Capítulo 3
Búsqueda de rutas

1. Presentación del capítulo

Muchos dominios se enfrentan a un problema de búsqueda de rutas, denominada *pathfinding* en inglés. Recordamos, en primer lugar, los GPS y los programas de búsqueda de itinerarios (en coche, en tren, en transporte público...), también los videojuegos en los que los enemigos deben alcanzar al jugador por la ruta más corta.

La búsqueda de rutas es, en realidad, un dominio más bien vasto. En efecto, muchos problemas pueden representarse bajo la forma de un grafo, como la sucesión de movimientos en un juego de ajedrez.

La búsqueda de una ruta puede verse, en este caso, como la búsqueda de la serie de movimientos que se deben realizar para ganar.

Este capítulo presenta, en primer lugar, los distintos conceptos de la teoría de grafos y las definiciones asociadas. A continuación, se presentan los algoritmos fundamentales con su funcionamiento y sus restricciones.

Más adelante, se exponen los principales dominios en los que se puede utilizar esta búsqueda de rutas y se presenta un ejemplo de implementación de estos algoritmos en C#, aplicado a una búsqueda de rutas en un entorno en 2D.

El capítulo termina con un resumen.

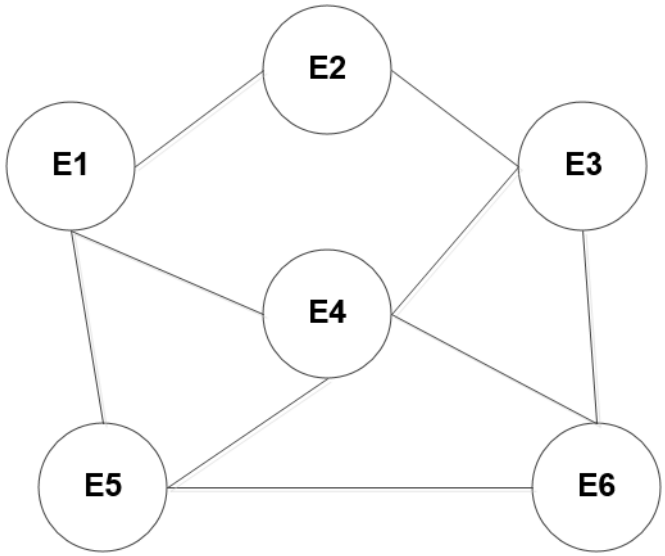
2. Rutas y grafos

Una ruta puede verse como un recorrido por un grafo. Los principales algoritmos se basan, por lo tanto, en la **teoría de grafos**.

2.1 Definición y conceptos

Un **grafo** es un conjunto de **nodos** o **vértices** (que pueden representar, por ejemplo, ciudades) ligados por **arcos**, que serán las rutas.

He aquí un grafo que representa las estaciones y los vínculos que existen entre ellas (en tren, sin trasbordo):



Las estaciones E1 a E6 son los nodos. El arco que va de E5 a E6 indica la presencia de un enlace directo entre estas dos estaciones. Se escribe (E5, E6) o (E6, E5) según el sentido deseado.

Por el contrario, para ir de E1 a E6 no existe ningún enlace directo. Será preciso pasar por E4 o por E5 si se desea realizar un único trasbordo, o por E2 y, a continuación, por E3 con dos trasbordos.

Una **ruta** permite alcanzar varios destinos unidos entre sí mediante arcos. De este modo, E1-E2-E3-E6 es una ruta de **distancia** 3 (la distancia es el número de arcos seguidos).

Hablamos de **circuito** cuando es posible partir de un nodo y volver a él. Aquí, el grafo contiene varios circuitos, como por ejemplo E1-E4-E5-E1 o E4-E5-E6-E4.

El **orden** de un grafo se corresponde con el número de destinos que contiene. Nuestro ejemplo contiene 6 estaciones, se trata por tanto de un grafo de orden 6.

Dos nodos se dice que son **adyacentes** (o vecinos) si existe un vínculo que permite ir de uno al otro. E5 es, por lo tanto, adyacente a E1, E4 y E6.

2.2 Representaciones

2.2.1 Representación gráfica

Existen varias maneras de representar un grafo. La primera es la **representación gráfica**, como hemos visto antes.

El orden y la situación de los nodos no son importantes. No obstante, se va a intentar buscar situar siempre los destinos de forma que el grafo sea lo más legible posible.

Se dice que el grafo está **orientado** si los arcos tienen un sentido, representando por ejemplo calles de sentido único en una ciudad. Si todos los arcos pueden tomarse en ambos sentidos, se dice que el grafo está **no orientado**, como es generalmente el caso en los grafos utilizados en la búsqueda de rutas.

2.2.2 Matriz de adyacencia

Las representaciones gráficas no siempre son prácticas, en particular cuando se trata de aplicar algoritmos o escribirlos en un ordenador.

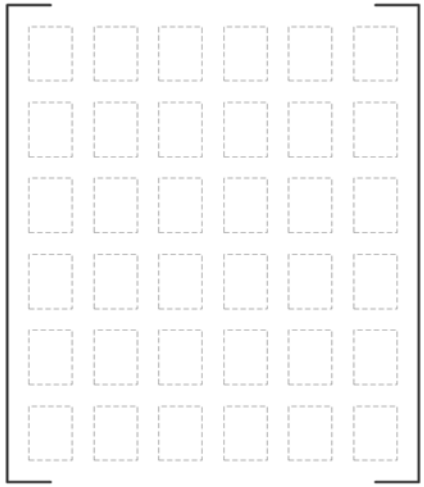
Se prefiere, a menudo, utilizar una matriz, llamada **matriz de adyacencia**.

Observación

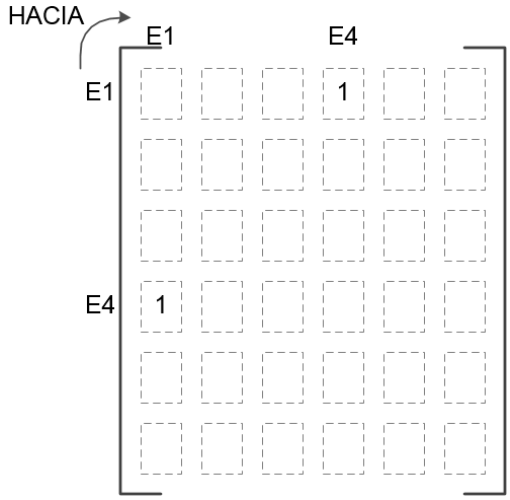
Una matriz es una estructura matemática particular que puede verse simplemente como una tabla de dos dimensiones.

En esta matriz, la ausencia de un arco se representa por un 0, y su presencia mediante un 1.

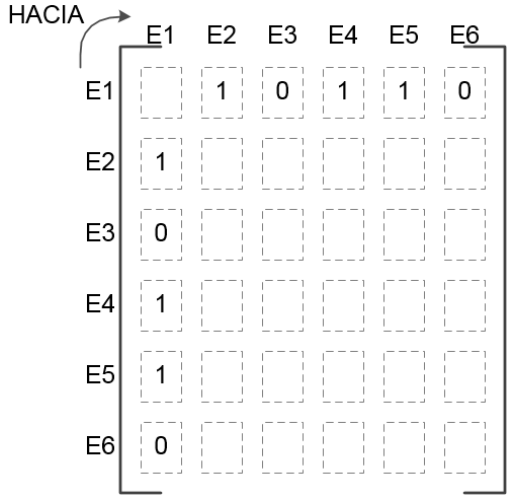
En el ejemplo de las estaciones, tenemos una matriz de 6 por 6 (puesto que existen 6 estaciones):



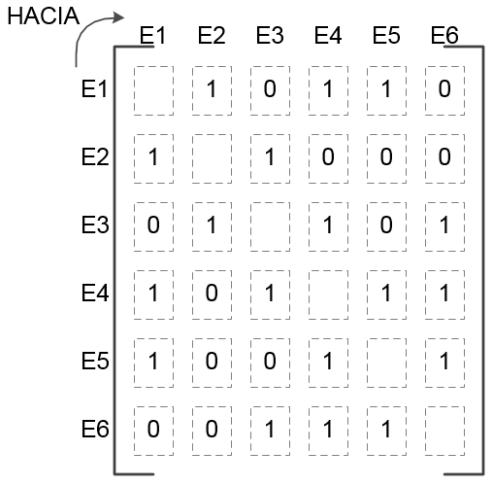
Vemos en el grafo que existe un vínculo entre E1 y E4. La casilla correspondiente al trayecto de E1 a E4 contiene, por lo tanto, un 1, igual que la de E4 a E1 (el trayecto es de doble sentido). Obtenemos la siguiente matriz:



Del mismo modo, existe un arco desde E1 hacia E2 y E5, pero no hacia E3 ni E6. Podemos completar nuestra matriz:

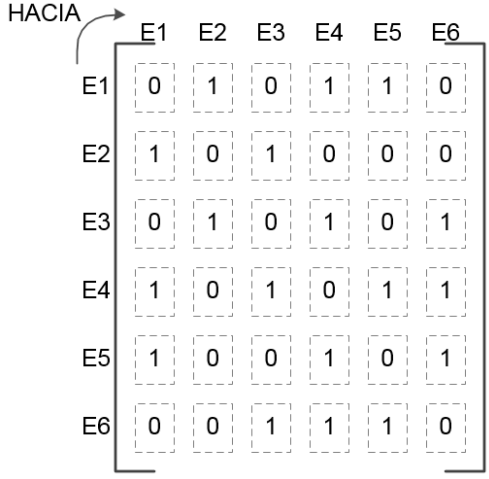


Hacemos lo mismo para los demás nodos y los demás arcos:



HACIA	E1	E2	E3	E4	E5	E6
E1		1	0	1	1	0
E2	1		1	0	0	0
E3	0	1		1	0	1
E4	1	0	1		1	1
E5	1	0	0	1		1
E6	0	0	1	1	1	

Solo nos queda la diagonal. Representa la posibilidad de ir desde un nodo hacia sí mismo, lo cual se denomina un bucle. En este caso, no existe ningún trayecto directo que permita ir de una estación a sí misma, de modo que rellenamos con 0 esta diagonal.



HACIA	E1	E2	E3	E4	E5	E6
E1	0	1	0	1	1	0
E2	1	0	1	0	0	0
E3	0	1	0	1	0	1
E4	1	0	1	0	1	1
E5	1	0	0	1	0	1
E6	0	0	1	1	1	0

La matriz de adyacencia está, ahora, completa.