)

procedure leearcos(var n,k:integer; var alto:boolean);
var
   i,j:integer;
   buf:arco;
   ban:boolean;

   begin
    k:=0;n:=0;ban:=false;
    writeln('*** PROGRAMA PARA DETECTAR GRAFICAS BIPARTITAS ***':66);
    writeln,writeln;
    write('Indica cuantos nodos son: ':20,' ');
    read(n);writeln;
    if n=0 then
       alto:=true
    else
    begin
 (* inicializacion de nodos *)
    for i:=1 to n do
        nodos[i]:=0;
```

```
      for i:=1 to 20 do
          write('----');
      writeln;l:=0;
      repeat
       l:=l+1;
       writeln('Escribe arco  A, B    (termina con 0 0)':50);
       read(buf.x,buf.y);writeln;
       if((buf.x>0)and(buf.x<=n)and(buf.y>0)and(buf.y<=n)) then
          begin
          arch[l].x:=buf.x; arch[l].y:=buf.y; arch[l].marca:=0;
          k:=k+1;
          end
       else
          begin
          if((buf.x=0)and(buf.y=0)) then
              ban:=true
          else
             writeln('    ***ERROR. Datos leidos: A=',buf.x,' B=',buf.y),
          end;
      until ban;
      ClrScr;
      writeln('Total de arcos leidos: ':40,k);writeln;
      i:=0;
      for l:=1 to k do
          begin
          i:=i+1;
          with arch[l] do
               write('  Arco No. ',l:2,'---(',x:2,',',y:2,')');
          if( i mod 3=0 ) then
              writeln;
          end;
      writeln;
      end;
    end;

procedure separa(var n,k:integer; var conexa:boolean);

label 100;

var
   cond,flag:boolean;
   buf:arco;
   i,j,tipo,vueltas,partes:integer;
   clase:char;
begin
   vueltas:=0;conexa:=true;partes:=1;
   (* se inicializan los nodos del primer arco *)
   nodos[arch[1].x]:=1; nodos[arch[1].y]:=-1; arch[1].marca:=1;
100 : cond:=true;
   repeat
    vueltas:=vueltas+1;
    for l:=1 to k do
```

```
  begin
  with arch[l] do
   begin
    if marca <> 1 then
        begin
         flag:=false; marca:=1;
         if nodos[x]=0 then
            begin
             if nodos[y]=0 then
                marca:=0
             else
                nodos[x]:=-nodos[y]
            end
         else
            begin
            if nodos[y]=0 then
               nodos[y]:=-nodos[x]
            else
               begin
               if nodos[x]<>-nodos[y]  then
                  begin
                  flag:=true;
                  cond:=false;
                  end;
               end;
            end;
        end;
   end;
  end;
  until flag or (vueltas>k);
 writeln;
 ClrScr;
 if cond then
    begin
    flag:=true;l:=1;
    while (l<k) and flag do
        begin
         l:=l+1;
         if arch[l].marca=0 then
            begin
            flag:=false;conexa:=false;partes:=partes+1;
            nodos[arch[l].x]:=1; nodos[arch[l].y]:=-1;
            arch[l].marca:=1;
            writeln;
            end;
        end;
    if flag then
       begin
       writeln;
       writeln('--- G R A F I C A  B I P A R T I T A ---':55);
       writeln;
       if not conexa then
```

```
            writeln('-----> GRAFICA DISCONEXA CON ':60,partes:3,
                    'COMPONENTES.');
         tipo:=1;
         for i:=1 to 2 do
             begin
             if tipo=1 then
                clase:='S'
             else
                clase:='T';
             writeln;writeln('NODOS TIPO ':30,clase,':');writeln;
             for j:=1 to n do
                 begin
                 if nodos[j]=tipo then
                    write('  ',j,',');
                 end;
             writeln;
             writeln(' **********************************':50);writeln;
             tipo:=-tipo;
             end;
         end;
      end
   else
      begin
      writeln;
      writeln('++++++ LA GRAFICA NO ES BIPARTITA ++++++':53);
      end;
   if not flag and cond then
      goto 100;
   end;

begin
(* PROGRAMA PRINCIPAL *)
alto:=false,conexa:=true;
repeat
 ClrScr;
 leearcos(n,k,alto);
 if not alto then
    begin
    espera;
    separa(n,k,conexa);
    espera;
    end;
until alto;
writeln;writeln;writeln('**** F I N  D E  P R O G R A M A ****':60);
(*regreso al menu*)
assign(filvar,'MENU.COM');
Execute(filvar);
end.
```

ARCHIVO: PROG02.PAS

```
program maxmatch;
(***********************************************************************)
(*                                                                     *)
(* Este programa lee una grafica bipartita y encuentra el aparea--    *)
(* miento con el mayor numero posible de arcos. Despliega ademas la*)
(* solucion dual al apareamiento maximal: un recubrimiento minimo     *)
(* de nodos de la grafica consultada.                                  *)
(* El programa se invoca desde el menu con la instruccion "CHAIN",    *)
(* y regresa al menu invocandolo con la instruccion "EXECUTE".        *)
(* El programa objeto es PROG02.CHN y el fuente es PROG02.PAS.        *)
(*                                                                     *)
(***********************************************************************)
type
     vertices=record
     tipo:char;
     etiq,exam,expo:integer;
     end;
     aristas=record
     x,y,apareado:integer;
     end;
     nodos=array[1..80] of vertices;
     arcos=array[1..400] of aristas;

var
   l,cont,numnodos,numarcos:integer;
   ar1:nodos; ar2:arcos; buf2:aristas;
   alto:boolean;
   FILVAR:FILE;
(* la siguiente instruccion incluye el archivo ESPERA.PAS *)
(*$I ESPERA.PAS *)

procedure leegrafica(var numnodos,numarcos:integer;var alto:boolean);
var
   k,j,p:integer;
   ban:boolean;
   contipo:char;

begin
k:=0; numnodos:=0; numarcos:=0;
ClrScr;writeln('*** MAXIMO APAREAMIENTO BIPARTITA ***':55);
writeln;writeln;
write('Indica el numero de nodos':20,' (termina con 0) : ');
read(numnodos);writeln;
if numnodos=0 then
   alto:=true
else
   begin
   for l:=1 to 20 do
       write('----');
```

```
(* se inicializan arreglos de nodos y arcos *)
for l:=1 to 80 do
     begin
     ar1[l].etiq:=0; ar1[l].exam:=0;
     ar1[l].expo:=1;
     end;
for l:=1 to 400 do
     begin
     ar2[l].x:=0; ar2[l].y:=0; ar2[l].apareado:=0;
     end;
(* se considera una grafica bipartita G=(S,T,V)   *)
 for contipo:='S' to 'T' do
      begin
      ban:=false;
      writeln('Escribe los nodos tipo ':30,contipo,
              '  (termina con 0):');writeln;
      repeat
      write('Nodo? ':20);
      read(p);writeln;
      if(p>0) and (p<=numnodos) then
         begin
         ar1[p].tipo:=contipo;
         k:=k+1;
         end
      else
         begin
         if p=0 then
            ban:=true
         else
            writeln('    ***ERROR. Dato leido: ',p,' fuera de rango');
         end;
      until ban;
      end;
ClrScr;
writeln('TOTAL DE NODOS LEIDOS: ':40,k);
if numnodos <> k then
   writeln('*** ATENCION: Se declararon ',numnodos,
           'nodos y se leyeron: ',k);
for contipo:='S' to 'T' do
     begin
     writeln('*** Nodos tipo "':40,contipo,'" ***');
     for l:=1 to k do
         begin
         if ar1[l].tipo=contipo then
            write(' ',l,',');
         end;
     writeln;writeln;
     for l:=1 to 20 do
         write('----');
     end;
espera;ClrScr;
ban:=false;
```

```
writeln('*** Inicia lectura de arcos ***':50);writeln;
repeat
writeln('Escribe arco: A, B            (termina con 0 0)':60);
read(buf2.x,buf2.y);writeln;
if(buf2.x>0)and(buf2.x<=k)and(buf2.y>0)and(buf2.y<=k) then
   begin
   numarcos:=numarcos+1;
   ar2[numarcos].x:=buf2.x;
   ar2[numarcos].y:=buf2.y;
   end
else
  begin
  if (buf2.x=0) and (buf2.y=0) then
      ban:=true
  else
     writeln('*** ERROR  Datos leidos: A=',buf2.x,' B=',buf2.y);
  end;
until ban;
ClrScr;
writeln('Total de arcos leidos: ':40,numarcos);writeln;
j:=0;
 repeat
 for l:=1 to 3 do
    begin
    if j<numarcos then
       begin
       j:=j+1;
       write('  Arco No. ',j:2,'---(',ar2[j].x:2,',',ar2[j].y:2,')');
       end
    end;
writeln;
until j=numarcos;
end;
end;

procedure marcalibre(var numnodos:integer);
begin
(* Esta subrutina pone la etiqueta cero (0) en los nodos "S" expues*)
(* tos y "borra" las etiquetas de los otros nodos, asignando el va-*)
(* lor: numnodos+1, que no corresponde a ningun nodo de la grafica *)
for l:=1 to numnodos do
    begin
    ar1[l].etiq:=numnodos+1;
    ar1[l].exam:=0;
    if (ar1[l].expo=1)and(ar1[l].tipo='S') then
        ar1[l].etiq:=0;
    end;
end;

procedure aumenta(var p:integer);
var
  bana,banb:boolean;
```

```
  j,q:integer;
begin
(* Con la trayectoria aumentante "P" localizada a partir del nodo  *)
(* "p" que esta subrutina recibe como parametro, se hace la actua- *)
(* lizacion del apareamiento en turno X sustituyendolo por la dife-*)
(* rencia simetrica:   (X-P) U (P-X).                              *)
(*                                                                 *)
bana:=false; j:=0;
 repeat
 j:=j+1;
 ar1[p].expo:=0;
 if ar1[p].etiq <> 0 then
    begin
    q:=ar1[p].etiq;
    banb:=false; l:=0;
     repeat
     l:=l+1;
     with ar2[l] do
          begin
          if( (x=p)and(y=q) )or( (x=q)and(y=p) )then
             begin
             apareado:=j mod 2;
             banb:=true;
             end;
          end;
     until banb;
    p:=q;
    end
 else
    bana:=true;
 until bana;
end;

procedure etiqueta(var numnodos,numarcos:integer);
var
  p:integer;
  flag,ban:boolean;
begin
(* esta subrutina etiqueta arboles alternantes; se detiene cuando -*)
(* los arboles son hungaros.                                       *)
(*                                                                 *)
marcalibre(numnodos);
 repeat
 flag:=true; p:=0;
  repeat
  p:=p+1;
  if (ar1[p].exam=0)and(ar1[p].etiq<=numnodos) then
      begin
      ar1[p].exam:=1; flag:=false;
      end;
  until(not flag)or(p=numnodos);
  if not flag then
```

```
      begin
      if ar1[p].tipo='S' then
         begin
         for l:=1 to numarcos do
             begin
             with ar2[l] do
                  begin
                  if( (x=p)or(y=p) )and(apareado=0) then
                     begin
                     if x=p then
                        begin
                        if ar1[y].etiq>numnodos then
                           ar1[y].etiq:=x;
                        end
                     else
                        begin
                        if ar1[x].etiq>numnodos then
                           ar1[y].etiq:=y;
                        end;
                     end;
                  end;
             end;
         end
      else
         begin
         if ar1[p].expo=1 then
            begin
            aumenta(p);marcalibre(numnodos);
            end
         else
            begin
            ban:=true;l:=1;
            while (l<numarcos)and(ban) do
                begin
                with ar2[l] do
                     begin
                     if( (x=p)or(y=p) )and (apareado=1) then
                        begin
                        if x=p then
                           ar1[y].etiq:=x
                        else
                           ar1[x].etiq:=y;
                        ban:=false;
                        end;
                     end;
                l:=l+1;
                end;
            end;
         end;
      end;
 until flag;
end;
```

```
begin
(*** PROGRAMA PRINCIPAL ***)
repeat
alto:=false;
leegrafica(numnodos,numarcos,alto);
if not alto then
 begin
 etiqueta(numnodos,numarcos);espera;
 ClrScr;writeln('*** ARCOS SOLUCION ***':42);writeln;
 cont:=0;
 for l:=1 to numarcos do
     begin
     if ar2[l].apareado=1 then
        begin
        writeln('  ARCO:  (':25,ar2[l].x:2,',',ar2[l].y:2,
                ') ---> Apareado');
        cont:=cont+1;
        end;
     end;
 writeln;writeln;
 writeln('Total de arcos apareados: ':60,cont);
 writeln;
 for l:=1 to 10 do
     write('--------');
 writeln('*** SOLUCION DUAL ***':42);writeln;
 writeln('Los siguientes nodos forman un recubrimiento minimo: ':75);
 writeln;cont:=0;
 for l:=1 to numnodos do
     begin
     with ar1[l] do
          begin
          if(  ((tipo='S')and(etiq>numnodos))or((tipo='T')and
               (etiq<=numnodos))   ) then
              begin
              cont:=cont+1;
              write(' Nodo: ',l:2,',');
              end;
           end;
     if(cont mod 5)=0 then
        writeln;
     end;
 espera;
 end;
until alto;
writeln;writeln;writeln;
writeln('+ + + FIN DE PROGRAMA + + +':50);
(* Regreso al menu *)
assign(filvar,'MENU.COM');
Execute(filvar)
end
```

ARCHIVO: PROG03.PAS

```
program hungaro;
(*************************************************************************)
(*                                                                       *)
(* Este programa lee una grafica bipartita con pesos asociados a        *)
(* sus arcos, y encuentra un apareamiento con peso total maximo.        *)
(* Proporciona tambien los valores optimos de las variables duales     *)
(* asociadas a los nodos de la grafica.                                 *)
(* El programa se invoca desde el menu con la instruccion "CHAIN",     *)
(* y regresa al menu con la instruccion "EXECUTE". El programa ob-     *)
(* jeto es PROG03.CHN y el fuente es PROG03.PAS.                        *)
(*                                                                       *)
(*************************************************************************)
type
     vertices=record
     tipo:char;
     etiq,exam,expo:integer;
     uv,pi:real;
     end;
     aristas=record
     x,y,apareado:integer;
     peso:real;
     end;
     nodos=array[1..80] of vertices;
     arcos=array[1..400] of aristas;

var
   l,cont,numnodos,numarcos:integer;
   pesomax,d1,d2,d,w:real;
   alto,termina:boolean;
   ar1:nodos; ar2:arcos; buf2:aristas;
   FILVAR:FILE;
(* la siguiente instruccion incluye el archivo ESPERA.PAS *)
(*$I ESPERA.PAS *)

procedure leegrafica(var numnodos,numarcos:integer;
                     var termina:boolean);
var
   k,j,p:integer;
   ban:boolean;
   contipo:char;

begin
(* los valores iniciales de mas o menos infinito que requiere el al*)
(* goritmo original se proporcionan con mas o menos 1e+37.          *)
k:=0; numnodos:=0; numarcos:=0;
writeln('*** APAREAMIENTO PONDERADO BIPARTITA ***':55);
writeln;writeln;
write('Indica cuantos nodos son    (termina con 0) ':20,' ');
read(numnodos);writeln;
```

```
if numnodos=0 then
   termina:=true
else
   begin
   for l:=1 to 20 do
       write('----');
(* se inicializan arreglos de nodos y arcos *)
for l:=1 to 80 do
   begin
   ar1[l].etiq:=0; ar1[l].exam:=0;
   ar1[l].expo:=1;ar1[l].uv:=0;ar1[l].pi:=0;
   end;
for l:=1 to 400 do
   begin
   ar2[l].x:=0;ar2[l].y:=0;ar2[l].apareado:=0;ar2[l].peso:=0.0;
   end;
for contipo='S' to 'T' do
   begin
   ban:=false;
   writeln('Escribe los nodos tipo ':30,contipo,' (termina con 0):');
   writeln;
   repeat
   write('Nodo? ':20);
   read(p);writeln;
   if (p>0) and (p<=numnodos) then
       begin
       ar1[p].tipo:=contipo;
       k:=k+1;
       end
   else
       begin
       if p=0 then
          ban:=true
       else
          writeln('    ***ERROR. Dato leido: ',p,' fuera de rango');
       end;
   until ban;
   end;
ClrScr;
writeln('TOTAL DE NODOS LEIDOS: ':40,k);
if numnodos <> k then
  writeln('***ATENCION: Se declararon ',numnodos,
         ' nodos y se leyeron: ',k);
numnodos:=k;
for contipo:='S' to 'T' do
   begin
   writeln('*** Nodos tipo "':40,contipo,'" ***');
for l:=1 to k do
   begin
   if ar1[l].tipo=contipo then
      write(' ',l,' ');
   end;
```

```
writeln;writeln;
for l :=1 to 20 do
    write('----');
   end;
espera;ClrScr;
ban:=false;
writeln('*** Inicia lectura de arcos ***':50);writeln;
pesomax:=-1e+37;
 repeat
 writeln('Escribe arco: A, B          (termina con 0 0)':70);
 read(buf2.x,buf2.y);
 if(buf2.x>0)and(buf2.x<=k)and(buf2.y>0)and(buf2.y<=k) then
   begin
   write('   ---)  Peso? ');read(buf2.peso);writeln;
   numarcos:=numarcos+1;
   ar2[numarcos].x:=buf2.x;
   ar2[numarcos].y:=buf2.y;
   ar2[numarcos].peso:=buf2.peso;
   if(buf2.peso>pesomax) then
      pesomax:=buf2.peso;
   end
 else
   begin
   if(buf2.x=0) and (buf2.y=0) then
      ban:=true
   else
      writeln('***ERROR.  Datos leidos: A=',buf2.x,' B=',buf2.y);
   end;
 until ban;
ClrScr;
writeln('Total de arcos leidos: ':40,numarcos);writeln;
j:=0;
 repeat
 for l:=1 to 2 do
    begin
    if j<numarcos then
       begin
       j:=j+1;
       write('     Arco# ',j:2,':(',ar2[j].x:2,',' ar2[j].y:2,
             ')  PESO= ',ar2[j].peso:8:2);
       end
    end;
 writeln;
 until j=numarcos;
 for l:=1 to numnodos do
    begin
    if ar1[l].tipo='S' then
       ar1[l].uv:=pesomax
    else
       ar1[l].pi:=1e+37;
    end;
 espera;
```

```
 end;
end;

procedure marcalibre(var numnodos:integer);
begin
(* esta subrutina pone la etiqueta cero (0) en todos los nodos "S"  *)
(* expuestos y "borra" las etiquetas en los otros nodos, asignando *)
(* el valor:  numnodos+1, que no corresponde a ningun nodo en la - *)
(* grafica.                                                          *)
for l:=1 to numnodos do
    begin
    ar1[l].etiq:=numnodos+1;
    ar1[l].exam:=0;
    if(ar1[l].expo=1)and(ar1[l].tipo='S') then
       ar1[l].etiq:=0;
    end;
end;

procedure aumenta(var p,numnodos:integer);
var
  bana,banb:boolean;
  j,q:integer;
begin
(* a partir de la trayectoria aumentante "P" que inicia en el nodo *)
(* "p" que esta subrutina recibe como parametro, se actualiza el a-*)
(* pareamiento en turno X, sustituyendolo por la diferencia simetri*)
(* ca:   (X-P) U (P-X).                                              *)
bana:=false; j:=0;
 repeat
 j:=j+1;
 ar1[p].expo:=0;
 if ar1[p].etiq <> 0 then
    begin
    q:=ar1[p].etiq;
    banb:=false; l:=0;
     repeat
     l:=l+1;
     with ar2[l] do
           begin
           if( (x=p)and(y=q) )or( (x=q)and(y=p) ) then
              begin
              apareado:=j mod 2;
              banb:=true;
              end;
           end;
     until banb;
    p:=q;
    end
 else
    bana:=true;
 until bana;
for l:=1 to numnodos do
```

```
     begin
     if ar1[1].tipo='T' then
        ar1[1].pi:=1e+37;
     end;
end;

procedure etiqueta(var numnodos,numarcos:integer);
var
   p:integer;
   pij:real;
   flag,ban:boolean;
begin
(* esta subrutina etiqueta arboles alternantes; se detiene cuando  *)
(* es necesario hacer cambios en las variables duales.             *)
 repeat
 flag:=true; p:=0; pij:=0.0;
  repeat
  p:=p+1;
  with ar1[p] do
      begin
      if((exam=0)and(etiq<=numnodos)) and
        ( (tipo='S')or((tipo='T')and(pi=0)) ) then
        begin
        exam:=1; flag:=false;
        end;
      end;
  until(not flag)or(p=numnodos);
  if not flag then
     begin
     if ar1[p].tipo='S' then
        begin
        for l:=1 to numarcos do
            begin
            with ar2[l] do
                 begin
                 if( (x=p)or(y=p) )and(apareado=0) then
                    begin
                    pij:=ar1[x].uv+ar1[y].uv-peso;
                    if (x=p) and (pij<ar1[y].pi) then
                       begin
                       ar1[y].etiq:=x; ar1[y].pi:=pij;
                       end;
                    if (y=p) and (pij<ar1[x].pi) then
                       begin
                       ar1[x].etiq:=y; ar1[x].pi:=pij;
                       end;
                    end;
                 end;
            end;
        end
     else
        begin
```

```
        if ar1[p].expo:=1 then
           begin
           aumenta(p,numnodos);marcalibre(numnodos);
           end
        else
           begin
           ban:=true; l:=1;
           while (l(numarcos)and(ban) do
               begin
               with ar2[l] do
                   begin
                   if( (x=p)or(y=p) )and(apareado=1) then
                      begin
                      if x=p then
                         ar1[y].etiq:=x
                      else
                         ar1[x].etiq:=y;
                      ban:=false;
                      end;
                   end;
               l:=l+1;
               end;
           end;
        end;
     end;
 until flag;
end;

procedure cambiodual(var numnodos:integer; var alto:boolean);
begin
(* esta subrutina hace los cambios en las variables duales; contie-*)
(* ne el criterio para encontrar la solucion optima, que se comuni-*)
(* ca al programa principal con variable logica "alto".           *)
(*                                                                *)
d:=1e+37; d1:=1e+37; d2:=1e+37; alto:=false;
for l:=1 to numnodos do
    begin
    with ar1[l] do
        begin
        if(tipo='S')and(uv(d1) then
           d1:=uv;
        if(tipo='T')and(pi)0)and(pi(d2) then
           d2:=pi;
        end;
    end;
d:=d1;
if d2(d1 then
   d1:=d2;
for l:=1 to numnodos do
    begin
    with ar1[l] do
        begin
```

```
        if(tipo='S')and(etiq<=numnodos) then
           uv:=uv-d;
        if(tipo='T')and(pi=0) then
           uv:=uv+d;
        if(tipo='T')and(pi>0) then
           pi:=pi-d;
        end;
    end;
if (d>=d1) then
    alto:=true;
end;

begin
(*** PROGRAMA PRINCIPAL ***)
repeat
 ClrScr;
 termina:=false;
 leegrafica(numnodos,numarcos,termina);marcalibre(numnodos);
 if not termina then
  begin
  alto:=false;
  while not alto do
     begin
     etiqueta(numnodos,numarcos);
     cambiodual(numnodos,alto);
     end;
  ClrScr;writeln('*** ARCOS SOLUCION ***':47);writeln;
  cont:=0; w:=0.0;
  for l:=1 to numarcos do
     begin
     if ar2[l].apareado=1 then
        begin
        writeln('  ARCO: (':25,ar2[l].x:2,',',ar2[l].y:2,')  Peso= ',
                ar2[l].peso:8:2,'     Apareado');
        cont:=cont+1; w:=w + ar2 [l].peso;
        end;
     end;
  writeln;writeln;
  writeln('Total de arcos apareados: ':30,cont,
          '     ---> Peso total* Maximo= ',w:8:2);
  writeln;
  for l:=1 to 10 do
      write('--------');
  writeln('*** SOLUCION DUAL ***':46);writeln;
  writeln('Valores optimos de las variables duales (asociadas a
           nodos):  ':75);
  writeln;cont:=0;
  for l:=1 to numnodos do
      begin
      with ar1[l] do
           begin
           cont:=cont+1;
```

```
            write(' Nodo: ',1:2,', Var. Dual U:',uv:8:2);
            end;
        if(cont mod 2)=0 then
           writeln;
        end;
    espera;
    end;
until termina;
writeln;writeln;writeln;
writeln('+ + + FIN DE PROGRAMA + + +':52);
(* Regreso al menu *)
assign(filvar,'MENU.COM');
Execute(filvar);
end.
```

ARCHIVO: PROG04.PAS

```
program bottleneck;
(*****************************************************************)
(*                                                               *)
(* Este programa lee una grafica bipartita con pesos asociados  *)
(* a sus arcos y encuentra la asignacion con el maximo valor en *)
(* el arco de peso minimo (Apareamiento Max-Min).                *)
(* El programa se invoca desde el menu con la instruccion:       *)
(* "CHAIN" y regresa al menu con la instruccion "EXECUTE".       *)
(* El programa objeto es PROG04.CHN y el fuente es PROG04.PAS.   *)
(*                                                               *)
(*****************************************************************)
type
     vertices=record
     tipo:char;
     etiq,exam,expo:integer;
     pi:real;
     end;
     aristas=record
     x,y,apareado:integer;
     peso:real;
     end;
     nodos=array[1..80] of vertices;
     arcos=array[1..400] of aristas;

var
    l,cont,numnodos,numarcos:integer;
    pesomax,d,w:real;
    termina:boolean;
    ar1:nodos; ar2:arcos; buf2:aristas;
    FILVAR:FILE;
(* la siguiente instruccion incluye el archivo ESPERA.PAS *)
(*$I ESPERA.PAS *)

procedure leegrafica(var numnodos,numarcos:integer;
                     var termina:boolean);
var
    k,j,p:integer;
    ban:boolean;
    contipo:char;

begin
(* los valores infinitos iniciales que requiere el algoritmo ori*)
(* ginal se proporcionan como el valor real:  1e+37             *)
(*                                                              *)
k:=0; numnodos:=0; numarcos:=0;
ClrScr;
writeln('*** APAREAMIENTO MAX-MIN (CUELLO DE BOTELLA) ***':60);
writeln;writeln;
write('Indica cuantos nodos son  (termina con 0): ':20,' ');
```

```
read(numnodos);writeln;
if (numnodos=0) then
   termina:=true
else
 begin
 for l:=1 to 20 do
     write('----');
(* se inicializan arreglos de nodos y arcos *)
 for l:=1 to 80 do
     begin
     ar1[l].etiq:=0;ar1[l].exam:=0;
     ar1[l].expo:=1;ar1[l].pi:=0;
     end;
 for l:=1 to 400 do
     begin
     ar2[l].x:=0; ar2[l].y:=0; ar2[l].apareado:=0; ar2[l].peso:=0.0;
     end;
 pesomax:=1e+37;
 for contipo:='S' to 'T' do
     begin
     ban:=false;
     writeln('Escribe los nodos tipo ':30,contipo,
             '    (termina con 0):');writeln;
     repeat
     write('Nodo? ':20);
     read(p);writeln;
     if (p>0) and (p<=numnodos) then
         begin
         ar1[p].tipo:=contipo;
         if contipo='T' then
            ar1[p].pi:=-1e+37;
         k:=k+1;
         end
     else
         begin
         if p=0 then
            ban:=true
         else
            writeln('     ***ERROR. Dato leido: ',p,' fuera de rango');
         end;
     until ban;
     end;
 ClrScr;
 writeln('TOTAL DE NODOS LEIDOS: ':40,k);
 if numnodos <>k then
   writeln('*** ATENCION: Se declararon ',numnodos,
           ' nodos y se leyeron ',k);
 numnodos:=k;
 for contipo:='S' to 'T' do
     begin
     writeln('*** Nodos tipo "':40,contipo,'" ***');
     for l:=1 to k do
```

```
          begin
          if ar1[1].tipo=contipo then
             write(' ',l,',');
          end;
     writeln;writeln;
     for l:=1 to 20 do
         write('----');
     end;
  espera;ClrScr;
  ban:=false;
  writeln('*** Inicia lectura de arcos ***':50);writeln;
   repeat
   writeln('Escribe arco: A, B      (termina con 0 0)':70);
   read(buf2.x,buf2.y);
   if(buf2.x>0)and(buf2.x<=k)and(buf2.y>0)and(buf2.y<=k) then
     begin
     write('  ---)  Peso? ');read(buf2.peso);writeln;
     numarcos:=numarcos+1;
     ar2[numarcos].x:=buf2.x;
     ar2[numarcos].y:=buf2.y;
     ar2[numarcos].peso:=buf2.peso;
     if(buf2.peso>pesomax) then
        pesomax:=buf2.peso;
     end
   else
     begin
     if (buf2.x=0) and (buf2.y=0) then
        ban:=true
     else
        writeln('*** ERROR. Datos leidos: A=',buf2.x,' B=',buf2.y);
     end;
   until ban;
  ClrScr;
  writeln('Total de arcos leidos: ':40,numarcos);writeln;
  j:=0;
   repeat
   for l:=1 to 2 do
       begin
       if j<numarcos then
          begin
          j:=j+1;
          write('    Arco# ',j:2,':(',ar2[j].x:2,',',ar2[j].y:2,
                ') PESO= ',ar2[j].peso:8:2);
          end
       end;
   writeln;
   until j=numarcos;
   espera;
   end;
  end;

procedure marcalibre(var numnodos:integer);
```

```
begin
(* esta subrutina pone la etiqueta cero (0) en todos los nodos  *)
(* "S" expuestos y "borra" las etiquetas en los otros nodos asig*)
(* nando el valor:  numnodos+1, que no corresponde a ningun nodo*)
(* de la grafica                                                *)
(*                                                              *)
for l :=1 to numnodos do
    begin
    ar1[l].etiq:=numnodos+1;
    ar1[l].exam:=0;
    if(ar1[l].expo=1)and(ar1[l].tipo='S') then
       ar1[l].etiq:=0;
    end;
end;


procedure aumenta(var p, numnodos:integer);
var
  bana,banb:boolean;
  j,q:integer;
begin
(* a partir de la trayectoria aumentante "P" que inicia en el *)
(* nodo "p" que esta subrutina recibe como parametro, se ac-  *)
(* tualiza el apareamiento en turno "X", sustituyendolo por la*)
(* diferencia simetrica:    (X-P) U (P-X)                     *)
(*                                                            *)
bana:=false; j:=0;
  repeat
  j:=j+1;
  ar1[p].expo:=0;
  if ar1[p].etiq <>0 then
     begin
     q:=ar1[p].etiq;
     banb:=false; l:=0;
     repeat
     l:+l+1;
     whith ar2[l] do
           begin
           if( (x=p) and (y=q) )or( (x=q)and(y=p) )  then
              begin
              apareado:=j mod 2;
              banb:=true;
              end;
           end;
     until banb;
    p:=q;
    end
   else
      bana:=true;
   until bana;
 for l:=1 to numnodos do
     begin
```

```
      if ari[l].tipo='T' then
         ari[l].pi:=-1e+37;
      end;
 end;


procedure etiqueta(var numnodos, numarcos: integer);
var
   p:integer;
   pij:real;
   flag,ban:boolean;

begin
(* esta subrutina etiqueta arboles alternantes;se detiene cuan-*)
(* do los arboles son hungaros                                  *)
(*                                                              *)
repeat
flag:=true; p:=0; pij:=0.0;
repeat
p:=p+1;
with ari[p] do
   begin
if (( exam=0)and(etiq<=numnodos))and((tipo='S')or((tipo='T')and
    (pi)=pesomax))) then
   begin
   exam:=1;flag:=false;
   end;
 end;
until (not flag)or( p=numnodos);
if flag then
   begin
   p:=0; d:=-1e+37;
   for l:=1 to numnodos do
    begin
    with ari[l] do
          begin
          if(exam=0)and(etiq<=numnodos)and(tipo='T')and
             (pi<pesomax)and(pi>=d) then
              begin
              d:=pi; p:=l; flag:=false;
              end;
           end;
   end;
if not flag then
   pesomax:=d;
end;
if not flag then
   begin
   if ari[p].tipo='S' then
      begin
      for l:=1 to numarcos do
      begin
```

```
        with ar2[l] do
        begin
        if ( (x=p)or(y=p) )and(apareado=0) then
           begin
           if(x=p)and(ar1[y].pi<peso)and(ar1[y].pi<pesomax) then
              begin
              ar1[y].etiq:=x; ar1[y].pi:=peso;
              end;
           if(y=p)and(ar1[x].pi<peso)and(ar1[x].pi<pesomax) then
              begin
              ar1[x].etiq:=y; ar1[x].pi:=peso;
              end;
           end;
        end;
       end;
       end
     else
       begin
       if ar1[p].expo=1 then
          begin
          aumenta(p,numnodos);marcalibre(numnodos);
          end
       else
          begin
          ban:=true; l:=1;
          while (l<numarcos)and(ban) do
              begin
              with ar2[l] do
                  begin
                  if( (x=p)or(y=p) )and(apareado=1) then
                     begin
                     if x=p then
                        ar1[y].etiq:=x
                     else
                        ar1[x].etiq:=y;
                     ban:=false;
                     end;
                  end;
             l:=l+1;
             end;
         end;
      end;
   end;
until flag;
 end;


 begin
 (*** PROGRAMA PRINCIPAL ***)
 repeat
  termina:=false;
  leegrafica (numnodos,numarcos,termina); marcalibre(numnodos);
```

```
begin
etiqueta(numnodos, numarcos);
ClrScr;writeln('*** ARCOS SOLUCION ***':47);writeln ;
cont:=0;w:=0.0;d:=1e+37;
for l:=1 to numarcos do
  begin
  if ar2[l].apareado=1 then
     begin
     if ar2[l].peso <= d then
        d:=ar2[l].peso;
     writeln('ARCO : (':25,ar2[l].x:2,',',ar2[l].y:2,')     Peso=',
            ar2[l].peso:8:2,'     Apareado');
     cont:=cont+1; w:=w + ar2[l].peso;
     end;
  end;
writeln;writeln;
writeln('Total de arcos apareados: ':30,cont,'---> Peso Total=',
        w:8:2);
writeln;
writeln('*** Valor maximo del Peso Minimo de los arcos = ':49,
        d:12:2);
espera;
end;
until termina;
writeln;writeln;writeln;writeln('+ + + FIN DE PROGRAMA + + +':52);
(* Regreso al menu *)
assing(filvar,'MENU.COM');
Execute (filvar);
end;
```

ARCHIVO· PROG05.PAS

```
program marriage;
(*****************************************************************)
(*                                                               *)
(* Este programa lee las preferencias de apareamiento en dos     *)
(* grupos de nodos de una grafica bipartita, y encuentra  el     *)
(* apareamiento optimo en el sentido de Gale-Shapley             *)
(* El programa se invoca desde el menu con la instruccion:       *)
(* "CHAIN" y regresa al menu con la instruccion "EXECUTE".       *)
(* El programa objeto es PROG05.CHN y el fuente es PROG05.PAS.   *)
(*                                                               *)
(*****************************************************************)
(* PROGRAMA PARA RESOLVER APAREAMIENTO GALE-SHAPLE *)
var
  i,j,el,m,n,s,t,w:integer;
  prefer,rank:array[1..100,1..100] of integer;
  prox,fiance:array[1..100] of integer;
(* La siguiente instruccion incluye el archivo ESPERA.PAS *)
  FILVAR:FILE;

(*$I ESPERA.PAS  *)

begin
repeat
 ClrScr;
 writeln('*** PROGRAMA PARA ENCONTRAR PAREJAS ESTABLES ***':60);
 writeln;
 writeln('       ( --- Apareamiento GALE/SHAPLEY --- )      ':60);
 writeln;
 writeln('Indica el numero de casos.    (termina con 0)');
 n:=0;
 read(n);writeln;
 if n<>0 then
  begin
  (* se inicializan vectores, y se asigna a cada mujer el hombre *)
  (* menos preferido que todos por medio del valor : n+1         *)
  (*                                                             *)
  for i:=1 to n do
      begin
      prox[i]:=0;
      fiance[i]:=n+1;
      end;
  for i:=1 to 80 do
      write('_');
  writeln;
  writeln('Indica las Preferencias de los Hombres:');writeln;
  for i:=1 to n do
      begin
      (* se leen las preferencias de los hombres; "prefer[i,j]"  *)
      (* es la mujer que en el lugar numero "j" prefiere el hom- *)
```

```
    (* bre "i".                                                     *)
    (*                                                              *)
    writeln(' +++ Hombre No. ':44,i);j:=0;
    repeat
        j:=j+1;
        writeln('*** Eleccion Numero  ',j);
        read(prefer[i,j]);writeln;
        if(prefer[i,j]<1) or (prefer[i,j]>n) then
          begin
          j:=j-1;
          writeln('*** ERROR Preferencia leida fuera de rango':60);
          end;
      until j=n;
   end;
writeln('#*#*#* Indica las preferencias de las Mujeres...':70);
for i:=1 to n do
    begin
    (* Se leen preferencias de las mujeres y se forma la tabla de  *)
    (* rangos para ellas; "rank[i,j]" representa el lugar de pre-  *)
    (* ferencia que el hombre "j" tiene para la mujer "i".         *)
    (*                                                             *)
    writeln('-*- Mujer Numero ':44,i);j:=0;
    repeat
        j:=j+1;
        writeln('-*-*-*- Eleccion Numero ',j);
        read(el);writeln;
        if(el<1) or (el>n) then
           begin
           j:=j-1;
           writeln('*** ERROR)Preferencia leida fuera del rango':60);
           end
        else
           rank[i,el]:=j;
        until j=n;
    rank[i,n+1]:=n+1;
    end;
(* Se despliegan datos leidos *)
ClrScr;
writeln;writeln('*** DATOS LEIDOS ':50);
writeln('------------------------------------------':60);
writeln(' --- Num. de casos: ',n);writeln;
writeln('Preferencias de los HOMBRES:':60);writeln;
for i:=1 to n do
    begin
    write('Hombre No. ',i,':');
    for j:=1 to n do
        write(' ',prefer[i,j]);
    writeln;
    end;
espera;
write('-*- Preferencias de las MUJERES -*-':60);writeln;
for i:=1 to n do
```

```
    begin
    m:=0;
    write('-*- Mujer No. ',i,': ');
    repeat
    m:=m+1;
    for j:=1 to n do
        begin
        if (rank[i,j]=m) then
           write(' ',j);
        end;
    until m=n;
    writeln;
    end;
espera;
(* inicia el algoritmo,"fiance[j]" es el hombre que se asigna de    *)
(* pareja para la mujer "j"                                          *)
(*                                                                   *)
for m:=1 to n do
    begin
    s:=m;
    repeat
     prox[s]:=prox[s]+1;
     w:=prefer[s,prox[s]];
     if rank[w,s]<rank[w,fiance[w]] then
        begin
        t:=fiance[w];
        fiance[w]:=s;
        s:=t;
        end;
    until s=n+1
    end;ClrScr;
writeln('*** S O L U C I O N  O P T I M A:':50);
writeln('-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=':55);
for j:=1 to n do
     writeln('Mujer # ':20,j,'<--- Hombre  # ',fiance[j]);
   writeln;
  espera;
 end;
until n=0;
writeln;writeln;writeln;
writeln('+ + + FIN DE PROGRAMA + + +':50);
(* Regreso al menu *)
assing(filvar,'MENU.COM');
Execute(filvar);
end.
```

* INDICE ALFABETICO *