

PRÁCTICAS DE MATEMÁTICA DISCRETA CON MaGraDa

Manuel Ángel Caballero Palomino / Violeta
Migallón Gomis / José Penadés Martínez

TD

TEXTOSDOCENTES

PUBLICACIONES

Universidad de Alicante

© los autores

© de la presente edición

Publicaciones de la Universidad de Alicante

Campus de San Vicente s/n

03690 San Vicente del Raspeig

Publicaciones@ua.es

<http://publicaciones.ua.es>

ISBN: 84-7908-641-6

Depósito Legal: A-1480-2001

Reservados todos los derechos. No se permite reproducir, almacenar en sistemas de recuperación de la información ni transmitir alguna parte de esta publicación, cualquiera que sea el medio empleado -electrónico, mecánico, fotocopia, grabación, etc.-, sin el permiso previo de los titulares de los derechos de la propiedad intelectual.

Edición electrónica:



M. Caballero, V. Migallón y J. Penadés

Prácticas de matemática discreta con MaGraDa

Índice

Portada

Créditos

Prólogo **9**

1 Introducción a MaGraDa **13**

1.1 Introducción 13

1.2 El modo de texto 17

1.2.1 Introducción de un grafo 19

1.2.2 Borrando un grafo ya existente 23

1.2.3 Seleccionando un grafo 23

1.2.4 Uso de ficheros para abrir o guardar un grafo 24

1.2.5 Modificación de un grafo 25

1.2.6 Salir de MaGraDa 27

1.2.7 Práctica 1 28

1.3 El modo gráfico 31

1.3.1 El lienzo 33

1.3.2 Introducción de un grafo 34

1.3.3 Práctica 2 41

2 Fundamentos **45**

2.1 Introducción 45

2.2 Cálculos básicos 46

Índice

2.2.1	Grado	46
2.2.2	Ver	50
2.2.3	Grafo (simple, cíclico, completo y conexo)	50
2.2.4	Grafo no dirigido asociado	52
2.2.5	Grafos isomorfos	53
2.2.6	Práctica 3	57
2.3	Matriz de adyacencia	68
2.3.1	Práctica 4	70
3	Estudio de la accesibilidad y conectividad	73
3.1	Introducción	73
3.2	Matriz de accesibilidad y matriz de acceso . . .	73
3.2.1	Vértices alcanzables desde otro	73
3.2.2	Vértices que alcanzan a otro	76
3.2.3	Accesibilidad y algoritmo de Warshall . .	78
3.2.4	Práctica 5	81
3.3	Cálculo de componentes conexas	89
3.3.1	Práctica 6	94
4	Problemas de recorrido de aristas y vértices	107
4.1	Introducción	107
4.2	Problema de recorrido de aristas	107
4.2.1	Práctica 7	112
4.3	Problema de recorrido de vértices	119
4.3.1	Práctica 8	123

Índice

5	Introducción a la estructura de árbol	129
5.1	Introducción	129
5.2	Definiciones y propiedades	129
5.2.1	Práctica 9	132
5.3	Árboles con raíz	140
5.3.1	Práctica 10	142
5.4	Notación polaca	149
5.4.1	Práctica 11	153
6	Grafos ponderados. Caminos críticos en grafos acíclicos	157
6.1	Introducción	157
6.2	Creación de un grafo ponderado	157
6.2.1	Práctica 12	160
6.3	Ecuaciones de Bellman	164
6.3.1	Caminos más cortos	164
6.3.2	Secuenciación de actividades (PERT) . .	172
6.3.3	Práctica 13	176
7	Caminos más cortos y árboles generadores de peso mínimo	191
7.1	Introducción	191
7.2	Caminos más cortos	191
7.2.1	Algoritmo de Dijkstra	191
7.2.2	Algoritmo de Floyd y Warshall	199

Índice

7.2.3	Práctica 14	204
7.3	Árboles generadores de peso mínimo	219
7.3.1	Algoritmo de Kruskal	219
7.3.2	Algoritmo de Prim	222
7.3.3	Práctica 15	227
Bibliografía		237

Prólogo

Este libro va dirigido al alumnado de las titulaciones de informática de la Universidad de Alicante y cubre parte de las prácticas de la asignatura Matemática Discreta. Para la realización de estas prácticas se ha elegido el paquete de software MaGraDa (**G**rafos para **M**atemática **D**iscreta). Esta aplicación se ha realizado en el Departamento de Ciencia de la Computación e Inteligencia Artificial, inicialmente como proyecto de la asignatura Sistemas Informáticos, por el ingeniero informático Manuel Ángel Caballero Palomino y bajo el asesoramiento de la profesora Violeta Migallón Gomis y el profesor José Penadés Martínez.

MaGraDa ha sido diseñado específicamente para fines docentes y cubre gran parte de los conceptos de la teoría de grafos que se estudian en la asignatura de Matemática Discreta, pretendiendo automatizar el trabajo con grafos y además que quién utilice MaGraDa se familiarice con ellos de manera eficaz. Los objetivos que se pretenden alcanzar con la realiza-

ción de las prácticas de este libro no se limitan únicamente al conocimiento de su manejo. Se desea también que al finalizar dichas prácticas, se hayan afianzado los conceptos estudiados en las clases teóricas, sabiendo plantear inicialmente el problema propuesto, para posteriormente, utilizando o no MaGraDa, poder obtener las conclusiones pertinentes.

Este libro comienza con un capítulo introductorio en el que se describe MaGraDa y se dan algunas ideas básicas para su manejo. El segundo capítulo se centra en la utilización de MaGraDa para resolver problemas básicos de grafos, relacionados con sus fundamentos. En el capítulo tercero se estudiará la accesibilidad y la conectividad de un grafo con MaGraDa. El capítulo cuarto está dedicado a los problemas de recorrido de aristas y vértices. En el capítulo quinto se tratará la estructura de árbol. Los capítulos sexto y séptimo están dedicados a los grafos ponderados. En el capítulo sexto se verá cómo introducir estos grafos con MaGraDa y la forma de obtener caminos críticos. Por último, en el capítulo séptimo, se tratan los problemas de caminos más cortos y árboles generadores de peso mínimo.

En este libro las prácticas van entrelazadas con las explicaciones. Nuestro consejo es que antes de una sesión de prácticas se lea el capítulo (o parte del capítulo) correspondiente a esa sesión, así como los apuntes de la parte teórica

correspondiente. De esa forma, antes de empezar la práctica y después de las explicaciones complementarias que se den en las clases, se podrán tratar las dudas surgidas.

Esperamos que la realización de estas prácticas ayude al alumnado de informática a comprender con más facilidad la matemática discreta.

Por último, mencionar que los autores han decidido que MaGraDa sea un software de dominio público, esperando que sea útil a toda la comunidad universitaria. Está disponible en <http://www.dccia.ua.es/dccia/inf/ asignaturas/MD/>.

Prólogo

1. Introducción a MaGraDa

1.1 Introducción

El paquete de software MaGraDa (**G**rafos para **M**atemática **D**iscreta) es una aplicación informática programada en lenguaje Java y diseñada específicamente para trabajar con grafos. MaGraDa trabaja con grafos tanto dirigidos como no dirigidos y ponderados como no ponderados, pero por el momento, sólo se trabajará con grafos no ponderados. Más adelante, en el capítulo 6, introduciremos los grafos ponderados. Según la filosofía de MaGraDa podríamos trabajar con un número ilimitado de vértices, pero hemos creído conveniente sólo permitir trabajar con grafos de hasta un máximo de 50 vértices. Este paquete es sencillo y cómodo de manejar, está basado en menús sobre pantalla y consta de dos pantallas de visualización:

(i) El modo de texto: Llamémoslo de esta forma para diferenciarlo del otro. Permite trabajar con los grafos de forma analítica. Es decir, trabajaremos en todo momento con

los datos del grafo, pero sin visualizarlo gráficamente.

(ii) El modo gráfico: Trabaja con los grafos de forma que pueden verse gráficamente.

Ambas pantallas de trabajo son prácticamente equivalentes en funcionalidad. Es decir, no hay ningún método que esté sólo implementado para una forma de trabajo exclusivamente. Sin embargo, la forma de ofrecer los resultados no es la misma en los dos modos. En cada uno de ellos se muestran los resultados intentando maximizar la comprensión de los mismos. Básicamente, podemos agrupar las aplicaciones que nos ofrece MaGraDa en tres partes:

- **Manejo de grafos (Grafo):** Ésta es la parte donde se pueden crear grafos nuevos o abrir grafos ya creados desde un fichero, modificarlos, borrarlos de memoria, seleccionarlos o guardarlos en un fichero para su tratamiento posterior.
- **Cálculos básicos:** Hay una serie de características o propiedades básicas de los grafos que se pueden averiguar fácilmente con esta serie de métodos, tales como el grado de un vértice, la matriz de adyacencia o pesos, o ver las aristas (o arcos) que pueda tener el grafo. También, en el caso de que el grafo sea dirigido, MaGraDa

nos ofrece la utilidad de obtener su correspondiente grafo no dirigido asociado. Para grafos no dirigidos obtiene un árbol generador de los muchos que pueda tener. Podemos estudiar, también desde este menú, si dos grafos son isomorfos, ver qué vértices alcanzan a otros, así como qué vértices son alcanzados por otros. Nos indica además si el grafo es simple, cíclico, completo o conexo. Otra aplicación de gran interés es el cálculo de componentes conexas.

- **Algoritmos:** Esta última parte es la más importante de la aplicación. Dispone de algoritmos muy conocidos en el mundo de los grafos, tales como los algoritmos de Warshall, Fleury, caminos más cortos en grafos acíclicos, PERT (Program Evaluation and Review Technique), Dijkstra, Floyd y Warshall, Kruskal y Prim. Sin duda lo más importante aquí es que MaGraDa los aplica sobre los grafos en curso, de manera que podamos ver los resultados intermedios para así entender mejor el funcionamiento del correspondiente algoritmo.

Una de las ventajas que ofrece MaGraDa es que puede tener en memoria varios grafos al mismo tiempo. De esta forma, se puede seleccionar aquel que más convenga en cada momento, sin tener que preocuparse por guardarlo en disco

antes. La aplicación los mantendrá en memoria y al acabar la sesión, el mismo programa será quien recuerde si se desea guardar los grafos en un archivo para su uso posterior. No obstante, es conveniente grabar los grafos en un archivo nada más crearlos por posibles fallos inesperados en el sistema.

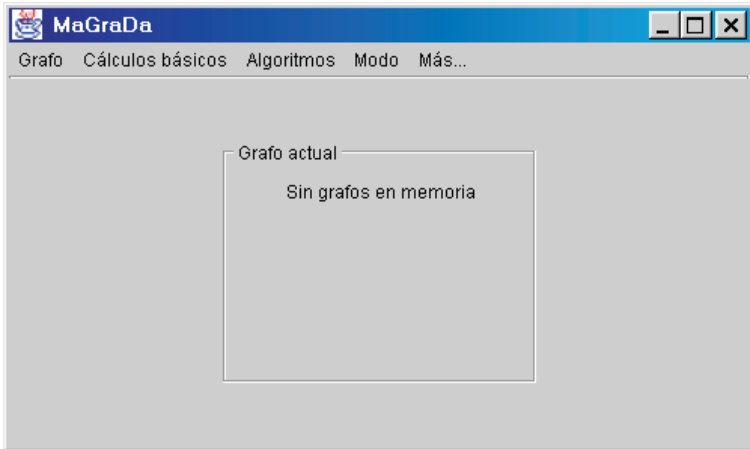
Llegados a este punto, lo mejor que podemos hacer es ir adentrándonos un poco más en cómo trabaja MaGraDa. Así nos daremos cuenta mucho mejor de sus posibilidades. En este capítulo, nos vamos a centrar en la parte correspondiente al manejo de datos. En la sección 1.2 veremos la forma de introducir y manipular un grafo desde el modo de texto, en la sección 1.3 estudiaremos la manera de hacer esto mismo desde el modo gráfico. Pero antes de nada ejecutemos MaGraDa. Recordemos que es una aplicación en lenguaje Java, por tanto, si no disponemos del icono correspondiente, tendremos que teclear `java -jar magrada.jar`. Aparecerá la siguiente pantalla:



1.2 El modo de texto

Una vez situados en la portada de MaGraDa, pulsando Comenzar, aparecerá la siguiente pantalla principal del programa desde la que se trabaja cuando estamos en el modo de texto:

Introducción a MaGraDa



Como vemos, MaGraDa busca ya desde el principio y con su pantalla principal una de sus principales características, la sencillez. Claramente podemos observar, si desplegamos los distintos menús, cómo estructura MaGraDa su trabajo. Los tres primeros menús engloban toda la potencia de la aplicación. El cuarto menú *Modo* nos permitirá pasar a trabajar al ya mencionado modo gráfico del programa y el último hace referencia al aspecto de las ventanas de MaGraDa, al logotipo del programa y a sus autores.

Además de los menús, la pantalla principal también ofrece información básica de los datos del grafo que en cada momento se tenga seleccionado. Notemos que cuando no hay ningún grafo en memoria, prácticamente todas las opcio-

nes de los tres primeros menús están inhabilitadas, excepto aquellas que nos permiten agregar grafos o bien salir del programa.

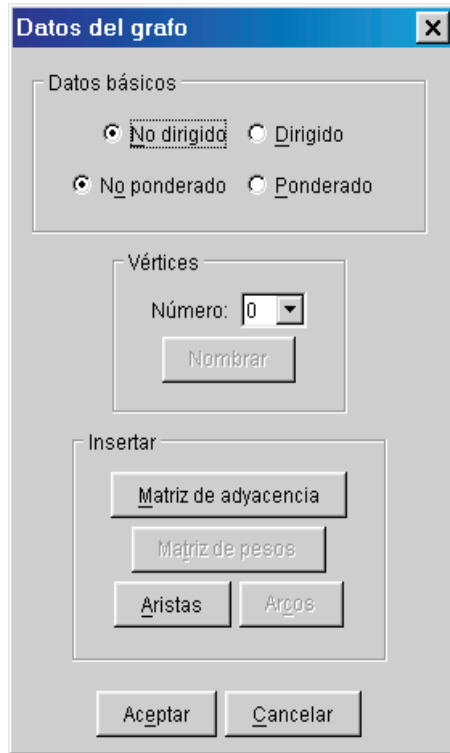
1.2.1 Introducción de un grafo

A continuación vamos a ver cómo se crea un grafo desde el modo de texto. Para ilustrar el proceso consideraremos el siguiente grafo no dirigido:

$$G = \{V, A\}, \quad V = \{a, b, c, d\},$$

$$A = \{\{a, a\}, \{a, b\}, \{a, c\}, \{b, c\}, \{b, d\}, \{c, d\}\}.$$

Para la creación de un grafo desde el modo de texto debemos situarnos en la opción Nuevo del menú Grafo. Aparecerá una pantalla como ésta:

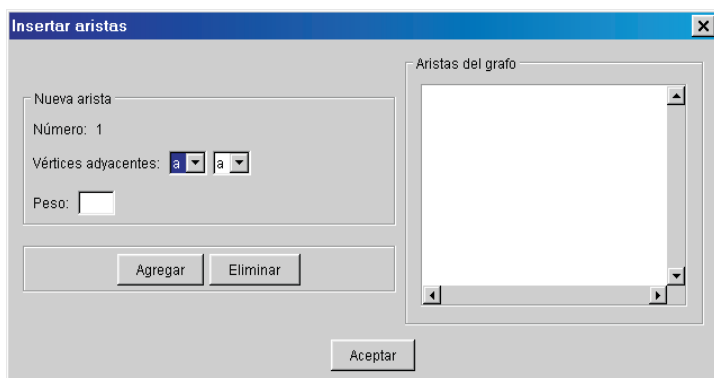


En ella insertamos los datos básicos de nuestro grafo. En una primera parte, le decimos si el grafo que se va a introducir es Dirigido o No dirigido y Ponderado o No ponderado. En nuestro ejemplo, por tanto, deberemos indicar que el grafo es no dirigido y no ponderado.

También debemos indicar el número de vértices que queremos que tenga el grafo (un número mayor que 0, lógicamente)

y nos permitirá etiquetarlos (o nombrarlos), si queremos, con el botón *Nombrar*. En nuestro caso indicaremos que el grafo va a tener cuatro vértices y los llamaremos a, b, c y d , respectivamente.

Por último, según los primeros datos, nos dejará insertar Aristas o Arcos o bien acceder directamente a la Matriz de adyacencia o a la Matriz de pesos para poder rellenarlas convenientemente. En nuestro caso, ya que se trata de un grafo no ponderado y no dirigido, podremos crear el grafo insertando las aristas o mediante la matriz de adyacencia. Por el momento nos limitaremos a insertar aristas. La introducción de un grafo mediante la matriz de adyacencia se verá más adelante. Si pulsamos el botón *Aristas* de la pantalla anterior nos saldrá algo así:



Introducción a MaGraDa

Se observa que hay dos cajas con todos los vértices en cada una de ellas. Sólo hay que ir seleccionando parejas de vértices. Como se trata de aristas, no importa el orden en el que pongamos los vértices de éstas. Una vez seleccionada una pareja, la agregamos e inmediatamente aparecerá en la ventana derecha, queriendo decir así, que ya pertenece al grafo. Si queremos eliminar alguna de las ya hechas, la formamos como si la fuéramos a introducir y pulsamos Eliminar.

En nuestro ejemplo, al final del proceso debe aparecer la pantalla de la siguiente forma:



Ahora si pulsamos Aceptar, volveremos a la pantalla inmediatamente anterior. Pulsando de nuevo Aceptar nos pedirá el nombre que le vamos a dar al grafo, por ejemplo grafo1. Una

vez dado el nombre y pulsando de nuevo *Aceptar* nos aparecerá en pantalla un resumen de las características del grafo creado. Recordemos no obstante que aunque el grafo está disponible en memoria, no ha sido guardado en un fichero.

Si el grafo hubiera sido dirigido, la forma de proceder sería similar, no obstante tendríamos que indicárselo en la pantalla correspondiente y serían arcos lo que tendríamos que introducir en el grafo. La manera de hacerlo es igual que antes. La diferencia es que aquí sí que importa el orden de los vértices a la hora de formar el arco. Así, el vértice de origen será el vértice inicial del arco y el vértice de destino el vértice final del arco.

1.2.2 Borrando un grafo ya existente

Cuando queramos borrar un grafo de memoria, sólo hay que situarse encima de la opción *Borrar* del menú *Grafo*. Aparecerá inmediatamente a la derecha un nuevo submenú con los nombres de todos los grafos que tengamos. No hay más que situarse encima del deseado y pulsar. Lo habremos eliminado. También habrá una opción suplementaria, *Todos*, por si queremos borrar todos los grafos de memoria.

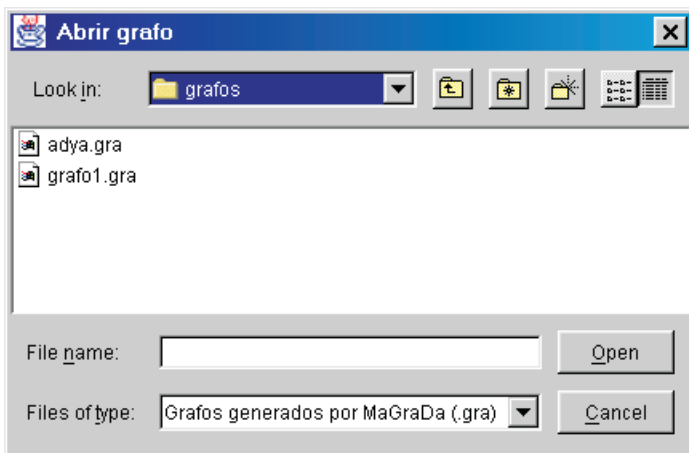
1.2.3 Seleccionando un grafo

Para poder trabajar con un grafo, tiene que ser el actual. Para ello tenemos la opción *Seleccionar*. Es similar a la ante-

rior. La diferencia está en que antes lo borrábamos y ahora lo seleccionamos para poder trabajar con él. Cuando seleccionemos un grafo distinto al actual, se actualizará la información que se presenta en la pantalla principal.

1.2.4 Uso de ficheros para abrir o guardar un grafo

Estas opciones son muy útiles y nos ahorrarán bastante trabajo. Al pulsar la opción **Abrir** nos presenta un selector de ficheros análogo a éste:



Esto nos ofrece la posibilidad de seleccionar los archivos con extensión “gra”, que son los generados por MaGraDa. Sólo hay que pinchar el que queramos y lo cargará en memoria al instante. Para guardar un grafo en un archivo se hará

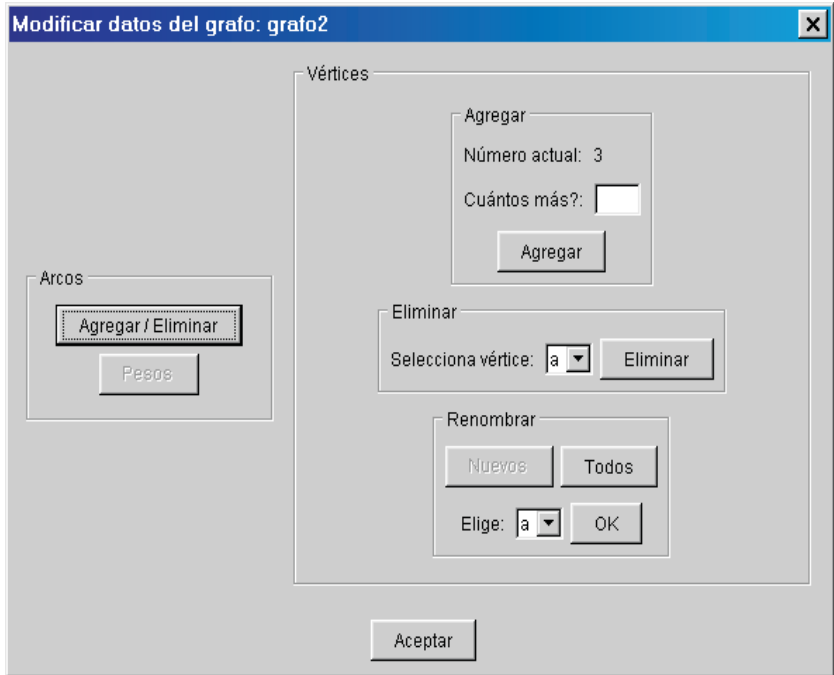
de forma análoga, salvo que tendremos que darle un nombre al archivo donde queremos guardar el grafo. Los nombres del grafo y del archivo donde lo guardamos no tienen, necesariamente, que coincidir. Además, también nos da la posibilidad de guardar todos los grafos que tengamos en memoria, aunque en este caso MaGraDa, por comodidad, toma el nombre del grafo para crear el fichero.

1.2.5 Modificación de un grafo

Una vez creado un grafo siempre existe la posibilidad de modificarlo. Previamente se selecciona el grafo que deseamos modificar si éste no es el actual, y se pulsa la opción *Modificar* del menú *Grafo*. Así por ejemplo, supongamos que se desea modificar el siguiente grafo dirigido que previamente se ha creado y denominado *grafo2*:

$$G = \{V, A\}, \quad V = \{a, b, c\}, \quad A = \{(a, b), (a, c), (b, c), (c, b)\}.$$

Observamos que se obtiene la siguiente pantalla:



Vemos que está dividida en dos partes:

- **Arcos** (o aristas en los no dirigidos): Nos dejará agregar nuevos arcos (aristas) o eliminarlos.
- **Vértices**: Nos dejará agregar nuevos vértices, eliminar aquellos que seleccionemos, así como renombrarlos o simplemente darles nombre si no tenían.

Así por ejemplo, supongamos que se desea modificar el grafo anterior (grafo2) añadiendo un nuevo vértice d y el arco (c, d) , y eliminando el vértice b . En este caso, el grafo resultante sería: $G = \{V, A\}$, $V = \{a, c, d\}$, $A = \{(a, c), (c, d)\}$. Notemos que ahora es éste el grafo que posee MaGraDa en memoria como grafo2.

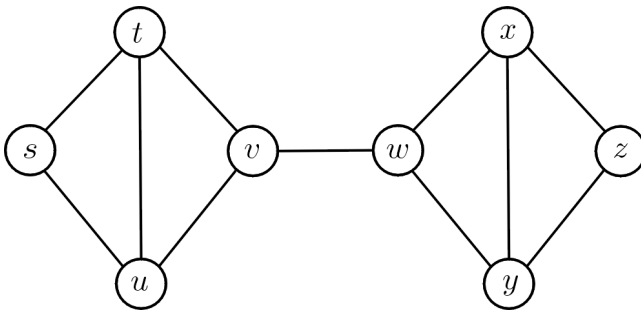
1.2.6 Salir de MaGraDa

Cuando queramos cerrar la sesión, ejecutamos la opción **Salir** del menú **Grafo**. Nos dirá si queremos guardar los grafos que tenemos antes de salir. Después, cerrará la aplicación hasta una nueva ocasión.

1.2.7 Práctica 1

En esta práctica nos vamos a familiarizar con la creación y manipulación de grafos desde el modo de texto, al mismo tiempo que empezamos a retener la terminología básica de grafos.

Problema 1.1. *Consideremos el siguiente grafo:*



(i) *Escribe dicho grafo en forma matemática.*

(ii) *Indica cuáles de las siguientes frases son verdaderas o falsas:*

- *El grafo es dirigido.*

- *El grafo es no dirigido.*
- *El grafo es mixto.*
- *Los vértices v y w son adyacentes, pero los vértices t y u no lo son.*

(iii) *Rellena los huecos: Este grafo tiene _____ vértices. Tiene _____ arcos y _____ aristas. El vértice v es adyacente con los vértices _____ y el vértice w con los vértices _____.*

(iv) *Crea el grafo con MaGraDa desde el modo de texto y guárdalo en un fichero con el nombre pro1.*

(v) *Añade al grafo pro1, desde el modo de texto, una arista que sea incidente con los vértices t y x y una arista en la que sus extremos coincidan y sean el vértice z . Además elimina los vértices v y w . Guarda el grafo modificado en un fichero con el nombre promo1 y completa las siguientes frases:*

Este grafo es un grafo _____ (dirigido, no dirigido o mixto) que se puede escribir de forma matemática como _____

_____.

Introducción a MaGraDa

Se observa, además, que contiene una arista denominada _____, ya que sus extremos coinciden. Por lo que este grafo no es _____, diremos entonces que es un _____.

1.3 El modo gráfico

El modo gráfico es la segunda forma de trabajo que ofrece MaGraDa. Las posibilidades que nos da son las mismas que en el modo de texto, aunque la forma de ver los resultados no siempre es la misma. Vamos a comentar un poco dichas posibilidades. Se tienen tres menús, además de un cuarto para poder retornar al modo de texto:

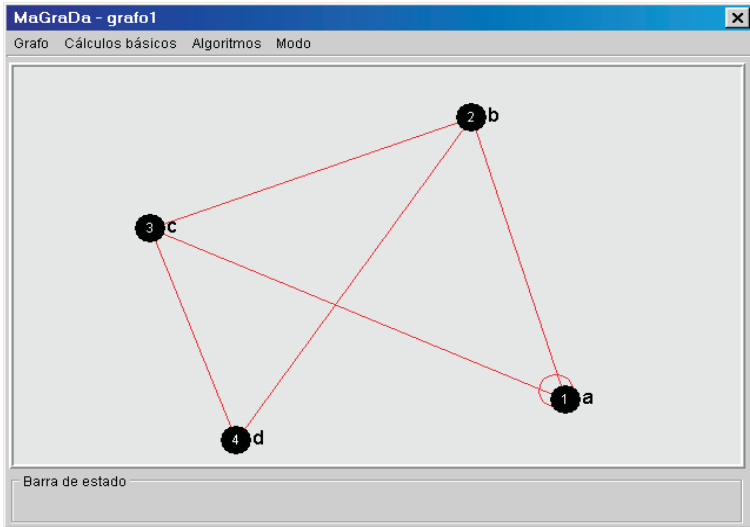
- **Grafo:** Este menú se utiliza para manipular grafos. Comentaremos posteriormente aquellas opciones cuya forma de trabajo difiera del modo de texto.
- **Cálculos básicos:** Es análogo al menú Cálculos básicos del modo de texto. Pero aquí se tratan las soluciones de un modo más gráfico.
- **Algoritmos:** Aparecen los mismos algoritmos que en el modo de texto, aunque como en el caso anterior la resolución de éstos se realiza de forma gráfica siempre que es posible.

En primer lugar, y para ponernos en situación vamos a ofrecer una pantalla que nos acerque lo antes posible a este modo de trabajo. Dicha pantalla muestra el grafo1 definido en la sección 1.2.1, es decir, el grafo:

$$G = \{V, A\}, \quad V = \{a, b, c, d\},$$

Introducción a MaGraDa

$$A = \{\{a, a\}, \{a, b\}, \{a, c\}, \{b, c\}, \{b, d\}, \{c, d\}\}.$$



Podemos diferenciar en la pantalla tres zonas:

- **Barra de menús:** Nos permite acceder a los servicios que ofrece la aplicación.
- **Lienzo:** Llamaremos lienzo a la zona de dibujo donde aparecerán los grafos.
- **Barra de estado:** Es la parte inferior. Mediante ella, MaGraDa intentará en todo momento informarnos y ayudarnos en el trabajo con los grafos.

1.3.1 El lienzo

Sobre el lienzo mostraremos los grafos de la siguiente forma:

- **Vértices:** Los representaremos mediante círculos negros. Dentro de cada uno colocaremos su número en blanco. Cuando apliquemos ciertos algoritmos, pueden cambiar de color para resaltarlos. También, cuando el grafo sea ponderado, podrán alojar en su interior un segundo número correspondiente al peso. Ya lo explicaremos más adelante.
- **Aristas:** Las representaremos mediante líneas rojas. Unirán dos vértices distintos, o bien el mismo formando un bucle. Entre dos vértices podrán existir varias aristas, apareciendo paralelas entre sí, conforme se van añadiendo.
- **Arcos:** Los representaremos mediante flechas azules. También podrá haber más de un arco entre el mismo par de vértices (en cualquier sentido).
- **Pesos:** Cuando el grafo sea ponderado, mostraremos los pesos de las aristas (o arcos) en blanco sobre cuadrados de color morado. Para evitar posibles colisiones, los cuadrados no se dibujarán en los centros de las

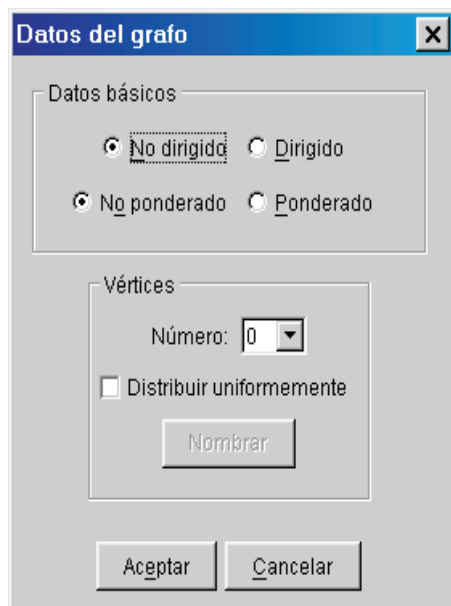
aristas (o arcos). A esta cuestión volveremos cuando tratemos dichos grafos.

- **Rectángulo FIN.** Cuando activemos cualquier algoritmo de forma que exista interacción con el grafo, aparecerá en la esquina superior derecha del lienzo un rectángulo coloreado con la palabra fin dentro (cada método o algoritmo tiene su propio color). Cuando queramos abandonar el método o algoritmo actual, no tendremos más que pinchar sobre él para que el lienzo vuelva a la normalidad.

Existen algoritmos que resaltan de color blanco determinadas aristas (o arcos) del grafo con diferentes intenciones. Ya lo explicaremos más adelante. Otro aspecto importante es que muchos métodos o algoritmos son bloqueados al ser activados otros y no se podrá acceder a ellos hasta pulsar fin.

1.3.2 Introducción de un grafo

Es desde el menú *Grafo*, como en el modo de texto, desde donde se crearán o modificarán grafos en el modo gráfico. Para crear un grafo nos situaremos en la opción *Nuevo*. Veamos una pantalla con dicha opción activada:



Como se puede observar, la pantalla que presenta la opción Nuevo del modo gráfico es análoga a su correspondiente en el modo de texto. Cabe destacar, no obstante, que en este caso, se puede decidir dónde poner los vértices o si se prefiere, disponerlos de forma concéntrica en forma de círculo en el centro del lienzo (activando la opción Distribuir uniformemente). Veamos con un ejemplo cómo proceder: Supongamos que se desea introducir el siguiente grafo dirigido

$$G = \{V, A\}, \quad V = \{a, b, c, d, e\},$$

$$A = \{(a, b), (b, a), (c, b), (c, a), (c, d), (c, e), (d, a), (c, c)\}.$$

Para ello realizamos los siguientes pasos:

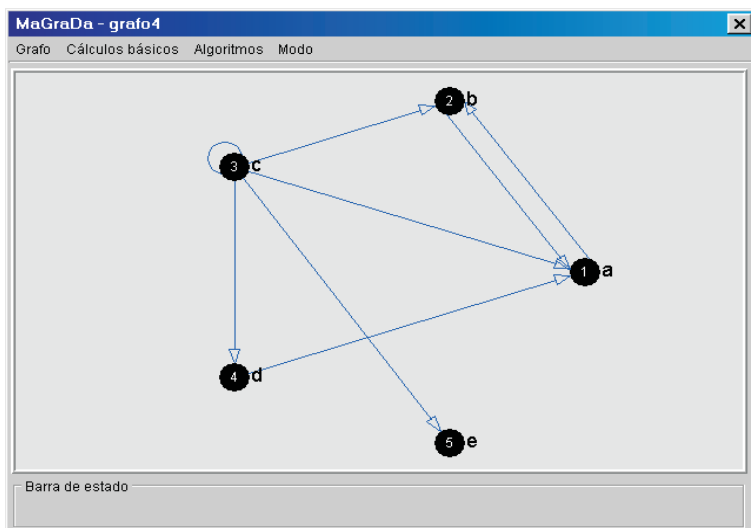
Paso 1. Se rellena la ventana anterior con los datos del grafo. Es decir, especificaremos el tipo de grafo que queremos crear y el número de vértices que tendrá dicho grafo. En caso de que queramos que sea MaGraDa quien disponga los vértices en el lienzo activaremos la opción `Distribuir uniformemente`. Pulsaremos `Aceptar` e introduciremos el nombre del grafo, por ejemplo `grafo4`.

Paso 2. Si se ha activado la opción `Distribuir uniformemente`, aparecerán los vértices en el lienzo. En caso contrario nos situaremos sobre el lugar donde queramos que esté cada vértice y pulsaremos el botón izquierdo del ratón para que éste aparezca. Hacemos esto hasta introducir todos los vértices deseados. Como se puede observar, dentro de cada vértice aparece un número, que es el nombre que por defecto le da MaGraDa a cada vértice. Si hemos dado nombres a los vértices y queremos verlos hay que activar la opción `Vértices con nombre`. Si deseamos que vuelvan a desaparecer sus nombres activaremos la opción `Vértices sin nombre`.

Paso 3. Una vez introducidos todos los vértices, MaGraDa pasa directamente al modo de introducción de aristas o

arcos, según sea el caso. Para especificar las aristas o arcos entre dos vértices nos situamos sobre un vértice, pulsamos el botón izquierdo del ratón y luego sobre el otro vértice, volviendo a pulsar el botón izquierdo del ratón. Repetiremos el proceso hasta especificar todas las aristas o todos los arcos deseados. Para finalizar el proceso pulsaremos fin (aparece en el ángulo superior derecho).

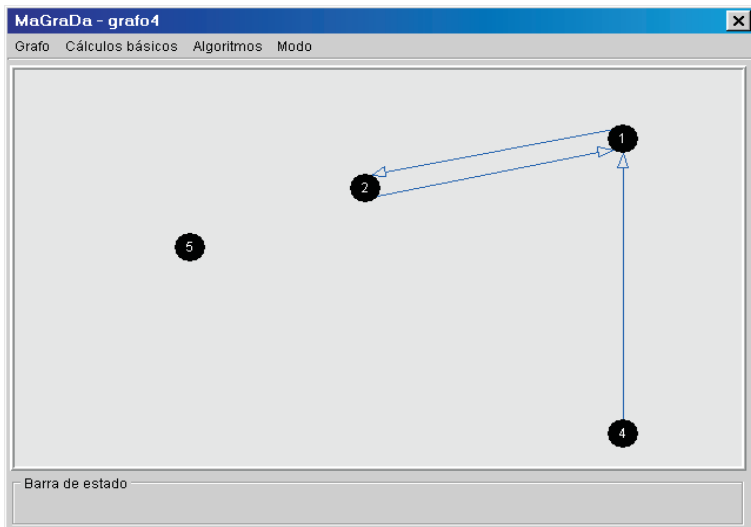
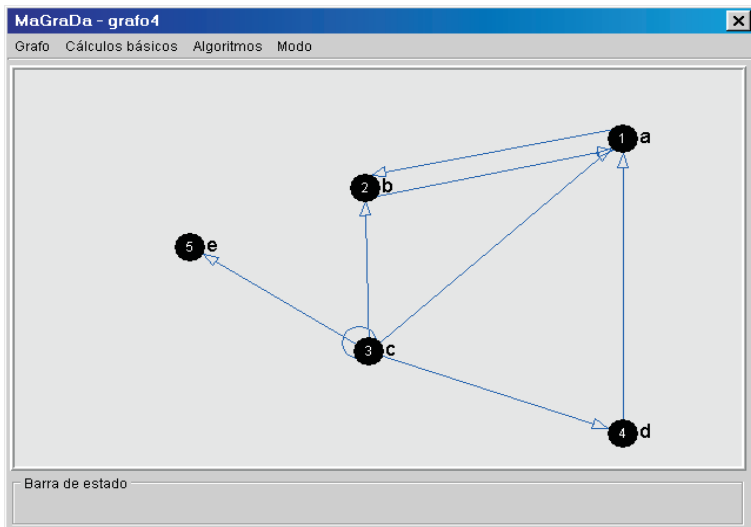
Una vez realizados todos los pasos, y suponiendo que se ha elegido la opción de que sea MaGraDa quien distribuya los vértices del grafo del ejemplo anterior, el grafo quedaría dibujado de la siguiente forma:



Notemos que se ha activado la opción que permite ver los nombres de los vértices.

Desde el modo gráfico y desde la opción `Modificar` del menú `Grafo` también se puede modificar el grafo. La forma de proceder es análoga a la de creación de un grafo, es decir, para añadir o eliminar vértices nos situamos en la opción correspondiente y situamos el ratón donde deseemos añadir el vértice o encima del vértice que se desea eliminar. Pulsamos entonces el botón izquierdo del ratón y se realizará la operación deseada. La forma de eliminar arcos o aristas es similar a la de agregarlos, que se ha visto cuando se explicó la forma de introducir un grafo desde el modo gráfico. Desde esta misma opción también se puede cambiar el nombre de los vértices y recolocarlos. En ambos casos nos debemos situar, de nuevo, sobre el vértice con el que deseemos trabajar y pulsar el botón izquierdo del ratón. Si estamos en la opción `Renombrar vértices` nos pedirá que le demos el nuevo nombre. Si lo que queremos es cambiar el vértice de sitio, tendremos que mover el cursor hasta el sitio deseado y pulsar, de nuevo, el botón izquierdo del ratón.

Como ejemplo de lo explicado, mostramos una primera imagen en la que hemos cambiado los vértices de sitio en el grafo⁴, mientras que en la segunda hemos eliminado el vértice *c* y hemos indicado que los vértices aparezcan sin su nombre.



Introducción a MaGraDa

El resto de opciones que nos ofrece este primer menú (Grafo) relacionadas con la manipulación de un grafo son análogas a las explicadas en la sección correspondiente al modo de texto. No obstante, para seleccionar o borrar un grafo lo hace ahora mediante nuevos diálogos y no con sub-menús como antes.

1.3.3 Práctica 2

Una vez entendida la forma de crear un grafo desde el modo gráfico, vamos a familiarizarnos con las distintas opciones del menú `Grafo` de este modo. Para ello, se ha de realizar la siguiente práctica. A la finalización de la misma, se estará en disposición de abordar problemas de grafos más complicados.

Problema 1.2. *Visualiza desde el modo gráfico los grafos guardados en los ficheros `pro1` y `promo1`, respectivamente, que se crearon desde el modo de texto en la práctica 1. Indica que sea MaGraDa quien sitúe los vértices y que veamos sus nombres. Presenta el resultado obtenido en tu práctica, dibujándolos a mano.*

- `pro1`

- promol

Problema 1.3. Consideremos el grafo no dirigido $G = (V, A)$,

$$V = \{1, 2, 3, 4\}, \quad A = \{\{1, 1\}, \{1, 2\}, \{2, 1\}, \{3, 3\}, \{3, 4\}, \{2, 4\}\}.$$

Realiza las siguientes cuestiones:

- Dibuja desde el modo gráfico este grafo, situando tu los vértices. Presenta el resultado obtenido en tu práctica, dibujándolo a mano y completa la siguiente frase:

Este grafo no es simple porque _____

_____.

- (ii) *Elimina, desde el modo gráfico, los dos bucles del grafo. Presenta el resultado obtenido en tu práctica, dibujándolo a mano y completa la siguiente frase:*

Este grafo es _____ (simple o multi-grafo) porque _____

_____.

- (iii) *Elimina, desde el modo gráfico, el vértice 1 y la arista incidente con los vértices 3 y 4. Presenta el resultado obtenido en tu práctica, dibujándolo a mano y completa la siguiente frase:*

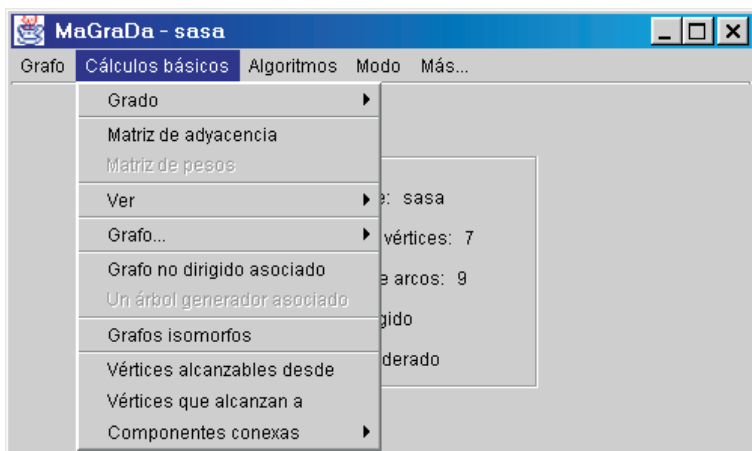
Introducción a MaGraDa

Este grafo es _____ (simple o multi-grafo) porque _____

2. Fundamentos

2.1 Introducción

En este capítulo vamos a ver gran parte de las posibilidades que ofrece el menú Cálculos básicos de MaGraDa, tanto desde el modo de texto como desde el modo gráfico. Este menú es el segundo de los tres menús más importantes del programa. Como se puede apreciar en la siguiente figura, nos ofrece múltiples opciones.



La característica que comparten todas ellas es que las operaciones que tienen que hacer con los grafos son, en principio, más sencillas que si les aplicamos los algoritmos del tercer menú. En la sección 2.2 comentaremos algunas de estas opciones con un poco de profundidad, y la sección 2.3 estará dedicada a la matriz de adyacencia de un grafo.

2.2 Cálculos básicos

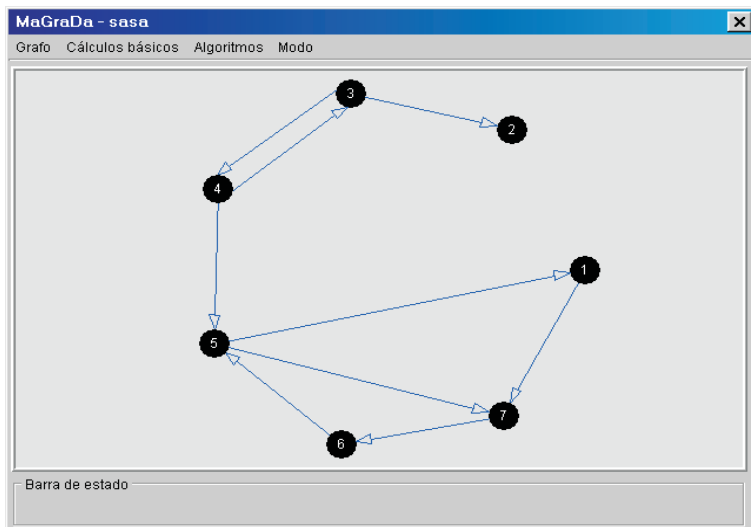
Las opciones del menú `Cálculos básicos` que se van a estudiar en esta sección son:

- Grado.
- Ver (aristas o arcos).
- Grafo (simple, cíclico, completo y conexo).
- Grafo no dirigido asociado.
- Grafos isomorfos.

En el resto de la sección vamos a comentar un poco cada una de estas opciones.

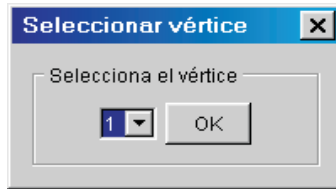
2.2.1 Grado

Para ilustrar esta opción dibujaremos con MaGraDa el siguiente grafo dirigido:



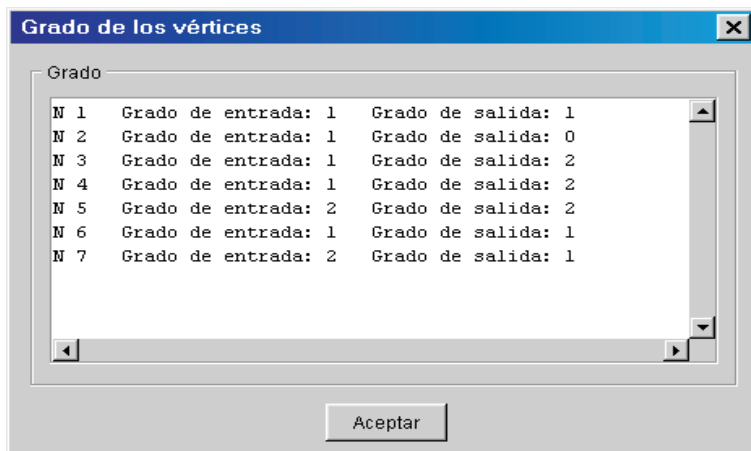
Recordemos que si el grafo es no dirigido hablaremos simplemente de grado de un vértice. Sin embargo, si como en nuestro caso, el grafo es dirigido habrá que diferenciar entre grado de entrada y grado de salida de un vértice. Esta opción está dividida en dos subopciones:

- **Grado de un sólo vértice:** Nos dirá el grado del vértice que queramos. Para ello, desde el modo de texto, nos presentará una pantalla de selección de vértices como ésta:



Notemos que si los vértices tuvieran nombre dado, en dicha pantalla aparecería el nombre. Como no es así, aparece sólo el número. En cuanto lo hayamos seleccionado, nos informará del grado del vértice en una pantalla de información.

- **Grado de todos los vértices:** Utilizamos esta opción, si queremos ver, directamente, desde el modo de texto, el grado de todos los vértices del grafo. Para nuestro ejemplo nos aparecerá algo así:



Aquí vemos cómo nos muestra el grado de todos los vértices del grafo de una manera sencilla y eficaz. Fijémonos, por ejemplo, en el vértice 3. Su grado de entrada es uno. Por tanto, sólo hay un arco en el grafo cuyo vértice final sea 3. Sin embargo el grado de salida es dos, por lo tanto hay dos arcos en el grafo cuyo vértice inicial es el 3.

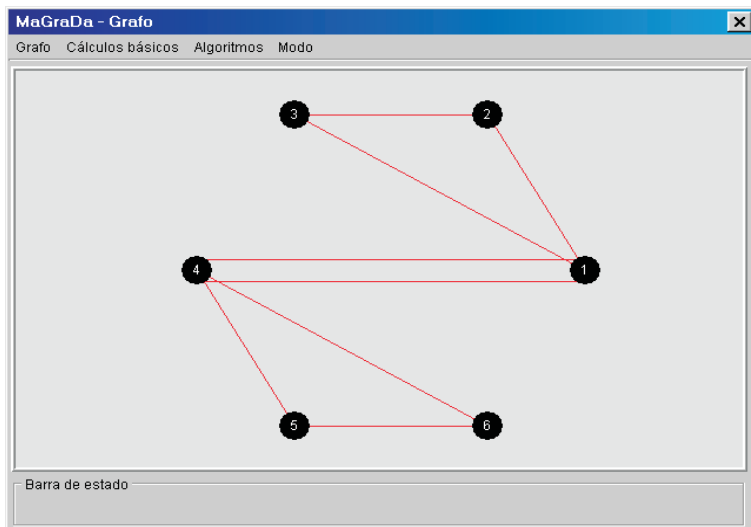
Desde el modo gráfico podemos realizar estos mismos cálculos. La diferencia estriba en que cuando se quiere calcular el grado de un vértice debemos situarnos encima de él y pulsar el botón izquierdo del ratón. Para salir de la opción se debe pulsar el recuadro fin situado en el ángulo superior derecho.

2.2.2 Ver

Cuando seleccionamos un grafo desde el modo gráfico, dicho grafo aparece en el lienzo. Sin embargo, desde el modo de texto sólo nos aparece información general sobre él, pero no cuáles son las aristas o arcos del grafo. Para ver dichas aristas o arcos simplemente ejecutaremos, desde el modo de texto, la opción `Ver`. Las pantallas que nos mostrará serán similares a las que el programa nos ofreció a la hora de introducir estos datos desde el modo de texto, aunque ahora son pantallas informativas. Si el grafo tiene nombrados sus vértices, MaGraDa nos mostrará la información usando estos nombres. En otro caso, usará los números.

2.2.3 Grafo (simple, cíclico, completo y conexo)

Son cuatro características importantes de los grafos. Cuando las ejecutamos, tanto desde el modo de texto como desde el modo gráfico, el programa realizará cálculos internos sobre el grafo actual para ofrecernos la información de forma inmediata. Si la respuesta es afirmativa, así nos lo hará saber en cada caso. Si no es así, también lo hará y además nos dirá la razón que le ha llevado a dar esa respuesta. Así por ejemplo, supongamos que se ha introducido con MaGraDa el siguiente grafo no dirigido:



Entonces MaGraDa nos dirá que el grafo no es simple, ya que hay más de una arista que une el mismo par de vértices, por ejemplo hay dos aristas uniendo los vértices 1 y 4.

Un grafo cíclico es aquel que contiene algún ciclo (también denominado circuito en el caso dirigido). Por tanto, como indica MaGraDa, el grafo que nos ocupa lo es, ya que por ejemplo, el vértice 1 aparece al menos en un ciclo, por ejemplo en el ciclo $\{1, 4\}$, $\{4, 1\}$. Animamos a observar el grafo y ver si dicho vértice aparece en algún otro ciclo.

Por otra parte los grafos completos son aquellos en los que hay al menos una arista o arco (según sea no dirigido o dirigido) uniendo cada par de vértices. Claramente el grafo

que nos ocupa no es completo ya que, como indica MaGraDa, por ejemplo, entre los vértices 1 y 5 no hay ninguna arista.

Un grafo es conexo, si todo par de vértices está conectado. En el grafo que nos ocupa, al ser no dirigido y relativamente pequeño, es fácil observar que esto ocurre, ya que no hay ningún vértice o vértices desconectados del resto, de hecho podríamos ir recorriendo, a lo largo de las aristas (con posibilidad de repetirlas), todos los vértices con un lápiz sin tener que levantar el lápiz del papel. No obstante, cuando el grafo es dirigido o de gran tamaño, esto no es tan fácil de ver. Para ello se dispone de algoritmos que se estudiarán en el capítulo 3. MaGraDa hace uso interno de estos algoritmos para dar solución a esta cuestión.

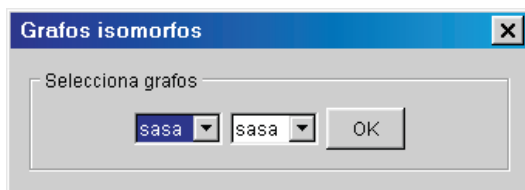
2.2.4 Grafo no dirigido asociado

Ésta es una opción sólo válida para grafos dirigidos. Se trata de que el programa genere un nuevo grafo, no dirigido, a partir del grafo dirigido que tengamos seleccionado. Conservará todas sus características con la diferencia de que los arcos los transformará en aristas. Cuando ejecutemos la opción, el programa nos preguntará qué nombre queremos darle al nuevo grafo, si dicho nombre es válido lo guardará en memoria y lo seleccionará como actual y a partir de ahí, si no hacemos otra cosa, trabajaremos con este grafo.

2.2.5 Grafos isomorfos

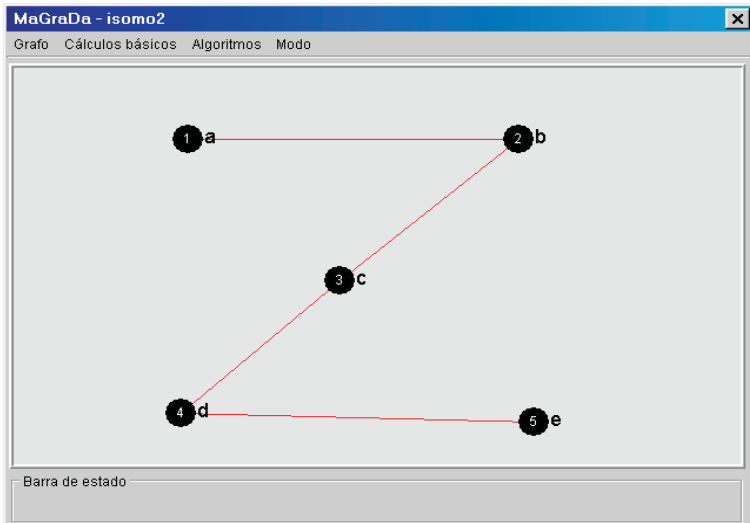
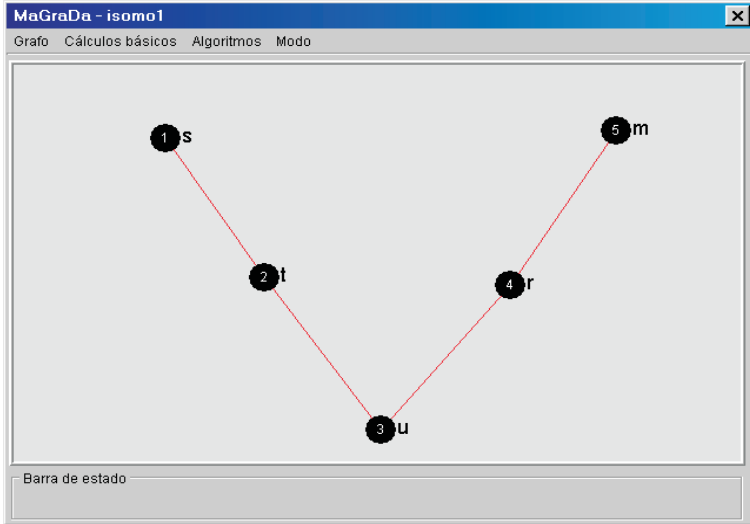
Supongamos que $G_1 = (V_1, A_1)$ y $G_2 = (V_2, A_2)$, son grafos no dirigidos (dirigidos). Se dice que G_1 y G_2 , son grafos isomorfos si existe una aplicación biyectiva $f : V_1 \rightarrow V_2$, tal que, $\{u, v\}$ es una arista de G_1 , si y sólo si, $\{f(u), f(v)\}$ es una arista de G_2 ((u, v) es un arco de G_1 , si y sólo si, $(f(u), f(v))$ es un arco de G_2).

Con MaGraDa podemos estudiar si dos grafos no ponderados son isomorfos desde la opción Grafos isomorfos del menú Cálculos básicos. Ésta es la única opción del programa que permite trabajar con más de un grafo al mismo tiempo. Puede ser que el grafo que MaGraDa tiene seleccionado no sea usado. Para ello, tanto desde el modo de texto como desde el modo gráfico, lo primero que hace MaGraDa es pedirnos dos grafos no ponderados de los que dispone en memoria. Así, nos ofrecerá una pantalla como ésta:



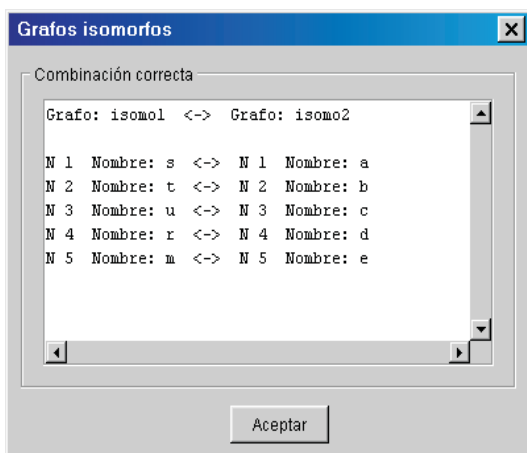
Seleccionaremos dos grafos, por ejemplo los siguientes grafos, isomo1 e isomo2 dibujados previamente con MaGraDa, que representan la letra V y Z , respectivamente.

Fundamentos



Si los grafos no son isomorfos por cualquier causa, nos lo dirá mediante un mensaje. Si por el contrario lo son, además de indicarlo, nos dará la aplicación biyectiva que ha dado lugar a que todas las condiciones para que sean isomorfos se den. Recordemos que se tienen que cumplir condiciones indispensables como que ambos grafos sean los dos dirigidos o los dos no dirigidos, que tengan el mismo número de vértices y que la sucesión de grados de los vértices sea la misma en ambos grafos.

En particular, ésta será la respuesta que nos dé MaGraDa, para el ejemplo considerado.



Por tanto, si nos fijamos en los nombres dados a los vértices de ambos grafos, obtenemos que una aplicación biyectiva

Fundamentos

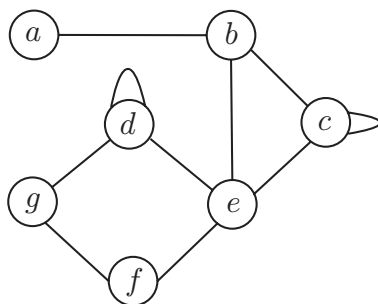
$f : V_1 \longrightarrow V_2$, con $V_1 = \{s, t, u, r, m\}$, el conjunto de vértices de isomo1 y $V_2 = \{a, b, c, d, e\}$, el conjunto de vértices de isomo2, que cumple las condiciones para que los grafos sean isomorfos viene definida por $f(s) = a$, $f(t) = b$, $f(u) = c$, $f(r) = d$, $f(m) = e$, ya que cumple que:

- $\{s, t\}$ es arista de isomo1 $\iff \{f(s), f(t)\} = \{a, b\}$ es arista de isomo2.
- $\{t, u\}$ es arista de isomo1 $\iff \{f(t), f(u)\} = \{b, c\}$ es arista de isomo2.
- $\{u, r\}$ es arista de isomo1 $\iff \{f(u), f(r)\} = \{c, d\}$ es arista de isomo2.
- $\{r, m\}$ es arista de isomo1 $\iff \{f(r), f(m)\} = \{d, e\}$ es arista de isomo2.

2.2.6 Práctica 3

Con la ayuda de MaGraDa se han de realizar los siguientes ejercicios relacionados con los fundamentos de grafos.

Problema 2.1. Consideremos el grafo no dirigido que aparece en la siguiente figura:



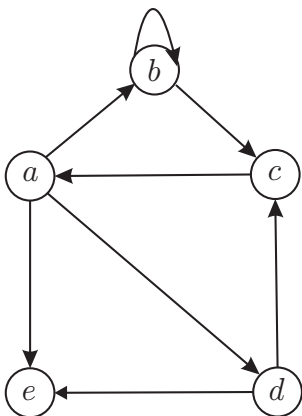
(i) Calcula a mano el grado de cada vértice. Comprueba el resultado con MaGraDa, desde el modo gráfico.

(ii) Relaciona el grado de los vértices con el número de aristas. Exprésalo también de forma matemática para cualquier grafo.

(Continuación)

(iii) Con la ayuda de MaGraDa, di si el grafo es completo o no y explica el porqué.

Problema 2.2. Consideremos el siguiente grafo dirigido:



- (i) Calcula a mano el grado de entrada y de salida de cada vértice. Comprueba el resultado con MaGraDa, desde el modo gráfico.

Fundamentos

(ii) *Relaciona el grado de los vértices (tanto de entrada como de salida) con el número de arcos. Exprésalo también de forma matemática para cualquier grafo.*

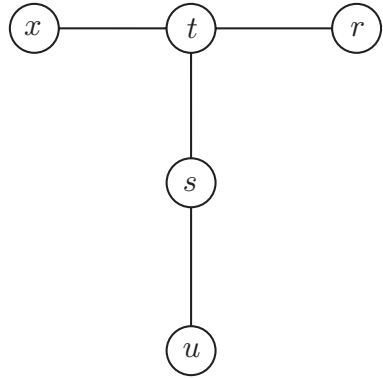
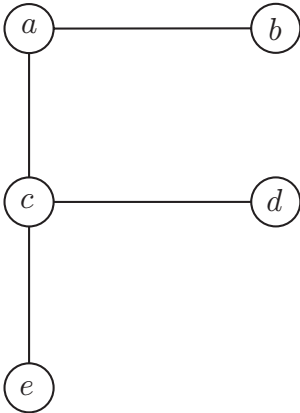
(iii) *Con la ayuda de MaGraDa, di si el grafo es cíclico o no y explica el porqué.*

(iv) *Con la ayuda de MaGraDa, di si el grafo es un multigrafo o no y explica el porqué.*

(v) *Con la ayuda de MaGraDa y tus conocimientos actuales, di si el grafo es conexo o no y explica el porqué.*

(vi) *Con la ayuda de MaGraDa, di si el grafo es débilmente conexo o no y explica el porqué.*

Problema 2.3. *La siguiente figura nos da dos grafos dibujados como las letras F y T.*



Comprueba y razona que los grafos F y T son grafos isomorfos.

(Continuación)

Fundamentos

Problema 2.4. *Consideremos que se tiene un grafo, que tiene 8 aristas y 5 vértices. Se sabe además que tiene 4 vértices de grado 3.*

(i) Explica razonadamente de qué grado es el otro vértice.

(ii) *Dibuja con MaGraDa un grafo no dirigido y sin bucles con esas características. Presenta dicho grafo en tu práctica.*

(iii) Desde MaGraDa, elimina de tu grafo el vértice que tiene grado 4. Razona, sin que el dibujo del grafo te sirva para tu explicación, cuántas aristas te quedarán en el grafo y comprueba que sigue cumpliéndose la relación matemática entre vértices y aristas. Comprueba con MaGraDa que tus razonamientos han sido válidos.

Problema 2.5. Consideremos que se tiene un grafo no dirigido que tiene 5 aristas y sólo tiene vértices de grado 3 y de grado 2.

(i) Explica cuántos vértices tiene de cada tipo.

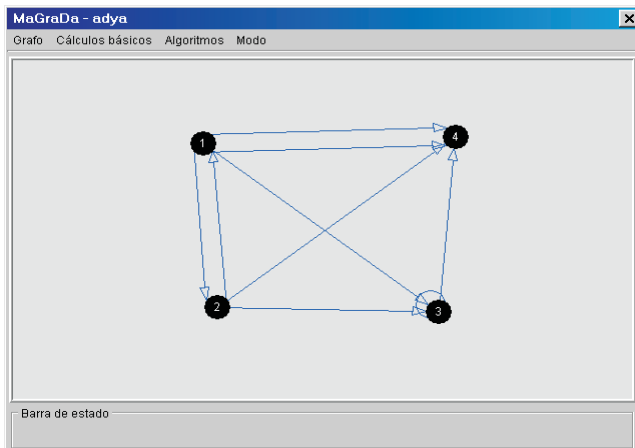
(Continuación)

(ii) Dibuja con MaGraDa un grafo no dirigido y simple que cumpla estas características y muestra con MaGraDa que tu razonamiento ha sido válido.

2.3 Matriz de adyacencia

Además de las formas de introducción de un grafo vistas en el capítulo anterior, MaGraDa nos permite introducirlo mediante su matriz de adyacencia. Esto se hará desde la opción Nuevo del menú Grafo del modo de texto, vista en el capítulo anterior, pulsando Matriz de adyacencia, una vez indicado cómo es el grafo y el número de vértices que va a tener.

Ya haya sido introducido el grafo de esta forma o mediante la introducción de sus arcos o aristas, MaGraDa nos da la posibilidad de acceder a dicha matriz de una forma muy fácil, desde la opción Matriz de adyacencia del menú Cálculos básicos. Vamos a ilustrar esto con un ejemplo. Supongamos que se ha introducido con MaGraDa el siguiente grafo dirigido:



Tanto desde el modo de texto como desde el modo gráfico, la matriz de adyacencia obtenida para dicho grafo viene dada en la siguiente pantalla:

	1	2	3	4
CERO	1	1	2	
1	CERO	1	1	
CERO	CERO	1	1	
CERO	CERO	CERO	CERO	

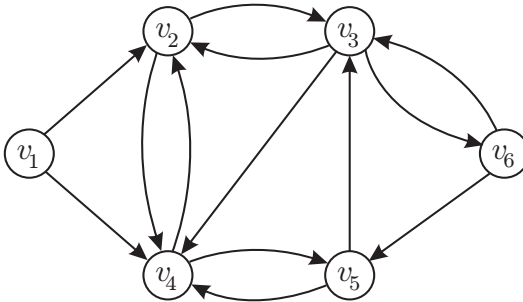
Aceptar

Una de las ventajas que MaGraDa ofrece en este método es que resalta de una manera significativa aquellas celdas de la matriz de adyacencia donde entre el par de vértices correspondientes no hay aristas (o arcos). Lo hace mediante la palabra cero. De esta forma, es mucho más fácil ver de inmediato las casillas que nos interesan. No obstante, si se introduce el grafo mediante su matriz de adyacencia, deberá utilizarse para ello números no negativos.

2.3.1 Práctica 4

En esta práctica vamos a familiarizarnos con el concepto de matriz de adyacencia y sus propiedades.

Problema 2.6. Consideremos el grafo dirigido de la siguiente figura:



(i) Introduce el grafo en MaGraDa mediante su matriz de adyacencia. Visualízala y escríbela en tu práctica.

(ii) *Usando dicha matriz explica cuánto vale el grado de salida y de entrada del vértice v_3 .*

(iii) *Usando de nuevo dicha matriz, explica cuántas cadenas de longitud 2 hay del vértice v_4 al vértice v_3 .*

Fundamentos

(iv) *Obtén, con la ayuda de la matriz de adyacencia, $\Gamma(t)$ y $\Gamma^{-1}(t)$, para cada vértice t del grafo, es decir, para $t = v_1, v_2, v_3, v_4, v_5, v_6$. Explica en qué te basas para calcular $\Gamma(t)$ y $\Gamma^{-1}(t)$.*

3. Estudio de la accesibilidad y conectividad

3.1 Introducción

Como se indicó en el capítulo anterior, cuando un grafo es pequeño y sobretodo si no es dirigido es fácil ver si dicho grafo es conexo. Sin embargo, conforme aumenta el tamaño del grafo, esto se hace más difícil. En este capítulo vamos a ver qué hace MaGraDa para ver si un grafo es conexo o no. En la sección 3.3 estudiaremos dos algoritmos que permiten calcular las distintas componentes conexas de un grafo. Recordemos que si se obtiene una única componente conexa el grafo será conexo y en caso de obtener más de una, el grafo será no conexo. Previamente, en la sección 3.2 estudiaremos algunos conceptos relacionados, como la accesibilidad, el acceso y las aristas de corte.

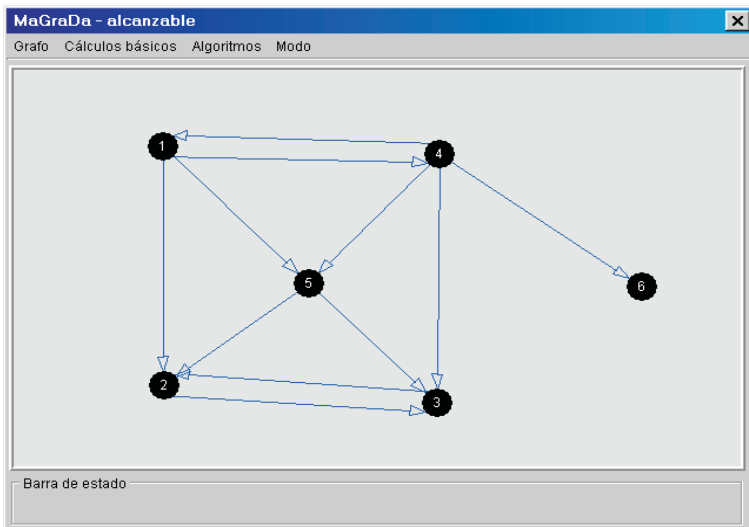
3.2 Matriz de accesibilidad y matriz de acceso

3.2.1 Vértices alcanzables desde otro

A partir de la matriz de adyacencia se puede obtener el conjunto de vértices que son alcanzables desde uno dado,

Estudio de la accesibilidad y conectividad

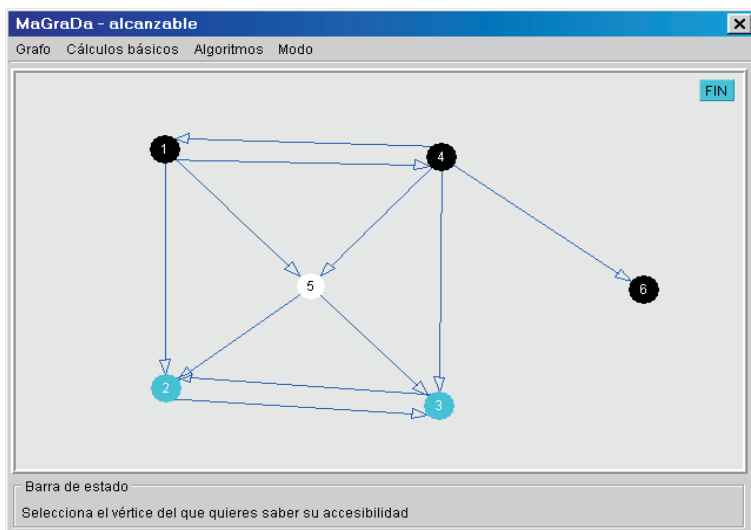
ya sea porque estén unidos directamente o porque exista un camino entre ellos. Con la ayuda de MaGraDa podemos saber cuál es dicho conjunto. Para ello debemos situarnos en la opción **Vértices alcanzables desde**, del menú **Cálculos básicos**. Por ejemplo, supongamos que se ha introducido con MaGraDa el siguiente grafo dirigido:



Desde el modo de texto, MaGraDa nos ofrecerá una pantalla que nos da una relación de todos los vértices del grafo con los vértices a los que cada uno de ellos puede alcanzar. Si los vértices tuvieran nombre, éstos aparecerían referenciados con ellos. Si no es así muestra los números y ya está.

Para salir, pulsaremos Aceptar.

Desde el modo gráfico, al pinchar el vértice que se desea estudiar, cambiarán de color todos los vértices que son alcanzables desde dicho vértice. Así por ejemplo, en la siguiente pantalla se observa que los vértices alcanzables desde el vértice 5 son el 2, el 3 y el propio 5, por tanto si denotamos por $R(v)$ al conjunto de vértices alcanzables desde el vértice v , obtenemos que $R(5) = \{2, 3, 5\}$.



Con el fin de familiarizarse con esta opción proponemos que se calcule $R(v)$, para el resto de vértices desde los dos modos, y a partir de los resultados obtenidos, la matriz de

accesibilidad R , que se necesitará para ilustrar algunos métodos.

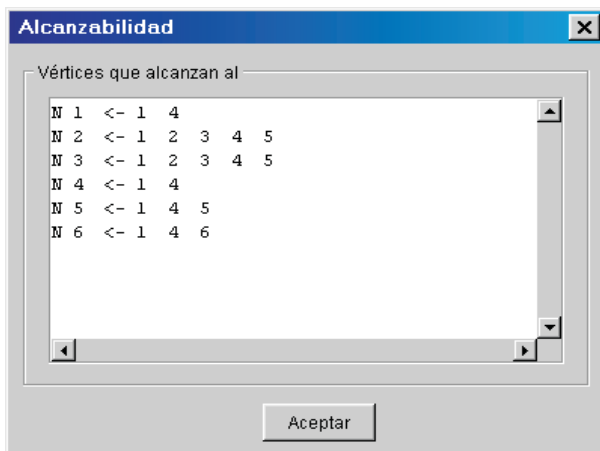
Recordamos que dicha matriz viene definida de la siguiente forma:

Definición 3.1. Sea $G = (V, A)$, un grafo dirigido tal que $V = \{x_i\}_{i=1}^n$. Llamamos matriz de accesibilidad asociada al grafo G a la matriz cuadrada de orden n definida por

$$R = [r_{ij}]_{1 \leq i, j \leq n} / r_{ij} = \begin{cases} 1 & \text{si } x_i \text{ alcanza a } x_j \\ 0 & \text{en otro caso} \end{cases}$$

3.2.2 Vértices que alcanzan a otro

Para ver qué vértices alcanzan a un vértice dado nos debemos situar en la opción `Vértices que alcanzan a`. La forma en que MaGraDa presenta los resultados es similar a la de la opción anterior. La diferencia está en que ahora, para cada vértice nos dirá cuáles son los vértices que lo alcanzan a él, bien directamente o mediante un camino. Así por ejemplo, para el grafo del ejemplo anterior desde el modo de texto obtendríamos:



En este caso si denotamos por $Q(v)$ al conjunto de vértices que alcanzan al vértice v , obtenemos que $Q(1) = \{1, 4\}$, $Q(2) = \{1, 2, 3, 4, 5\}$, $Q(3) = \{1, 2, 3, 4, 5\}$, $Q(4) = \{1, 4\}$, $Q(5) = \{1, 4, 5\}$, $Q(6) = \{1, 4, 6\}$.

Recordemos ahora el concepto de matriz de acceso:

Definición 3.2. Sea $G = (V, A)$ un grafo dirigido tal que $V = \{x_i\}_{i=1}^n$. Llamamos matriz de acceso asociada al grafo G a la matriz cuadrada de orden n definida por

$$Q = [q_{ij}]_{1 \leq i, j \leq n} / q_{ij} = \begin{cases} 1 & \text{si } x_i \text{ es alcanzable desde } x_j \\ 0 & \text{en otro caso} \end{cases}$$

De aquí se deduce por tanto que la matriz de acceso del grafo que nos ocupa es

$$Q = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 \end{bmatrix}.$$

3.2.3 Accesibilidad y algoritmo de Warshall

En la sección anterior se ha tratado la forma de calcular la matriz de accesibilidad de un grafo, a partir de la matriz de adyacencia. A continuación vamos a ver otra forma, más económica, de calcular la matriz de accesibilidad: el algoritmo de Warshall. Construiremos una sucesión de matrices R_0, R_1, \dots, R_n , donde

- El valor de n es el número de vértices del grafo.
- La matriz R_0 es una matriz calculada a partir de la matriz de adyacencia A , del grafo, donde los elementos distintos de 0 se han cambiado por unos.
- La matriz R_n es la matriz de accesibilidad pero sin considerar los caminos de longitud cero. Para obtener la matriz de accesibilidad, hay que añadir unos en la diagonal principal, y así incluir dichos caminos.

Si denotamos a los elementos de esta sucesión de matrices por

$$R_k = [r_{ij}^{(k)}]_{1 \leq i, j \leq n}, \quad k = 0, 1, 2, \dots, n,$$

estos elementos pueden ser calculados de acuerdo con la siguiente condición:

$$r_{ij}^{(k)} = 1 \iff \begin{cases} r_{ij}^{(k-1)} = 1 \\ \text{ó} \\ r_{ik}^{(k-1)} = r_{kj}^{(k-1)} = 1 \end{cases} \\ k = 1, 2, \dots, n.$$

Así, por ejemplo, consideremos un grafo simple G con cuatro vértices, cuya matriz de adyacencia es:

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} = R_0.$$

La sucesión de matrices que genera el algoritmo de Warshall es:

$$R_1 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad R_2 = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix},$$

$$R_3 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad R_4 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

El procedimiento ilustrado en el ejemplo anterior produce el siguiente algoritmo para calcular la matriz R_n de un grafo con n vértices, a partir de su matriz de adyacencia A .

ALGORITMO DE WARSHALL

$R \leftarrow R_0$

Para $k = 1$ hasta n

 Para $i = 1$ hasta n

 Para $j = 1$ hasta n

$R(i, j) \leftarrow R(i, j) \vee (R(i, k) \wedge R(k, j))$

MaGraDa realiza este algoritmo desde la opción `Warshall` del menú `Algoritmos`. Tanto desde el modo de texto como desde el modo gráfico tiene dos posibilidades de uso:

- **Por pasos:** Nos indica los distintos pasos, presentando la sucesión de matrices correspondientes, cada una de ellas, a cada iteración del algoritmo. El último paso consistirá en transformar en 1, los elementos diagonales nulos de R_n , para así, obtener la matriz de accesibilidad R tal y como se ha definido aquí, ya que por convenio todo vértice es alcanzado por él mismo por un camino de longitud 0. Para una mejor lectura, MaGraDa indica los elementos nulos de esta matriz con la palabra `cero`.
- **Resultado final:** Sólo muestra la matriz de accesibilidad, obtenida a partir de dicho algoritmo.

3.2.4 Práctica 5

A continuación, con la ayuda de la siguiente práctica, vamos a profundizar en los conceptos de accesibilidad y acceso, y en la utilización del algoritmo de Warshall.

Problema 3.1. *Consideremos un grafo dirigido con el conjunto de vértices $V = \{1, 2, 3, 4, 5, 6\}$ y cuya matriz de adyacencia es la siguiente:*

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}.$$

- (i) *Calcula a mano y razonadamente, sin usar el algoritmo de Warshall (es decir a partir de la matriz de adyacencia), $R(v)$, para $v = 1, 2, 3, 4, 5, 6$, y completa la frase.*

Estudio de la accesibilidad y conectividad

(Continuación)

El vértice 3 alcanza a los vértices _____ y el vértice 4 a _____.

(ii) *Con la ayuda del apartado anterior explica cómo se puede obtener la matriz de accesibilidad y escríbela en tu práctica.*

Estudio de la accesibilidad y conectividad

(iii) *Calcula razonadamente, con la ayuda del apartado anterior, la matriz de acceso.*

(iv) *Calcula razonadamente, con la ayuda del apartado anterior, $Q(v)$, para $v = 1, 2, 3, 4, 5, 6$, y completa la frase.*

Los únicos vértices que alcanzan al vértice 1 son los vértices _____ . Sin embargo al vértice 3 lo alcanzan _____ .

Problema 3.2. *Consideremos un grafo dirigido con siete vértices numerados del 1 al 7, cuya matriz de adyacencia es la siguiente:*

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

- (i) *Calcula a mano y de forma razonada la matriz de accesibilidad del grafo, usando el algoritmo de Warshall. Asegúrate que el resultado es correcto con MaGraDa.*

Estudio de la accesibilidad y conectividad

(Continuación)

(Continuación)

Estudio de la accesibilidad y conectividad

(ii) A partir de dicha matriz, calcula la matriz de acceso y completa la siguiente frase.

Como en la tercera columna de la matriz de acceso hay unos en las filas _____ esto significa que el vértice _____ alcanza a los vértices _____.

3.3 Cálculo de componentes conexas

En esta sección vamos a recordar dos formas de calcular las componentes conexas de un grafo. Consideremos para ello un grafo dirigido $G = (V, A)$. El primer método corresponde con el siguiente algoritmo:

MÉTODO 1.

Etapa 1. Inicializar $i \leftarrow 1$, $V^{(1)} = V$.

Etapa 2. Tomar $v_i \in V^{(i)}$.

Etapa 3. Calcular $R(v_i) \cap Q(v_i)$.

Hacer $V^{(i+1)} = V^{(i)} \sim R(v_i) \cap Q(v_i)$.

Hacer $i \leftarrow i + 1$.

Etapa 4. Si $V^{(i)} = \emptyset$, entonces STOP.

En otro caso, volver a la etapa 2.

Notemos que si el grafo es no dirigido, tendremos en el algoritmo que $Q = R$.

Para ilustrar su funcionamiento, vamos a considerar el grafo del ejemplo de la sección 3.2.1 y los resultados obtenidos previamente:

Etapa 1. $V^{(1)} = \{1, 2, 3, 4, 5, 6\}$.

Etapa 2. Tomamos, por ejemplo, el vértice $1 \in V^{(1)}$.

Etapa 3. $R(1) = \{1, 2, 3, 4, 5, 6\}$ y $Q(1) = \{1, 4\}$, por tanto, $R(1) \cap Q(1) = \{1, 4\}$. Esta será la primera componente conexa. Como no contiene todos los vértices del grafo, éste no es conexo.

Ahora hacemos $V^{(2)} = V^{(1)} \sim R(1) \cap Q(1) = \{1, 2, 3, 4, 5, 6\} \sim \{1, 4\} = \{2, 3, 5, 6\}$.

Como $V^{(2)} \neq \emptyset$, volvemos a la etapa 2, es decir elegimos de nuevo un vértice, pero ahora de $V^{(2)}$, por ejemplo el 2, y realizamos de nuevo el proceso:

- De esta forma se obtiene $R(2) \cap Q(2) = \{2, 3\} \cap \{1, 2, 3, 4, 5\} = \{2, 3\}$ y por tanto la segunda componente conexa.
- Ahora hacemos $V^{(3)} = V^{(2)} \sim R(2) \cap Q(2) = \{2, 3, 5, 6\} \sim \{2, 3\} = \{5, 6\} \neq \emptyset$.
- Elegimos entonces un vértice de $V^{(3)}$, por ejemplo el 5, y razonando de forma análoga a como hasta ahora se obtiene $R(5) \cap Q(5) = \{2, 3, 5\} \cap \{1, 4, 5\} = \{5\}$ y por tanto la tercera componente conexa.
- Ahora hacemos $V^{(4)} = V^{(3)} \sim R(5) \cap Q(5) = \{6\} \neq \emptyset$. Como queda un único vértice claramente ésta será la última componente conexa. Pero finalizemos el

proceso: $R(6) \cap Q(6) = \{6\} \cap \{1, 4, 6\} = \{6\}$ y por tanto $V^{(5)} = V^4 \sim R(6) \cap Q(6) = \emptyset$.

Con lo que el algoritmo ha finalizado, y resumiendo, podemos decir que el grafo no es conexo y tiene cuatro componentes conexas definidas por los siguientes conjuntos de vértices: $C_1 = \{1, 4\}$, $C_2 = \{2, 3\}$, $C_3 = \{5\}$ y $C_4 = \{6\}$. El método anterior no está implementado en MaGraDa, pero se puede hacer uso de las utilidades de las que dispone para aplicarlo a mano.

El segundo método viene definido de la siguiente manera:

MÉTODO 2.

Para calcular las componentes conexas mediante este método lo que se hace es calcular $R \otimes Q$, donde \otimes representa un producto elemento a elemento. Entonces la componente conexa de un vértice x_i se calcula viendo qué columnas tienen un 1 en la fila i .

Este algoritmo lo resuelve MaGraDa desde la opción Componentes conexas del menú Cálculos básicos. Desde el modo de texto, en esta opción, aparecen dos subopciones:

- **Por pasos:** Nos mostrará cómo calcula las componentes conexas por pasos. Primero la matriz de accesibilidad R , luego su traspuesta Q , y luego la matriz $R \otimes Q$,

Estudio de la accesibilidad y conectividad

resultante de multiplicar elemento a elemento las dos primeras matrices. Por último muestra las componentes conexas. Este resultado será la interpretación de la matriz $R \otimes Q$.

- **Resultado final:** Nos ofrece directamente las componentes conexas del grafo, es decir, la última parte de lo que nos ofrecía la subopción anterior.

Desde el modo gráfico también aparecen dos opciones pero no exactamente iguales:

- **Por pasos:** Nos mostrará cómo calcula las componentes conexas por pasos, de forma análoga a como lo hizo desde el modo de texto
- **Pinchando en el grafo:** Al pinchar un vértice enciende del mismo color todos los vértices que están en la misma componente conexa que el pinchado.

Así, si al grafo de nuestro ejemplo le aplicamos este método, MaGraDa obtiene las matrices

$$R = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad Q = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 \end{bmatrix},$$

y

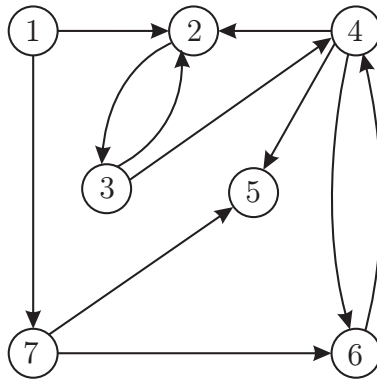
$$R \otimes Q = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

Para calcular, por ejemplo, la componente conexa que contiene al vértice 1, nos situamos en la primera fila, como sólo aparecen unos en la primera y cuarta columna de dicha fila, la componente conexa estará formada por los vértices $\{1, 4\}$. De forma análoga se obtendrían el resto de componentes conexas, que como ya vimos son $\{2, 3\}$, $\{5\}$ y $\{6\}$.

3.3.1 Práctica 6

En esta práctica vamos a tratar el concepto de conectividad y el cálculo de las componentes conexas de un grafo.

Problema 3.3. Consideremos el siguiente grafo dirigido:



- (i) *Calcula las componentes conexas de dicho grafo usando el método 1. Ayúdate cuando sea posible de los resultados que obtiene MaGraDa.*

(Continuación)

Estudio de la accesibilidad y conectividad

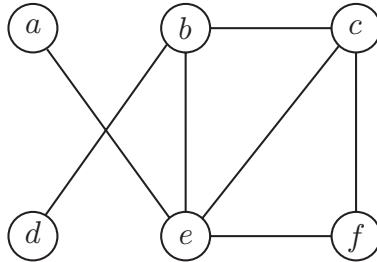
(ii) Calcula por pasos, con la ayuda de MaGraDa, las componentes conexas de dicho grafo usando el método 2. Explica el procedimiento seguido y comprueba que el resultado coincide con el del apartado anterior.

Problema 3.4. *Dibuja en tu práctica un grafo simple no dirigido que tenga seis vértices y sea conexo, de forma que si quitas una arista concreta del grafo, el grafo ya no sea conexo. Estas aristas se denominan aristas de corte. Comprueba con MaGraDa que efectivamente dicha arista es de corte y dibuja en tu práctica cuáles son las componentes conexas (obtenidas con MaGraDa) resultantes de eliminarla.*

Problema 3.5. *Dibuja con MaGraDa un grafo simple y no dirigido que tenga cinco vértices y sea conexo, de forma que si quitas cualquier arista del grafo, el grafo siga siendo conexo. Esto significa que tu grafo no tiene aristas de corte. Dibuja el grafo en tu práctica. Comprueba con MaGraDa que el grafo inicial es conexo y estudia el número de aristas que tienes que quitar del grafo como mínimo para que tenga dos componentes conexas. Escribe en tu práctica cuáles son dichas componentes.*

(Continuación)

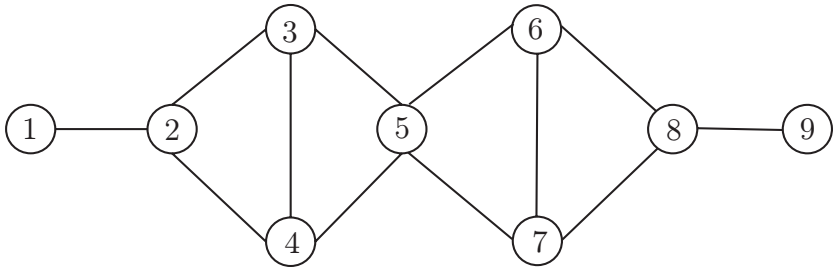
Problema 3.6. *Consideremos el grafo de la figura siguiente:*



Con la ayuda de MaGraDa explica qué aristas del grafo son de corte y cuáles no. Dibuja en tu práctica las componentes conexas que se crean.

(Continuación)

Problema 3.7. Decimos que un vértice de un grafo conexo es un vértice de corte, si al eliminarlo del grafo, el grafo resultante no es conexo. Consideremos el grafo de la siguiente figura:



Muestra con la ayuda de MaGraDa si el grafo de la figura tiene vértices de corte. Dibuja las componentes conexas resultantes en tu práctica.

(Continuación)

Estudio de la accesibilidad y conectividad

Problema 3.8. Pon un ejemplo de grafo no dirigido conexo tal que al eliminar cualquier arista se desconecte el grafo. Comprueba el resultado con MaGraDa. Completa, además, las siguientes frases respecto a este tipo de grafos:

(i) El grafo _____ (sí o no) puede tener bucles porque

(ii) El grafo _____ (sí o no) puede ser un multigrafo

fo porque _____

_____.

(iii) *Un grafo de este tipo con n vértices debería tener*
_____ *aristas porque* _____

_____.

Estudio de la accesibilidad y conectividad

4. Problemas de recorrido de aristas y vértices

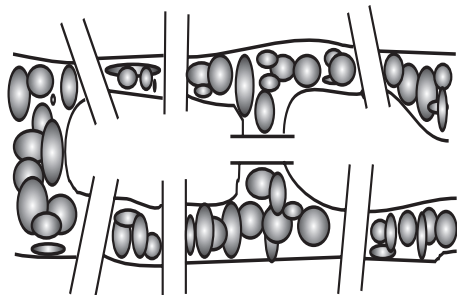
4.1 Introducción

En este capítulo, vamos a ver cómo utilizar MaGraDa para estudiar los problemas de recorrido de aristas y de vértices. La sección 4.2 está dedicada al problema de recorrido de aristas y en concreto al *algoritmo de Fleury*. En la sección 4.3 estudiaremos el problema de obtención de caminos y ciclos hamiltonianos.

4.2 Problema de recorrido de aristas

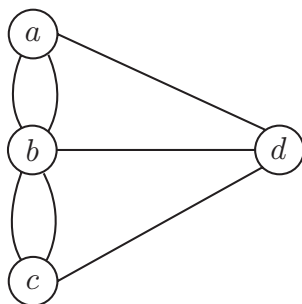
Durante el siglo XVIII, Königsberg fue una populosa y rica ciudad de la zona oriental de Prusia. Esta ciudad estaba dividida en cuatro zonas por el río Pregel. Siete puentes comunicaban estas zonas, como se muestra en la figura:

Problemas de recorrido de aristas y vértices



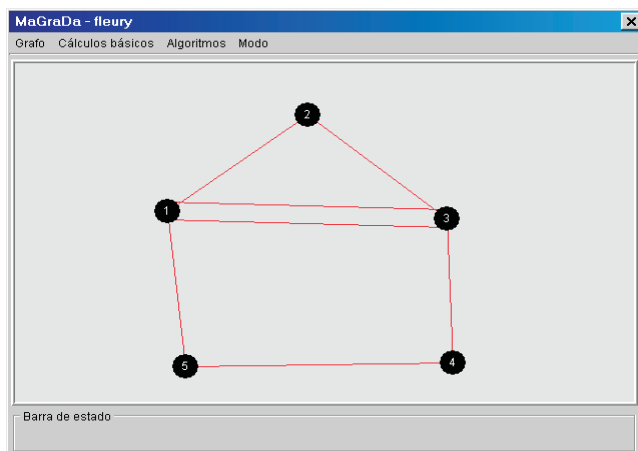
Se cuenta que los habitantes destinaban sus paseos dominicales intentando hallar un punto inicial para poder pasear por toda la ciudad, cruzando cada puente exactamente una vez y regresando a dicho punto.

Leonardo Euler, un matemático suizo natural de Basilea dio la solución a este problema. Para ello representó las cuatro zonas de la ciudad y los siete puentes como el multigrafo de la siguiente figura:



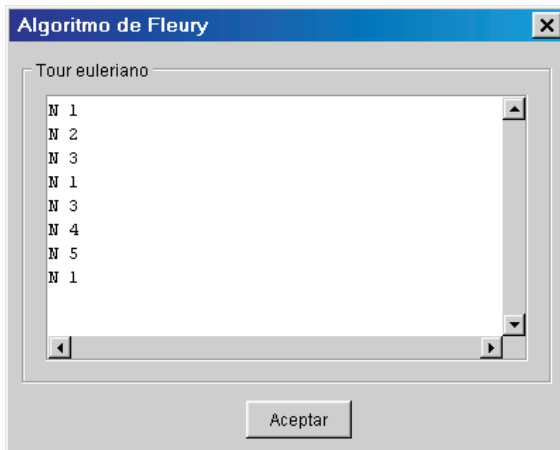
Entonces, Euler planteó el problema en los siguientes términos. ¿Es posible, partiendo de un vértice cualquiera, dibujar la figura volviendo siempre al mismo punto de partida, sin repasar ninguna línea ya trazada y sin levantar el lápiz del papel? La respuesta fue no, ya que la figura representa un grafo no dirigido que tiene vértices de grado impar, de hecho todos los vértices de este grafo tienen grado impar. Éste fue el comienzo de la teoría de grafos relativa a los caminos y tours eulerianos, es decir, caminos y ciclos, respectivamente, que recorren todas las aristas del grafo exactamente una vez.

A continuación vamos a estudiar la obtención de caminos y tours eulerianos en un grafo. Para ello consideremos el siguiente grafo no dirigido introducido desde el modo gráfico con MaGraDa:



Problemas de recorrido de aristas y vértices

Este grafo es conexo y no tiene vértices de grado impar, por tanto es euleriano. Para la obtención del tour euleriano con MaGrada, tanto desde el modo de texto como desde el modo gráfico, nos situaremos en la opción Algoritmo de Fleury del menú Algoritmos. Esta opción nos permite aplicar el algoritmo de Fleury para obtener tours o caminos eulerianos. Cuando ejecutemos esta opción MaGraDa verá si el grafo actual es euleriano, o tiene algún camino euleriano. Si es así, desde el modo de texto nos ofrecerá una pantalla con una sucesión de vértices que indican la forma de recorrer las aristas del grafo para formar el tour o camino. Así, en nuestro ejemplo, el tour obtenido por MaGraDa viene indicado de la siguiente forma:



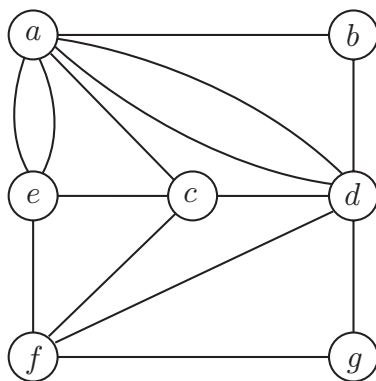
Es decir, el tour encontrado, empieza en el vértice 1 y recorre todas las aristas del grafo terminando de nuevo en el vértice 1.

Desde el modo gráfico, si el grafo tiene algún tour o camino euleriano no es el programa quien lo calcula y lo muestra, sino que es quien está trabajando con MaGraDa quien lo compone. MaGraDa generará una copia del grafo e iremos pinchando de vértice en vértice hasta acabar. Si la arista o arco recorrido es legal, lo eliminará (del grafo copia, el original se lo guarda). Si no es legal, nos avisará y tendremos que elegir otra alternativa. Cuando esta copia del grafo se quede sin aristas (o arcos), se habrá confeccionado el tour o camino y se mostrará una pantalla con el orden de los vértices que se eligieron. Cuando esto acabe, la copia del grafo se borrará y se mostrará de nuevo el grafo original con la totalidad de aristas (o arcos).

4.2.1 Práctica 7

A continuación vamos a familiarizarnos con los conceptos relativos a recorrido de aristas realizando la siguiente práctica.

Problema 4.1. *Consideremos el grafo de la siguiente figura:*



(i) *Con la ayuda de MaGraDa, indica si el grafo es conexo.*

(ii) *Utilizando el teorema de caracterización de un grafo euleriano y con la ayuda de MaGraDa, razona si dicho grafo es euleriano.*

Problemas de recorrido de aristas y vértices

(iii) Si la respuesta al apartado anterior es afirmativa, construye razonadamente y por pasos un tour euleriano desde el modo gráfico y escríbelo en tu práctica.

Problema 4.2. Obtén con MaGraDa el grafo resultante de eliminar la arista $\{c, d\}$ del grafo del ejercicio anterior. Dibújalo en tu práctica y realiza las siguientes cuestiones.

- (i) *Razona si el grafo es euleriano.*
- (ii) *Razona si el grafo tiene un camino euleriano.*
- (iii) *Si la respuesta al apartado anterior es afirmativa, construye razonadamente y por pasos un camino euleriano desde el modo gráfico y escríbelo en tu práctica.*

Problemas de recorrido de aristas y vértices

Problema 4.3. *Queremos saber si es posible que un insecto camine por las aristas de un cubo de manera que sólo pase una vez por cada arista. Para ello se deben seguir los siguientes pasos:*

(i) *Dibuja el grafo correspondiente con MaGraDa y en tu práctica.*

(ii) *Estudia con MaGraDa si el grafo es conexo, y si es así aplica el teorema correspondiente para contestar a la pregunta inicial. Ayúdate siempre que puedas de MaGraDa.*

Problema 4.4. *Introduce con MaGraDa el grafo dirigido cuya matriz de adyacencia es la siguiente y contesta a las siguientes cuestiones:*

$$\begin{bmatrix} 0 & 0 & 1 & 1 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 \end{bmatrix}$$

- (i) *Visualiza el grafo desde el modo gráfico, y preséntalo en tu práctica.*

Problemas de recorrido de aristas y vértices

(ii) *Comprueba con la ayuda de MaGraDa y el teorema correspondiente, si el grafo tiene un tour euleriano o camino euleriano.*

(iii) *Calcula razonadamente, si existe, un camino euleriano con MaGraDa desde el modo gráfico. Escríbelo en tu práctica.*

4.3 Problema de recorrido de vértices

Mediante MaGraDa, no podemos calcular directamente un camino o ciclo hamiltoniano (camino o ciclo que recorre todos los vértices del grafo exactamente una vez, respectivamente), pero podemos ayudarnos de esta aplicación para estudiar si existe dicho camino o ciclo, usando las reglas conocidas para tal fin, y encontrarlos en caso de que existan. Dado un grafo G con conjunto de vértices V , dichas reglas pueden enunciarse de la siguientes forma:

Regla 1. Si G es un grafo no conexo, no posee ciclos hamiltonianos.

Regla 2. Si G es un grafo con n vértices, entonces un camino hamiltoniano debe tener exactamente $n - 1$ aristas, y un ciclo hamiltoniano n aristas.

Regla 3. Si v es un vértice del grafo, entonces un camino hamiltoniano debe tener al menos una arista incidente con v y como mucho dos.

Regla 4. Si G es un grafo hamiltoniano, entonces $d_G(v) \geq 2$, $\forall v \in V$.

Regla 5. Si $v \in V$ tiene grado 2, entonces las dos aristas incidentes con v deben aparecer en cualquier ciclo hamiltoniano del grafo G .

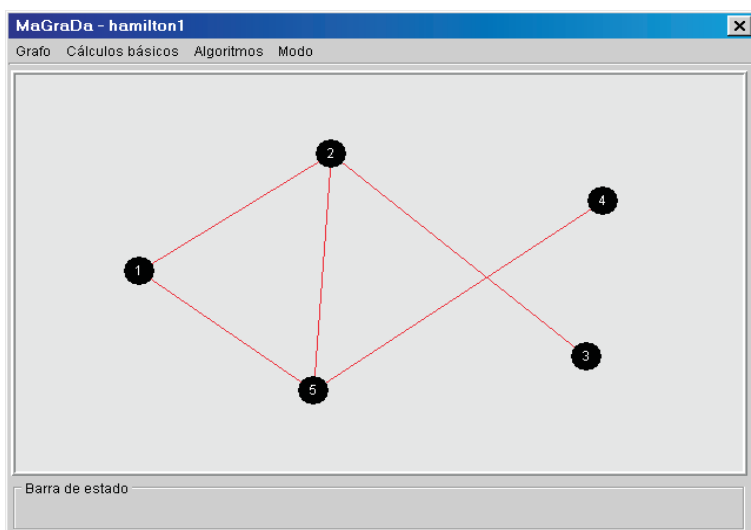
Regla 6. Si $v \in V$ tiene grado mayor que 2, entonces cuando se intenta construir un ciclo hamiltoniano, una vez que se

Problemas de recorrido de aristas y vértices

pase por v , las aristas no utilizadas incidentes con v se dejan de tener en cuenta.

Regla 7. Al construir un ciclo o camino hamiltoniano para un grafo G , no se puede dar el caso de obtener un ciclo para un subgrafo de G a menos que contenga todos los vértices de G .

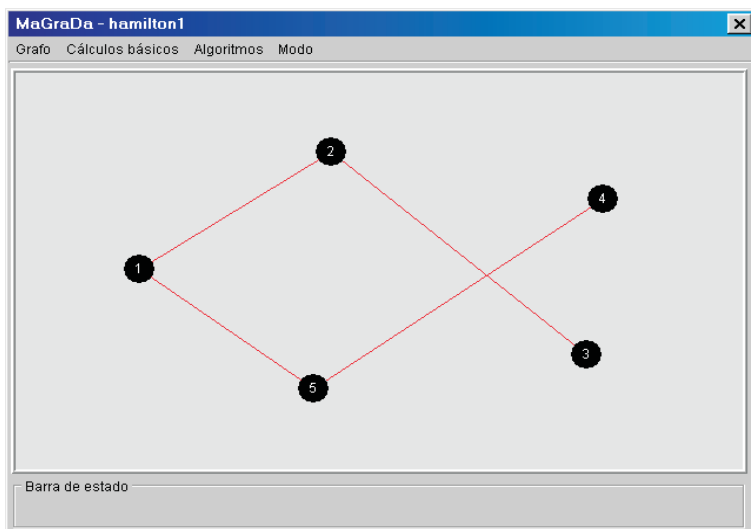
A continuación vamos a ver algunos ejemplos. Consideremos, en primer lugar, el siguiente grafo obtenido con MaGraDa:



Este grafo es conexo pero, según la regla 4, no es hamiltoniano, es decir no contiene un ciclo hamiltoniano ya que no

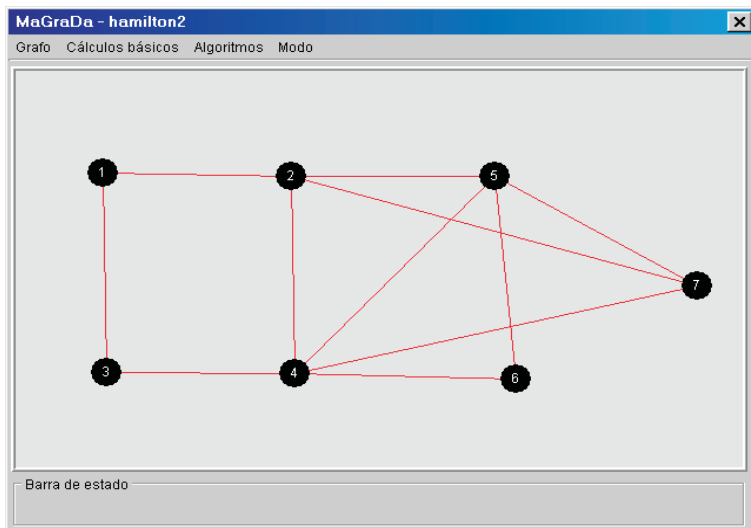
todos sus vértices tienen grado mayor o igual que dos. Así por ejemplo, los vértices 4 y 3 tienen grado uno.

Por otro lado, si tuviera un camino hamiltoniano, según la regla 2 dicho camino debería tener 4 aristas. Vamos a intentar su construcción. En primer lugar, atendiendo a la regla 3, las aristas $\{5, 4\}$ y $\{2, 3\}$, tendrían que estar en el camino. A partir de aquí es fácil ver que un camino hamiltoniano está formado por esas dos aristas y las aristas $\{1, 2\}$ y $\{1, 5\}$. Es decir el camino hamiltoniano vendría representado por la siguiente figura:



Veamos otro ejemplo. Consideremos el siguiente grafo dibujado con MaGraDa.

Problemas de recorrido de aristas y vértices



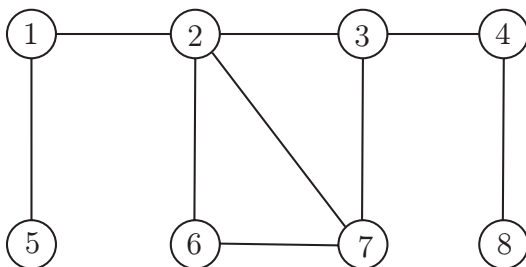
Este grafo es conexo, y si existe un ciclo hamiltoniano debe tener 7 aristas ya que tiene 7 vértices. Atendiendo a la regla 5, las aristas $\{1, 2\}$, $\{1, 3\}$, $\{3, 4\}$, $\{4, 6\}$ y $\{5, 6\}$, deben estar en el ciclo.

Teniendo en cuenta lo anterior y atendiendo ahora a la regla 6, sabemos que las aristas $\{2, 4\}$, $\{4, 7\}$ y $\{4, 5\}$ no pueden estar en el ciclo hamiltoniano. Nos quedan por tanto sólo tres aristas para poder construir el ciclo. Sin embargo, según la regla 7, la arista $\{2, 5\}$ no puede pertenecer al ciclo hamiltoniano. Nos quedan por tanto las aristas $\{2, 7\}$ y $\{7, 5\}$, que si las introducimos obtendremos un ciclo hamiltoniano.

4.3.1 Práctica 8

A continuación se plantean algunos problemas que ayudarán a comprender las reglas básicas para construir caminos y ciclos hamiltonianos.

Problema 4.5. *Consideremos el grafo de la siguiente figura:*



- (i) *Di cuántas aristas debería tener un ciclo hamiltoniano, si existiera.*

- (ii) *Calcula con MaGraDa el grado de cada vértice.*

Problemas de recorrido de aristas y vértices

- (iii) *Aplicando la regla 5, deja en el grafo sólo las aristas que sabes que deberían estar en el ciclo necesariamente, si existiera.*
- (iv) *De las aristas restantes y usando la regla 7, explica cuál de ellas no podría estar en el ciclo hamiltoniano.*
- (v) *A la vista de los resultados y usando la regla 6 explica si podemos encontrar un camino o ciclo hamiltoniano.*
- (vi) *Di, a la vista del grafo, qué única regla podrías haber utilizado para decir, directamente, que el grafo no era hamiltoniano. Razona la respuesta.*

Problema 4.6. *Consideremos un grafo cuya matriz de adyacencia viene dada por la siguiente matriz:*

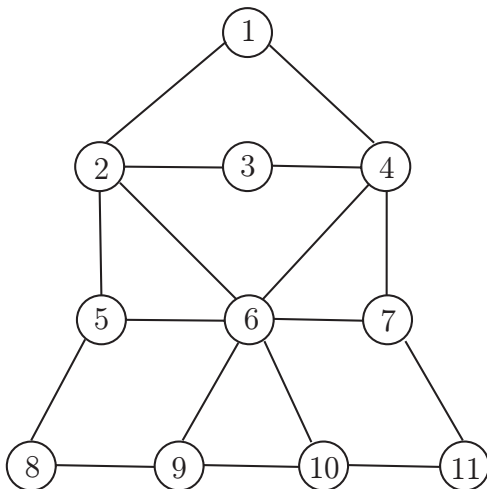
$$\begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 \end{bmatrix}.$$

Introduce este grafo con MaGraDa, dibújalo en tu práctica y ayudándote de las reglas conocidas, razona la existencia o no de un ciclo y camino hamiltoniano, obteniéndolo en caso de existir.

Problemas de recorrido de aristas y vértices

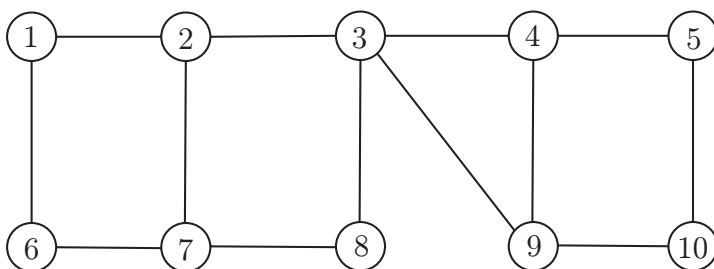
(Continuación)

Problema 4.7. *Razona utilizando MaGraDa y las reglas conocidas, la existencia o no de un ciclo hamiltoniano, encontrándolo en su caso, para el siguiente grafo:*



Problemas de recorrido de aristas y vértices

Problema 4.8. *Razona utilizando MaGraDa y las reglas conocidas, la existencia o no de un ciclo hamiltoniano y la existencia o no de un camino hamiltoniano para el siguiente grafo. Encuentra en su caso, razonadamente, un ciclo y camino hamiltoniano.*



5. Introducción a la estructura de árbol

5.1 Introducción

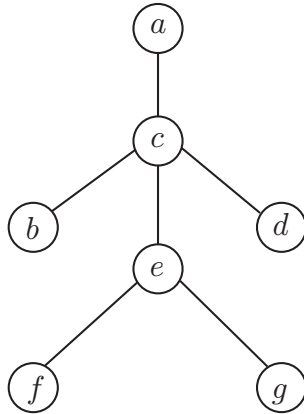
En este capítulo vamos a ver cómo utilizar MaGraDa, para hacer un estudio de los conceptos básicos relacionados con un tipo de grafo particular, los árboles. En la sección 5.2 se definirá dicho concepto y se ahondará sobre las propiedades de dichos grafos. La sección 5.3 está dedicada a los árboles enraizados. Como aplicación de estos conceptos, en la sección 5.4, introduciremos la notación polaca para la representación de expresiones algebraicas.

5.2 Definiciones y propiedades

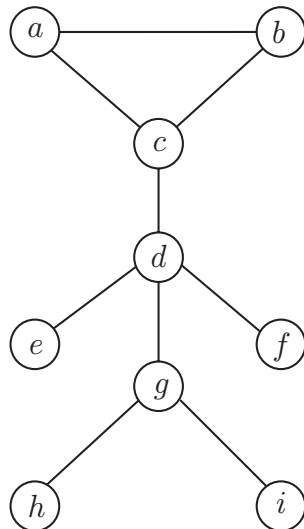
Recordemos que un árbol es un grafo no dirigido conexo y acíclico. Por tanto podemos ayudarnos de MaGraDa para ver si un grafo es árbol. Para ello utilizaremos las opciones Cíclico y Conexo del menú Grafo de Cálculos básicos. Veamos un ejemplo. Consideremos los grafos no dirigidos:

Introducción a la estructura de árbol

(i) Grafo a:

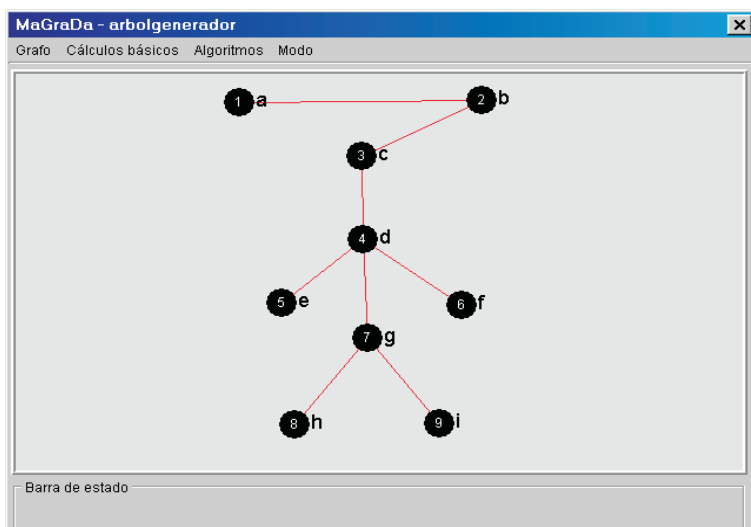


(ii) Grafo b:



Como se puede comprobar con MaGraDa ambos grafos son conexos, sin embargo el grafo a es acíclico y el grafo b no lo es, por tanto sólo es árbol el grafo a.

Por otro lado, dado un grafo no dirigido, se llama árbol generador de dicho grafo a un subgrafo generador del grafo que además es árbol. Con MaGraDa se puede obtener un árbol generador de un grafo desde la opción Un árbol generador asociado del menú Cálculos básicos. Así por ejemplo, el árbol generador que obtiene MaGraDa para el grafo b viene dibujado en la siguiente pantalla:

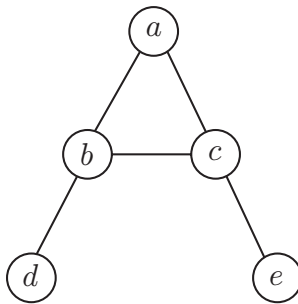


5.2.1 Práctica 9

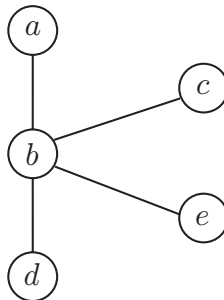
Para la realización de la siguiente práctica, se deben conocer y haber entendido el concepto de árbol y los teoremas relacionados con este concepto.

Problema 5.1. Consideremos los grafos de las siguientes figuras:

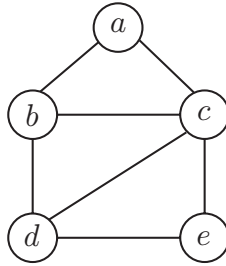
- Grafo A:



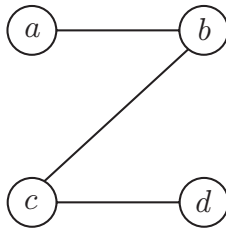
- Grafo B:



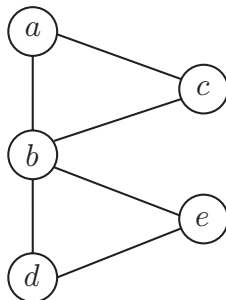
- Grafo C:



- Grafo D:



- Grafo E:



Introducción a la estructura de árbol

(i) Razona, con la definición, cuáles de los grafos anteriores son árboles.

(ii) Introduce dichos grafos con MaGraDa y calcula el grado de cada vértice para cada grafo.

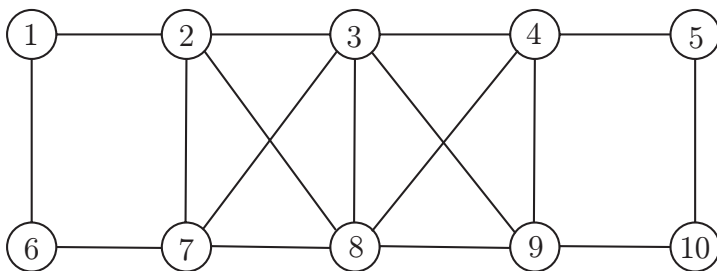
Vértice	<i>a</i>	<i>b</i>	<i>c</i>	<i>d</i>	<i>e</i>
Grafo A					
Grafo B					
Grafo C					
Grafo D					*
Grafo E					

(iii) Completa las siguientes frases:

Atendiendo a los grados de los vértices podemos asegurar que los grafos _____ no son árboles porque _____.

Atendiendo al número de aristas podemos asegurar que los grafos _____ no son árboles porque _____.

Problema 5.2. Consideremos el grafo de la figura. Introduce dicho grafo con MaGraDa y contesta a las siguientes cuestiones:



(i) ¿Por qué no es un árbol? Razónalo a partir de la definición.

(ii) Contesta a la pregunta anterior pero sin usar la definición, es decir, aplicando los teoremas conocidos sobre

Introducción a la estructura de árbol

árboles. Hazlo de dos formas distintas, usando MaGraDa, si lo crees conveniente, para ayudarte a comprobar las condiciones de los teoremas utilizados.

(iii) *¿Podemos obtener un árbol generador del grafo? ¿Por qué? Usa para tu razonamiento el teorema que creas conveniente.*

(iv) *Desde MaGraDa, obtén un árbol generador de dicho grafo. Dibújalo en tu práctica y explica por qué lo es.*

Problema 5.3. *Realiza las siguientes cuestiones:*

- (i) *Dibuja un árbol con 4 aristas con MaGraDa y en tu práctica.*

- (ii) *Di, razonándolo, cuántos vértices debe tener necesariamente este árbol.*

- (iii) *Quita una arista a tu árbol. El grafo resultante ya no es un árbol. Explica el porqué, usando la definición. Comprueba con MaGraDa la condición que ya no cumple para ser árbol.*

Introducción a la estructura de árbol

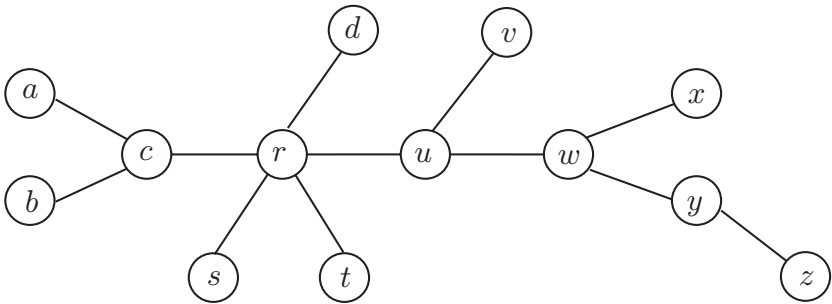
(iv) Repite el apartado anterior con una arista distinta.

(v) Crees que para volver a conseguir un árbol, sería siempre suficiente quitar también un vértice. Razona la respuesta.

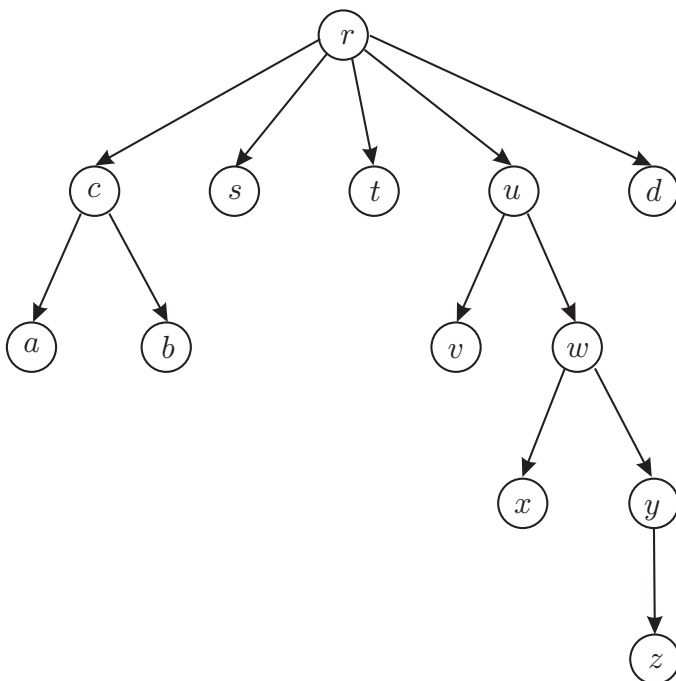
Problema 5.4. *Se sabe que cierto árbol sólo tiene vértices de grado 1, 2 y 3. Si dicho árbol tiene un vértice de grado 2, y dos de grado 3, razona cuántos vértices tendrá de grado 1. Introduce con MaGraDa un árbol con estas características y dibújalo en tu práctica. Comprueba con MaGraDa que es conexo y acíclico.*

5.3 Árboles con raíz

Recordemos que si T es un árbol, podemos elegir un vértice r_0 de dicho árbol y al ser conexo definir un grafo dirigido $T(r_0)$ donde todos los arcos sean extremos finales de un camino que se inicia en r_0 . A este grafo se le llama *árbol enraizado* en r_0 . Veamos un ejemplo. Consideremos el árbol de la figura:



A partir de él podemos construir el siguiente árbol enraizado en r :



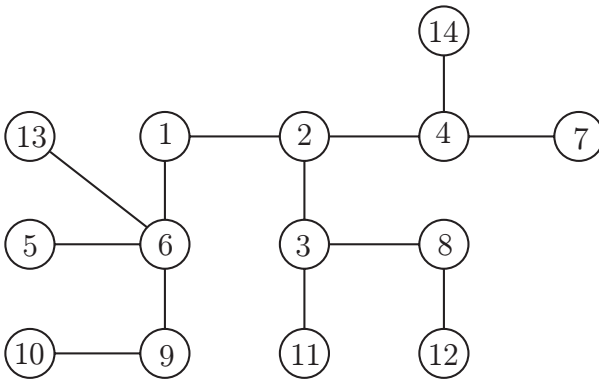
Queremos hacer notar que por comodidad, al dibujar un árbol enraizado, se suelen obviar las flechas de los arcos.

A la vista de este árbol podemos decir, por ejemplo, que los vértices a y b son hijos del vértice c , que los vértices a, b, s, t, d, v, x, z no tienen descendientes y por tanto son terminales y que los vértices r, c, u, w, y son internos.

5.3.1 Práctica 10

A continuación vamos a familiarizarnos con los conceptos relacionados con los árboles enraizados.

Problema 5.5. Consideremos el grafo de la siguiente figura:



(i) Dibuja con MaGraDa y en tu práctica, con la forma típica de árbol enraizado, el grafo dirigido $T(1)$.

(ii) *Calcula la matriz de adyacencia con la ayuda de MaGraDa.*

(iii) *Con el uso de esa matriz, completa las siguientes frases:*

El vértice 6 es hijo de _____ porque en la matriz de adyacencia observamos que _____

_____ . Los hermanos del vértice 5 son _____

porque en la matriz de adyacencia observamos que

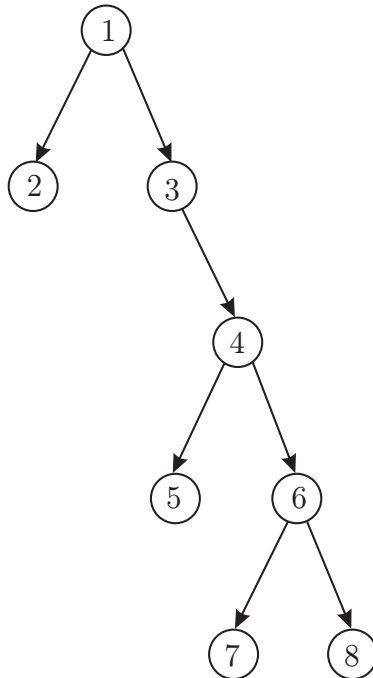
_____ .

Introducción a la estructura de árbol

(iv) El vértice 11 no tiene hijos porque en la matriz de adyacencia observamos que _____

_____.

Problema 5.6. Consideremos el grafo de la siguiente figura:



(i) Con la ayuda de MaGraDa, calcula la matriz de accesi-

bilidad y contesta a las siguientes cuestiones razonadamente, usando dicha matriz:

- *Di qué vértices son internos.*

- *A la vista del apartado anterior, explica cuántos vértices tendrá terminales. Además, usando la matriz de accesibilidad, indica de forma razonada, cuáles son.*

Introducción a la estructura de árbol

- *Explica cuáles son los descendientes del vértice 3.*

- *Explica cuál es la altura de este árbol. Usa también, si lo crees necesario, la matriz de adyacencia para dar tu contestación.*

Problema 5.7. *El primer domingo del año 2001, Héctor inició una cadena de cartas, enviando una carta a cada uno de sus dos amigos. Cada persona que reciba una carta debe enviar 2 copias a 2 personas, el domingo siguiente a la llegada de la carta. Después de los 3 primeros domingos, contesta a las siguientes cuestiones:*

- (i) *Dibuja con MaGraDa un grafo que represente esta situación.*

Introducción a la estructura de árbol

(ii) *Conociendo el número total de vértices del árbol y sin mirar el grafo, razona cuántas cartas se han mandado.*

(iii) *¿Cuántos vértices internos tiene el árbol? Conociendo el número de vértices del árbol y el número de vértices internos, razona cuántas cartas se mandan el último domingo.*

5.4 Notación polaca

Los recorridos en preorden, postorden e inorden dan modos de listar los vértices de un árbol enraizado. Si etiquetamos los vértices obtenemos una lista de etiquetas. Si las etiquetas de los vértices terminales son números y las etiquetas de los vértices internos son signos de operaciones binarias, nuestra lista puede dar una representación de una expresión aritmética. Por ejemplo, $5 * 4 / 2$ determina el número 10, sin embargo la lista $5 + 4 * 2$ parece ambigua ya que podría determinar el número 18 o el número 13.

La notación polaca es un método de definir operaciones algebraicas sin usar paréntesis, utilizando listas obtenidas de árboles. Es importante que las listas determinen por completo los árboles etiquetados correspondientes y sus expresiones. Veamos un ejemplo para recordar esto. Consideremos la siguiente expresión:

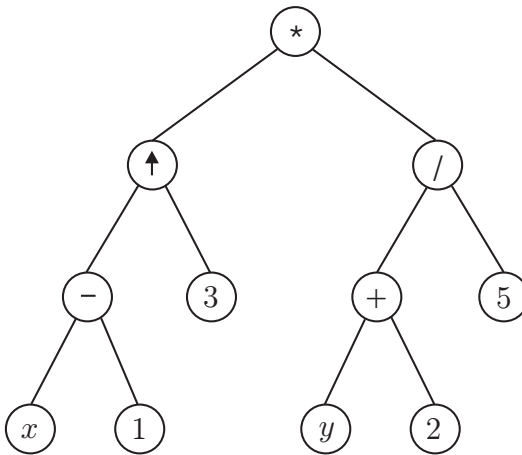
$$(x - 1)^3 * \frac{(y + 2)}{5}.$$

Dicha expresión puede ser representada por un árbol binario, en donde una operación actúa sobre los subárboles a la izquierda y a la derecha. Notemos que en un árbol binario hay diferencia entre el subárbol a la izquierda y el subárbol a la derecha de un vértice, es decir el subárbol a la derecha y el subárbol a la izquierda de un vértice corresponden al operan-

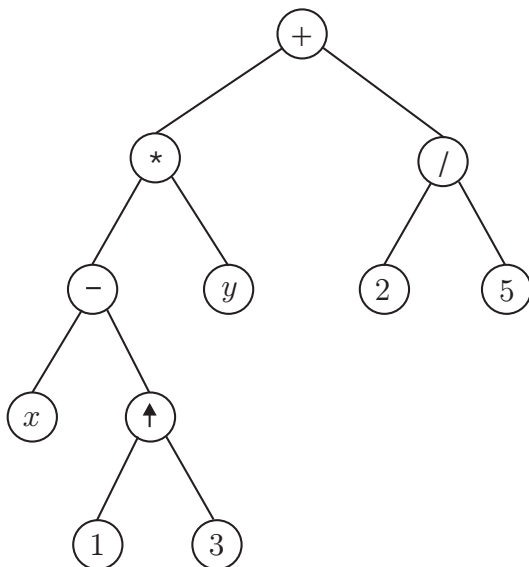
Introducción a la estructura de árbol

do izquierda y derecha respectivamente de una operación (es decir, no es lo mismo $x - 1$, que $1 - x$).

Así el árbol correspondiente a la expresión anterior es:



Si recorremos este árbol aplicando inorden da $x - 1 \uparrow 3 * y + 2 / 5$, que es la expresión original sin paréntesis, y se denomina *notación infija*. Este listado es ambiguo ya que podría provenir de otro árbol. Por ejemplo del árbol:



que corresponde con la expresión matemática

$$(x - 1^3) * y + \frac{2}{5}.$$

Sin embargo, si el árbol inicial hubiera sido recorrido usando preorden o postorden, las operaciones se hubieran podido evaluar sin ninguna ambigüedad, ya que dichas representaciones no necesitan paréntesis ni convención alguna que especifique el orden de las operaciones y es por ello que muchos compiladores las usan en lugar de la representación infija. Estas formas de notación algebraica se denominan *notación*

Introducción a la estructura de árbol

polaca directa (o *prefija*) y *notación polaca inversa* (o *postfija*), respectivamente. Como ejemplo, podemos decir que para nuestro árbol inicial se obtiene:

- Notación polaca directa: $* \uparrow - x \ 1 \ 3 / + y \ 2 \ 5$.
- Notación polaca inversa: $x \ 1 - 3 \uparrow y \ 2 + 5 / *$.

5.4.1 Práctica 11

Ahora se realizarán algunos ejercicios relacionados con la notación polaca.

Problema 5.8. *Dibuja un árbol enraizado, a partir del cual se pueda obtener la expresión dada en notación polaca inversa: $6\ 2 / 1 - 5\ 3 + *$. Dibújalo en tu práctica. A partir de dicho árbol calcula por pasos su notación polaca directa.*

Problema 5.9. *Dibuja un árbol enraizado, a partir del cual se pueda obtener la expresión dada en notación polaca directa: $/ * 2 + x 5 \uparrow + 3 z y$. A partir de dicho árbol calcula su notación polaca inversa.*

Problema 5.10. *Dibuja un árbol enraizado que corresponda con la expresión algebraica*

$$\frac{x^y}{3} + (z - 5)^7.$$

A partir de dicho árbol calcula su notación polaca inversa y su notación polaca directa.

Problema 5.11. *Dibuja un árbol enraizado que corresponda con la expresión algebraica*

$$\left(\frac{(x^2 - 1)^3}{y - 5} \right) \left(\frac{4z + 7}{8 - t} \right).$$

A partir de dicho árbol calcula su notación polaca inversa y su notación polaca directa.

6. Grafos ponderados. Caminos críticos en grafos acíclicos

6.1 Introducción

En este capítulo vamos a introducir los grafos ponderados. En la sección 6.2 veremos cómo construir un grafo ponderado con MaGraDa. Posteriormente en la sección 6.3 veremos cómo obtener caminos más cortos y el algoritmo PERT, para grafos acíclicos, basándonos en las ecuaciones de Bellman.

6.2 Creación de un grafo ponderado

Dado un grafo simple, se dice que dicho grafo es ponderado si las aristas o arcos del grafo tienen asociado un peso. Como en el caso de grafos no ponderados, los grafos ponderados pueden crearse tanto desde el modo de texto como desde el modo gráfico. Vamos a explicar la forma de hacerlo desde ambos modos:

- **Desde el modo de texto:** La creación de un grafo ponderado desde el modo de texto se realiza, como en el caso de los grafos no ponderados, desde la opción Nuevo.

Grafos ponderados. Caminos críticos en grafos acíclicos

Al indicar que el grafo es ponderado, nos permitirá crear este grafo de dos maneras, mediante la introducción de los arcos o aristas, o mediante su matriz de pesos. Si lo hacemos de la primera forma, además de pedirnos los vértices extremos de cada arista o arco, nos pedirá también su peso. Queremos hacer notar que esta versión de MaGraDa sólo trabaja con pesos enteros no negativos. Además, la introducción de un grafo mediante su matriz de pesos, está limitada a grafos que no tienen peso cero en ninguna arista o arco, ya que internamente utiliza ese valor para indicar que no existe arista o arco entre el correspondiente par de vértices. De hecho, si se crea el grafo de esta forma y luego se desea visualizar la matriz de pesos desde el menú de Cálculos básicos, observaremos que los ceros los ha transformado en infinito. Cabe mencionar también que, debido al tipo de algoritmos con los que MaGraDa trabaja dentro del contexto de grafos ponderados, es irrelevante tener más de una arista o arco uniendo el mismo par de vértices y con pesos distintos, por lo que en el caso de grafos ponderados MaGraDa sólo permite una única arista o arco entre el mismo par de vértices (recordemos además que desde el punto de vista teórico un grafo ponderado es un grafo simple). Obviamente, en el caso dirigido, se en-

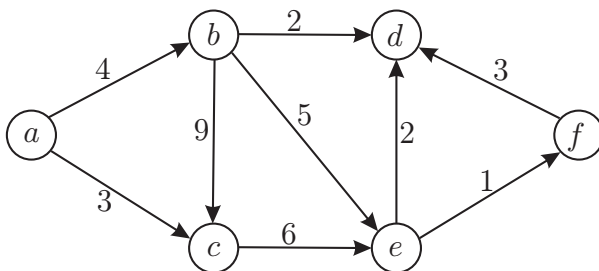
tiende que puede haber dos arcos entre un mismo par de vértices pero con sentidos contrarios.

- **Desde el modo gráfico:** La forma de introducir un grafo ponderado desde el modo gráfico es análoga al caso de grafos no ponderados. La diferencia estriba en que hay que decir que es ponderado, cuando especifiquemos el tipo de grafo. Además, como el grafo es ponderado, a cada arista o arco tendremos que asociarle un peso. Cuando se introduce cada arista o arco, el propio MaGraDa nos preguntará su peso. Como ya se adelantó en el capítulo 1 el peso se muestra en blanco sobre cuadrados de color morado. Hacemos notar que, al igual que desde el modo de texto, sólo se pueden utilizar pesos enteros no negativos y que MaGraDa sólo admite una única arista o arco entre un mismo par de vértices.

6.2.1 Práctica 12

Para familiarizarnos con la creación de grafos ponderados se plantean los siguientes ejercicios.

Problema 6.1. *La siguiente figura representa un grafo ponderado.*



- (i) *Introduce este grafo desde el modo gráfico, dejando a MaGraDa que sea él, el que sitúe los vértices en el lienzo. Dibuja en tu práctica el grafo resultante.*

(ii) *A la vista del grafo, calcula en tu práctica la matriz de pesos y explica su significado. Comprueba tu resultado con MaGraDa.*

Problema 6.2. *Supongamos que tenemos un grafo ponderado dirigido, cuya matriz de pesos es:*

$$\begin{bmatrix} \infty & 2 & 1 & 9 & \infty & \infty & \infty \\ 9 & \infty & 8 & \infty & 4 & 2 & \infty \\ 1 & 1 & \infty & 1 & \infty & 1 & \infty \\ 3 & \infty & 1 & \infty & \infty & 1 & 1 \\ \infty & 1 & \infty & \infty & \infty & 1 & \infty \\ \infty & 3 & 1 & 5 & 1 & \infty & 1 \\ \infty & \infty & \infty & 4 & \infty & 3 & \infty \end{bmatrix}$$

Grafos ponderados. Caminos críticos en grafos acíclicos

(i) Calcula la matriz de adyacencia. Comprueba el resultado con MaGraDa.

(ii) Razona cuáles son las similitudes entre la matriz de adyacencia y la matriz de pesos.

(iii) *Completa las siguientes frases: El grafo que nos ocupa tiene _____ vértices, _____ aristas y _____ arcos. En particular, el vértice 3 es adyacente con los vértices _____. El vértice 5 es el vértice final de los arcos _____ y _____ con pesos _____ y _____, respectivamente, porque _____*

_____. Además los arcos de mayor peso son los arcos _____ y _____, cuyo peso es _____.

6.3 Ecuaciones de Bellman

En esta sección vamos a estudiar cómo utilizar las ecuaciones de Bellman para obtener caminos más cortos y más largos en un grafo acíclico.

6.3.1 Caminos más cortos

En un grafo ponderado se llama camino más corto entre dos vértices dados al camino de peso mínimo entre dichos vértices. Supondremos que el grafo es dirigido y que los pesos asociados a los arcos son todos no negativos. En caso de que el grafo fuera no dirigido se transformaría cada arista en dos arcos (una en cada sentido) y se resolvería el problema como si el grafo fuera dirigido.

MaGraDa obtiene este camino para grafos acíclicos usando internamente las ecuaciones de Bellman. Por tanto, el método que vamos a tratar aquí no va a ser válido para grafos no dirigidos.

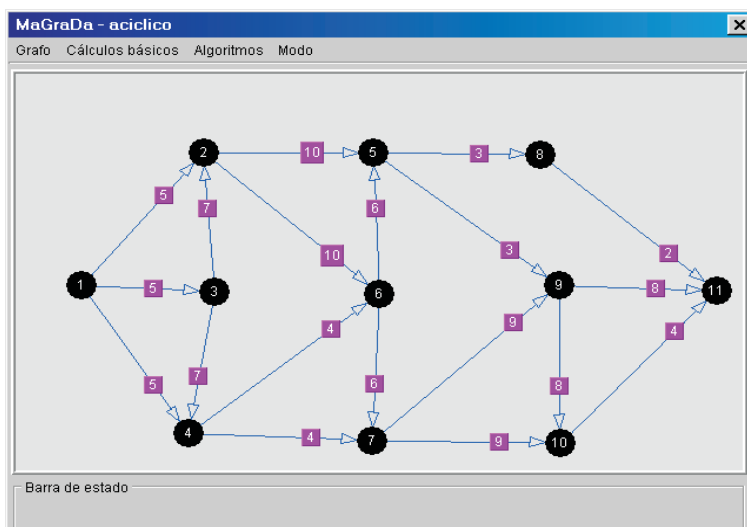
Para ello se suponen los vértices numerados del 1 al n , de forma que w_{ij} representa el peso del arco (i, j) y que el vértice 1 es el origen del camino. Por último denotaremos por u_j el peso del camino más corto del vértice 1 al vértice j . Entonces, las ecuaciones de Bellman vienen dadas por la siguiente expresión:

$$u_1 = 0,$$

$$u_j = \min_{k < j} \{u_k + w_{kj}\}, \quad j = 2, 3, \dots, n.$$

Recordemos que dichas ecuaciones se pueden aplicar si la numeración considerada de los vértices del grafo cumple que si (i, j) es un arco del grafo, entonces $i < j$, y esta numeración o reetiquetación de vértices es posible encontrarla si y sólo si el grafo es acíclico. Es por ello que las ecuaciones de Bellman no son aplicables a grafos no dirigidos.

Para entender el uso de las ecuaciones de Bellman para la obtención de caminos más cortos, utilizaremos el siguiente grafo ponderado:



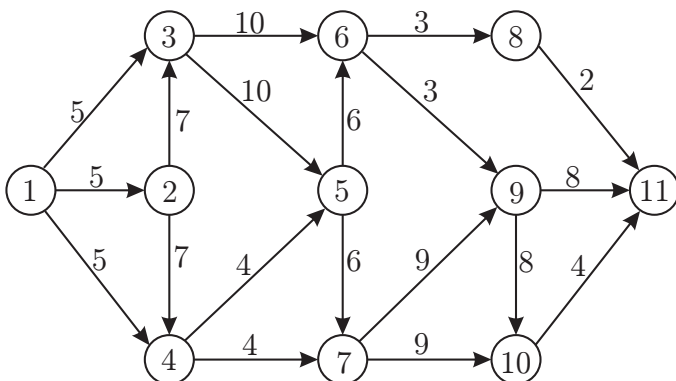
Este grafo, como se puede comprobar con MaGraDa es acíclico.

Grafos ponderados. Caminos críticos en grafos acíclicos

co, sin embargo la numeración actual de los vértices no permite aplicar las ecuaciones de Bellman. Así por ejemplo, existe un arco del vértice 3 al 2 y sin embargo 3 no es menor que 2. Por tanto, para aplicar dichas ecuaciones debemos renumerar los vértices. Esto lo hace MaGraDa, tanto desde el modo gráfico como desde el modo de texto en la opción `Renumerar de Caminos más cortos en grafos acíclicos`. Si aplicamos dicha opción, MaGraDa presenta una pantalla con la renumeración que va a usar y el vértice origen de los caminos que va a buscar, que será el 1, según su numeración.

Si observamos los resultados que aparecen en dicha pantalla, vemos que ha realizado sólo los siguientes cambios: El vértice 2 ha pasado a ser el 3 y viceversa, y el vértice 5 ha pasado a ser el 6 y viceversa. Notemos que sin embargo esto lo hace internamente. Es decir el grafo que presenta MaGraDa en pantalla sigue siendo el mismo.

Si ahora aplicamos el algoritmo al grafo con la nueva renumeración, que viene representada en la siguiente figura:



se obtiene lo siguiente:

$$u_1 = 0.$$

$$u_2 = \min\{u_1 + w_{12}\} = \min\{0 + 5\} = 5.$$

$$u_3 = \min\{u_1 + w_{13}\} = \min\{0 + 5\} = 5.$$

$$u_4 = \min\{u_1 + w_{14}\} = \min\{0 + 5\} = 5.$$

$$u_5 = \min\{u_3 + w_{35}, u_4 + w_{45}\} = \min\{5 + 10, 5 + 4\} = 9.$$

$$u_6 = \min\{u_3 + w_{36}, u_5 + w_{56}\} = \min\{5 + 10, 9 + 6\} = 15.$$

$$u_7 = \min\{u_4 + w_{47}, u_5 + w_{57}\} = \min\{5 + 4, 9 + 6\} = 9.$$

$$u_8 = \min\{u_6 + w_{68}\} = \min\{15 + 3\} = 18.$$

$$u_9 = \min\{u_6 + w_{69}, u_7 + w_{79}\} = \min\{15 + 3, 9 + 9\} = 18.$$

Grafos ponderados. Caminos críticos en grafos acíclicos

$$u_{10} = \min\{u_7 + w_{7,10}, u_9 + w_{9,10}\} = \min\{9 + 9, 18 + 8\} = 18.$$

$$\begin{aligned} u_{11} &= \min\{u_8 + w_{8,11}, u_9 + w_{9,11}, u_{10} + w_{10,11}\} \\ &= \min\{18 + 2, 18 + 8, 18 + 4\} = 20. \end{aligned}$$

Hacemos notar que, según las ecuaciones de Bellman, por ejemplo, $u_5 = \min\{u_1 + w_{15}, u_2 + w_{25}, u_3 + w_{35}, u_4 + w_{45}\}$, pero como $w_{15} = w_{25} = \infty$, hemos obviado los correspondientes valores $u_1 + w_{15}$ y $u_2 + w_{25}$, en el cálculo del mínimo realizado anteriormente. Esto es válido en todos los casos.

Una vez que tenemos los pesos de cada uno de los caminos más cortos del vértice 1 a los restantes, vamos a identificar los caminos:

- $u_1 = 0$, el peso del camino más corto del vértice 1 al 1 tiene peso cero porque es de longitud cero.
- $u_2 = 5$, el peso del camino más corto del vértice 1 al 2 es 5 y dicho camino está formado por el arco (1, 2).
- $u_3 = 5$, el peso del camino más corto del vértice 1 al 3 es 5 y dicho camino está formado por el arco (1, 3).
- $u_4 = 5$, el peso del camino más corto del vértice 1 al 4 es 5 y dicho camino está formado por el arco (1, 4).

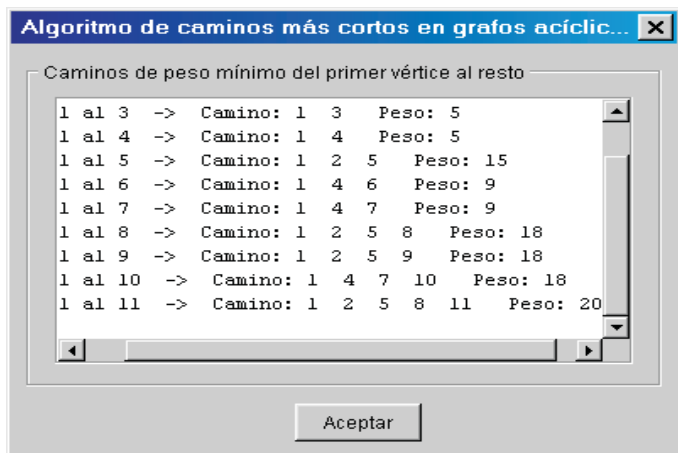
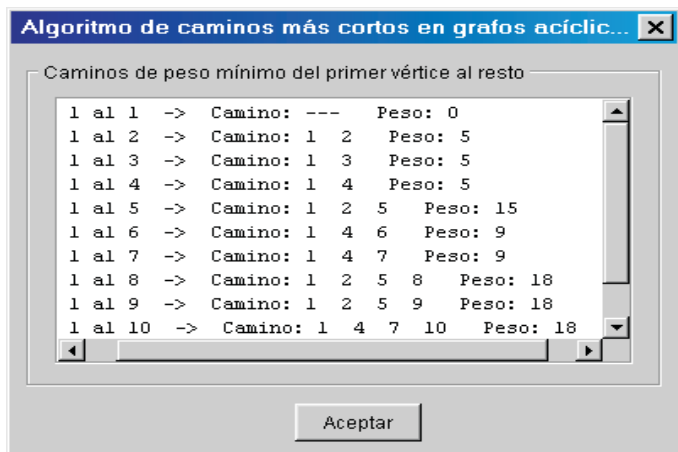
- $u_5 = 9$, el peso del camino más corto del vértice 1 al 5 es 9 y dicho camino está formado por el camino más corto del vértice 1 al 4 más el arco $(4, 5)$. Es decir, por los arcos $(1, 4)$ y $(4, 5)$.
- $u_6 = 15$, el peso del camino más corto del vértice 1 al 6 es 15 y dicho camino está formado, por ejemplo, por el camino más corto del vértice 1 al 3 más el arco $(3, 6)$. Es decir, por los arcos $(1, 3)$ y $(3, 6)$. Notemos que en este caso podríamos haber elegido también el camino más corto del vértice 1 al 5 más el arco $(5, 6)$, porque el peso es el mismo.
- $u_7 = 9$, el peso del camino más corto del vértice 1 al 7 es 9 y dicho camino está formado por el camino más corto del vértice 1 al 4 más el arco $(4, 7)$. Es decir, por los arcos $(1, 4)$ y $(4, 7)$.
- $u_8 = 18$, el peso del camino más corto del vértice 1 al 8 es 18 y dicho camino está formado por el camino más corto del vértice 1 al 6 más el arco $(6, 8)$. Es decir, por los arcos $(1, 3)$, $(3, 6)$ y $(6, 8)$.
- $u_9 = 18$, el peso del camino más corto del vértice 1 al 9 es 18 y dicho camino está formado, por ejemplo, por el camino más corto del vértice 1 al 6 más el arco $(6, 9)$. Es

Grafos ponderados. Caminos críticos en grafos acíclicos

decir, por los arcos $(1, 3)$, $(3, 6)$ y $(6, 9)$. Notemos que en este caso podríamos haber elegido también el camino más corto del vértice 1 al 7 más el arco $(7, 9)$, porque el peso es el mismo.

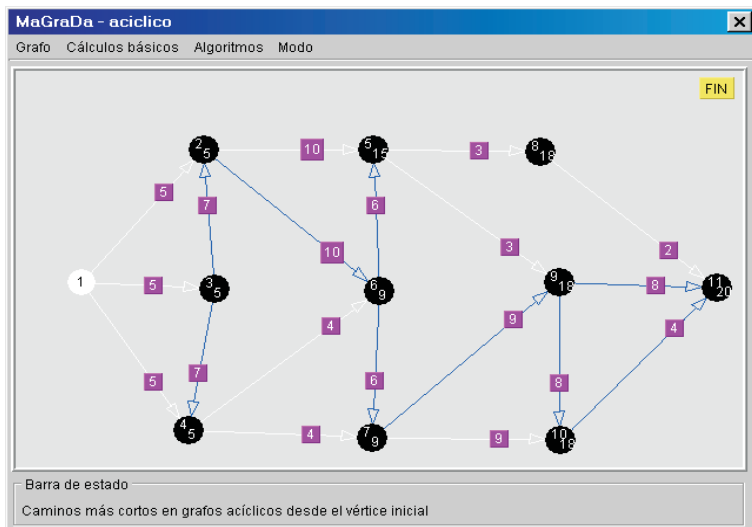
- $u_{10} = 18$, el peso del camino más corto del vértice 1 al 10 es 18 y dicho camino está formado por el camino más corto del vértice 1 al 7 más el arco $(7, 10)$. Es decir, por los arcos $(1, 4)$, $(4, 7)$ y $(7, 10)$.
- $u_{11} = 20$, el peso del camino más corto del vértice 1 al 11 es 20 y dicho camino está formado por el camino más corto del vértice 1 al 8 más el arco $(8, 11)$. Es decir, por los arcos $(1, 3)$, $(3, 6)$, $(6, 8)$ y $(8, 11)$.

Una vez obtenidos los caminos más cortos del vértice 1 a los restantes en el grafo reenumerado, hay que cambiar de nuevo la reenumeración para obtener dichos caminos en el grafo inicial. Es decir, hay que cambiar de nuevo el vértice 3 por el 2 y viceversa, y el 6 por el 5 y viceversa, en todos los resultados. De esta forma se obtiene el resultado final, que presenta MaGraDa, cuando se utiliza la opción `Aplicar` del algoritmo. Es decir, MaGraDa desde el modo de texto da la pantalla explicativa que mostramos a continuación en dos partes:



Desde el modo gráfico, MaGraDa da la información de la siguiente forma:

Grafos ponderados. Caminos críticos en grafos acíclicos



Como se puede observar los caminos más cortos se indican con flechas blancas, y dentro de cada vértice aparece, además de su número, el peso del camino más corto del vértice 1 a dicho vértice.

6.3.2 Secuenciación de actividades (PERT)

Se denomina camino más largo entre dos vértices dados, al camino de peso máximo entre dichos vértices. Un ejemplo típico de obtención de un camino más largo en un grafo dirigido, es la secuenciación de actividades, también llamada red de actividades.

Así por ejemplo, un proyecto de construcción grande se

suele dividir en varias actividades menores; estas actividades están relacionadas, en el sentido de que algunas no pueden comenzar hasta que otras hayan finalizado. Por ejemplo, para la construcción de una casa son necesarios cimientos, muros, carpintería, tejados, instalación eléctrica, Una actividad como la instalación eléctrica, no puede empezar hasta haber completado otras actividades. La realización de este tipo de proyectos hace necesaria una planificación racional de las actividades a realizar que se denomina PERT (Program Evaluation and Review Technique). Para planificar un proyecto de esta clase podemos representar cada actividad de las que se compone el proyecto por un vértice. Si para realizar una actividad i es necesario haber realizado antes la actividad j , se incluirá un arco de j a i . A cada arco (j, i) , se le asociará un peso w_{ji} , que indica el tiempo que debe transcurrir desde el comienzo de la actividad j al comienzo de la actividad i . En este grafo se incluyen dos vértices artificiales, que representan respectivamente el inicio y el final del proyecto que se unirán, respectivamente, con los vértices con grado de entrada cero y con los vértices con grado de salida cero. Es evidente que el camino más largo (de mayor peso), en el grafo construido, representa el tiempo mínimo necesario para completar el proyecto en su totalidad. A este camino se le denomina *camino crítico* ya que las actividades que incluye determinan el tiem-

po total de realización del proyecto y cualquier retraso en la ejecución de una de ellas, implica un retraso en la terminación del proyecto. Hacemos notar que el grafo de un PERT tiene que ser acíclico, ya que de lo contrario el proyecto sería impracticable. Por tanto, podemos aplicar las ecuaciones de Bellman, reemplazando las operaciones de minimización por maximización y así obtener el camino crítico y su peso, de la siguiente forma:

$$u_1 = 0,$$
$$u_j = \max_{k < j} \{u_k + w_{kj}\}, \quad j = 2, 3, \dots, n.$$

MaGraDa resuelve este problema desde la opción PERT del menú Algoritmos. Tanto desde el modo de texto como desde el modo gráfico ésta se divide en dos subopciones similares a las de la sección anterior:

- **Renumerar:** Se ha de ejecutar en primer lugar para tener acceso a la segunda opción. Una vez que se sabe que el grafo es acíclico tenemos que prepararlo (renumerarlo) para poder ejecutar el algoritmo. Es decir, hay que darle una nueva numeración a los vértices asegurando que ningún vértice sea extremo final de un arco cuyo vértice inicial tenga un número superior a él.
- **Aplicar:** Una vez ejecutada la opción anterior, el grafo está preparado para aplicar el algoritmo PERT. MaGra-

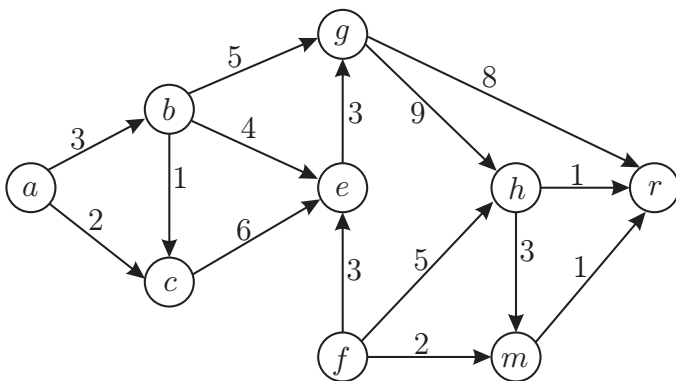
Da nos dará el camino más largo del vértice inicial al resto y sus pesos y en particular el camino crítico.

Queremos hacer notar que la forma de presentar los resultados es análoga a la vista en la sección anterior. Es decir, desde el modo de texto al ejecutar `Renumerar` nos presenta una pantalla explicativa de la renumeración utilizada para aplicar el algoritmo y al ejecutar `Aplicar` nos da los caminos más largos del vértice 1 al resto y su peso, en el grafo inicial (es decir, con la numeración original). Desde el modo gráfico nos resalta en blanco el camino crítico de dicho grafo.

6.3.3 Práctica 13

A continuación se propone una serie de ejercicios que ayudarán a comprender los algoritmos vistos en esta sección.

Problema 6.3. *Deseamos calcular los caminos más cortos del vértice a a los restantes en el grafo de la siguiente figura:*



Para ello resuelve las siguientes cuestiones:

- (i) *Razona si el grafo es acíclico, calculando a mano y razonadamente la reenumeración de los vértices que hace MaGraDa.*

(Continuación)

Grafos ponderados. Caminos críticos en grafos acíclicos

(ii) Calcula los caminos más cortos del vértice a al resto y su peso con MaGraDa. Explica cómo se han obtenido, usando las ecuaciones de Bellman.

Problema 6.4. *Vamos a ver un ejemplo sencillo que nos muestra un tipo de situación en la que aparece un PERT. Consideremos que un cocinero prepara una comida sencilla de curry y arroz. Suponemos que este cocinero cuenta con la ayuda suficiente para realizar simultáneamente aquellos pasos que sea posible. La receta de curry de este cocinero cuenta con los siguientes pasos.*

- *Cortar la carne (aproximadamente 11 minutos).*
- *Picar la cebolla (aproximadamente 2 minutos).*
- *Pelar y cortar las patatas (aproximadamente 6 minutos).*
- *Adobar la carne, la cebolla y las especias (aproximadamente 30 minutos).*
- *Calentar el aceite (4 minutos). Freír las patatas (15 minutos). Freír la semilla de cominos (2 minutos). Consideraremos esto como una única actividad ya que deben hacerse secuencialmente.*
- *Freír la carne adobada (aproximadamente 4 minutos).*
- *Hornear la carne frita y las patatas (aproximadamente 60 minutos).*

Grafos ponderados. Caminos críticos en grafos acíclicos

- *Además hay que cocinar el arroz (aproximadamente 18 minutos).*

Contesta a las siguientes cuestiones:

- (i) *Construye un grafo dirigido que represente la red de actividades para realizar esta comida de curry y arroz. Dibújalo en tu práctica e introdúcelo con MaGraDa.*

(ii) *Usa MaGraDa para reenumerar los vértices, si es necesario, y poder aplicar el PERT. Dibuja en tu práctica el grafo con la nueva reenumeración.*

Grafos ponderados. Caminos críticos en grafos acíclicos

(iii) Aplica al grafo reenumerado las ecuaciones de Bellman a mano, identificando los caminos más largos del vértice inicial al resto y sus pesos.

(Continuación)

Grafos ponderados. Caminos críticos en grafos acíclicos

(iv) *A la vista de los resultados obtenidos en el apartado anterior y teniendo en cuenta la numeración inicial de vértices, explica cuánto tiempo se necesitará como mínimo para tener hecha la comida e identifica razonadamente el camino crítico. Ayúdate de los resultados que obtiene MaGraDa.*

(v) *Completa las siguientes frases: El tiempo mínimo que se necesita para poder freír la carne es de _____ minutos. La cocción del aceite podríamos haberla retrasado _____ minutos sin retrasar la terminación de la receta.*

Problema 6.5. *Vamos a analizar un PERT sobre la construcción de una casa, abarcando sólo las primeras actividades típicas. Empezamos haciendo una descripción de las actividades primeras y el tiempo estimado para cada una de ellas.*

- *Pedido de los ladrillos (1 día).*
 - *Pedido del equipo y entrega (7 días).*
 - *Pedido del hormigón (1 día).*
 - *Explicación del solar (1 día).*
 - *Excavaciones (2 días).*
 - *Entrega del hormigón (3 días).*
 - *Realización de los cimientos (3 días).*
 - *Entrega de los ladrillos (20 días).*
 - *Diseño y pedido de la carpintería (14 días).*
 - *Construcción de los muros (11 días).*
 - *Entrega de la carpintería (13 días).*
- (i) *Estudia las dependencias existentes entre estas actividades y dibuja en tu práctica el grafo resultante de esta parte del proyecto de realización de una casa. Introdúcelo con MaGraDa.*

(Continuación)

- (ii) *Obtén el tiempo mínimo necesario para la consecución de esta primera parte de la obra. Identifica el camino crítico y explica cómo se obtiene, paso por paso, ayudándote con MaGraDa.*

(Continuación)

(iii) Completa las siguientes frases: El tiempo mínimo necesario para poder realizar los cimientos es de _____ días. El tiempo que se puede retrasar la construcción de cimientos sin retrasar todo el proyecto es de _____ días.

Grafos ponderados. Caminos críticos en grafos acíclicos

Problema 6.6. La tabla siguiente es una lista de las actividades $A, B, C, D, E, F, G, H, I$, de un proyecto y para cada una de ellas, el tiempo en días necesario y las actividades que deben completarse antes de poder iniciarse.

Actividad	A	B	C	D	E	F	G	H	I
Tiempo necesario	3	6	8	7	5	11	3	3	2
Prerrequisitos	–	–	A, B	C, E	B	E	D	F, G	B

- (i) Dibuja el grafo resultante.
- (ii) Calcula paso a paso el mínimo número de días en que puede completarse el proyecto. Comprueba el resultado con MaGraDa.

(Continuación)

Grafos ponderados. Caminos críticos en grafos acíclicos

(iii) Identifica razonadamente el camino crítico y su peso, explicando su significado.

(iv) Explica razonadamente cuántos días se puede retrasar la actividad E sin afectar la duración total del proyecto.

7. Caminos más cortos y árboles generadores de peso mínimo

7.1 Introducción

En el capítulo 6 se estudió una forma de encontrar caminos más cortos entre dos vértices, en grafos acíclicos, usando para ello, las ecuaciones de Bellman. En este capítulo y concretamente en la sección 7.2 veremos dos algoritmos que permiten obtener estos caminos sin la restricción de que el grafo sea acíclico: el algoritmo de Dijkstra y el algoritmo de Floyd y Warshall. La sección 7.3 está dedicada a la obtención de árboles generadores de peso mínimo, utilizando para ello los algoritmos de Kruskal y Prim.

7.2 Caminos más cortos

7.2.1 Algoritmo de Dijkstra

El algoritmo de Dijkstra encuentra los caminos más cortos y sus pesos desde un vértice cualquiera, que denotaremos por comodidad como vértice 1, al resto. Recordemos que a partir de las ecuaciones de Bellman podíamos calcular cami-

Caminos más cortos y árboles generadores de peso mínimo

nos más cortos, siempre y cuando el grafo fuera acíclico. Con el algoritmo de Dijkstra, no es necesaria esta restricción. Esto permite, por tanto, calcular caminos más cortos tanto en grafos dirigidos como en no dirigidos. En el caso no dirigido podemos trabajar transformando cada arista en dos arcos, uno en cada sentido, y resolver el algoritmo de Dijkstra como si de un grafo dirigido se tratase. Esto no lo podríamos haber hecho con las ecuaciones de Bellman porque el grafo dirigido resultante sería cíclico. Recordamos también que se supone que los vértices están numerados del 1 al n y que los pesos asociados a los arcos o aristas deben ser no negativos ($w_{ij} \geq 0$).

En el algoritmo de Dijkstra se asignan varias etiquetas a los vértices del grafo. En algún momento algunos vértices podrán tener etiquetas variables y el resto etiquetas fijas. Denotaremos al conjunto de vértices con etiqueta fija por P y al conjunto de vértices con etiqueta variable por T . Con estas consideraciones dicho algoritmo puede ser definido de la siguiente forma:

ALGORITMO DE DIJKSTRA

Paso 1. Inicialización.

$$P = \{1\}, T = \{2, 3, \dots, n\}$$

$$u_1 = 0$$

$$u_j = w_{1j}, \quad j \in \Gamma(1)$$

$$u_j = \infty, \quad j \notin \Gamma(1)$$

Paso 2. Designación de etiqueta variable como fija.

$$\text{Determinar } k \in T / u_k = \min_{j \in T} \{u_j\}$$

$$\text{Hacer } T := T \sim \{k\} \text{ y } P := P \cup \{k\}$$

Si $T = \emptyset$, STOP; u_j es el peso del camino más corto de 1 a j , $j = 2, 3, \dots, n$

Paso 3. Actualización.

$$\forall j \in \Gamma(k) \cap T, \quad u_j := \min\{u_j, u_k + w_{kj}\}$$

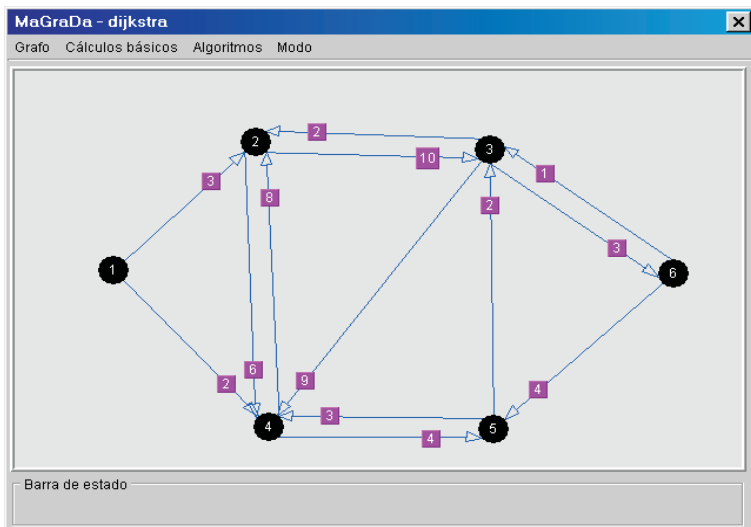
Ir al paso 2.

Queremos hacer notar que con este algoritmo realmente se pueden calcular los caminos más cortos entre todos los pares de vértices tomando como origen, en cada caso, cada uno de los vértices del grafo. Por otro lado, cabe mencionar que cuando un vértice j está etiquetado con etiqueta fija, la correspondiente variable u_j ya indica el peso del camino más corto del vértice 1 al j . Así, si quisieramos aplicar este algoritmo para calcular un sólo camino más corto entre, por ejemplo, los vértices 1 y j , el algoritmo finalizaría cuando $j \in P$.

MaGraDa resuelve este algoritmo tanto desde el modo de texto como desde el modo gráfico, en la opción Dijkstra del

Caminos más cortos y árboles generadores de peso mínimo

menú Algoritmos. Para comprender la forma en que MaGraDa presenta los resultados, vamos a considerar el siguiente grafo que ha sido previamente creado con MaGrada.



Desde el modo de texto, al aplicar el algoritmo de Dijkstra mediante MaGraDa, éste nos pide que seleccionemos el vértice inicial de los caminos más cortos que vamos a calcular. Consideraremos en este caso que se ha elegido el vértice 1. MaGraDa ofrecerá, en primer lugar, la siguiente pantalla:

N IT.	1	2	3	4	5	6
1	0	3	INFINITO	2 (1,4)	INFINITO	INFINITO
2	-	3 (1,2)	INFINITO	-	6	INFINITO
3	-	-	13	-	6 (4,5)	INFINITO
4	-	-	8 (5,3)	-	-	INFINITO
5	-	-	-	-	-	11 (3,6)

Veamos a qué corresponde la primera iteración: En ella empezamos realizando el paso 1, es decir, se ha tomado el 1 como etiqueta fija y el resto variables. En la fila 1, para cada columna j , aparece el peso $u_j = w_{1j}$, es decir, se ha tomado $P = \{1\}$, $T = \{2, 3, 4, 5, 6\}$, y los pesos obtenidos han sido $u_1 = 0$, $u_2 = 3$, $u_4 = 2$, y el resto ∞ .

A continuación, en el paso 2, el algoritmo designaba una etiqueta variable $k \in T$ como fija, que era aquella que satisfacía que $u_k = \min_{j \in T} \{u_j\}$. Esta etiqueta viene determinada por MaGraDa en cada iteración por la columna en la que aparece entre paréntesis un arco. Este arco corresponde al último arco del camino del vértice 1 al k . Así, en nuestro ejemplo se ha obtenido $u_4 = \min\{2, 3\} = 2$, y como el arco (1, 4) que

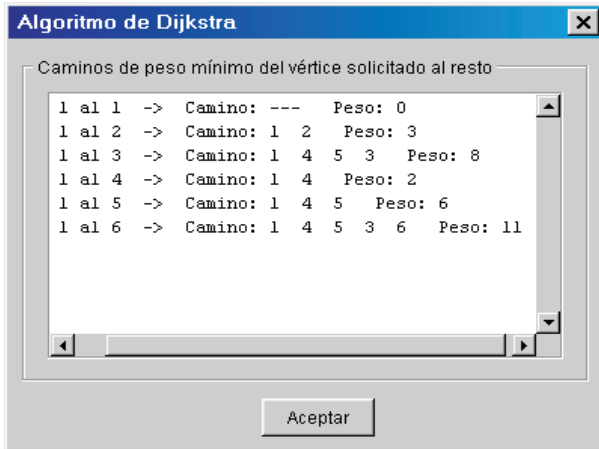
Caminos más cortos y árboles generadores de peso mínimo

aparece en la casilla, tiene como vértice inicial el 1, el camino más corto del vértice 1 al 4 está formado únicamente por el arco $(1, 4)$ y su peso es 2. Con esta información tendríamos que $P = \{1, 4\}$ y $T = \{2, 3, 5, 6\}$. Como $T \neq \emptyset$, debemos continuar con el paso 3 de actualización. Para ello debemos calcular para todo $j \in \Gamma(k) \cap T$, $u_j = \min\{u_j, u_k + w_{kj}\}$. Como $\Gamma(k) \cap T = \Gamma(4) \cap T = \{2, 5\} \cap \{2, 3, 5, 6\} = \{2, 5\}$, sólo tenemos que actualizar u_2 y u_5 . Los pesos del resto de vértices con etiqueta variable (el 3 y el 6) se mantienen igual, por tanto en este caso, seguirán siendo ∞ . Observamos en la segunda iteración que los pesos obtenidos son $u_2 = \min\{u_2, u_4 + w_{42}\} = \min\{3, 2 + 8\} = 3$ y $u_5 = \min\{u_5, u_4 + w_{45}\} = \min\{\infty, 2 + 4\} = 6$. Por tanto, como muestra MaGraDa, en la segunda iteración, si volvemos al paso 2, la etiqueta que ahora va a pasar a fija es $k = 2$ y en este caso, el peso $u_2 = 3$, se ha obtenido con el arco $(1, 2)$. Por tanto, el camino más corto del vértice 1 al 2 es el arco $(1, 2)$. Siguiendo el proceso, ahora tendríamos $P = \{1, 4, 2\}$ y $T = \{3, 5, 6\}$. La actualización de los pesos aparece ahora en la iteración 3. De ella se deduce que la nueva etiqueta que va a pasar de variable a fija es $k = 5$ y $u_5 = 6$. Para obtener el camino más corto del vértice 1 al 5 debemos fijarnos en el par que aparece al lado del número. En este caso dicho arco es el $(4, 5)$. Como el vértice inicial del arco no es 1, para obtener el camino más corto debemos ir

hacia atrás y encontrar el camino más corto entre 1 y 4, que ya vimos que era el arco $(1, 4)$. Por tanto, resumiendo, el camino más corto del vértice 1 al 5 está formado por los arcos $(1, 4)$ y $(4, 5)$. De forma análoga seguiríamos el algoritmo obteniendo en la iteración 4 la actualización de los pesos que da lugar a que la nueva etiqueta que se transformará en fija es $k = 3$ y que el camino más corto del vértice 1 al 3 está formado por los arcos $(1, 4)$, $(4, 5)$ y $(5, 3)$ y su peso es 8. Por último, en la iteración 5, una vez actualizados los pesos se obtiene que el camino más corto del vértice 1 al 6 tiene peso 11 y está formado por los arcos $(1, 4)$, $(4, 5)$, $(5, 3)$ y $(3, 6)$. Como ya no quedan vértices con etiqueta variable se ha finalizado el algoritmo, obteniendo todos los caminos más cortos desde el vértice 1 al resto.

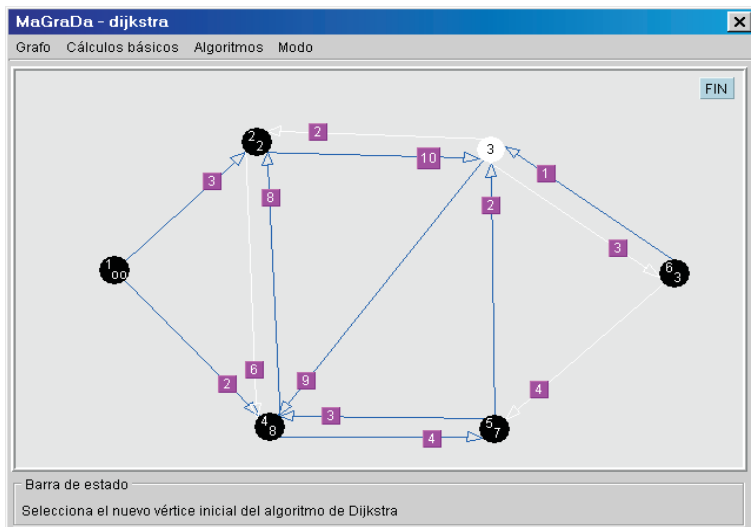
Estos caminos y sus pesos vienen dados por MaGraDa pulsando en la pantalla anterior `Ver caminos`. Concretamente MaGraDa presentará la siguiente pantalla:

Caminos más cortos y árboles generadores de peso mínimo



Queremos hacer notar que puede ocurrir que no siempre haya caminos entre cada par de vértices.

Desde el modo gráfico, MaGraDa actúa de forma análoga. Sin embargo, para obtener los caminos más cortos de un vértice al resto habrá que pinchar dicho vértice. Así, MaGraDa nos presentará una pantalla con las iteraciones igual que la del modo de texto. Si pulsamos *Ver caminos*, éstos nos los presentará de forma gráfica en el grafo en color blanco, como muestra la siguiente pantalla, en la que aparecen los caminos más cortos del vértice 3 a los restantes junto con su peso. Este peso aparece dentro del vértice final de cada camino.



7.2.2 Algoritmo de Floyd y Warshall

Aunque se pueden calcular los caminos más cortos entre todos los pares de vértices aplicando Dijkstra tomando como origen, en cada caso, cada uno de los vértices del grafo, el método de Floyd y Warshall es más eficiente en general para este propósito.

Llamaremos u_{ij} al peso del camino más corto del vértice i al j . Utilizaremos las variables $u_{ij}^{(m)}$, que indican el peso del camino más corto del vértice i al j con la restricción de que dicho camino no contenga los vértices $m, m + 1, \dots, n$ (exceptuando a los extremos i y j en su caso). Es decir, el camino que representa la variable $u_{ij}^{(m)}$ no debe contener como

Caminos más cortos y árboles generadores de peso mínimo

internos ninguno de los vértices $m, m + 1, \dots, n$.

Estas variables pueden calcularse recursivamente utilizando las ecuaciones:

$$\begin{aligned}u_{ij}^{(1)} &= w_{ij}, \quad \forall i, j \\u_{ij}^{(m+1)} &= \min \{ u_{ij}^{(m)}, u_{im}^{(m)} + u_{mj}^{(m)} \}, \quad \forall i, j, \\m &= 1, 2, \dots, n.\end{aligned}$$

Es posible ver que

$$u_{ij} = u_{ij}^{(n+1)},$$

con lo que tendremos los pesos de los caminos más cortos entre todos los pares de vértices.

Para facilitar la construcción de los caminos más cortos una vez calculados sus pesos, se puede utilizar otra matriz

$$\Theta^{(m)} = [\theta_{ij}^{(m)}]_{1 \leq i, j \leq n},$$

donde $\theta_{ij}^{(m)}$ representa el vértice anterior al j en el camino más corto del vértice i al j en la iteración m .

Inicialmente $\theta_{ij}^{(1)} = i$, si $u_{ij}^{(1)} < +\infty$, y

$$\theta_{ij}^{(m+1)} = \begin{cases} \theta_{ij}^{(m)} & \text{si } u_{ij}^{(m+1)} = u_{ij}^{(m)} \\ \theta_{mj}^{(m)} & \text{si } u_{ij}^{(m+1)} < u_{ij}^{(m)} \end{cases}$$

Tanto desde el modo de texto como desde el modo gráfico, MaGraDa resuelve este algoritmo desde la opción Floyd-Warshall del menú Algoritmos. Dentro de dicha opción tenemos dos subopciones:

- **Por pasos:** Este método muestra cómo se van calculando las matrices $[u_{ij}^{(m)}]$ y $\Theta^{(m)}$, para cada iteración m .
- **Entre dos vértices:** Nos da directamente el camino más corto entre el par de vértices seleccionado. Desde el modo de texto nos da una pantalla donde debemos seleccionar dichos vértices y desde el modo gráfico tendremos que pinchar los vértices que van a ser extremos del camino.

Vamos a ver cómo interpretar los resultados obtenidos con MaGraDa. Supongamos que deseamos calcular los caminos más cortos entre cada par de vértices para el grafo de este capítulo. Una vez realizadas todas las iteraciones del algoritmo obtenemos las siguientes matrices de pesos y caminos, respectivamente:

Caminos más cortos y árboles generadores de peso mínimo

Algoritmo de Floyd-Warshall. Iteración número 7

Matriz de pesos

1	2	3	4	5	6
INFINITO	3	8	2	6	11
INFINITO	12	10	6	10	13
INFINITO	2	4	8	7	3
INFINITO	8	6	7	4	9
INFINITO	4	2	3	7	5
INFINITO	3	1	7	4	4

Aceptar Ver matriz de caminos

Algoritmo de Floyd-Warshall. Iteración número 7

Matriz de caminos

1	2	3	4	5	6
-	1	5	1	4	3
-	3	2	2	4	3
-	3	6	2	6	3
-	4	5	5	4	3
-	3	5	5	4	3
-	3	6	5	6	3

Aceptar Ver matriz de pesos

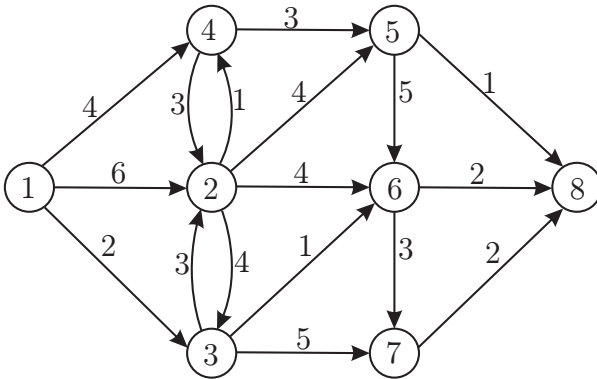
En la primera matriz aparece el peso de los caminos más cortos entre cada par de vértices. Así, por ejemplo, para calcular el peso del camino más corto del vértice 4 al vértice 6, nos

debemos situar en la cuarta fila y sexta columna de la primera matriz, con lo que se obtiene que el peso es 9. Para identificar el camino nos situamos en la misma posición de la siguiente matriz (matriz de caminos), como aparece un 3, eso significa que el último arco del camino es el $(3, 6)$. Ahora, siguiendo en la cuarta fila nos situamos en la columna 3, obtenemos un 5, por tanto ya tenemos los arcos $(5, 3)$ y $(3, 6)$. De nuevo en la cuarta fila, nos situamos en la columna 5 y obtenemos un 4, lo que quiere decir que hemos finalizado el proceso y el camino más corto está formado por los arcos $(4, 5)$, $(5, 3)$ y $(3, 6)$.

7.2.3 Práctica 14

A continuación presentamos una serie de ejercicios relativos a la búsqueda de caminos más cortos entre vértices utilizando los algoritmos vistos en esta sección.

Problema 7.1. *Introduce con MaGraDa, el siguiente grafo ponderado:*

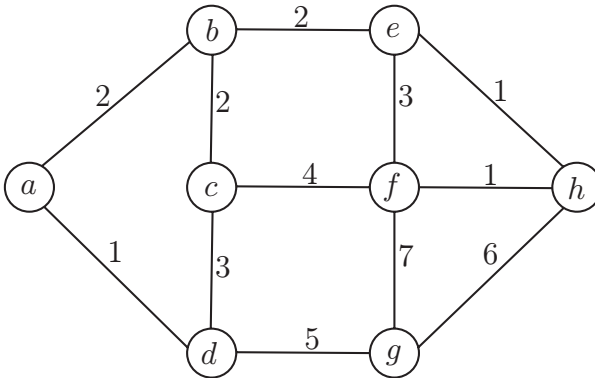


Obtén los caminos más cortos del vértice 1 a los restantes y sus pesos, paso por paso, mediante el algoritmo de Dijkstra. Ayúdate de MaGraDa para comprobar los resultados.

(Continuación)

Caminos más cortos y árboles generadores de peso mínimo

Problema 7.2. Consideremos el siguiente grafo ponderado:

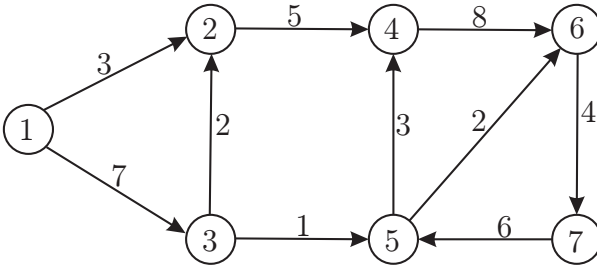


Calcula, de forma razonada, los caminos más cortos y sus pesos del vértice a al resto, basándote en la matriz de iteraciones que proporciona MaGraDa para el algoritmo de Dijkstra. Escribe dicha matriz en tu práctica .

(Continuación)

Caminos más cortos y árboles generadores de peso mínimo

Problema 7.3. Consideremos el siguiente grafo ponderado:



(i) Aplica el método de Floyd y Warshall. Hazlo a mano explicando el procedimiento y luego compruébalo con MaGraDa.

(Continuación)

Caminos más cortos y árboles generadores de peso mínimo

(ii) A la vista de las matrices que salen, identifica el camino más corto entre los vértices 2 y 5 y su peso. Haz lo mismo entre los vértices 7 y 6.

(iii) Aplicando el algoritmo de Floyd y Warshall como creas conveniente y razonadamente, usa MaGraDa para calcular la matriz de pesos y matriz de caminos que nos permiten identificar el camino más corto entre los vértices 1 y 6 y su peso con la restricción de no contener los vértices 3 y 5. Identifica dicho camino y su peso. Para ello tendrás que hacer antes una reordenación de los vértices y matriz de pesos, y decir en qué iteración de Floyd y Warshall tienes que parar.

Caminos más cortos y árboles generadores de peso mínimo

Problema 7.4. Consideremos un grafo simple ponderado con conjunto de vértices $V = \{1, 2, 3, 4, 5, 6\}$, y cuya matriz de pesos es:

$$\begin{bmatrix} \infty & 2 & \infty & 3 & 8 & \infty \\ \infty & \infty & 1 & 2 & 2 & \infty \\ 1 & \infty & \infty & 3 & \infty & \infty \\ \infty & \infty & \infty & \infty & 3 & \infty \\ 1 & \infty & 7 & \infty & \infty & 1 \\ 3 & \infty & \infty & \infty & \infty & \infty \end{bmatrix}$$

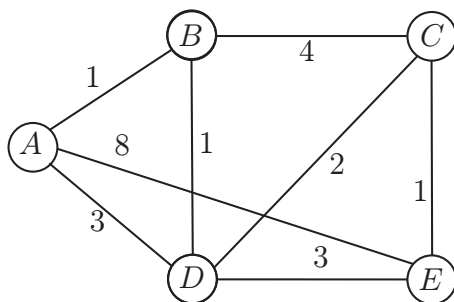
- (i) Usando MaGraDa, aplica el algoritmo de Floyd y Warshall y obtén, de forma razonada, el camino más corto entre los vértices 3 y 5 y su peso.

(Continuación)

Caminos más cortos y árboles generadores de peso mínimo

(ii) Con la ayuda de MaGraDa, utiliza de forma razonada el algoritmo de Floyd y Warshall, para calcular el camino más corto del vértice 3 al 6 con la condición de no utilizar los vértices 1 y 2 como internos. Para ello tendrás que hacer antes una reordenación de los vértices y matriz de pesos, y decir en qué iteración de Floyd y Warshall tienes que parar.

Problema 7.5. Una red informática conecta 5 puntos A, B, C, D y E tal y como indica el siguiente grafo ponderado, en donde los pesos asignados a las aristas representan el tiempo en milisegundos que se tarda en transmitir una palabra de un punto a otro.



Se sabe además, que los enlaces en los puntos B y D fallan en ocasiones, es decir, cuando este fallo se produce un mensaje no puede pasar por los puntos B y D .

Se desea saber qué ruta deben seguir los mensajes entre cada par de puntos para que el tiempo de transmisión sea mínimo, en los siguientes casos:

- Cuando los enlaces en los puntos B y D no fallan (ruta inicial).
- Cuando los enlaces en los puntos B y D fallan (ruta alternativa).

Caminos más cortos y árboles generadores de peso mínimo

Aplica el algoritmo que creas conveniente para resolver estas cuestiones. Expresa las soluciones generales en forma matricial, e indica razonadamente sólo la ruta inicial y la alternativa para la conexión de A a C.

(Continuación)

Caminos más cortos y árboles generadores de peso mínimo

(Continuación)

7.3 Árboles generadores de peso mínimo

Sea G un grafo ponderado y no dirigido. Diremos que T es un árbol generador de peso mínimo, si T es un árbol generador tal que la suma de los pesos asociados a sus aristas es mínima.

En esta sección vamos a ver cómo utilizar los algoritmos de Kruskal y Prim para obtener árboles generadores de peso mínimo. Este tipo de algoritmos nos permiten resolver problemas reales cuya solución sea la obtención de dicho árbol, como por ejemplo la construcción de un sistema de carreteras de coste mínimo que enlace una serie de ciudades, la creación de un sistema de interconexión de coste mínimo entre ordenadores o el problema de enviar un mensaje desde un punto de una red informática a los demás en un tiempo mínimo.

7.3.1 Algoritmo de Kruskal

Dado $G = (V, A)$, un grafo no dirigido con n vértices y con pesos w_i asociados a cada arista $e_i \in A$, $i = 1, 2, \dots, m$, el algoritmo de Kruskal, para la obtención de árboles generadores de peso mínimo, consiste en lo siguiente:

Caminos más cortos y árboles generadores de peso mínimo

Paso 1. $T = \emptyset$.

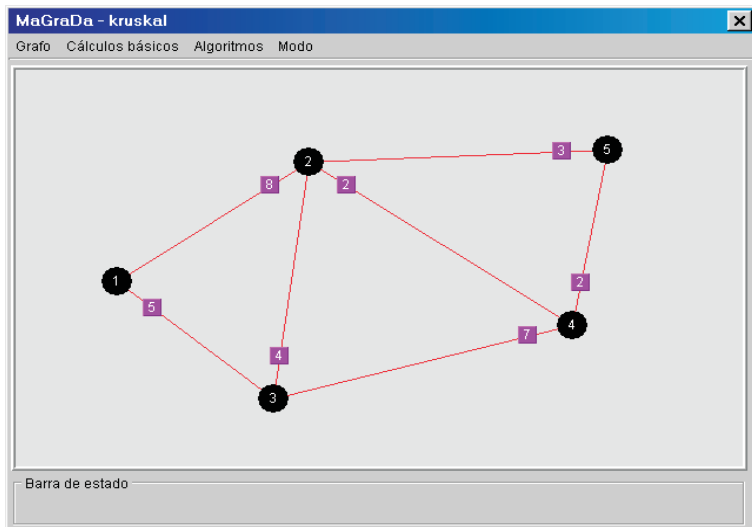
Paso 2. Ordenar en orden creciente las aristas de G atendiendo a sus pesos, es decir,

$$e_1, e_2, \dots, e_m / w_1 \leq w_2 \leq \dots \leq w_m.$$

Paso 3. Añadir aristas en T de forma ordenada siempre que no se formen ciclos hasta tener en T , $n - 1$ aristas.

MaGraDa resuelve este algoritmo desde la opción `Kruskal` del menú `Algoritmos`. La solución que nos da es clara. Nos dirá qué aristas del grafo son las que pertenecen al árbol generador de peso mínimo. Desde el modo de texto lo hace con dos pantallas. En una nos ofrece una relación de las aristas del grafo con sus pesos ordenados de menor a mayor y nos dice de cada una de ellas si la puede coger para el árbol o no, porque forma ciclo. En la otra pantalla nos dice claramente qué aristas va a coger y cuál es el peso del árbol obtenido. Desde el modo gráfico la primera pantalla es la misma pero la segunda presenta el árbol de forma gráfica resaltando las aristas en color blanco.

Así por ejemplo, para el siguiente grafo creado con MaGraDa,

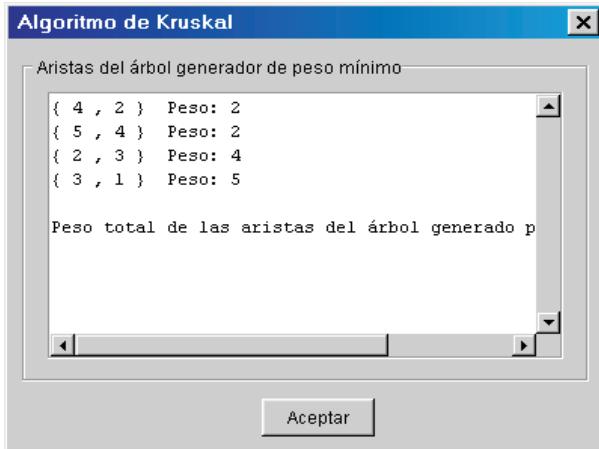


desde el modo de texto obtenemos las siguientes pantallas:

The image shows a window titled "Algoritmo de Kruskal" with a table listing the edges and their weights. The table has three columns: "Aristas", "Peso", and "Aristas en el árbol". The first row is highlighted in yellow.

Aristas	Peso	Aristas en el árbol
{ 4 , 2 }	2	*
{ 5 , 4 }	2	*
{ 2 , 5 }	3	FORMA CICLO
{ 2 , 3 }	4	*
{ 3 , 1 }	5	*
{ 3 , 4 }	7	FORMA CICLO
{ 1 , 2 }	8	FORMA CICLO

Below the table is a button labeled "Aceptar".



En la primera pantalla vemos que hay 3 aristas que formarían ciclo en el grafo, por tanto no las colocará en el árbol. En la segunda pantalla vemos las aristas definitivas que van a formar el árbol generador de peso mínimo.

7.3.2 Algoritmo de Prim

Sea G un grafo no dirigido ponderado con n vértices, entonces el algoritmo de Prim para la obtención de un árbol generador de peso mínimo consiste en realizar los siguientes pasos:

Paso 1. $T = \emptyset$, $U = \{v^*\}$, $v^* \in V(G)$,
 $L(u) = w_{uv^*}$ (∞ si \nexists arista) $\forall u \in V(G)$.

Paso 2. Encontrar $u^* \in V(G)$, tal que

$$L(u^*) = \min_{u \notin U} \{L(u)\}.$$

Paso 3. Añadir u^* a U , es decir, $U := U \cup \{u^*\}$.

Añadir la arista e incidente con u^* con peso $L(u^*)$ a T , es decir, $T := T \cup \{e\}$.

Paso 4. Si $\text{card}(U) = n$, STOP.

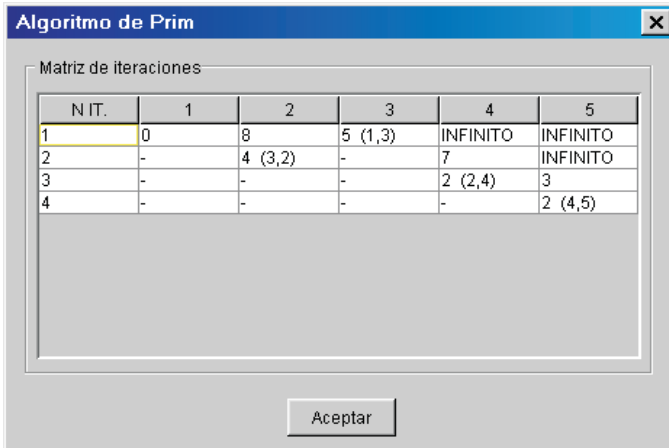
Si $\text{card}(U) < n$, hacer

$$L(u) := \min \{L(u), w_{u^*u}\}, \quad \forall u \notin U,$$

e ir al paso 2.

Para resolver este algoritmo mediante MaGraDa debemos seleccionar la opción Prim del menú Algoritmos. Tanto desde el modo de texto como desde el modo gráfico nos permite elegir el vértice a partir del cual vamos a construir el árbol generador de peso mínimo, es decir, el vértice v^* . La solución nos la presenta en dos pantallas. La primera, la matriz de iteraciones, resume los pasos del algoritmo mediante los que se va a obtener el árbol. Así, para nuestro ejemplo, partiendo del vértice 1 obtendríamos la siguiente pantalla:

Caminos más cortos y árboles generadores de peso mínimo



Matriz de iteraciones

N IT.	1	2	3	4	5
1	0	8	5 (1,3)	INFINITO	INFINITO
2	-	4 (3,2)	-	7	INFINITO
3	-	-	-	2 (2,4)	3
4	-	-	-	-	2 (4,5)

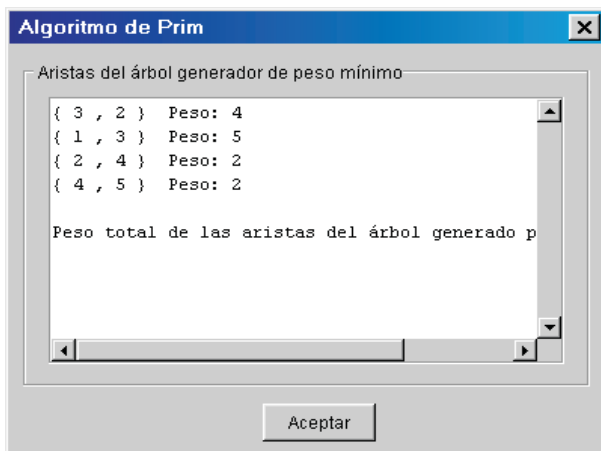
Aceptar

Notemos que esta pantalla tiene mucha similitud con la que se obtenía en el algoritmo de Dijkstra para la obtención de caminos más cortos. No en vano, los algoritmos de Dijkstra y Prim son muy similares. En cada iteración, cuando se alcanza el mínimo, se pone la arista que lo ha conseguido. Vamos a explicarlo de forma algo más detallada. Como hemos partido del vértice 1, en la primera iteración se obtiene $L(2) = w_{12} = 8$, $L(3) = w_{13} = 5$, y $L(4) = L(5) = \infty$. Claramente el mínimo se obtiene para $u^* = 3$. Por tanto, la primera arista que debemos introducir en el árbol es la arista $\{1, 3\}$. Tenemos entonces, $T = \{\{1, 3\}\}$ y $U = \{1, 3\}$. Si ahora pasamos a la segunda iteración, obtenemos $L(2) = \min\{L(2), w_{32}\} = \min\{8, 4\} = 4$, $L(4) = \min\{L(4), w_{34}\} = \min\{\infty, 7\} = 7$ y

$L(5) = \min\{L(5), w_{35}\} = \min\{\infty, \infty\} = \infty$. Por tanto, en esta iteración, el mínimo se obtiene para $u^* = 2$ y la arista que debemos introducir en el árbol es la arista $\{3, 2\}$. Tenemos entonces, $T = \{\{1, 3\}, \{3, 2\}\}$ y $U = \{1, 3, 2\}$. Pasamos ahora a la tercera iteración y de forma análoga obtenemos $L(4) = \min\{L(4), w_{24}\} = \min\{7, 2\} = 2$ y $L(5) = \min\{L(5), w_{25}\} = \min\{\infty, 3\} = 3$. Por tanto, en esta iteración, el mínimo se obtiene para $u^* = 4$, y la arista que debemos introducir en el árbol es la arista $\{2, 4\}$. Tenemos entonces, $T = \{\{1, 3\}, \{3, 2\}, \{2, 4\}\}$ y $U = \{1, 3, 2, 4\}$. Por último, en la iteración cuarta se obtiene $L(5) = \min\{L(5), w_{45}\} = \min\{3, 2\} = 2$. Por tanto, en esta iteración, el mínimo se obtiene en el único vértice que nos queda $u^* = 5$, y la arista que debemos introducir en el árbol es la arista $\{4, 5\}$. Tenemos entonces, $T = \{\{1, 3\}, \{3, 2\}, \{2, 4\}, \{4, 5\}\}$ y $U = \{1, 3, 2, 4, 5\}$. Por tanto, se ha finalizado el proceso y en T aparecen las aristas que forman el árbol generador de peso mínimo.

Estas aristas que forman el árbol generador de peso mínimo se mostrarán luego de forma más clara desde el modo de texto en otra pantalla de la siguiente forma:

Caminos más cortos y árboles generadores de peso mínimo

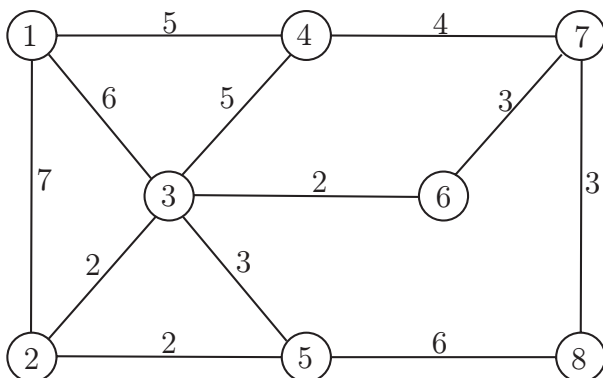


Si estamos en el modo gráfico presentará dicho árbol de forma gráfica.

7.3.3 Práctica 15

A continuación presentamos una serie de ejercicios relativos a la obtención de árboles generadores de peso mínimo.

Problema 7.6. *Se desea construir una red de ordenadores acoplada sin precisión para un sistema de ocho ordenadores. El grafo de la siguiente figura muestra la situación:*



Los ordenadores están representados por los vértices del grafo; las aristas representan líneas de transmisión a considerar para conectar cierto par de ordenadores. El peso asociado a cada arista indica el coste proyectado para construir esa línea de transmisión específica. El objetivo es conectar todos los ordenadores minimizando el coste total de la construcción.

Caminos más cortos y árboles generadores de peso mínimo

(i) Explica razonadamente qué problema de grafos debes resolver para obtener dicha conexión.

(ii) Resuelve el problema aplicando razonadamente a este grafo el algoritmo de Kruskal y ayudándote de MaGraDa.

(iii) *Resuelve el problema aplicando razonadamente a este grafo el algoritmo de Prim y ayudándote de MaGraDa.*

Caminos más cortos y árboles generadores de peso mínimo

Problema 7.7. *La siguiente tabla informa de la distancia en kilómetros entre cada dos ciudades de una serie de ciudades de España.*

	<i>Granada</i>	<i>Albacete</i>	<i>Alicante</i>	<i>Madrid</i>	<i>Valencia</i>
<i>Albacete</i>	363	-	-	-	-
<i>Alicante</i>	353	171	-	-	-
<i>Madrid</i>	434	251	422	-	-
<i>Valencia</i>	519	191	166	352	-
<i>Barcelona</i>	858	540	515	621	349

Se va a construir un sistema de carreteras que comunique estas seis ciudades y se desea determinar qué carreteras deberán construirse para que el coste de construcción sea mínimo. Se ha supuesto que el coste de construcción de un kilómetro de carretera es el mismo entre dos ciudades cualesquiera.

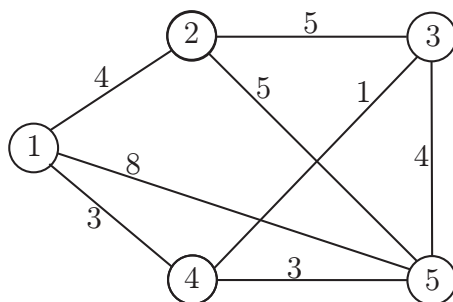
- (i) *Resuelve el problema mediante el algoritmo de Prim, de forma razonada y ayudándote de MaGraDa.*

(Continuación)

Caminos más cortos y árboles generadores de peso mínimo

(ii) Utiliza MaGraDa y el algoritmo de Kruskal para resolver el problema con la condición de que incluya una carretera que una de forma directa Madrid con Valencia.

Problema 7.8. Se desea establecer una red informática que conecte 5 puntos a_1, a_2, a_3, a_4 y a_5 . Las posibilidades de conexión vienen dadas en el siguiente grafo ponderado, en donde los pesos asignados a las aristas representan el coste de construcción de la línea directa correspondiente.



- (i) Usa el algoritmo de Prim, de forma razonada y ayudándote de MaGraDa, para determinar qué líneas deben construirse para que el coste total sea mínimo.

Caminos más cortos y árboles generadores de peso mínimo

(Continuación)

(ii) *Por otro lado se prevé que el tráfico entre los puntos a_3 y a_5 sea muy intenso, por lo que se desea saber qué líneas deben construirse de manera que exista una comunicación directa entre a_3 y a_5 y con coste mínimo. Modifica el algoritmo de Prim (explicando dicha modificación) y aplícalo para resolver esta cuestión. Ayúdate de MaGraDa para comprobar tu solución.*

Caminos más cortos y árboles generadores de peso mínimo

(Continuación)

Bibliografía

- [1] Manuel Abellanas y Dolores Lodaes. *Análisis de algoritmos y teoría de grafos*. Ra-ma, 1990.
- [2] Manuel Abellanas y Dolores Lodaes. *Matemática discreta*. Ra-ma, 1990.
- [3] Norman L. Biggs. *Matemática discreta*. Vicens Vives, 1994.
- [4] John A. Bondy y Uppaluri S. Murty. *Graph theory with applications*. Mc Millan Press, 1976.
- [5] Nicos Christofides. *Graph theory. An algorithmic approach*. Academic Press, 1975.
- [6] Pedro M. Cuenca. *Programación en Java*. Anaya, 1998.
- [7] Paul F. Dierker y William L. Voxman. *Discrete mathematics*. HBJ, 1986.

Bibliografía

- [8] Shimon Even. *Graph algorithms*. Computer Science Press, 1979.
- [9] Agustín Froufe. *JAVA 2: Manual de usuario y tutorial*. Ra-Ma, Segunda edición, 2000. Disponible en <http://usuarios.tripod.es/froufe/>.
- [10] Ronald Gould. *Graph theory*. The Benjamin/Cummings Publishing Company, INC., 1988.
- [11] Ralph P. Grimaldi. *Matemáticas discreta y combinatoria. Una introducción con aplicaciones*. Tercera edición. Addison-Wesley Longman, 1998.
- [12] Jonathan Gross y Jay Yellen. *Graph theory and its applications*. CRC Press, 1999.
- [13] Richard Johnsonbaugh. *Matemáticas discretas*. Grupo Editorial Iberoamericana, 1988.
- [14] Bernard Kolman y Robert Busby. *Estructuras de matemática discreta para la computación*. Prentice Hall Hispanoamericana, 1986.
- [15] Seymour Lipschutz. *Matemática discreta*. McGraw-Hill, 1990.
- [16] Russell Merris. *Graph theory*. Wiley-Interscience, 2000.

- [17] Sun Microsystems. The Java Tutorial. Tutorial on line, disponible en <http://java.sun.com/docs/books/tutorial/>.
- [18] Mike Morgan. *Descubre Java 1.2*. Prentice Hall, 1998.
- [19] Joe L. Mott, Abraham Kandel, y Theodore P. Baker. *Discrete mathematics for computer scientists and mathematicians*. Prentice-Hall, 1986.
- [20] J. Wilson Robin. *Introduction to graph theory*. Longman, 1985.
- [21] Kenneth A. Ross y Charles R. Wright. *Matemáticas discretas*. Prentice Hall, 1990.
- [22] Srikanta M.N. Swamy y K. Thulasiraman. *Graphs, networks and algorithms*. John Wiley, 1981.
- [23] Jean P. Tremblay y Ram Manohar. *Discrete mathematical structures with applications to computer Science*. MacGraw-Hill, 1975.