Capítulo 2

Introducción a la Teoría de Grafos

2.1. Conceptos Fundamentales de Grafos

Partiremos nuestro estudio un par de ejemplos que sugerirán una definicón para lo que es un *grafo* y motivarán el tipo de aplicaciones para los que se utilizan. El primero que veremos se suele citar como el que dio inicio a la teoría de grafos.

Ejemplo: Los Puentes de Königsberg. La ciudad de Königsberg (hoy conocida como Kaliningrado) estaba localizada en el este de Prussia. La ciudad tenía una isla que formaba el río Pregel al cruzarla, y antes de dejar la ciudad el río se bifurcaba dando paso a dos causes distintos. Las regiones formadas por el río estaban unidas con siete puentes. Un diagrama simplificado de la ciudad puede verse en la figura 2.1.

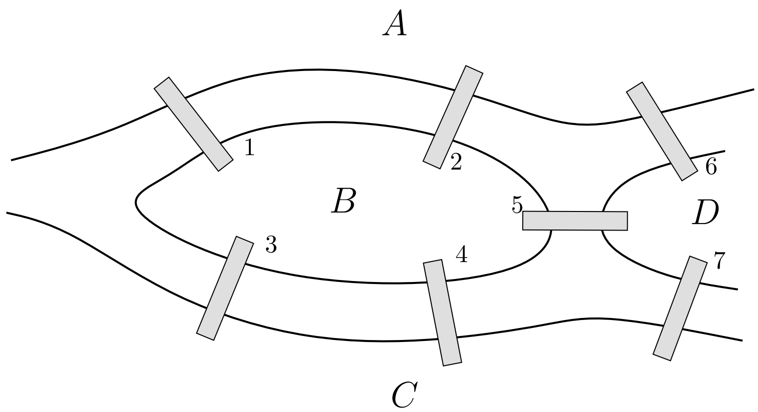


Figura 2.1: Diagrama de la ciudad de Königsberg.

Los habitantes de Königsberg se preguntaban si existía alguna forma de salir de casa, recorrer la ciudad pasando por todos los puentes una vez por cada uno, y regresar a casa. En la figura 2.2 se ha hecho una representación simplificada de la ciudad. Cada punto representa una de las regiones, y cada trazo a un puente. El problema puede reducirse entonces al de dibujar la figura 2.2 sin levantar el lápiz y sin repetir ningún trazo, partiendo desde uno de los puntos y volviendo al inicial.

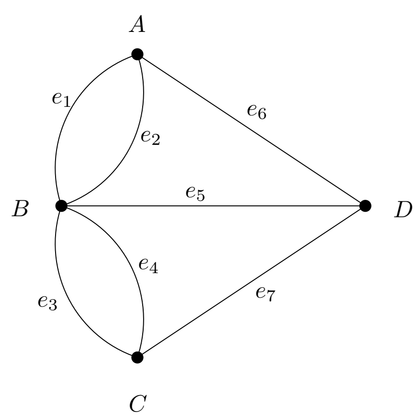


Figura 2.2: Representación simplificada de la ciudad de Königsberg.

Con esta reformulación no es difícil argumentar que la respuesta al problema de los puentes es no. Lo primero es notar que para dibujar la figura 2.2 debemos para cada punto que no es el incial, "entrar" por un trazo y "salir" por otro trazo (distinto). Si notamos en la figura el punto D por ejemplo, tiene tres trazos "incidiendo" en él, por lo que después de que se pase por D una vez (se "entre y salga" de D), la próxima vez que se llegue a D no se podrá salir. Lo mismo pasa con los puntos A y C. El punto B es un poco diferente, dado que tiene 5 trazos, se podrá "entrar y salir" dos veces, cuando se llegue por tercera vez a B ya no se podrá salir. El problema entonces surge porque los puntos tienen una cantidad impar de trazos. Dado que el problema exige que el punto inicial sea igual al punto final, se puede concluir, por la misma razón, que tampoco es posible partir de ninguno de estos puntos ya que el trazo por el que se sale inicialmente de un punto debe ser distinto al con el que se llega finalmente.

El problema entonces tiene que ver con la paridad de los trazos de cada punto. Basta con que uno de los puntos tenga una cantidad impar de trazos para que la figura no se pueda dibujar siguiendo las reglas pedidas. Finalmente es imposible recorrer la ciudad completa de Königsberg pasando por todos los punetes y volver a casa. El primero que dio esta respuesta fue el matemático suizo L. Euler(1707–1783) en el año 1735.

Ejemplo: El ejemplo anterior era un poco radical porque todos sus puntos tenían una cantidad impar de trazos. La figura 2.3 tampoco se puede dibujar sin repetir trazos y volviendo al punto de partida.

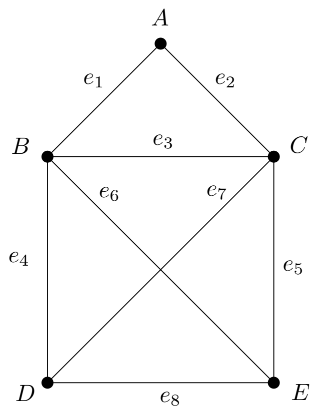


Figura 2.3: Figura que tampoco se puede dibujar cumpliendo las reglas.

La razón es la misma que antes, existen puntos con cantidad impar de trazos, en este caso los puntos D y E tienen ambos tres trazos. Basta con que uno de los puntos de la figura tenga una cantidad impar de trazos para que esta no se pueda dibujar siguiendo las restricciones. ¿Qué pasa si una figura tiene todos sus puntos con una cantidad par de trazos? En este caso nuestra argumentación inicial no sería aplicable si quisiéramos mostrar que no se puede dibujar. El resultado interesante que veremos más adelante, no dirá que para que una figura se pueda dibujar siguiendo las restricciones, simplemente basta con que todos sus puntos tengan una cantidad par de trazos. De allí se concluirá que una figura se puede dibujar sin repetir trazos y volviendo al punto de partida, si y sólo si cada punto tiene una cantidad par de trazos.

Los anteriores ejemplo motivan nuestra definición de grafo.

Def: Un **grafo** G está compuesto por un conjunto de **vértices** que llamaremos $V(G)$, un conjunto de **aristas** que llamaremos $E(G)$, y una relación que a cada arista $e \in E(G)$ le asigna un par de vértices no necesariamente distintos de $V(G)$.

Para representar un grafo se usan puntos para dibujar vértices y trazos para dibujar aristas, cada arista se dibuja como un trazo entre los vértices con los que se encuentra relacionada.

Ejemplo: En ejemplo de la figura 2.2 el conjunto de vértices es $\{A, B, C, D\}$ y el conjunto de aristas es $\{e_1, e_2, e_3, e_4, e_5, e_6, e_7\}$. La asignación entre aristas y vértices se puede obtener de la figura, por ejemplo, la arista e_5 está relacionada con los vértices B y D.

En ejemplo de la figura 2.3 el conjunto de vértices es $\{A, B, C, D, E\}$ y el conjunto de aristas es $\{e_1, e_2, e_3, e_4,$

$e_5, e_6, e_7, e_8\}$. La asignación entre aristas y vértices se puede obtener de la figura, por ejemplo, la arista e_5 está relacionada con los vértices C y E.

Una diferencia importante entre los grafos de las figuras 2.2 y 2.3, es que en el segundo, cada arista está relacionada con un par de vértices distintos. Nuestra definición de grafo también permite por ejemplo que una arista esté relacionada con un par de vértices iguales. En la figura 2.4 se muestra un grafo con aristas de este tipo. En este grafo el conjunto de vértices es $\{A, B, C, D\}$ y el conjunto de aristas es $\{e_1, e_2, e_3, e_4, e_5, e_6\}$. La

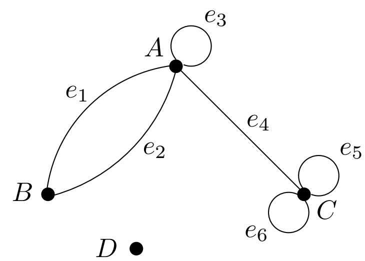


Figura 2.4: Grafo con rulos y aristas paralelas.

arista e_6 por ejemplo está relacionada con el vértice C (o con el par de vértices C y C). Otra cosa interesante de este grafo es que ninguna arista está relacionada con el vértice D, esto para nada escapa de nuestra definición.

Def: Un **rulo** en un grafo, es una arista que está relacionada con sólo un vértice. Dos aristas son **paralelas** si sus pares de vértices relacionados son iguales.

Un grafo de dice **simple** si no tiene rulos ni aristas paralelas. El grafo de la figura 2.3 es simple, mientras que los de las figuras 2.2 y 2.4 no lo son.

Una arista en un grafo simple puede verse como un par no ordenado de vértices. Si la arista e está relacionada con los vértices u y v, escribiremos $e = uv$ o $e = vu$. Así podremos decir que un grafo simple es un par $G = (V(G), E(G))$ donde los elementos de $E(G)$ son pares no ordenados de elementos de $V(G)$. En el grafo de la figura 2.3, podemos decir por ejemplo que $e_4 = BD$ y que $e_7 = CD$, luego el grafo es $G = (V(G), E(G))$ con $V(G) = \{A, B, C, D, E\}$ y $E(G) = \{AB, AC, BC, BD, CE, BE, CD, DE\}$. En un grafo simple entonces no es necesario que las aristas tengan nombre, basta identificar al par de vértices que relacionan. Cuando en un grafo simple G exista una arista $uv \in E(G)$ diremos que u y v son **vecinos** o vértices **adyacentes**, así por ejemplo en el grafo de la figura 2.3, A y B son vértices vecinos, al igual que D y E.

Nosotros estudiaremos principalmente grafos simples. A menos que se explicite otra cosa, cuando nos refiramos a un grafo nos estaremos refiriendo a un grafo simple con un conjunto no vacío finito de vértices.

2.1.1. Isomorfismos y Clases de Grafos

¿Cuando dos grafos son estructuralmente equivalentes? Por ejemplo, ¿cuál es la diferencia entre los dos grafos de la figura 2.5? Ciertamente los dibujos se ven distintos, sin embargo comparten algunas cosas como que ambos tienen la misma cantidad de vértices y la misma cantidad de aristas. ¿Pero será que se ven distintos simplemente por la forma en que lo dibujamos? ¿Podremos dibujarlos de manera que se "vean" iguales? La respuesta es sí. En la figura 2.6 se muestra como se pueden "mover" los vértices de G_1 de manera que se "vea" igual a G_2. Lo que estamos haciendo es simplemente "llevando" v_3 a la posición que ocupa w_1 y v_4 a

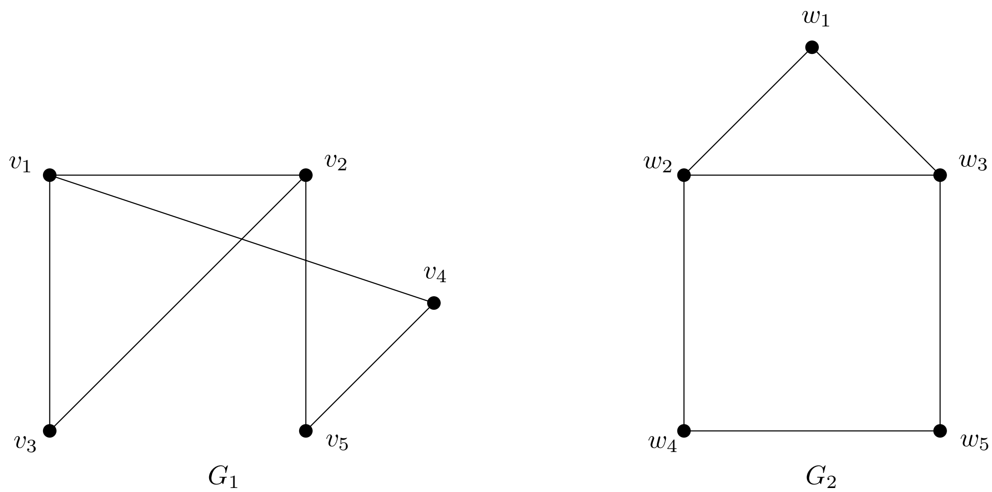


Figura 2.5: ¿Cuál es la diferencia entre G_1 y G_2?

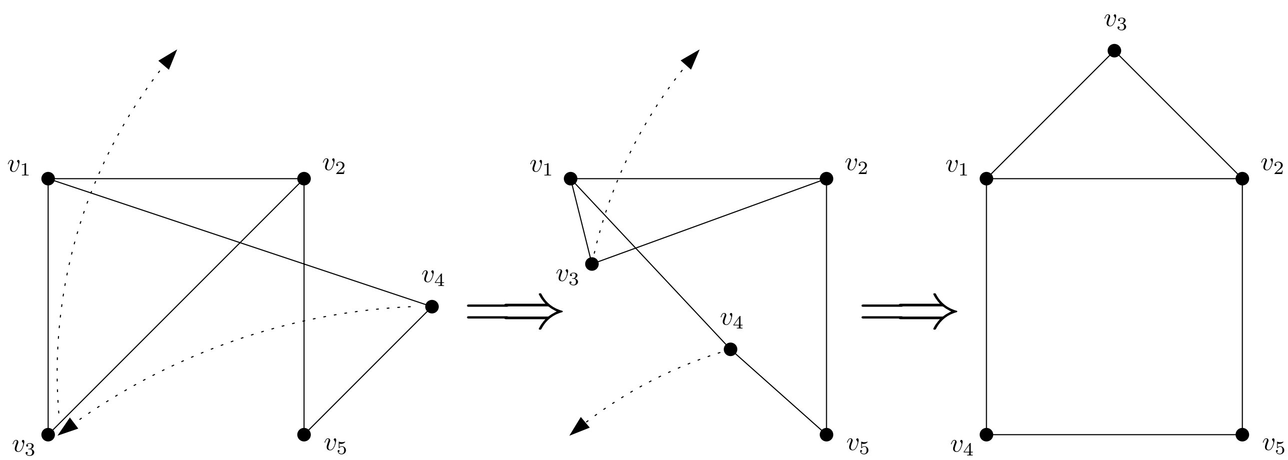


Figura 2.6: Transformación de G_1.

la posición que ocupa w_4. Si ahora hacemos un "renombre" de los vértices de G_1 siguiendo la siguiente regla:

$$\begin{array}{ccc} v_1 & \rightarrow & w_2 \\ v_2 & \rightarrow & w_3 \\ v_3 & \rightarrow & w_1 \\ v_4 & \rightarrow & w_4 \\ v_5 & \rightarrow & w_5 \end{array}$$

obtenemos exactamente a G_2. Esto motiva nuestra definición de equivalencia entre grafos que llamaremos **isomorfismo**.

Def: Dos grafos G_1 y G_2 se dicen **isomorfos** si existe una función biyectiva f desde $V(G_1)$ a $V(G_2)$, $f : V(G_1) \rightarrow V(G_2)$, tal que si $uv \in E(G_1)$ entonces $f(u)f(v) \in E(G_2)$, o sea, si hay una arista entre el par de vértices u y v en G_1, entonces hay una arista entre sus imágenes $f(u)$ y $f(v)$ en G_2. Cuando se cumplan estas condiciones, diremos que f es un **isomorfismo** entre G_1 y G_2. Escribiremos $G_1 \cong G_2$ cuando G_1 y G_2 sean isomorfos. No es difícil notar que $\cong$ es una relación de equivalencia entre grafos.

Ejemplo: Los grafos G_1 y G_2 de la figura 2.5 son isomorfos. Para demostrarlo basta encontrar una función f biyectiva que cumpla con ser un isomorfismo entre G_1 y G_2. La función f es la que ya detallamos:

$$\begin{array}{ccc} & f & \\ v_1 & \rightarrow & f(v_1) = w_2 \\ v_2 & \rightarrow & f(v_2) = w_3 \\ v_3 & \rightarrow & f(v_3) = w_1 \\ v_4 & \rightarrow & f(v_4) = w_4 \\ v_5 & \rightarrow & f(v_5) = w_5 \end{array}$$

Primero f es claramente biyectiva. Ahora debemos comprobar que efectivamente es un isomorfismo, para esto debemos chequear que para cada par de vértices que forman una arista en G_1, sus imágenes también forman una arista en G_2.

$$\begin{array}{ll} v_1v_2 \in E(G_1), & f(v_1)f(v_2) = w_2w_3 \in E(G_2) \\ v_1v_3 \in E(G_1), & f(v_1)f(v_3) = w_2w_1 \in E(G_2) \\ v_1v_4 \in E(G_1), & f(v_1)f(v_4) = w_2w_4 \in E(G_2) \\ v_2v_3 \in E(G_1), & f(v_2)f(v_3) = w_3w_1 \in E(G_2) \\ v_2v_5 \in E(G_1), & f(v_2)f(v_5) = w_3w_5 \in E(G_2) \\ v_5v_4 \in E(G_1), & f(v_5)f(v_4) = w_5w_4 \in E(G_2) \end{array}$$

Finalmente f es un isomorfismo de donde concluimos que $G_1 \cong G_2$.

Más adelante veremos técnicas que nos ayudarán a determinar cuándo dos grafos no son isomorfos, por ahora el alumno puede notar que por ejemplo, una condición necesaria (pero no suficiente) para que dos grafos sean isomorfos es que tengan la misma cantidad de vértices y la misma cantidad de aristas.

Otro punto interesante del isomorfismo de grafos y que tiene que ver con computación, es que hasta el día de hoy, nadie ha podido encontrar un algoritmo "eficiente" para determinar si dos grafos cualquiera son o no isomorfos. Volveremos a este punto cuando en el siguiente capítulo definamos la noción de eficiencia de un algoritmo.

La relación $\cong$ es una relación de equivalencia, como tal define clases de equivalencias sobre el conjunto de los grafos. Estudiaremos algunas de estas clases de equivalencia y les daremos nombre.

2.1.2. Algunas Clases de Grafos

Comenzaremos con un par de definiciones.

Def: Un **camino** es un grafo simple cuyos vértices pueden *ordenarse en una linea* de manera tal que dos vértices son adyacentes si y sólo si son consecutivos en la lista. Un **ciclo** es un grafo simple cuyos vértices pueden disponerse *en círculo* de manera que dos vértices son adyacentes si y sólo si aparecen en posiciones consecutivas en un círculo. Un ejemplo de camino y ciclo se muestra en la figura 2.7

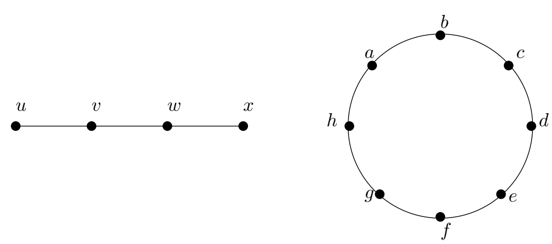
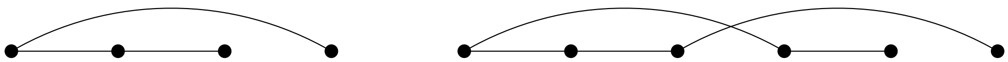


Figura 2.7: Un camino (izquierda) y un ciclo (derecha).

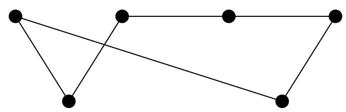
Def: La *clase de equivalencia* de todos los caminos con n vértices la llamaremos P_n. La *clase de equivalencia* de todos los ciclos con n vértices la llamaremos C_n. En general en vez de hablar de *clase de equivalencia* de grafos, simplemente hablaremos de un grafo particular representante de esta clase, tal que al dibujarlo no nombraremos sus vértices. Siguiendo esta norma, en la figura 2.8 aparecen P_4 y P_6. En ella P_4 y



P_4 P_6

Figura 2.8: Clases de equivalencia para el camino de 4 y 6 vértices.

P_6 se han dibujado a propósito en una disposición no lineal, para enfatizar que lo que importa es su estructura más que el dibujo. En la figura 2.9 aparece C_6.

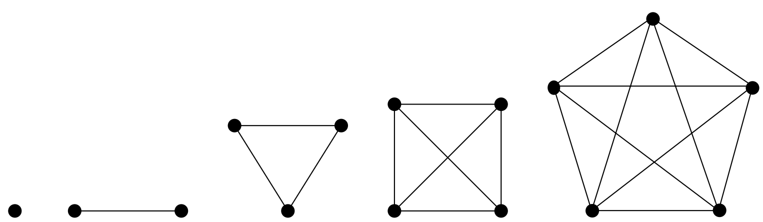


C_6

Figura 2.9: Ejemplo del ciclo con 6 vértices.

Otra clase de grafos importantes es el grafo completo.

Def: Un **grafo completo** es un grafo simple en el que todos los pares de vértices son adyacentes. Al grafo completo de n vértices le llamaremos K_n. En la figura 2.10 se pueden ver a los grafos K_n para $n = 1, 2, 3, 4, 5$.



K_1 K_2 K_3 K_4 K_5

Figura 2.10: Grafos completos.

Def: Un grafo G se dice **bipartito** si $V(G)$ se puede agrupar en dos conjuntos disjuntos V_1 y V_2, $V_1 \cap V_2 = \phi$, $V_1 \cup V_2 = V(G)$, tal que toda arista en $E(G)$ une a un vértice de V_1 con uno de V_2. Esto quiere decir que dos vértices de V_1 no pueden ser adyacentes, lo mismo con V_2.

Ejemplo: El grafo G de la figura 2.11 es un grafo bipartito. El conjunto de vértices de G es $V(G) = \{t, u, v, w, x, y, z\}$, que se puede separar en los conjuntos $V_1 = \{t, u, v, w\}$ y $V_2 = \{x, y, z\}$ tal que toda arista en $E(G)$ une a un vértice de V_1 con uno de V_2. En general, cuando dibujemos un grafo bipartito haremos una clara separación entre las particiones de los vértices (V_1 y V_2) dibujando los vértices de una de las particiones "arriba" de los vértices de la otra partición. En la figura 2.12 se ha seguido esta norma para dibujar nuevamente a G.

Ejemplo: Los grafos bipartitos generalmente se usan para modelar problemas de *asignación* de recursos o tareas. Podemos suponer que hay vértices de un grafo representando personas y tareas, y que un vértice p correspondiente a una persona es adyacente a un vértice t correspondiente a una tarea si es que la persona p

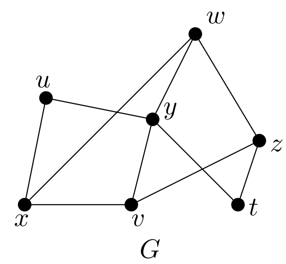


Figura 2.11: Ejemplo de un grafo bipartito

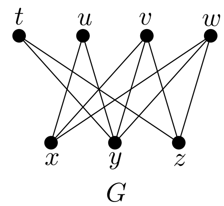


Figura 2.12: El mismo grafo bipartito haciendo una clara diferencia en las particiones.

está capacitada para realizar la tarea t. Un grafo de estas características siempre será bipartito. Un ejemplo se ve en la figura 2.13. Una pregunta que se puede hacer sobre este tipo de grafos es si existe alguna forma de asignar las tareas de manera tal que toda puedan ser realizadas simultáneamente. En el grafo de ejemplo esto

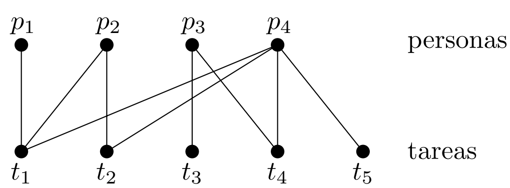


Figura 2.13: Un grafo para modelar un problema de asignación de tareas.

no es posible (¿por qué?). Más adelante en el curso veremos algunos resultado que nos permitirán establecer cuándo y cuándo no se puede hacer una asignación en grafos de este tipo.

Def: Un grafo **bipartito completo** es un grafo bipartito en que cada uno de los vértices de una de las particiones es adyacente con cada uno de los vértices de la otra partición. Cuando las particiones tengan n y m vértices, llamaremos $K_{n,m}$ al grafo bipartito completo. En la figura 2.14 se muestra un diagrama para $K_{2,3}$.

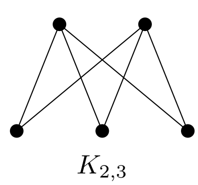


Figura 2.14: El grafo bipartito completo cuyas particiones tienen tamaño 2 y 3.

En los siguientes ejemplos estudiaremos dos grafos muy importantes en la teoría, el hipercubo y el grafo de Petersen.

Ejemplo: El Hipercubo. Generalmente se utilizan grafos para representar la *topología* de un conjunto de computadores (o procesadores) conectados por red. Esta red de computadores podrá ejecutar algoritmos *en paralelo* y la eficiencia de estos algoritmos muchas veces tiene que ver con la forma en la que los computadores se encuentran conectados para poder interactuar. Un tipo de red muy eficiente y para el cual existen una cantidad considerable de algoritmos paralelos, es la llamada red de **hipercubo**.

Un hipercubo n–dimensional, que llamaremos H_n, es un grafo simple en el que sus vértices han sido numerados en binario desde el 0 al $2^n - 1$, o sea cada vértice tiene un nombre compuesto por n bits. Una arista conecta a un par de vértices del hipercubo si estos difieren exactamente en un bit. Así por ejemplo, el hipercubo de 3 dimensiones, H_3, tiene 8 vértices, $V(H_3) = \{000, 001, 010, 011, 100, 101, 110, 111\}$ y es tal que hay una arista entre 010 y 110, entre 111 y 011, etc. Un diagrama de H_3 se puede ver en la figura 2.15.

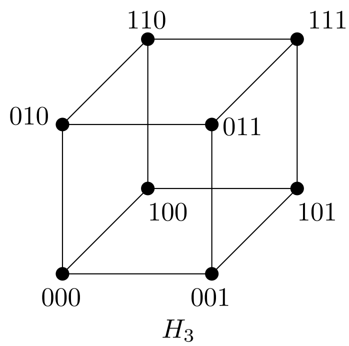


Figura 2.15: El hipercubo de 3 dimensiones.

Una primera observación es que H_n es un grafo bipartito para todo n. De hecho si tomamos V_1 como todos los vértices de $V(H_n)$ que tienen una cantidad par de $1's$ y V_2 como todos los vértices que tienen una cantidad impar de $1's$ es claro que toda arista en $E(H_n)$ unirá siempre a un vértice de V_1 con uno de V_2.

Otra observación importante es que un hipercubo n–dimensional puede ser creado en forma recursiva a partir de dos hipercubos de dimensión $n - 1$. Supongamos que tenemos dos hipercubos de dimensión $n - 1$, H' y H'', cada uno de ellos tiene 2^{n-1} vértices numerados desde el 0 al $2^{n-1} - 1$. Podemos crear un hipercubo n–dimensional a partir de H' y H'' añadiendo una arista entre cada par de vértices de H' y H'' que tienen el mismo número binario asignado y posteriormente cambiar los nombres de los vértices de $V(H')$ agregándoles un 0 al inicio, y los de $V(H'')$ agregándoles un 1 al inicio. El caso base de esta construcción es el hipercubo H_1 que tiene vértices 0 y 1, y una arsita uniéndolos. La figura 2.16 muestra la construcción de H_4 a partir de dos instancias de H_3.

Esta última forma de construcción nos da un manera de establecer algunas propiedades de conteo recursivas acerca de H_n. Por ejemplo podemos contar cuántas aristas tiene H_n a partir de establecer una fórmula recursiva para $|E(H_n)|$ en función de $|E(H_{n-1})|$, usando la construcción no es difícil obtener que:

$$\begin{array}{rcl} |E(H_n)| & = & 2^{n-1} + 2|E(H_{n-1})| \\ |E(H_1)| & = & 1 \end{array}$$

Como ejercicio el alumno podría obtener una fórmula explicita para $|E(H_n)|$ usando la recurrencia anterior.

Ejemplo: El Grafo de Petersen. El grafo de Petersen, que llamaremos Pet es un grafo simple cuyos vértices son los subconjuntos de dos elementos de $\{0, 1, 2, 3, 4\}$, por ejemplo $\{0, 1\}$, $\{2, 4\}$ y $\{1, 3\}$ son vértices en el grafo de Petersen. Dos vértices A y B del grafo de Petersen están unidos por una arista si ocurre que $A \cap B = \phi$. Formalmente podemos definir entonces al grafo de Petersen, como un grafo $G = (V(G), E(G))$

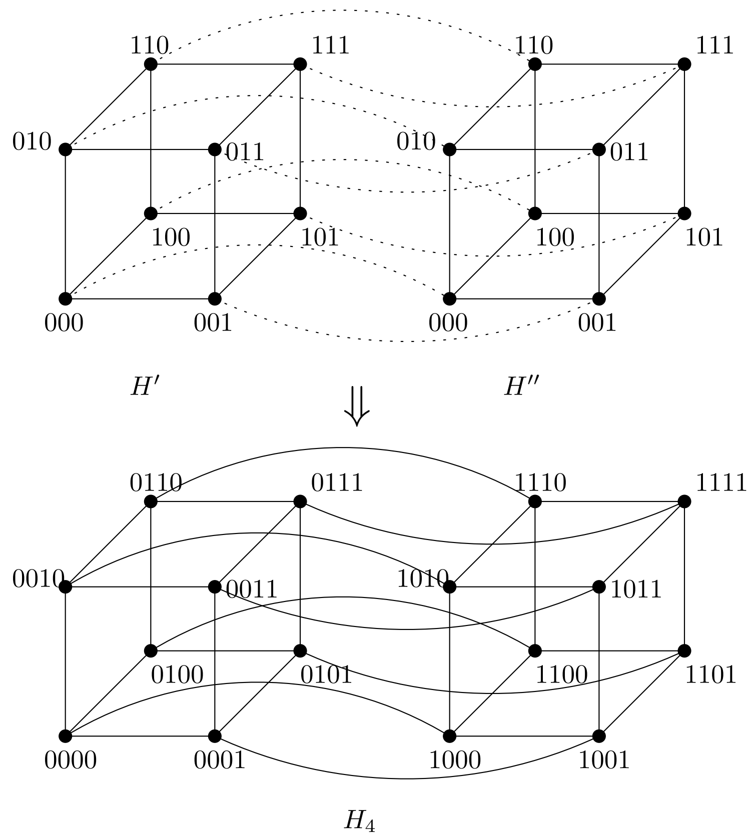


Figura 2.16: Construcción de H_4 a partir de dos copias de H_3.

tal que:

$$V(G) = \{A \mid A \subseteq \{0,1,2,3,4\} \wedge |A| = 2\}$$
$$E(G) = \{AB \mid A, B \in V(G) \wedge A \cap B = \phi\}$$

En la figura 2.17 se muestra tres formas distintas de dibujar al grafo de Petersen. Para mostrar que ellos son efectivamente isomorfos al grafo de Petersen, basta con nombrar sus vértices como subconjuntos de tamaño 2 de $\{0,1,2,3,4\}$ y verificar que sus aristas unen a los vértices correspondientes.

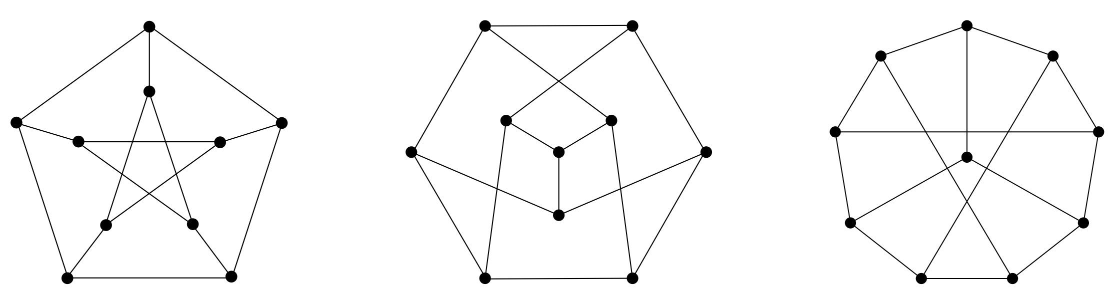


Figura 2.17: Tres formas de dibujar el grafo de Petersen.

Se pueden establecer varias propiedades interesantes del grafo de Petersen, sin necesidad de analizar sus dibujos, usando sólo su definición abstracta. Primero, el grafo tiene exactamente 10 vértices, que es la cantidad de subconjuntos de dos elementos que se pueden hacer sobre el conjunto $\{0,1,2,3,4\}$. Otra propiedad tiene que ver con la cantidad de vecinos que cada vértice tiene en el grafo de Petersen. Sea A un vértice cualquiera, sabemos que $|A| = 2$ y por lo tanto el conjunto $A' = \{0,1,2,3,4\} - A$ es tal que $|A'| = 3$. Cada vértice de A se obtendrá a partir de un subconjunto de tamaño 2 de A', como hay exactamente 3 de tales subconjuntos, A tiene 3 vecinos. Otra propiedad interesante es la siguiente, dados dos vértices no adyacentes en el grafo, ellos tienen exactamente un vecino en común. Esto se puede demostrar con el siguiente argumento, dados dos vértices distintos A y B, estos no son vecinos si ocurre que $A \cap B \neq \phi$, esto quiere decir que el conjunto $A \cup B$ tiene exactamente 3 elementos. Un vértice que sea simultáneamente vecino de A y B debe ser entonces un subconjunto de dos elementos del conjunto $\{0,1,2,3,4\} - (A \cup B)$, y dado que este conjunto tiene exactamente 2 elementos, él representa al único vecino común entre A y B.

Def: Dado un grafo $G = (V(G), E(G))$, diremos que $H = (V(H), E(H))$ es un **subgrafo** de G si $V(H) \subseteq V(G)$, $E(H) \subseteq E(G)$, y en $E(H)$ aparecen sólo aristas que unen a vértices de $V(H)$. Cuando H sea subgrafo de G, escribiremos $H \subseteq G$.

Un **clique** en un grafo G es un conjunto de vértices $K \subseteq V(G)$ en que para cada par de vértices $u, v \in K$, la arista $uv \in E(G)$. Un **conjunto independiente** en un grafo G es un conjunto de vértices $K \subseteq V(G)$ tal que para cada par de vértices $u, v \in K$, la arista $uv \notin E(G)$. El tamaño de un clique o de un conjunto independiente es la cantidad de vértices que lo componen.

Ejemplo: Para el grafo completo K_n, se cumple que $\forall i \leq n$, $K_i \subseteq K_n$. También se cumple que cualquier subconjunto de $V(K_n)$ es un clique en K_n. Los únicos conjuntos independientes en K_n son los conjuntos compuestos por un único vértice de $V(K_n)$.

Para el grafo bipartito completo $K_{n,m}$ con particiones V_1 y V_2, tanto V_1 como V_2 son conjuntos independientes. El clique más grande que se puede encontrar en $K_{n,m}$ está compuesto por dos vértices ¿por qué?. Podemos decir también que C_3 nunca es subgrafo de $K_{n,m}$, ¿puede ocurrir que $C_4 \subseteq K_{n,m}$? ¿puede ocurrir que $C_5 \subseteq K_{n,m}$?

Def: Dado un grafo $G = (V(G), E(G))$ definimos el **complemento** de G como el grafo $\overline{G} = (V(G), E(\overline{G}))$, en donde el conjunto de arista cumple con $uv \in E(\overline{G}) \Leftrightarrow uv \notin E(G)$, o sea $\overline{G}$ se obtiene a partir de los vértices de G agregando una arista entre cada par de vértices no vecinos en G. Un grafo G se dice **autocomplementario** si ocurre que $G \cong \overline{G}$.

El siguiente teorema nos da una relación entre cliques, conjuntos independientes y el grafo complemento.

Teorema 2.1.1: Dado un grafo $G = (V(G), E(G))$ y un subconjunto $V \subseteq V(G)$, entonces V es un clique en G si y sólo si V es un conjunto independiente en $V(\overline{G})$.

Demostración: Supongamos que $V \subseteq V(G)$ es un clique en G, esto quiere decir que para todo $u, v \in V$ ocurre que $uv \in E(G)$. Por la definición de $\overline{G}$, sabemos que para todo $u, v \in V$ ocurre que $uv \notin E(\overline{G})$ por lo tanto V es un conjunto independiente en $\overline{G}$. La implicación inversa se obtiene de manera similar. □

Ejemplo: En la figura 2.18 se muestran tres grafos y sus grafos complemento. En ella podemos ver que por ejemplo P_4 es autocomplementario, que P_5 no es autocomplementario, y que el complemento de K_4 es un grafo de 4 vértices *aislados*, ninguno es vecino de otro.

Ejemplo: Dado un conjunto de 6 personas ¿es cierto que siempre ocurre que hay, o bien tres personas que se conocen mutuamente, o bien tres personas que se desconocen mutuamente? La respuesta a esta pregunta es SI, y lo justificaremos modelando el problema con grafos.

Podemos usar un grafo G con 6 vértices $\{p_1, p_2, p_3, p_4, p_5, p_6\}$, cada uno representando a una persona, y agregar la arista p_ip_j si las personas p_i y p_j se conocen. Queremos demostrar entonces que en cualquier grafo G de 6 vértices ocurre que este o contiene un clique de tamaño 3, o contiene un conjunto independiente de tamaño 3. Similarmente y usando el teorema 2.1.1 podemos decir que el problema es equivalente a demostrar

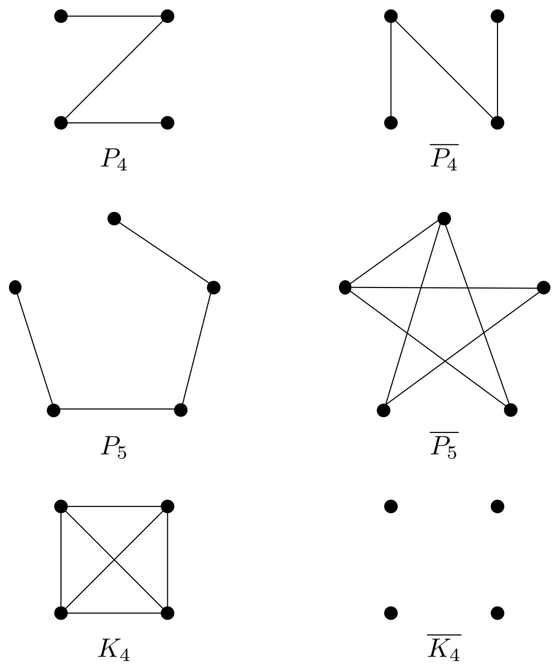


Figura 2.18: Algunos grafos y sus complementos

que para cualquier grafo de 6 vértices ocurre que, o G tiene un clique de tamaño 3, o $\overline{G}$ tiene un clique de tamaño 3. En la figura 2.19 se muestra un posible grafo de 6 vértices junto a su complemento. En el podemos

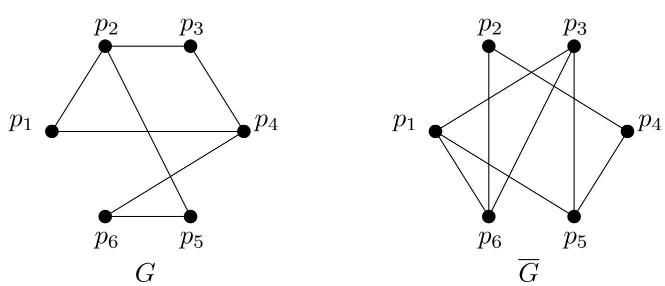


Figura 2.19: Conocidos mutuos y desconocidos mutuos

ver que G no tiene un clique de tamaño 3, pero que $\overline{G}$ si lo tiene, por lo tanto hay tres personas que se desconocen mutuamente (de hecho hay dos de estos grupos, $\{p_1, p_3, p_6\}$ y $\{p_1, p_3, p_5\}$).

Lo que sigue de la demostración la haremos por contradicción suponiendo que ni G ni $\overline{G}$ tienen un clique de tamaño 3. La primera observación que haremos es la siguiente: si miramos una persona en particular, esta o se conoce con al menos tres personas, o se desconoce con al menos tres personas, esto es equivalente a decir que dado un vértice v cualquiera ocurre que, o la cantidad de vecinos de v en G es mayor o igual a 3, o la cantidad de vecinos de v en $\overline{G}$ es mayor o igual a 3, esto es inmediato del hecho de que todas las aristas que faltan en G aparecen en $\overline{G}$ (y vice versa).

Enfoquémonos en un vértice en particular p_i y supongamos que su cantidad de vecinos es mayor o igual a 3 en G, entonces existen otros tres vértices distintos a p_i y distintos entre sí, p_j, p_k y p_l que son vecinos de p_i. Dado que estamos suponiendo que G no tiene un clique de tamaño 3, entonces necesariamente en G p_j no es vecino de p_k, p_j no es vecino de p_l, y p_k no es vecino de p_l, lo que implica que en $\overline{G}$ los vértices p_j, p_k y p_l forman un clique de tamaño tres contradiciendo nuestra suposición de que $\overline{G}$ no tiene un clique de tamaño

3 (ver figura 2.20) Si por el contrario resulta que el vértice particular p_i en el que nos estamos enfocando tiene menos de tres vecinos en G, necesariamente este tiene una cantidad de vecinos mayor o igual a 3 en $\overline{G}$ y podemos usar exactamente el mismo argumento partiendo de $\overline{G}$ y contradiciendo la suposición de que G no tiene un clique de tamaño 3.

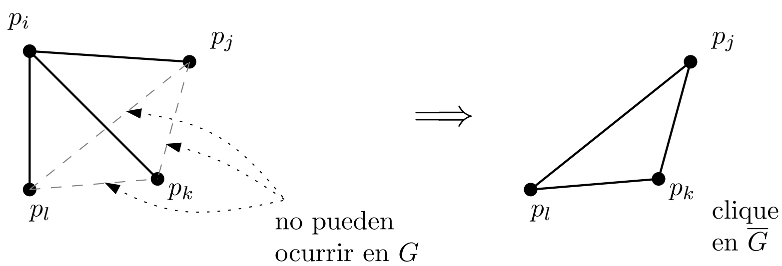


Figura 2.20: Conocidos y desconocidos mutuos.

2.1.3. Representación Matricial

Podemos usar una matriz M_G para representar cualquier grafo simple G usando la matriz que representa a la relación *ser vecino de* entre los vértices de G, $V(G)$. Claramente la relación de ser vecino es simétrica por lo que se cumplirá que $M_G = (M_G)^T$, además, dado que G es un grafo simple, la diagonal de G tendrá sólo 0's. A la matriz M_G se le llama **matriz de adyacencia** de G. Si G tiene n nodos entonces M_G será una matriz de $n \times n$.

Ejemplo: Para el grafo G de la figura 2.19 la matriz de adyacencia M_G resulta

$$M_G = \begin{array}{c} \\ p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \end{array} \begin{array}{c} \begin{array}{cccccc} p_1 & p_2 & p_3 & p_4 & p_5 & p_6 \end{array} \\ \left[\begin{array}{cccccc} 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 \end{array}\right] \end{array}$$

en donde las filas se han organizado en el orden de los índices de los vértices. Por su parte la matriz para el grafo $\overline{G}$ de la misma figura es

$$M_{\overline{G}} = \begin{array}{c} \\ p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \end{array} \begin{array}{c} \begin{array}{cccccc} p_1 & p_2 & p_3 & p_4 & p_5 & p_6 \end{array} \\ \left[\begin{array}{cccccc} 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 \end{array}\right] \end{array}$$

No es difícil justificar que si M_G es la matriz de adyacencia para G, entonces la matriz de adyacencia para $\overline{G}$ se puede obtener de M_G intercambiando todos los 0's por 1's excepto en la diagonal.

La matriz de adyacencia es una forma en que un grafo se le puede entregar a un computador para realizar cierta tarea sobre él. ¿Qué pasa cuando G no es un grafo simple? Supongamos que G es un grafo no necesariamente simple, pero sin rulos, entonces podríamos usar una matriz de adyacencia M_G *extendida* con

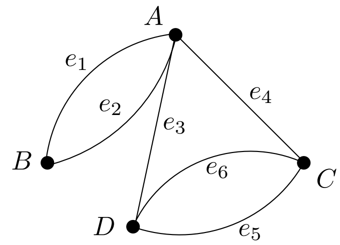


Figura 2.21: Un grafo sin rulos.

no solo 0's y 1's, en que la posición $[M_G]_{(i,j)} = n$ si hay n aristas incidiendo en los vértices que representados por i y j.

Ejemplo: Para el gafo de la figura 2.21 la matriz de adyacencia *extendida* resulta

$$\begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 2 & 1 & 1 \\ 2 & 0 & 0 & 0 \\ 1 & 0 & 0 & 2 \\ 1 & 0 & 2 & 0 \end{bmatrix} \end{array}$$

en ella las filas se han organizado en el orden alfabético de los vértices.

Otra forma de representar un grafo matricialmente es usando lo que se llama la **matriz de incidencia**. En ella las filas se etiquetan con los vértices de G y las columnas con las aristas de G. Si suponemos que G es un grafo no necesariamente simple, pero sin rulos, para cada columna representante de una arista, habrán dos 1's uno por cada vértice extremo de la arista. Si G tiene n vértices y m aristas, entonces su matriz de incidencia será una matriz de $n \times m$.

Ejemplo: Para el grafo de la figura 2.21 la matriz de incidencia asociada será

$$\begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccccc} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 \end{array} \\ \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix} \end{array}$$

Las matrices de adyacencia e incidencia de un grafo G nos sirven para obtener propiedades del grafo y algunas otras propiedades interesantes de conteo. Por ejemplo, en un grafo simple si sumamos una fila de la matriz, el resultado es la cantidad de vecinos que tiene el vértice asociado a esa fila. La siguiente definición tiene que ver con estas propiedades.

Def: El **grado** de un vértice v en un grafo G sin rulos, es la cantidad de aristas de $E(G)$ que están asociadas a v. Cuando G sea un grafo simple el grado de v coincidirá con la cantidad de vecinos. Al grado del vértice v en el grafo G lo denotaremos por $\delta_G(v)$. Cuando el grafo que estemos usando quede claro por el contexto, usaremos simplemente $\delta(v)$.

La primera implicación interesante para el grado de un vértice tiene que ver con las matrices de adyacencia e incidencia. Sea G un grafo sin rulos con n vértices y m aristas. Si al vértice v_i se asocia la fila i de la matriz de adyacencia M_G entonces se cumple que

$$\delta_G(v_i) = \sum_{j=1}^{n} [M_G]_{(i,j)}.$$

Si al vértice v_i se asocia la fila i de la matriz de incidencia A_G entonces se cumple que

$$\delta_G(v_i) = \sum_{j=1}^{m} [A_G]_{(i,j)}.$$

Otra propiedad que se puede obtener a partir de la matriz de incidencia tiene que ver con la relación entre los grados de los vértices y la cantidad de arista de un grafo. Supongamos que tenemos un grafo G sin rulos con n vértices $v_1, v_2, \ldots, v_n$ y m aristas $e_1, e_2, \ldots, e_m$. Si tomamos la matriz de incidencia de G, A_G y sumamos todos los 1's que aparecen en ella, esta suma resulta a partir de la doble sumatoria

$$\sum_{i=1}^{n} \sum_{j=1}^{m} [A_G]_{(i,j)} \tag{2.1}$$

y por el resultado anterior tenemos que

$$\sum_{i=1}^{n} \sum_{j=1}^{m} [A_G]_{(i,j)} = \sum_{i=1}^{n} \delta_G(v_i), \tag{2.2}$$

lo único que hicimos fue reemplazar la sumatoria interna por el grado del vértice correspondiente a la fila que se está sumando. Ahora si en la expresión (2.1) invertimos el orden de las sumatorias obtenemos

$$\sum_{j=1}^{m} \sum_{i=1}^{n} [A_G]_{(i,j)}.$$

Si miramos la nueva sumatoria interna $\sum_{i=1}^{n} [A_G]_{(i,j)}$ su resultado es siempre 2 para cualquier j, esto porque én ella se están sumando los 1's que aparecen en una columna determinada de A_G, que sabemos que es 2 porque hay un 1 por cada vértice extremo de la arista correspondiente a esa columna, con lo que obtenemos

$$\sum_{j=1}^{m} \sum_{i=1}^{n} [A_G]_{(i,j)} \sum_{j=1}^{m} 2 = 2m. \tag{2.3}$$

Dado que el orden en que se realizen las sumatorias no altera el resultado, a partir de (2.3) y (2.2) obtenemos la igualdad

$$\sum_{i=1}^{n} \delta_G(v_i) = 2m,$$

es decir, la suma total de los grados de los vértices es igual al doble de la cantidad de aristas. Un elegante argumento de conteo del mismo resultado se puede ver en el siguiente diagrama:

$$\begin{array}{cc}
 & \begin{array}{ccccc} e_1 & e_2 & e_3 & \cdots & e_m \end{array} \\
\begin{array}{c} v_1 \\ v_2 \\ \vdots \\ v_n \end{array} & \left[\begin{array}{ccccc} & & & & \\ & & A_G & & \\ & & & & \end{array} \right] \left. \begin{array}{l} = \delta(v_1) \\ = \delta(v_2) \\ \quad \vdots \\ = \delta(v_n) \end{array} \right\} \qquad \sum_{i=1}^{n} \delta(v_i) \\
 & \underbrace{\begin{array}{ccccc} \shortparallel & \shortparallel & \shortparallel & & \shortparallel \\ 2 & 2 & 2 & \cdots & 2 \end{array}}_{\displaystyle 2 \times m} \qquad\qquad \downarrow \\
 & \qquad 2 \times m \qquad \rightarrow \qquad \sum_{i=1}^{n} \delta(v_i) = 2m
\end{array}$$

A_G es la matriz de incidencia de G. Para contar todos los 1's que aparecen en ella podemos sumar una a una las filas, obteniendo la suma de los grados $\sum_{i=1}^{n} \delta_G(v_i)$, o sumar una a una las columnas obteniendo el doble de la cantidad de aristas, $2 \times m$, de donde concluimos que $\sum_{i=1}^{n} \delta_G(v_i) = 2m$.

Por la importancia del resultado anterior, lo estableceremos en el siguiente teorema.

Teorema 2.1.2: Dado un grafo G sin rulos siempre se cumple que

$$\sum_{v\in V(G)} \delta_G(v) = 2|E(G)|,$$

o sea, que la sumatoria de los grados de todos los vértices de un grafo, es igual a dos veces la cantidad de aristas del grafo.

Demostración: La demostración se sigue de la discusión previa al teorema. $\square$

Este teorema nos permite establecer un par de corolarios muy importantes.

Corolario 2.1.3: En un grafo G sin rulos siempre existe una cantidad par de vértices de grado impar.

Demostración: La primera observación es que la suma de los grados de todos los vértices de G es par. Ahora, dividamos los vértices de G en dos grupos disjuntos, los que tienen grado par, digamos $u_1, u_2, \dots, u_p$ y los que tienen grado impar, $w_1, w_2, \dots, w_q$, o sea, G tiene p vértices de grado par y q vértices de grado impar. Sean ahora

$$\begin{array}{rcl} P & = & \delta(u_1) + \delta(u_2) + \cdots + \delta(u_p) \\ I & = & \delta(w_1) + \delta(w_2) + \cdots + \delta(w_q). \end{array}$$

Dado que el resultado de $P + I$ es la suma de los grados de todos los vértices, necesariamente $P + I$ es par. Ahora, P es claramente par ya que es la suma de sólo números pares por lo que I es par también. Dado que I es una suma de q números impares, para que I sea par, necesariamente q debe ser un número par, de donde se concluye que G tiene una cantidad par de vértices de grado impar. $\square$

Los siguientes ejemplo aplican directamente estos resultados.

Ejemplo: Se quiere organizar un campeonato de fútbol con 25 equipos de manera tal que cada equipo juegue 5 partidos, cada uno contra un equipo distinto. ¿Hay forma de realizar el campeonato? La respuesta es NO y se obtiene como una consecuencia de los resultados anteriores.

Podemos modelar el problema como un grafo de 25 vértices cada uno representando a un equipo distinto. El grafo sería tal que hay una arista entre dos vértices si corresponden a dos equipos que jugarán un partido en el campeonato. Para cumplir la regla de que cada equipo juegue con exactamente 5 equipos distintos, el grafo debiera ser tal que cada uno de los 25 vértices tuviese grado 5, lo que sabemos que no puede ocurrir ya que tendría una cantidad impar de vértices de grado impar. Por lo tanto el campeonato no puede realizarse siguiendo estas reglas.

Ejemplo: En el departamento de informática de una empresa trabajan 15 empleados, uno de ellos es la secretaria del departamento y otro es el jefe del departamento, ambos se saludan todos los días y saludan a todos los demás empleados. Cada uno de los restantes empleados del departamento asegura que diariamente se saluda con exactamente 3 de sus compañeros (sin contar a la secretaria y el jefe) ¿Es esto posible?

En este caso podemos modelar el problema con un grafo de 15 vértices, uno por empleado, con una arista entre vértices correspondientes a empleados que se saludan. Dos de los vértices son distinguidos, los correspondientes al jefe y la secretaria, y tienen grado 14 (se saludan con todos). Los restantes 13 vértices, correspondientes a los demás empleados, debieran tener grado 5 cada uno, ya que se supone que saludan al jefe a la secretaria y a tres de sus compañeros. Por los resultado anteriores, no puede existir tal grafo ya que tendría una cantidad impar de vértices de grado impar lo que sabemos no puede ocurrir.

2.2. Caminos y Ciclos

Este tema ya lo motivamos con el ejemplo de los puentes de Königsberg de la sección anterior. En esta sección nos interesará estudiar algunas propiedades importantes de los grafos que tienen que ver con *caminos* y *ciclos* sobre ellos. Comenzaremos con un par de definiciones importantes para nuestro siguiente estudio.

Def: Una **caminata** en un grafo (no necesariamente simple) G es una secuencia de vértices y aristas $(v_0, e_1, v_1, e_2, v_2, \ldots, e_k, v_k)$ tal que para $1 \leq i \leq k$, la arista e_i une a los vértices v_{i-1} y v_i. Una **caminata cerrada** en un grafo es una caminata en la que el primer y último vértices son iguales ($v_0 = v_k$). Cuando el grafo es simple, se pueden omitir los nombres de las aristas entre vértices en la representación de una caminata.

Un **camino** en un grafo G, es una caminata en la que no se repiten aristas. Un **ciclo** en un grafo G, es una caminata cerrada en la que no se repiten aristas. (Estas definiciones no debieran causar confusión con las definiciones de P_n y C_n que son clases de grafos, aquí estamos definiendo camino y ciclo *en un grafo dado*, son subgrafos de un grafo dado.)

Una caminata o camino que comience en u y termine en v lo llamaremos caminata $u - v$ o camino $u - v$. El largo de una caminata, caminata cerrada, camino y ciclo, se obtiene a partir de la cantidad de aristas. Por simplicidad supondremos que un camino (o ciclo, o caminata) de largo 0 es un camino (o ciclo o caminata) compuesto por un único vértice sin aristas.

Ejemplo: En el grafo de la figura 2.22 las secuencia:

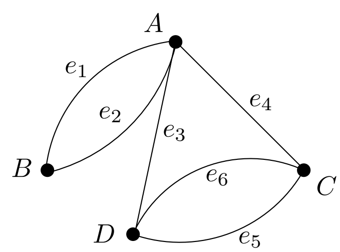


Figura 2.22: Grafo sin rulos

- $(D, e_6, C, e_5, D, e_6, C, e_4, A, e_1, B, e_1, A)$ es una caminata pero no un camino, su largo es 6.
- $(D, e_6, C, e_5, D, e_6, C, e_4, A, e_1, B, e_1, A)$ es una caminata cerrada pero no un ciclo., su largo es 6.
- $(D, e_6, C, e_5, D, e_3, A, e_2, B)$ es un camino, su largo es 4.
- $(D, e_6, C, e_5, D, e_3, A, e_2, B, e_1, A, e_3, D)$ es un ciclo, su largo es 6.
- (D, e_3, B, e_1, C) no es una caminata.

En ellos es necesario nombrar las aristas dado que el grafo no es simple.

En el grafo simple de la figura 2.23 podemos decir que las secuencias:

- $(v_1, v_4, v_5, v_2, v_3, v_4, v_5)$ es una caminata pero no un camino, su largo es 6.
- $(v_1, v_4, v_5, v_2, v_3, v_4, v_5, v_4, v_1)$ es una caminata cerrada pero no un camino, su largo es 8.
- (v_1, v_4, v_6) es un camino, su largo es 3.
- $(v_1, v_2, v_5, v_4, v_3, v_1)$ es un ciclo, su largo es 5.
- (v_1, v_5, v_6) no es una caminata.

En este caso no se nombran las arista ya que quedan implícitas por el hecho de sur un grafo simple.

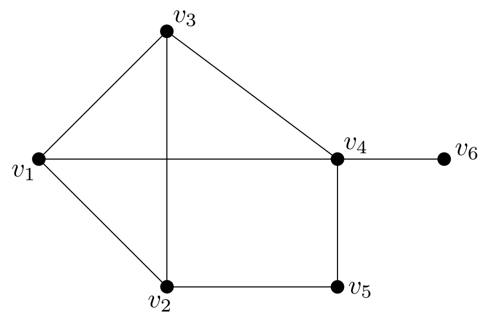


Figura 2.23: Grafo simple

2.2.1. Conectividad

Def: Un grafo G se dice **conexo** si para cada par de vértices $u, v \in V(G)$ existe un camino que contiene tanto a u como a v. No es difícil notar que si existe un camino que contiene a un par de vértices u y v, entonces existe un camino cuyo vértice inicial es u y final es v. Cuando esto ocurra lo denotaremos por $u \sim v$. La relación $\sim$, o como la llamaremos "existe un camino entre", es una relación de equivalencia sobre los vértices de un grafo. A un grafo que no sea conexo le llamaremos disconexo.

Dado un grafo G, el subgrafo de G compuesto por todos los caminos que contienen un vértice particular se llama **componente conexa** de G. La componente conexa de G a la que pertenece un vértice v particular, contiene a todos los vértices de G que están relacionados con v mediante $\sim$, es decir, todos los vértices de la clase de equivalencia de v.

Ejemplo: En la figura 2.24, G_1 es un grafo conexo, mientras que G_2 no es conexo, por ejemplo no existe un camino entre los vértices v_1 y v_8. En la misma figura, G_2 tiene 3 componentes conexas, $\{v_2, v_6\}$, $\{v_1, v_3, v_5, v_7, v_9\}$, $\{v_4, v_8, v_{10}\}$. En la figura 2.25 se ha dibujado G_2 de manera de mostrar claramente sus componentes conexas.

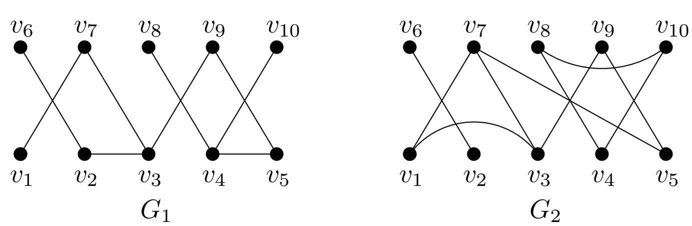


Figura 2.24: G_1 es un grafo conexo, mientras que G_2 no lo es.

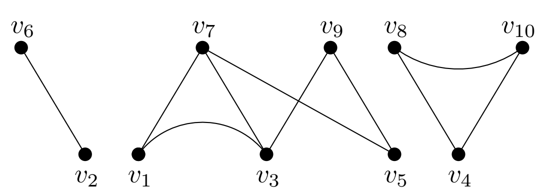


Figura 2.25: Componentes conexas de G_2.

¿Cuántas aristas puedo agregarle al grafo G_2 de la figura 2.25 de tal manera que este siga siendo simple y teniendo tres componentes conexas? ¿Cuántas aristas puedo sacar de G de manera que este siga teniendo tres componentes conexas? No es difícil notar que una nueva arisita agregada a un grafo puede disminuir la cantidad de componentes conexas a lo más en 1, no las modifica si la arista es entre vértices de una misma componente y une dos componentes si la arista es entre vértices de distintas componentes. De la misma forma, sacar una arista de un grafo puede aumentar la cantidad de componentes conexas a lo más en 1. Estas observaciones nos ayudan en el siguiente teorema.

Teorema 2.2.1: Un grafo G con n vértices y k aristas tiene al menos $n-k$ componentes conexas.

Demostración: Un grafo G con n vértices puede tener como máximo n componentes conexas, cuando no tiene ninguna arista, cada nueva arista que se le agregue puede reducir la cantidad de componentes a lo más en 1, por lo que luego de agregar k aristas la cantidad de componentes se ha reducido como mínimo a $n-k$, por lo que la cantidad de componentes siempre se mantiene mayor o igual a $n-k$. $\square$

El anterior teorema implica que si se quiere un grafo conexo de n vértices, entonces al menos $n-1$ aristas son necesarias. Una definición motivada también por la discusión anterior es la siguiente.

Def: Una **arista de corte** en un grafo G es una arista tal que su eliminación aumenta la cantidad de componentes. Escribiremos $G-e$ para representar al grafo que resulta de eliminar la arista e a G.

Un **vértice de corte** en un grafo G es un vértice tal que su eliminación aumenta la cantidad de componentes. Al eliminar un vértice v de un grafo, sacamos v y todas sus aristas incidentes, al grafo resultante lo llamamos $G-v$.

Ejemplo: En el grafo G_1 de la figura 2.24, todas las aristas son de corte, y sus vértices de corte son $v_2, v_3, v_4, v_5, v_7, v_9$. Un grafo como G_1 en el que todas sus aristas son de corte, es un grafo bien particular y lo estudiaremos en la siguiente sección. En el grafo G_2 de la figura 2.24 no hay vértices de corte, y la única arista de corte es v_2v_6.

Ejemplo: Las aristas y vértices de corte así como la conectividad son conceptos muy importantes a la hora de diseñar una red (o grafo) de comunicación, como una red de computadores que intercambian datos, una red de repetidores de telefonía, o incluso una red de conexiones de vuelo entre ciudades (aquí lo comunicado serían pasajeros entre ciudades).

Tomemos el ejemplo de una red de computadores en la que necesitamos que todos los terminales sean capaces de comunicarse entre sí. En el grafo asociado a esta red, cada vértice representa a un computador y una arista al cable que conecta un par de computadores. La primera observación es que este grafo debe ser conexo para que todos los computadores puedan establecer comunicación. ¿Que pasa si este grafo tiene algún vértice de corte? Esto significaría que existe algún computador que, de fallar, haría imposible la comunicación entre algunos de los computadores que no han fallado. ¿Qué pasa si este grafo tiene una arista de corte? Esto significaría que existe alguna conexión (cable) que de fallar haría imposible la comunicación entre alguno de los computadores de la red. Es interesante que al diseñar una red con un grafo asociado que no tenga ni puntos de corte ni aristas de corte, esta red será automáticamente *tolerante a la falla* de un computador o un cable de comunicación.

En el ejemplo de los vuelos entre ciudades, en el grafo asociado cada vértice representaría a un aeropuerto en alguna ciudad, y una arista representaría la existencia de algún vuelo directo entre un par de aeropuertos. Si existe un vértice de corte, querría decir que existe un aeropuerto que de ser cerrado dejará a algunas personas sin poder viajar por avión a su destino. Algo similar pasa con las aristas de corte en este caso.

Caracterizaremos una arista de corte en términos de ciclos en el grafo.

Teorema 2.2.2: Una arista en un grafo G es de corte si y sólo si no pertenece a ningún ciclo en G.

Demostración: Nos centraremos en un grafo G conexo, la demostración se aplicará entonces para cada componente conexa de G.

Sea $e = uv$ una arista que pertenece a un ciclo C en G, al eliminar e los únicos caminos afectados en $G - e$ son los que contenían a la arista $e = uv$, pero dado que esta pertenece a un ciclo C, los caminos afectados pueden completarse con la porción restante de C, luego la arista e no puede ser de corte.

Supongamos ahora que $e = uv$ no es una arista de corte, o sea que $G - e$ sigue siendo conexo, esto quiere decir que existe un camino digamos P entre u y v en $G - e$. El camino P junto con la arista uv forman un ciclo en G.

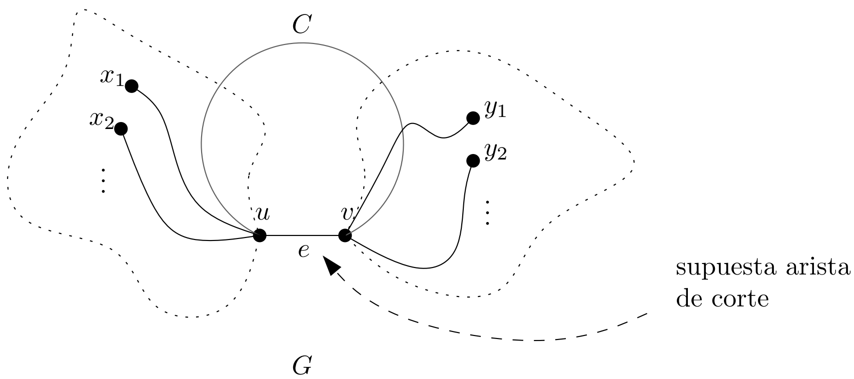


Figura 2.26: Una arista es de corte sólo si no pertenece a un ciclo.

La idea central de la demostración se ve en la figura 2.26, claramente si $e = uv$ pertenece a un ciclo su eliminación no desconectará a G. □

Ejemplo: En el grafo G_1 de la figura 2.24, ninguna de sus aristas pertenece a un ciclo ya que G_1 no tiene ningún ciclo, luego todas sus aristas son de corte. En el grafo G_2 de la misma figura, la única arista que no pertenece a un ciclo es v_2v_6 por lo tanto es la única arista de corte.

2.2.2. Grafos Bipartitos

En esta sección estudiaremos una caracterización de grafos bipartitos en función de ciclos y caminos.

Lema 2.2.3: En un grafo simple G, toda caminata cerrada de largo impar, contiene un ciclo de largo impar.

Demostración: Lo haremos usando un argumento inductivo en el largo de la caminata cerrada. La caminata cerrada más pequeña de largo impar que se puede hacer en un grafo simple es un ciclo de tres vértices, esta caminata ya es un ciclo así que el caso base se comprueba. Ahora tomemos una caminata C cerrada de largo l impar y supongamos como hipótesis de inducción que toda caminata cerrada de largo impar menor a l tiene un ciclo de largo impar. Si en C no se repiten vértices entonces C ya es un ciclo de largo impar y comprobamos lo que queríamos, si por otro lado en C se repite un vértice, digamos v, entonces podemos partir C en dos caminatas cerradas distintas que comienzan en v, C' y C'' como se muestra en la figura 2.27. No puede ocurrir que simultáneamente C' y C'' tengan largo par ya que entonces C no podría tener largo impar, por lo que al menos una de ellas es una caminata cerrada de largo impar estrictamente menor a l y por HI contiene un ciclo de largo impar, que también será un ciclo de largo impar contenido en C comprobando lo que queríamos. □

El anterior teorema sólo se aplica para caminatas cerradas de largo impar, una caminata cerrada de largo par podría no contener un ciclo de largo par, por ejemplo podría ser una caminata que repitiera todas las aristas dos veces cada una. El anterior lema nos servirá para demostrar el siguiente teorema.

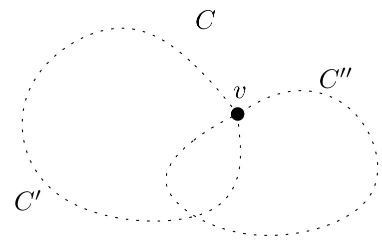


Figura 2.27: Una caminata cerrada que repite un vértice, dividida en dos caminatas cerradas

Teorema 2.2.4: Un grafo conexo (simple) G es bipartito si y sólo si no contiene ningún ciclo de largo impar.

Demostración: ($\Rightarrow$) Supongamos que G contiene un ciclo de largo impar, digamos $C = (v_1, v_2, \ldots, v_{k-1}, v_k, v_1)$, con k un natural impar, demostraremos que G no puede ser bipartito. Supongamos que G es bipartito con particiones V_1 y V_2 y supongamos sin pérdida de generalidad que $v_1 \in V_1$. Dado que C es un ciclo, necesariamente $v_i v_{i+1} \in E(G)$ para $1 \leq i < k$ y $v_k v_1 \in E(G)$, por lo que debe ocurrir que $v_2 \in V_2$, $v_3 \in V_1$, $v_4 \in V_2$, etc. En general debe ocurrir que para los vértices del ciclo C, $v_i \in V_1$ si i es impar, y $v_i \in V_2$ si i es par, luego $v_k \in V_1$ lo que es una contradicción con el hecho de suponer que V_1 es una partición que contiene a v_1 ya que $v_k v_1 \in E(G)$.

($\Leftarrow$) Supongamos que G no contiene ningún ciclo de largo impar, demostraremos que es posible definir una partición de los vértices de G en dos conjuntos independientes. Sea v un vértice cualquiera de $V(G)$, definimos $V_1 = \{u \in V(G) \mid \text{ existe un camino de largo impar de } v \text{ a } u\}$, y $V_2 = \{u \in V(G) \mid \text{ existe un camino de largo par de } v \text{ a } u\}$. Si existiera una arista entre dos vértices de V_1 digamos u_1 y u_2, entonces, dado que existen caminos $v - u_1$ y $v - u_2$ ambos de largo impar, existiría una caminata cerrada de largo impar formada por los dos caminos anteriores más la arista $v_1 v_2$ y por el lema 2.2.3 existiría un ciclo de largo impar contradiciendo nuestra suposición. Si existiera una arista entre dos vértices de V_2 digamos w_1 y w_2, entonces, dado que existen caminos $v - w_1$ y $v - w_2$ ambos de largo par, existiría una caminata cerrada de largo impar formada por los dos caminos anteriores más la arista $w_1 w_2$ y nuevamente por el lema 2.2.3 existiría un ciclo de largo impar contradiciendo nuestra suposición. Finalmente no existe arista entre vértices de V_1 y no existe aristas entre vértices de V_2 y como G es conexo se tiene que $V_1 \cup V_2 = V(G)$ por lo que G es bipartito con particiones V_1 y V_2. $\square$

No es difícil notar que si G no es conexo pero si bipartito, la demostración se aplica a cada componente conexa, y también ocurrirá que G no tendrá ciclos de largo impar. Para demostrar entonces que un grafo G es bipartito basta dividir los vértices de G en dos conjuntos independientes. Para demostrar que un grafo G no es bipartito basta encontrar un ciclo de largo impar en G.

El alumno podría diseñar un algoritmo para determinar si un grafo es o no bipartito. El algoritmo debiera ser tal que si el grafo es bipartito entonces el output entregue los conjuntos de vértices que conforman cada partición, y si el grafo no es bipartito el output entregue una secuencia de vértices que formen un ciclo de largo impar en el grafo.

2.2.3. Ciclos y Caminos Eulerianos

En el ejemplo de los puentes de Königsberg los habitantes del pueblo querían encontrar una forma de recorrer la ciudad pasando una vez por cada puente y volviendo al lugar inicial. Modelamos el problema usando un grafo y dijimos que el problema era equivalente a intentar "dibujar" el grafo sin repetir las aristas y volviendo al vértice de donde se inició el dibujo. Ya argumentamos que si algún vértice del grafo tenía grado impar entonces tal dibujo no se podía lograr. En esta sección completaremos el resultado, demostrando que esta condición es también suficiente para que un grafo pueda dibujarse siguiendo las restricciones, o sea que un grafo se puede dibujar sin repetir trazos y volviendo al vértice inicial, si y sólo si ninguno de sus vértices tiene grado impar. Iniciaremos formalizando algunos conceptos.

Def: Un **ciclo Euleriano** en un grafo G, es un ciclo que contiene a todas las aristas de G y a todos los vértices de G. Note que es importante que en un ciclo no se repitan aristas, pero perfectamente pueden repetirse vértices. Diremos que G es un **grafo Euleriano** si contiene un ciclo Euleriano.

Según la definición anterior, el problema de los puentes de Königsberg es equivalente a determinar si su grafo asociado es o no Euleriano. En la figura 2.28 se muestra un grafo que si es Euleriano, de hecho contiene el siguiente ciclo Euleriano: $(v_1, v_2, v_6, v_7, v_3, v_9, v_{10}, v_4, v_8, v_9, v_5, v_7, v_1)$.

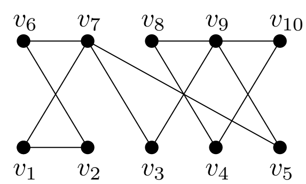


Figura 2.28: Grafo Euleriano.

Estableceremos ahora el teorema importante de esta sección.

Teorema 2.2.5: Un grafo G (sin rulos) es Euleriano si y sólo si es conexo y todos sus vértices tienen grado par.

Demostración: ($\Rightarrow$) Supongamos que G es Euleriano, entonces G tiene un ciclo que contiene a todas las aristas y todos los vértices, supongamos que este ciclo parte (y termina) en un vértice particular v. El ciclo necesita una arista para "salir" de v y otra para "llegar" finalmente a v, y cada vez que v aparece nuevamente en el ciclo necesita dos aristas distintas más (una para entrar y otra para salir), un diagrama se ve en la figura 2.29. Esto implica que $\delta(v)$ necesariamente es par, ya que todas sus aristas incidentes aparecen en este

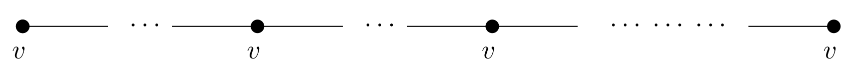


Figura 2.29: Un vértice v particular del ciclo Euleriano y sus artistas incidentes.

ciclo una vez cada una. Este ciclo se puede pensar que parte (y termina) en cada uno de los otros vértices del grafo concluyendo que todos tienen grado par. Es claro también que si existe tal ciclo, el grafo G es conexo ya que dentro del mismo ciclo se pueden encontrar caminos entre todos los pares de vértices de G.

($\Leftarrow$) Supongamos que G es conexo y que todos sus vértices tiene grado par. Demostraremos por inducción en el número de aristas de G que G tiene un ciclo Euleriano.

B.I. Para la base de inducción debemos tomar el grafo con el menor número de aristas conexo y con todos sus vértices de grado par. Vamos a dividir en dos casos. Un primer caso sería un grafo compuesto por un único vértice de grado 0, claramente este grafo tiene un ciclo Euleriano compuesto por el único vértice del grafo. El grafo más pequeño con más de un vértice que cumple las propiedades es el que tiene dos vértices y dos aristas, es claro que este grafo tiene un ciclo Euleriano Con la base de inducción podemos ir un poco más lejos, no es difícil demostrar además que cualquier grafo conexo con 2 vértices, ambos de grado par siempre contendrá un ciclo Euleriano. Todos estos casos base se muestran en la figura 2.30

H.I. Supongamos como hipótesis de inducción que cualquier grafo conexo cuyos vértices tienen grado par y que tiene menos de n aristas tiene un ciclo Euleriano.

T.I. Sea G un grafo conexo cuyos vértices tienen grado par y que tiene exactamente n aristas. Dado que ya mostramos en la BI los casos en que G tiene sólo uno o dos vértices, podemos concentrarnos en los casos en que G tiene al menos 3 vértices distintos. Dado que G es conexo y tiene al menos tres vértices,

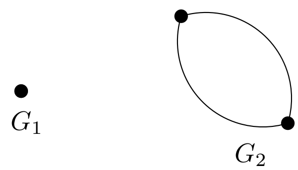


Figura 2.30: Los grafos más pequeños conexos y tal que sus vértices tienen grado par.

necesariamente debe existir un camino de largo 2 con aristas e_1 y e_2, que contiene tres vértices, digamos v_1, v_2 y v_3, un diagrama de esto se ve en la figura 2.31.

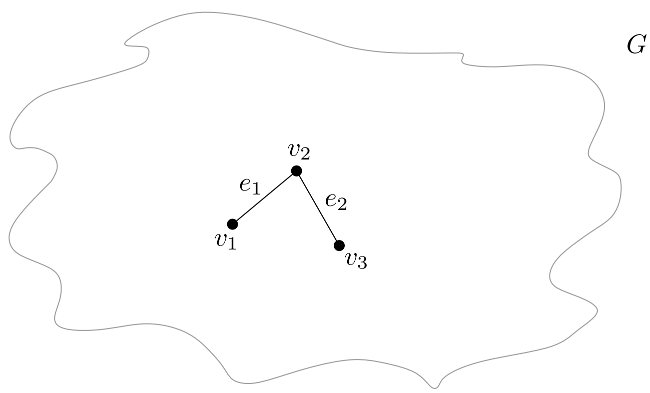


Figura 2.31: Camino de largo 2 en un grafo con al menos 3 vértices.

Podemos crear un nuevo grafo G' a partir de eliminar las aristas e_1 y e_2 de G, y agregar una nueva arista e entre los vértices v_1 y v_3. Un diagrama de G' se ve en la figura 2.32. La primera observación

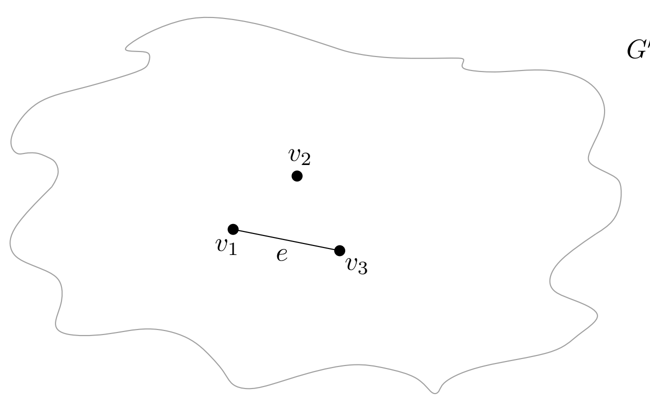


Figura 2.32: Creación de G' a partir de la eliminación de e_1 y e_2 y la inserción de e'.

es que G' tiene estrictamente menos aristas que G. Otra cosa que se puede observar es que los únicos vértices que pudieron haber visto afectados sus grado en G son v_1, v_2 y v_3. Tanto para v_1 y v_3 su grado en G' es el mismo que en G, para v_2 el grado se ha disminuido en 2, por lo que, dado que en G los tres vértices tenían grado par, en G' también ocurrirá que tienen grado par y por lo tanto G' tiene todos sus vértices de grado par. En este punto "casi" podemos aplicar la hipótesis de inducción a G', dado que no tenemos seguridad que después del cambio, G' sea un grafo conexo. Nos pondremos entonces en ambos casos, cuando G' resulta ser conexo y cuando no. Si G' es conexo entonces cumple con la HI y por lo tanto tiene un ciclo Euleriano digamos C' que contiene a todas las aristas de G' y por lo tanto contiene a e, o sea C' es un ciclo Euleriano en G' que contiene la subsecuencia (v_1, e, v_3). A partir de C' podemos generar un ciclo para G eliminando la arista e y añadiendo las aristas e_1 y e_2, o sea cambiando la secuencia (v_1, e, v_3) de C' por la secuencia $(v_1, e_1, v_2, e_2, v_3)$ este nuevo ciclo es claramente un ciclo que contiene todas las aristas de G, por lo tanto G tiene un cilco Euleriano. Un diagrama de esta construcción se ve en la figura 2.33.

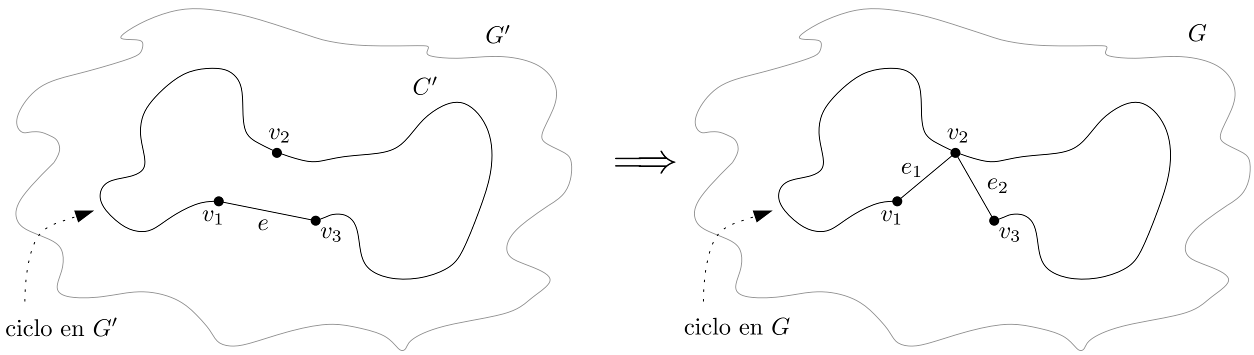


Figura 2.33: Construcción de un ciclo Euleriano para G a partir de uno para G'

Si por otra parte G' no es conexo, dado que G es conexo, G' tendrá dos componentes conexas, una que contendrá a v_2, y otra que contendrá a v_1 y v_3. Cada una de estas componentes tendrá sólo vértices de grado par y estrictamente menos aristas que G, por lo tanto a cada componente se le aplica la HI. Por HI entonces existe un ciclo, digamos C' que podemos suponer que comienza y termina en v_2 y contiene a todas las aristas de una de las componentes de G', o sea C'' es una secuencia de la forma $(v_2, \dots, v_2)$. Por HI también existe otro ciclo C'' que pasa por todas las aristas de la otra componente de G' y que por lo tanto contiene a e, o sea C'' contiene la subsecuencia $(v_1, e, v, 3)$. Podría entonces crearse un ciclo para G "insertando" C' en C'' entre v_1 y v_3 añadiendo las aristas e_1 y e_2. Un diagrama de esta construcción se ve en la figura 2.34.

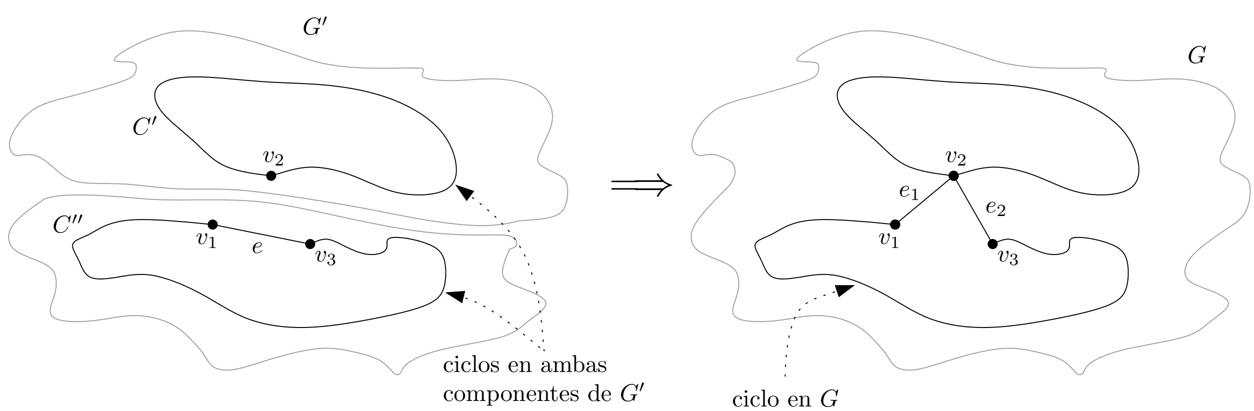


Figura 2.34: Construcción de un ciclo Euleriano para G a partir de los ciclos en las dos componentes de G'

Por inducción se sigue entonces que cualquier grafo conexo cuyos vértices tiene todos grado par es Euleriano, o sea, contiene un ciclo Euleriano.

□

El alumno debiera notar que la demostración además nos entrega un algoritmo para encontrar un ciclo Euleriano en un grafo. El algoritmo inicialmente realizaría el paso de cambiar un par de aristas por una sola como en la figura 2.32 y recursivamente debiera encontrar los ciclos del grafo resultante para luego construir un ciclo para el grafo inicial. Los casos base serían un grafo con un único vértice a con grafo con dos vértices y dos aristas.

Ejemplo: Supongamos que se tiene un tablero de ajedrez de $n \times n$ ¿Existe algún valor de n para el cuál el caballo pueda moverse desde un casillero, hacer todas las movidas que son posibles para él en el tablero una vez cada una y volver al casillero inicial? La respuesta es no, el problema puede modelarse como un grafo, cada casillero representa un vértice y cada movida posible del caballo una arista. Lo que se quiere entonces es encontrar un ciclo Euleriano. No es difícil notar que para ningún n (excepto claro $n = 1$ existirá un ciclo Euleriano. Los casos $n = 2$ y $n = 3$ resultan en grafos no conexos y para $n \leq 4$ el vértice correspondiente a la posición $(1, 2)$ del tablero tiene grado 3, luego existe un vértice de grado impar y por lo tanto no existirá un ciclo Euleriano.

La siguiente definición relaja un poco la noción de ciclo Euleriano.

Def: Un **camino Euleriano** en un grafo G es un camino no cerrado (el vértice inicial es distinto al final) que contiene a todos los vértices y a todas las aristas de G. Recuerde que para ser un camino, no debe repetir aristas de G.

Ejemplo: El grafo de la figura 2.35 no tiene un ciclo Euleriano ya que por ejemplo el vértice v_3 tiene grado 3, pero si tiene un camino Euleriano, de hecho el camino $(v_3, v_2, v_1, v_5, v_2, v_4, v_3, v_5, v_4)$ contiene a todos los vértices y a todas las aristas.

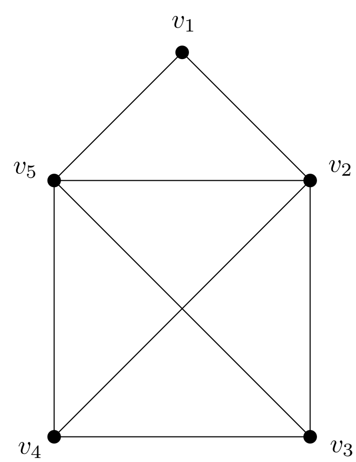


Figura 2.35: Un grafo que contiene un camino Euleriano.

La noción de camino Euleriano captura mucho más nuestra intuición de "poder dibujar una figura sin levantar el lápiz", ¿existirá alguna caracterización para grafos que tengan caminos Eulerianos? La respuesta es sí y la establecemos en el siguiente teorema.

Teorema 2.2.6: Un grafo G tiene un camino Euleriano si y sólo si es conexo y contiene exactamente dos vértices de grado impar.

Demostración: ($\Leftarrow$) Supongamos que en un grafo conexo hay dos vértices de grado impar, digamos u y v. Si al grafo se le agrega una nueva arista para unir u con v, el grafo resultante tiene todos sus vértices de grado par por lo que, por el teorema 2.2.5, existirá un ciclo Euleriano en él. Si a este ciclo se le "borra" la arista recién agregada resulta un camino que pasa por todos los vértices y aristas del grafo original, o sea un camino Euleriano.

($\Rightarrow$) Primero si en el grafo existe un camino Euleriano, entonces el grafo es necesariamente conexo (existe un camino entre cada par de vértices). Supongamos ahora que el camino Euleriano parte en un vértice v y termina en u, entonces al agregar una arista nueva uv al camino, se forma un nuevo grafo que contiene un ciclo Euleriano (se cierra el camino). Por el teorema 2.2.5 el nuevo grafo necesariamente tiene todos sus vértices de grado par, por lo que en el grafo inicial los únicos vértices de grado impar eran u y v, por lo tanto el grafo tiene exactamente dos vértices de grado impar. $\square$

2.2.4. Ciclos Hamiltonianos

Considere los grafos de la figura 2.36 ¿Es posible encontrar en alguno de ellos, un ciclo que contenga a todos los vértices una véz a cada uno (excepto por el inicial y final)? Después de probar un poco nos damos cuenta de que en G_1 si existe tal ciclo, por ejemplo $(v_1, v_2, v_3, v_4, v_5, v_1)$, pero que en G_2 y en G_3 es imposible encontrar un ciclo de estas características.

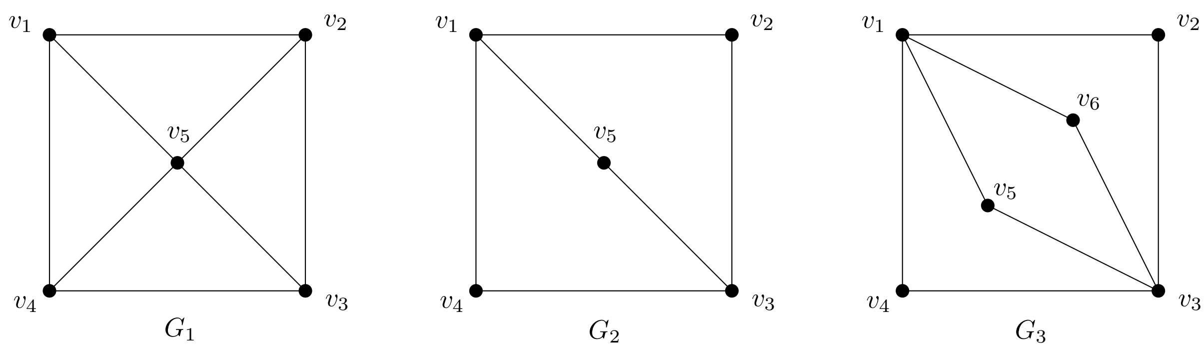


Figura 2.36: Solo G_1 tiene un ciclo Hamiltoniano.

Def: Un ciclo en un grafo G se dice **ciclo Hamiltoniano** si contiene a todos los vértices de G una única vez a cada uno (excepto por el vértice inicial y final). A un grafo que contenga un ciclo Hamiltoniano lo llamaremos **grafo Hamiltoniano**.

Con esta definición podemos decir que el grafo G_1 de la figura 2.36 es Hamiltoniano, pero que los grafos G_2 y G_3 de la misma figura no lo son. Una primera pregunta que podemos hacernos, dada la similitud del problema, es si existe alguna relación entre grafos Eulerianos y grafos Hamiltonianos. Rápidamente podemos darnos cuenta que no hay una relación directa, por ejemplo, en la figura 2.36 el grafo G_1 es Hamiltoniano pero no Euleriano, G_2 no es ni Hamiltoniano ni Euleriano, y G_3 es Euleriano pero no Hamiltoniano.

¿Existirá alguna propiedad simple de chequear para determinar si un grafo es o no Hamiltoniano? Hasta el día de hoy nadie ha sido capaz de encontrar una tal propiedad y es bastante poco probable que se encuentre. Por otra parte nadie ha podido demostrar que no exista una propiedad simple de chequear. Desde el punto de vista computacional lo anterior nos quiere decir que no existe un procedimiento rápido para determinar si un grafo cualquiera es o no Hamiltoniano, estamos condenados a tener que probar todas las posibilidades para poder responder SI o NO. La verdad es que si la respuesta es SI, posiblemente no tengamos que probar todas las posibilidades basta con que encontremos un ciclo Hamiltoniano para que nuestra búsqueda acabe, el tema es que si la respuesta es NO, tendremos que asegurarnos de que ninguna posibilidad nos entrega un ciclo Hamiltoniano. Este es un contraste radical con el de determinar si un grafo es Euleriano, para lo que sabemos que existe un algoritmo muy simple y rápido que responde SI o NO. El problema de determinar si un grafo tiene o no un ciclo Hamiltoniano es un problema *computacionalmente difícil* y se cree que no puede ser resuelto de manera *eficiente* por un computador. Más adelante en nuestro estudio nos enfocaremos con más detalle en este tipo de problemas y formalizaremos las nociones de *eficiencia* y *dificultad computacional*.

Ejemplo: Supongamos que se tiene un tablero de ajedrez de $n \times n$ ¿Para qué valores de n puede un cabello moverse partiendo de un casillero cualquiera, pasando por todos los otros casilleros una única vez por cada uno y volviendo al casillero inicial? Nuevamente el problema puede modelarse como un grafo, cada casillero representa un vértice y cada movida posible del caballo una arista. El problema se resume a determinar si el grafo resultante es o no Hamiltoniano. Puede demostrarse que el grafo resultante es Hamiltoniano para todo n par mayor o igual a 6, sin embargo no hay una propiedad simple de chequear para asgurar esto (como en el caso de un ciclo Euleriano) y la demostración debe ser realizada por separado para cada caso.

2.3. Árboles y Grafos en Computación

Un árbol es una clase especial de grafo y su importancia es tal en las aplicaciones computacionales que lo estudiaremos en una sección por separado. Junto con el estudio de árboles surge el de problemas de optimización sobre grafos que también tocaremos en esta sección.

2.3.1. Árboles

Supongamos que tenemos una red de computadores como la de la figura 2.37 donde los trazos entre computadores representan cables directos entre ellos. Esta red cumple con la propiedad de que cualquier computador

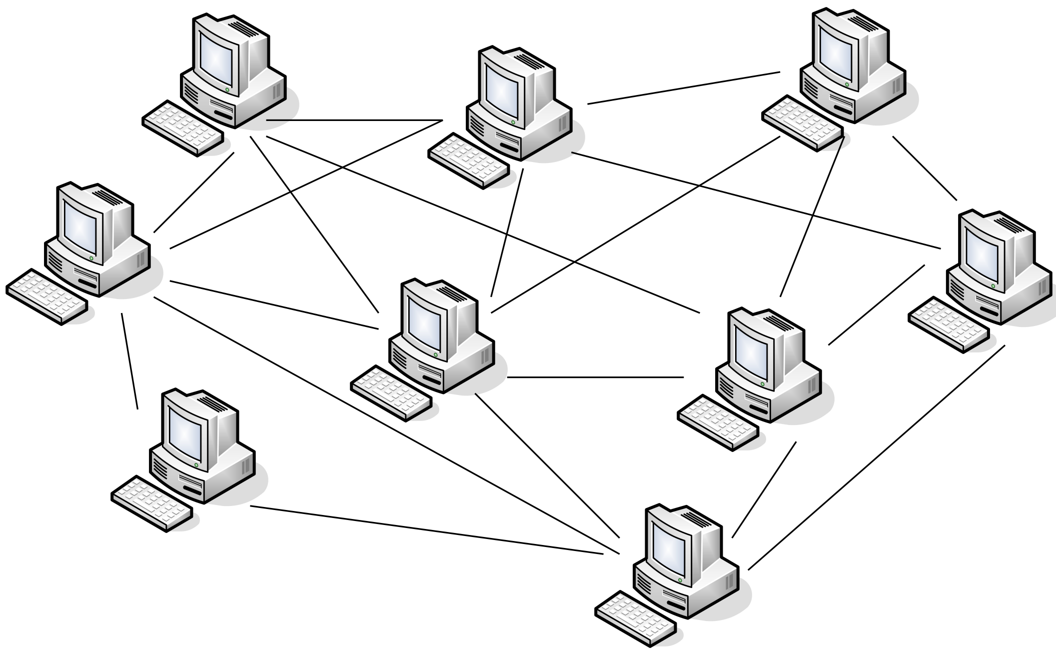


Figura 2.37: Una red de computadores.

puede enviar información a cualquier otro en la red (no necesariamente en forma directa), o sea, el grafo asociado a la red es conexo. Supongamos que quisiéramos construir una red con la propiedad anterior (todos los computadores se puedan comunicar entre ellos) pero minimizando la cantidad de conexiones directas entre computadores. Un posible resultado se ve en la figura 2.38. Esta red cumple la misma propiedad anterior, todo computador puede enviar información a cualquier otro computador en la red. Si nos enfocamos en el grafo asociado a esta nueva red de computadores, una característica crucial que lo diferencia con el anterior es que dado cualquier par de vértices (computadores en la red) existe un único camino que los une. A un grafo con estas características se le llama **árbol** y su definición se formaliza a continuación.

Def: Un grafo $T = (V(T), E(T))$ es un **árbol** si para cada par de vértices $u, v \in V(T)$ existe un único camino de u a v. Es inmediato de la definición que un árbol es siempre un grafo conexo, dado que para cada par de vértices *existe* un camino (que "de paso" es único).

Ejemplo: El grafo correspondiente a la red de la figura 2.38 es un árbol. El grafo de la figura 2.39 también es un árbol, para comprobarlo basta con notar que entre cualquier par de vértices existe un único camino.

En computación generalmente usamos una clase particular de árboles en los que un vértice particular se distingue de los demás, a este vértice se le llama **raíz** del árbol.

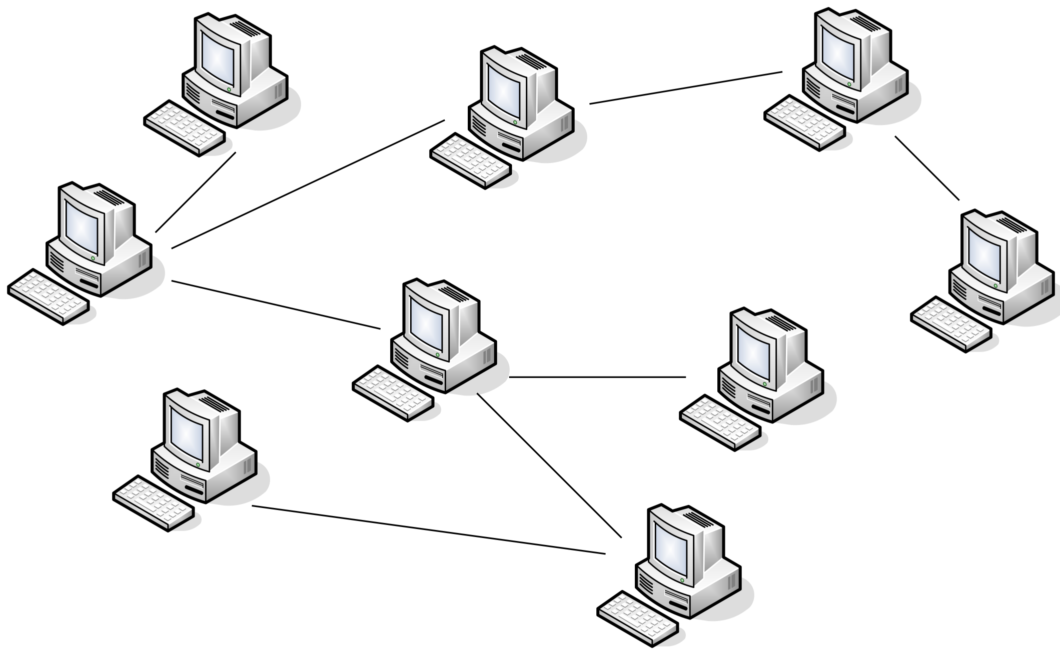


Figura 2.38: Una red de computadores con el mínimo número de conexiones directas.

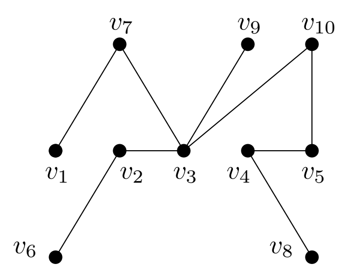


Figura 2.39: Un árbol.

Def: Un **árbol con raíz** (o árbol enraizado, *rooted tree* en inglés) es un árbol $T = (V(T), E(T))$ en que uno de sus vértices $r \in V(T)$ se ha distinguido de los demás. Al vértice distinguido r se le llama **raíz** del árbol. Los vértices en un árbol (con o sin raíz) que tienen grado igual a 1 se llaman **hojas** (también consideraremos hoja a un vértice de grado 0). Definiremos más conceptos más adelante.

Cuando dibujemos un árbol con raíz, el vértice correspondiente a la raíz se dibujará siempre "arriba" y los vértices hoja se dibujarán "abajo". El nombre de árbol se ve motivado por el resultado de dibujar un grafo de estas características, como se puede ver en la figura 2.40.

Los siguientes teoremas nos entregan caracterizaciones de árboles, formas alternativas de definirlos.

Teorema 2.3.1: Un grafo T es un árbol si y sólo si T es conexo y no tiene ciclos.[1]

Demostración: ($\Rightarrow$) Primero si T es un árbol es por definición conexo, nos falta demostrar entonces que un árbol no puede tener ciclos. Supongamos que T tuviese un ciclo, y sea C un ciclo en T que pasa por los vértices u y v. Supongamos que C parte (y termina) en u, entonces C es de la forma $(u, \ldots, v, \ldots, u)$, por lo que se puede dividir en dos porciones, una para ir de u a v, digamos p_1, y otra (distinta ya que un ciclo no repite aristas) para ir de v a u, digamos p_2. Resulta entonces que p_1 y p_2 son dos caminos distintos entre u

[1] Aquí nos referimos claramente a ciclos compuestos por más de un único vértice, o sea, a ciclos de largo mayor a 0.

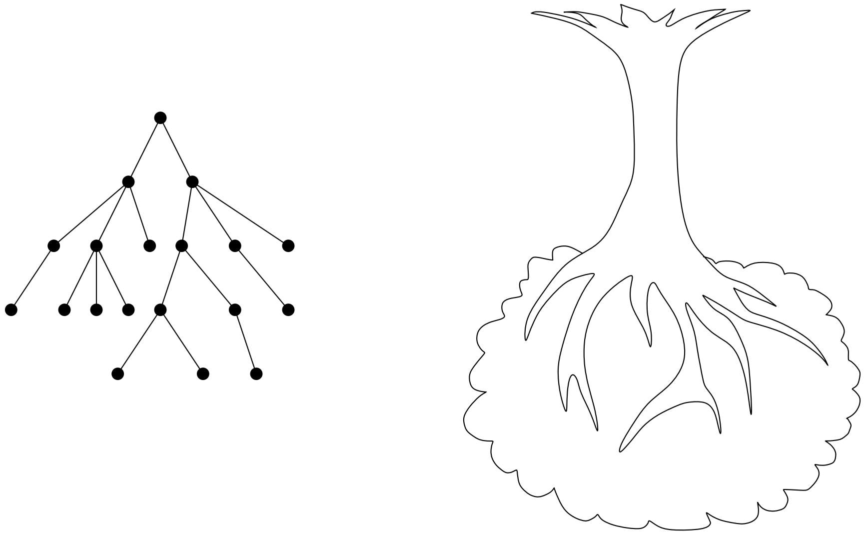


Figura 2.40: Árboles (izquierda) en computación y (derecha) en el mundo real... bueno, casi.

y v en T, lo que contradice el hecho de que T es un árbol. La figura 2.41 muestra un diagrama del anterior argumento. Finalmente T no puede tener ciclos.

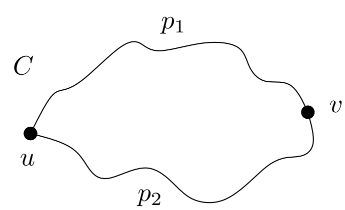


Figura 2.41: Formación de dos caminos distintos entre u y v a partir de un ciclo que los contiene.

($\Leftarrow$) Como T es conexo, para cada par de vértices existe un camino que los une, falta demostrar que ese camino es único. Supongamos entonces que T no tiene ciclos pero que sin embargo existe un par de vértices con dos caminos distintos uniéndolos en T. Sea u y v estos vértices y sean p_1 y p_2 los dos caminos distintos en T que unen a u con v. Dado que estos caminos son distintos entonces ambos tienen al menos tres vértices. Sea x el vértice anterior al primer vértice que diferencia a p_1 y p_2 (note que x_1 está en p_1 y en p_2). Sea y el vértice siguiente a x que pertenece simultáneamente a p_1 y p_2. Un diagrama de esto se ve en la figura 2.42. El camino entre x e y a través de p_1 junto con el camino entre x e y a través de p_2 forman un ciclo en T lo

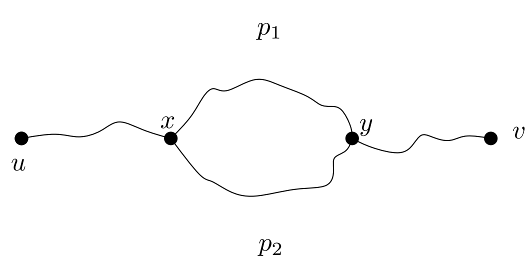


Figura 2.42: Dos caminos distintos entre el mismo par de vértices siempre contienen un ciclo.

que contradice nuestra hipótesis de que T no tiene ciclos. Finalmente no pueden existir dos caminos distintos entre u y v, de donde concluimos que para todo par de vértices en T existe un único camino que los une y por lo tanto T es un árbol. □

Corolario 2.3.2: Un grafo T es un árbol si y sólo si todas sus aristas son de corte, o sea, para cualquier arista e el grafo $T - e$ no es conexo.

Demostración: ($\Rightarrow$) En la sección anterior (teorema 2.2.2) demostramos que una arista es de corte si y sólo si no pertenece a ningún ciclo en el grafo. Ahora, T es un árbol si y sólo si T es conexo y no tiene ningún ciclo, si y sólo si todas sus aristas cumplen con la propiedad de no pertenecer a un ciclo, si y sólo si, todas sus aristas son de corte. □

Este corolario debió ser evidente, dado que un árbol tiene el mínimo número de aristas necesarias para que el grafo sea conexo, el sacar cualquier arista desconecta al grafo. El siguiente corolario no es tan evidente.

Corolario 2.3.3: Todo árbol es un grafo bipartito.

Demostración: En la sección anterior (teorema 2.2.4) demostramos que un grafo es bipartito si y sólo si no contiene ciclos de largo impar. Si T es un árbol T no contiene ningún ciclo y por lo tanto T es bipartito. □

La siguiente es una propiedad muy simple de los árboles lo que nos permitirá hacer demostraciones por inducción sobre ellos.

Lema 2.3.4: Sea T un árbol y v una hoja de T, entonces el grafo $T - v$ es también un árbol.

Demostración: Para demostrar que el grafo $T - v$ es un árbol debemos comprobar que para cualquier par de vértices en $T - v$, existe un único camino que los une. Sea u y w dos vértices en T distintos de v, y sea la secuencia $P = (u, u_1, u_2, \dots, u_n, w)$ el único camino en T que une a u con w. Es claro que el vértice v no aparece en P ya que todos los vértices de P (excepto u y w) deben tener grado al menos 2, luego si eliminamos v de T no afecta al camino entre u y w, luego el camino $P = (u, u_1, u_2, \dots, u_n, w)$ entre u y w en $T - v$. Como la demostración la hicimos en general para un par de vértices cualquiera, en $T - v$ existe un único camino entre todo par de vértices y por lo tanto $T - v$ también es un árbol. □

Con el anterior lema podemos demostrar el siguiente teorema que es una caracterización muy simple de un árbol

Teorema 2.3.5: Un grafo T con n vértices es un árbol si y sólo si es conexo y tiene exactamente $n - 1$ aristas.

Demostración: ($\Rightarrow$) Si T es un árbol con n vértices, entonces claramente es conexo, falta demostrar que tiene exactamente $n - 1$ aristas, lo haremos por inducción en n.

- **B.I.** Si $n = 1$ tenemos un árbol con sólo un vértice y sin aristas, por lo que se cumple la propiedad (la cantidad de aristas es igual a $0 = 1 - 1 = n - 1$).
- **H.I.** Supongamos que un árbol con n vértices tiene exactamente $n - 1$ aristas.
- **T.I.** Sea ahora T un árbol con $n+1$ vértices, queremos demostrar que T tiene exactamente $(n+1)-1 = n$ aristas. Centrémonos en una hoja v cualquiera de este árbol. Por el lema anterior $T - v$ también es un árbol y tiene exactamente n vértices por lo que se aplica la HI, luego $T - v$ tiene exactamente $n - 1$ aristas. Dado que v es una hoja, v tiene grado 1 en T y por lo tanto T tiene exactamente una arista más que $T - v$, o sea T tiene exactamente n aristas, por lo que se cumple la propiedad.

Por inducción simple se sigue que todo árbol con n vértices tiene exactamente $n - 1$ aristas.

($\Leftarrow$) En la sección anterior demostramos en el teorema 2.2.1 que un grafo con n vértices y k aristas tiene al menos $n - k$ componentes conexas. Si T es un grafo conexo con n vértices y exactamente $n - 1$ aristas y tomamos una arista e cualquiera de T, entonces dado que $T - e$ tiene $n - 2$ aristas, por el teorema 2.2.1,

$T - e$ tiene al menos dos componentes conexas y por lo tanto e es una arista de corte. Dado que elegimos e como una arista cualquiera, T cumple con que todas sus aristas son de corte y por lo tanto (corolario 2.3.2) T es un árbol.

□

En nuestro ejemplo inicial nos preguntábamos por la mínima cantidad de conexiones directas necesarias para hacer una red de computadores de manera tal que cualquier computador pudiera enviar información a cualquier otro en la red. Dado que esto se logra cuando el grafo asociado a la red es un árbol, el teorema anterior nos dice que si tenemos n computadores, la mínima cantidad de estas conexiones será $n - 1$. Este teorema nos da también una forma rápida de chequear si un grafo conexo es o no un árbol, sólo debemos verificar que tenga exactamente una arista menos que vértices.

2.3.2. Árboles en Computación

Ya mencionamos que en computación generalmente usamos árboles con raíz. Esta noción motiva varias otras definiciones sobre árboles en aplicaciones computacionales.

Def: Sea $T = (V(T), E(T))$ un árbol con raíz r, note que, dado que T es un árbol, para cada vértice $x \in V(T)$ existe un único camino de r a x. Definimos lo siguiente:

- El largo del único camino entre r y un vértice cualquiera x se llama **profundidad** de x. La raíz r tiene profundidad 0.
- El máximo de las profundidades de todos los vértices de T se llama **altura del árbol** (o **profundidad del árbol**).
- La **altura** de un vértice x es la altura del árbol menos la profundidad de x.
- El conjunto de vértices que aparecen en el único camino de r a x se llaman **ancestros** de x. Note que x es un ancestro de sí mismo. A veces usaremos el término ancestros propios de x para referirnos a los ancestros de x distintos de x.
- El ancestro propio de x de mayor profundidad se llama **padre** de x. Un vértice x es **hijo** de su padre. Note que r no tiene padre y que los vértices hoja no tienen hijos.
- Si dos vértices x e y tienen el mismo padre diremos que x e y son **hermanos**.

La figura 2.43 muestra gráficamente estas definiciones.

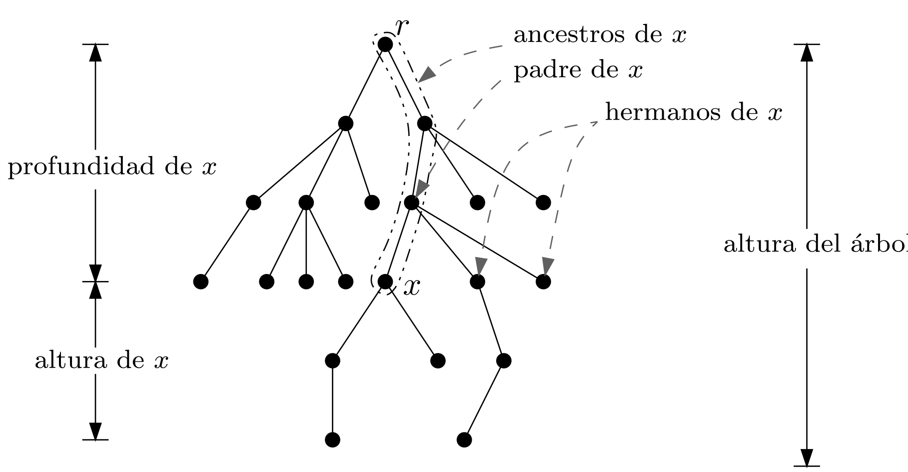


Figura 2.43: Definiciones sobre un árbol

Muchas aplicaciones de ordenamiento y búsqueda en computación usan estructuras de datos basadas en un tipo especial de árboles, los llamado árboles binarios.

Def: Un árbol con raíz se dice **árbol binario** si todo vértice tiene a lo más grado 3, o equivalentemente (y cómo se usa más comunmente en computación), si todo vértice tiene a lo más 2 hijos. En un árbol binario, los hijos de un vértice particular se distinguen en **hijo izquierdo** e **hijo derecho**, lo que también normará las posiciones en las que se dibujen los hijos de cierto vértice.

Note que si a un árbol binario se le elimina una hoja, el grafo que resulta es también un árbol binario (la demostración se deja como ejercicio).

Estudiaremos algunas propiedades de los árboles binarios y veremos un caso particular muy útil de este tipo de árboles.

Teorema 2.3.6: La cantidad de hojas de un árbol binario es uno más la cantidad de vértices con exactamente dos hijo.

Demostración: Usaremos la propiedad descrita con anterioridad acerca de que un árbol binario al que se le saca una hoja sigue siendo binario, con esto haremos una demostración por inducción en la cantidad de vértices del árbol binario.

B.I. El caso base es un árbol compuesto por sólo un vértice, la raíz. Un árbol de estas características tiene sólo una hoja y ningún vértice con dos hijos, luego cumple la propiedad.

H.I. Supongamos que un árbol binario con n vértices tiene una hoja más que vértices con dos hijos.

T.I. Sea T un árbol binario con $n+1$ vértices, queremos demostrar que T tiene una hoja más que vértices con dos hijos. Sea v una hoja cualquiera de T, sabemos que $T-v$ es también un árbol binario y tiene exactamente n vértices (uno menos que T) por lo que $T-v$ cumple con HI, o sea tiene una hoja más que vértices con dos hijos. Supongamos que $T-v$ tiene k vértices con dos hijos entonces por HI tiene $k+1$ hojas ¿qué podemos decir de T? Lo que podamos decir dependerá de si v (la hoja que le sacamos) tenía o no un hermano.

- Si v tiene un hermano en T, entonces el padre de v es un vértice con dos hijos en T. Ahora, en el árbol $T-v$, el vértice que era padre de v tiene sólo un hijo. Lo anterior quiere decir que T tiene exactamente un vértice más con dos hijos que $T-v$, o sea que T tiene exactamente $k+1$ vértices con dos hijos. Ahora también ocurre que T tiene exactamente una hoja más que $T-v$, o sea que T tiene $k+2$ hojas. Hemos concluido que T tiene $k+2$ hojas y $k+1$ vértices con dos hijos y por lo tanto cumple con la propiedad.
- Si v no tiene hermano, entonces el vértice padre de v en T se convierte en una hoja en el árbol $T-v$, lo que quiere decir que T y $T-v$ tienen exactamente la misma cantidad de hojas, $k+1$. El único vértice que ve afectado su cantidad de hijos en $T-v$ es el padre de v, este tiene exactamente un hijo en T y 0 hijos en $T-v$ por lo que la cantidad de vértices con dos hijos en T es también la misma que en $T-v$ e igual a k. Hemos concluido que T tiene $k+1$ hojas y k vértices con dos hijos y por lo tanto cumple con la propiedad.

Por inducción en la cantidad de vértices se sigue que todo árbol binario tiene exactamente una hoja más que vértices con dos hijos.

□

Ejemplo: Un torneo de eliminación simple es un torneo en que cada competencia se realiza entre dos participantes, el participante que pierde la competencia se va del torneo. El último participante que permanece en el torneo es el ganador. Un ejemplo típico de un torneo de eliminación simple es un campeonato de tenis.

Un torneo de eliminación simple se puede representar por un árbol binario, los participantes iniciales corresponden a las hojas del árbol, el padre de dos vértices corresponde al participante ganador en la competencia realizada entre sus dos hijos, y el participante representado por la raíz del árbol corresponde al ganador del

Figura 2.44: Cuartos de final del campeonato mundial de futbol.

Figura 2.45: Un torneo de eliminación simple en general con participantes libres en ciertas rondas.

torneo. Los "niveles" del árbol, o sea las distintas alturas de los vértices, representan las "rondas" del torneo, y vértices hojas que no se encuentre a altura 0 corresponden a participantes que pasan "libre" algunas rondas (esto suele ocurrir por ejemplo en los torneos de tenis). En la figura 2.44 se ve el resultado de un "hipotético" campeonato mundial de futbol desde los cuartos de final, representado por un árbol binario. En la figura 2.45 se ve un torneo más general con participantes que pasan libres ciertas rondas.

Queremos responder la siguiente pregunta, si tenemos n participantes en un torneo de eliminación simple ¿cuántos partidos (competencias entre dos participantes) en total se deberán realizar durante el torneo? Se podría pensar que esto dependerá de la organización particular del torneo, sin embargo, si nos fijamos que cualquier torneo se puede representar por un árbol binario podremos responder esta pregunta independiente de las decisiones de la organización. Ya hemos dicho que cada hoja del árbol asociado corresponde a un participante inicial en el torneo. Si un vértice del árbol tiene dos hijos, este corresponde al ganador de un partido entre sus dos vértices hijos, por lo que podríamos asociar un partido con cada vértice que tiene dos hijos en el árbol. Finalmente, el teorema anterior nos dice que si el un árbol hay n hojas, entonces la cantidad de vértices con dos hijos es exactamente $n-1$, luego $n-1$ será la cantidad de partidos necesaria realizar durante el campeonato. Por ejemplo, 100 participantes del torneo darán lugar a 99 partidos en total.

Una clase especial de árboles binarios nos ayudan para establecer casos límites y cotas para algunos algoritmos de búsqueda, estos son los árboles binarios completos que definiremos a continuación.

Def: Diremos que T es un **árbol binario completo** si es un árbol binario que cumple con que todos sus vértices que no son hoja tienen exactamente dos hijos y que todas las hojas se encuentran a la misma profundidad.

Si T es un árbol (no necesariamente binario completo) y v es un vértice cualquiera de T, el **sub-árbol con raíz en** v será el sub-grafo de T compuesto por v, todos los vértices que tienen a x como ancestros, y todas las aristas que conectan a estos vértices. Es claro que un sub-árbol es también un árbol, que un sub-árbol de un árbol binario es también un árbol binario, y que un sub-árbol de un árbol binario completo es también un árbol binario completo.

Teorema 2.3.7: Un árbol binario completo de altura H tiene exactamente 2^H hojas

Demostración: Por inducción en la altura del árbol □

Corolario 2.3.8: Un árbol binario completo de altura H tiene exactamente $2^{H+1}-1$ vértices.

Demostración: 2^H hojas $\Rightarrow 2^H-1$ vértices con dos hijos $\rightarrow 2^H+2^H-1=2^{H+1}-1$ vértices en total. □

Corolario 2.3.9: En un árbol binario completo con n vértices su altura es menor o igual que $\log_2(n)$.

Demostración: Si la altura es H, claramente $n \geq 2^H \Rightarrow \log_2(n) \geq H$. □

2.3.3. Grafos en Computación

Generalmente en computación nos interesa resolver problemas de optimización sobre grafos para esto el modelo de grafos se extiende un poco. Las siguientes definiciones introducen el tema.

Def: Un **grafo con peso** es una estructura $G=(V(G),E(G),w)$ donde $V(G)$ es un conjunto de vértices, $E(G)$ es un conjunto de aristas (tal como en un grafo normal) y w es una función de *peso* (o *costo*) $w: E(G) \rightarrow \mathbb{N}$ que a cada arista e de G le asigna un valor que llamaremos peso $w(e)$ (a veces la función puede tener otro conjutno de llegada como $\mathbb{Z}$ o incluso $\mathbb{R}$). Cuando dibujamos un grafo con peso, generalmente etiquetamos cada arista con su peso correspondiente.

El peso o costo de un camino (caminata, ciclo) en un grafo con peso, es la sumatoria de los pesos de las aristas que componen el camino (caminata, ciclo). Cuando queremos representar un grafo con peso usando una matriz, usamos una matriz en que cada posición tiene el peso de la arista correspondiente, 0's en la diagonal, e ∞ si la arista correspondiente no existe (representando que la arista existe pero su peso es demasiado alto como para ser tomada en cuenta).

En la figura 2.46 se muestra un ejemplo de grafo con peso cuya matriz asociada es:

$$\begin{bmatrix} 0 & 9 & 6 & 2 & \infty & 3 \\ 9 & 0 & 2 & \infty & 3 & 8 \\ 6 & 2 & 0 & 4 & 4 & 1 \\ 2 & \infty & 4 & 0 & 7 & \infty \\ \infty & 3 & 4 & 7 & 0 & 5 \\ 3 & 8 & 1 & \infty & 5 & 0 \end{bmatrix}$$

Sobre un grafo como este se pueden hacer muchas preguntas de optimización como por ejemplo cuál es el camino de menor peso total entre un par de vértices. En la figura para los vértices u y v, la respuesta es

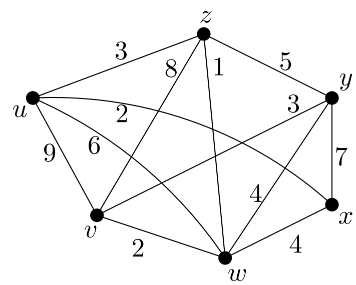


Figura 2.46: Ejemplo de un grafo con peso.

(u, z, w, v) de peso total 6. El anterior problema suele llamarse el problema del camino más corto. Cuando estudiamos árboles los introducimos como el grafo con menos aristas que mantiene la propiedad de ser conexo, nos podríamos preguntar en este contexto cuál es el árbol asociado al grafo que tiene menor peso total (siendo el peso total del árbol la suma de los pesos de todas las aristas). En el grafo de la figura el árbol es el compuesto por las aristas zw, ux, vw, uz, vy. El anterior problema suele llamarse el problema del árbol de cobertura de costo mínimo. Para los dos problemas anteriores existen métodos eficientes que los resuelven, estos métodos suelen estudiarse en detalle en un curso de Algoritmos y Estructuras de Datos. Otra pregunta que puede hacerse es acerca del ciclo Hamiltoniano de peso total mínimo, sin embargo nuevamente ocurre que para este problema no existe un algoritmo eficiente.

Existen otros tipos de grafos que se usan generalmente para problemas de búsqueda.

Def: Un **grafo dirigido** es una estructura $G = (V(G), E(G))$ muy similar a un grafo normal, pero en que la relación "ser vecino de" no necesariamente es simétrica, de echo en un gafo dirigido el conjunto $E(G)$ es un conjunto de pares ordenados de vértices, $E(G) \subseteq V(G) \times V(G)$.

Las definiciones de camino y ciclo se extienden a **camino dirigido** y **ciclo dirigido**.

En un grafo normal (no dirigido) estudiamos el problema de la componente conexa. En un grafo dirigido podemos estudiar un problema similar, cuál conjunto de vértices cumplen la propiedad de que entre cada par de ellos existe un camino dirigido. Este es el problema de encontrar las componentes **fuertemente conexas** de un grafo.

Se pueden hacer modelos que combinen las dos definiciones anteriores, grafos dirigidos con peso. En el se pueden plantear problemas como el de camino dirigido más corto, flujos máximos (suponiendo que el peso de una arista dirigida es la capacidad de un canal), etc.

(Falta completar...)