Maestría en Ciencia de Datos del ITAM

Teoría de Grafos para Análisis de Datos

Ricardo Mansilla

17 de enero de 2016

Índice

Introducción

Con la rápida expansión de internet y aparición reciente de nuevas tecnologías que engendrán un volumen masivo de datos, tanto a nivel personal como colectivo, la industria ha reconocido la necesidad de dedicar cada vez mas recursos a la creación de tecnologías y técnicas de análisis de datos para procesar estos grandes volúmenes con baja latencia.

La aplicación de teorías y métodos de uso limitado al estudio de problemas puramente académicos en casos de aplicación real son cada vez mas frecuentes. Parte importante de estos problemas consiste en capturar en estructuras tan compactas y eficientes como sea posible, la interacción y correlación de los elementos (datos) que forman parte de nuestro sistema a estudiar. Las gráficas han demostrado ser en más de un campo de la matemática abstracta una de dichas estucturas extremadamente eficiente. Es por eso que algunos profesionales de la Ciencia de Datos han reconocido el poder que implica su uso y cada vez con más frecuencia las involucran en sus modelos. Problemas importantes del *machine learning* y de la ciencia de datos en general se basan en el uso de estructuras de grafos.

En este curso pretende establecer un camino posible (puesto que en la literatura no existe) para definir y establecer la teoría necesaria en el análisis de datos y el machine learning usando estructuras gráficas.

1. Teoría de Grafos

En muchos problemas que aparecen como casos de estudios en el mundo real, es necesario modelar la interacción entre los entes que forman parte del sistema a estudiar. Por ejemplo, algunos de estos casos podrían consistir en como se relacionan usuarios de un servicio *online* cualquiera. La manera en que se conectan los autores de artículos académicos a través de las citas mutuas que hacen en los mismos. O en un caso mas general, la proximidad bajo alguna medida de distancia que pueden tener dos puntos en algún espacio métrico. En todos ellos, tenemos una colección de entes, que llamamos vértices o nodos, y realciones entre ellos, que pueden verse como una colección de segmentos abstractos que unen a estos entes y modelan su relación, estas son llamadas aristas o flechas. El estudio de las estructuras abstractas de este tipo es lo que llamamos Teoría de Grafos.

Las relaciones entre nuestros entes pueden ser simétricas o no. Es decir, dado un conjunto V y una relación

$$R = \{(a,b) \mid a,b \in V\}$$

se dice que esta es simétrica si para todo $(a,b) \in R \Rightarrow (b,a) \in R$. En otras palabras si la relación es mutua. Por ejemplo, la relación "*padre de*" entre un conjunto de familiares no es simétrica. Puesto que si *A es padre de B*, entonces *B no es padre de A*. Sin embargo la relación "*amigo de*" obviamente lo es. La estructura de grafos que modela una relación no simétrica es una **gráfica dirigida**.

1.1. Grafos y subgrafos

1.1.1. Grafos

De manera simple y suficientemente formal podemos definir una grafo como

Definición 1.1. *Sea un conjunto de elementos V y un conjunto de pares E de elementos de V, entonces definimos una grafo como el par ordenado $G = (V,E)$.*

Dicho de otra forma, sea el conjunto V y el conjunto $E = \{x,y \mid x,y \in V\}$. Entonces definimos $G = (V,E)$.

Al conjunto V se le llama conjunto de vértices y a E el conjunto de aristas.

En general tomaremos el conjunto V como un conjunto finito de elementos a menos que se indique lo contrario.

Los elemento de cada par ordenado que forma una arista se llaman los **extremos** de la arista. Notemos entonces que cada arista está definida por sus extremos unívocamente. Y que además, no podemos tener aristas que tengan extremos fuera del conjunto V.

En la literatura se hace referencia a que la palabra "*grafo*" proviene del hecho de que la estructura anterior puede ser representada de forma gráfica dibujando un punto por cada vértice en V y uniéndolos a través de una linea por cada arista que existe en E. Es posible además encontrar referencias a los términos **gráfica** o **red**.

1.1.2. Subgrafos

Usando la terminología anterior podemos definir lo que es un subgrafo.

Definición 1.2. *Sea $G = (V, E)$, una gráfica (lo que implica que V es un conjunto de vértices y E de aristas). Si tomamos $V_s \subset V$ y $E_s \subset E$ de manera que*

$$\forall e \in E_s, e = (x, y) \Rightarrow x, y \in V_s$$

entonces $G_s = (V_s, E_s)$ es un subgrafo de G.

Dicho de otra manera, si tomamos un subconjunto V_s del conjunto de vértices y un subconjunto E_s de aristas de forma que cada arista en E_s contenga sus extremos en V_s, entonces el grafo formado por $G_s = (V_s, E_s)$ es un subgrafo de G.

Es bastante claro que el requerimiento de que los extremos de cada elemento en E_s estén contenidos en V_s puesto que esta definición necesita ser compatible con la aterior.

Por otro lado, es importante notar, que para que un grafo S sea subgrafo de G (denotado como $S \subseteq G$), es necesario que $\forall e \in S \Rightarrow e \in G$. Es decir, un subgrafo no puede tener aristas que no existan en el grafo original.

1.1.3. Matrices de adyacencia e inicidencia

Existen otras representaciones de los grafos que son bastante comunes en la literatura. Estas representaciones son menos intuitivas pero bastante mas eficiente cuando uno esta haciendo cómputo con grafos. Ambas son representaciones matriciales. La primera de ellas es conocida como la **matriz de adyacencia**. La matriz de adyacencia es básicamente una matriz cuadrada, de entradas binarias, con tantas filas como nodos. Las entradaas de esta matriz son **1**'s si los vértices correspondientes estan conectados y **0**'s en caso contrario.

Dicho de manera mas formal

Definición 1.3. *Sea $G = (V, E)$ un grafo. Entonces puesto que V es finito podemos ordenar sus elementos en una secuencia $v_1, v_2, \ldots, v_n$, lo que nos permite construir la matriz cuadrada M_G de dimensión $n \times n$. En dicha matriz tenemos que*

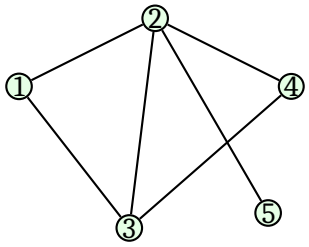
$$M_G(v_i v_j) = 1 \Leftrightarrow (v_i, v_j) \in E$$

y $M_G(v_i v_j) = 0$ en caso contrario. De esta forma tenemos una definición biunívoca de una matriz "equivalente" al grafo G.

La palabra equivalencia debe usarse con cuidado. En general lo anterior es cierto, pero hay que tener en cuenta el límite del significado de equivalencia en la teoría que se está trabajando.

Es importante notar que cada fila de esta matriz tiene tantos **1**'s como aristas inciden en el vértice, es decir, la fila i tiene tantos **1**'s como elementos de E tengan a v_i como extremo. Esta propiedad es fundamental para el desarrollo posterior de nuestra teoría.

Supongamos que tenemos el siguiente grafo



Entonces tenemos una matriz de adyacencia equivalente

$$\mathrm{M}_G = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{pmatrix} 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \end{pmatrix} \end{array}$$

La matriz que se muestra arriba tiene todos los elementos de su diagonal iguales a **1**. Esto es un convenio que se establece en la teoría de grafos usual, significando que cada vértice esta unido consigo mismo. A veces sin embargo es conveniente establecer este elemento en la diagonal como **0**. Es decir, escribiríamos

$$\mathrm{M}_G = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

Nótese también que estas matrices son simétricas ($M_G = M_G^t$), ya que la relación codificada en el grafo también lo es, es decir el grafo no es dirigida.

Existe otra matriz de gran importancia y es la **matriz de incidencia**. Esta matriz nos dice como se relacionan los vértices y las aristas, es decir, nos permite codificar en una estructura algebráica única las etiquetas de ambos conjuntos. Usando el mismo ejemplo de la gráfica anterior y suponiendo que tenemos sus aristas ordenadas, la matriz correspondiente sería

$$\mathrm{I}_G = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{cccccc} e1 & e2 & e3 & e4 & e5 & e6 \end{array} \\ \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \end{array}$$

Es importante ver que esta matriz debe tener en cada columna dos y solamente dos **1**'s puesto que cada arista tiene solo dos extremos. De nuevo, el numero de **1**'s en cada fila nos dice la cantidad de aristas que tienen a dicho vértice como extremo.

Ambas matrices son extremadamente útiles para codificar una estructura de grafo en una unidad de cómputo. La equivalencia entre cada grafo y sus matrices correspondientes es de gran utilidad pues muchos de los resultados mas conocidos y usados han sido probados a través de las matrices de adyacencia correspondientes. El área que estudia esto es conocida como Teoría Algebraica de Grafos.

1.1.4. Caminos y ciclos

Los grafos son buenos entre otras cosas para modelar la dinámica de algunos sistemas. Muchas de estas técnicas exigen la existencia de definiciones formales que permitan establecer a los nodos como "estados" y a las aristas como la posibilidad de "cambiar de estado" lo que se encarga de generar la dinámica buscada.

Definamos primero una simple función que nos será de utilidad mas adelante

Definición 1.4. *Sea $\phi_G : E \longrightarrow V$, de manera que dada $e = (v_1, v_2) \in E$, $\phi_G(e) = \{v_1, v_2\} \subset V$. Es decir la función que envía cada aristas en sus extremos. A esta la llamaremos* **aplicación de extremos** *en G.*

Definición 1.5. *Sea $G = (V, E)$ un grafo, y $\alpha = e_1 e_2 \dots e_k$ una secuencia ordenada de aristas de G tales que $\forall e_i$,*

$$\phi_G(e_i) \cap \phi_G(e_{i-1}) \neq \emptyset,$$
$$\phi_G(e_i) \cap \phi_G(e_{i+1}) \neq \emptyset$$

y

$$\phi_G(e_i) \cap \phi_G(e_{i-1}) \neq \phi_G(e_i) \cap \phi_G(e_{i+1}).$$

Esto es lo mismo que decir que en la secuencia α cada arista comparte exactamente un vértice con la próxima en la secuencia, distinto del que comparte con la anterior (si existe). A esto lo llamamos un **camino** *en G.*

La longitud de un camino es la antidad de aristas que forman parte de él, y se denota como $l(\alpha)$

Definición 1.6. *Si en la definición anterior todas las e_i's son distintas, entonces se dice que α es un* **recorrido** *o* **circuito**.

En general no haremos diferencia entre caminos y circuitos teniendo en cuenta que para fines prácticos solo nos interesan los circuitos (caminos sin repeticiones), que nosotros llamamos caminos.

Definición 1.7. *Sea $G = (V, E)$ un grafo y $\alpha = e_1 e_2 \dots e_k$ un camino en G, tomemos*

$$\{o_\alpha\} = \phi(e_1) - \phi(e_2)$$

y

$$\{f_\alpha\} = \phi(e_k) - \phi(e_{k-1})$$

*entonces se dice que el camino α "***conecta** o_α **con** l_α *" o que "***va desde** o_α **hasta** l_α *". Lo anterior podemos denotarlo como*

$$\alpha : e_1 \sim e_k$$

Nos podemos referir de muchas formas a lo anterior, pero la idea básica es que cada camino empieza en un vértice y termina en otro. Es claro que puede existir más de un camino que conecten dos vértices. Por otro lado, hay un hecho importantísimo que tiene que ver con el conjunto de dichos caminos

Definición 1.8. *Sean $G = (V, E)$ un grafo, $v_1, v_2 \in E$ y $C = \{\alpha \mid \alpha : v_1 \sim v_2\}$. Entonces existe α_0, tal que $\forall \alpha \in C$, $l(\alpha_0) \leq \alpha$. Este camino α_0 es llamado* **camino mínimo** *de v_1 a v_2.*

Más adelante veremos un algoritmo para calcular el camino mínimo, por ahora pensemos en la posibilidad de que un camino regrese en algún momento a su origen.

Definición 1.9. *Sea $G = (V, E)$ un grafo y $\alpha = e_1 e_2 \dots e_k$ un camino en G, entonces si $o_\alpha = f_\alpha$ se dice que α es un* **ciclo**.

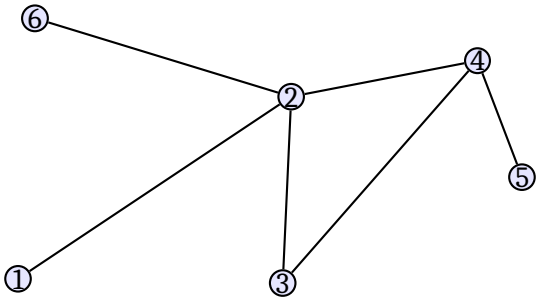
Por último daremos una definición estructural de suma importancia aprovechando la de ciclo

Definición 1.10. *Sea G un grafo. Si G no tiene ciclos entonces se dice que G es un* **árbol**.

Para enunciar algunas de las propiedades básicas de un grafo necesitamos definir un concepto de suma importancia conocido como el **grado de un vértice**.

Definición 1.11. *Sea $G = (V, E)$ un grafo y $v \in V$. Sea además $k = |\{e \in E \; v \in \phi(e)\}|$, es decir el número de aristas que tienen a v como extremo. Se dice entonces que k es el* **grado** *de v, y se denota como $\sigma(v) = k$.*

Examinemos el siguiente grafo y veámos algunas de las propiedades que hemos definido



El vértice **2** tiene grado 4, mientras que el **5** tiene solo grado 1. Es interesante notar también, por ejemplo que existen solo dos caminos que comiencen en el vértice **1** y terminen en **5**. El camino $\alpha = \{(2,3),(3,4),(4,2)\}$ es un ciclo. Por último, si quitáramos la arista $(3,4)$ del grafo tendríamos un árbol.

1.1.5. Propiedades y resultados básicos

Comencemos esta sección con algunas propiedades estructurales de los grafos.

Definición 1.12. *Sea $G = (V, E)$ un grafo, se define la* **distancia** *entre dos vértices $v_1, v_2 \in V$ como la longitud del camino mínimo(1.8) entre v_1 y v_2. Esta distancia se denota como $d(v_1, v_2)$.*

La distancia es de hecho una métrica en el grafo puesto que el camino mínimo es invariante ya que solo recorremos a cada camino en el conjunto posible en dirección contraria, la distancia de cualquier vértice a si mismo es nula a través del camino trivial y la desigualdad del triángulo es consecuencia de la minimalidad de la longitud del camino que da la distancia.

Definición 1.13. *Sea $G = (V, E)$ un grafo y $v \in V$. Se define la* **excentricidad** *de v como*

$$max_w\{d(v, w)\}$$

donde $w \in V - \{v\}$. Es decir, la distancia máxima de v al resto de los vértices. Esto se denota como $\epsilon(v)$.

Intuitivamente uno podría decir que si un vértice tiene la excentricidad mas chica está cerca del "centro" del grafo.

Definición 1.14. *La menor de las excentricidades es el* **radio** *del grafo. Esto es*

$$\rho(G) = min_v\{\epsilon(v)\}$$

con $v \in V$. Por tanto el **centro** *del grafo es el vértice donde se alcanza el* **radio**, *en otras palabras, si*

$$\epsilon(v_0) = \rho(G)$$

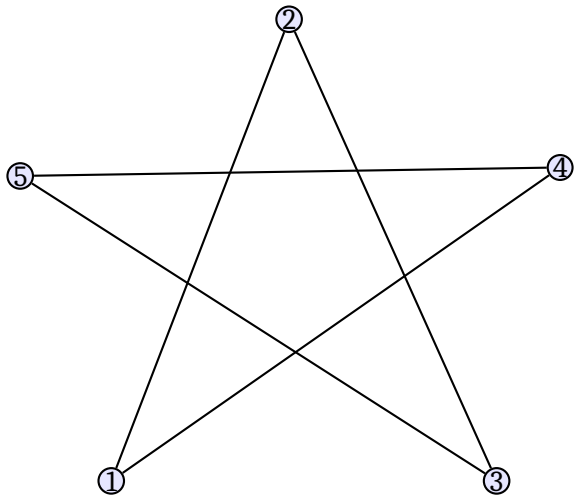
entonces a v_0 se le llama **centro**.

Definición 1.15. *Sea $G = (V, E)$ un grafo, entonces*

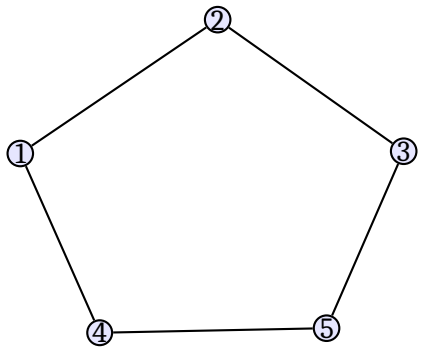
$$\delta(G) = max_v\{\epsilon(v)\}$$

con $v \in V$ es conocido como el **diámetro** *del grafo. La* **circunferencia** *es la longitud del ciclo más largo en G.*

Observemos el grafo siguiente



Este es un ejemplo de grafo **2-regular**, lo que significa que todos sus vértices tienen el mismo grado igual a 2. La excentricidad de cualquiera de sus vértices es 2, por tanto el radio del grafo es también igual a 2, lo que convierte a todos su vértices en centro del grafo. Finalmente la circunferencia es igual a 5, ya que de hecho el grafo es un ciclo de longitud 5, es decir el grafo anterior es "exactamente" el grafo



En matemáticas cuando decimos que dos estrucutras abstractas son la misma, en general se refiere a que son equivalentes bajo alguna aplicación que preserva su estructura. Estas aplicaciones son conocidas como isomorfismos. De manera similar definimos los isomorfimos de grafos

Definición 1.16. *Sean $G_1 = (V_1, E_1)$ y $G_2 = (V_2, E_2)$ dos grafos, entonces se dice que son* **isomorfos** *si existe una funciones γ y ψ que cumplen*

$$\gamma : V_1 \longrightarrow V_2$$

y γ es una biyección de conjuntos,

$$\psi : E_1 \longrightarrow E_2$$

y ψ es tal que

$$e = (v, w),\ \psi(e) = (\gamma(v), \gamma(w))$$

con $(\gamma(v), \gamma(w)) \in E_2$. Cuando $G_1 = G_2$ entonces a (γ, ψ) se le llama **automorfismo**.

En este sentido podemos reescribir lo anterior, y decir que los grafos **2**-regulares exhibidos arriba son isomorfos entre ellos (**ejercicio**). Uno de los problemas mas importantes de la teoría de la computación (por no decir el que más al momento), es equivalente al problema de decidir si dos grafos son isomorfos o no. Lo que sugiere que en general ésta no es tarea fácil.

Además de nuestras propiedades básicas tenemos algunos resultados que relacionan las definiciones que hemos estado estudiando.

Teorema 1.1. *Sea $G = (V, E)$ un grafo, entonces*

$$\sum_{v \in V} \sigma(v) = 2|E|$$

donde $\sigma(v)$ es el grado del vértice v.

En otras palabras, la suma de los grados de un grafo es exactamente dos veces la cardinalidad del conjunto de aristas. La intuición de la prueba es que básicamente cuando contamos el grado de los vértices contamos cada arista exactamente dos vértices, puesto que la arista tiene dos extremos.

De este resultado podemos derivar algunos corolarios

Corolario 1.1. *Sea $G = (V, E)$ un grafo, entonces el número de vértices de grado impar es par.*

Demostración: La demostración es bastante simple usando el resultado anterior. Puesto que

$$\sum_{v \in V} \sigma(v)$$

es un número par, el número de términos impares en la suma debe ser par. □

Corolario 1.2. *Sea $G = (V, E)$ un grafo regular, es decir $\forall v \in V$, $\sigma(v) = k$, con k una constante, entonces*

$$k|V| = 2|E|.$$

□

De la misma forma tenemos cotas superiores e inferiores para el número de aristas basados en los grados de los vértices

Corolario 1.3. *Sea $G = (V, E)$ un grafo, entonces si*

$$max_{v \in V}\{\sigma(v)\} \leq k$$

se cumple

$$k|V| \geq 2|E|.$$

La relación simétrica también es cierta, si tenemos que

$$max_{v \in V}\{\sigma(v)\} \geq k$$

se cumple

$$k|V| \leq 2|E|.$$

□

Sobre los árboles tenemos dos teoremas importantes

Teorema 1.2. *Si G es un grafo sin ciclos (un árbol), entonces entre dos vértices cualesquiera solo existe un camino.* □

Teorema 1.3. *Si G es un árbol entonces se cumple que $|E| = |V| - 1$*

Demostración: La demostración es simple si usamos el principio de inducción sobre el número de vértices. Es decir, supongamos que $|V| = 1$, entonces el resultado es obvio.

Ahora supongamos que el resultado se cumple cuando $|V| = n$. Tomamos un grafo sin ciclos $A = (V_0, E_0)$, con $|V_0| = n+1$ y dos vértices $u, v \in V_0$, tales que $(u,v) \in E_0$. De esta manera, al hacer $E_0 - (u,v)$ tenemos dos subgrafos A_1, A_2, que son a su vez árboles puesto que $\alpha = (u,v)$ es el único camino $\phi(\alpha) = \{u,v\}$ (ya que A es un árbol). Así se cumple

$$E(A_1) = V(A_1) - 1$$

$$E(A_2) = V(A_2) - 1$$

de lo que se deduce

$$E(A) = E(A_1) + E(A_2) + 1 = V(A_1) + V(A_2) - 1 = V(A) - 1.$$

□

1.1.6. Aplicaciones: El problema del camino mínimo

En esta sección veremos una aplicación de los conceptos definidos hasta el momento. La aplicación hace uso de un algoritmo diseñado por Dijkstra en 1959 para encontrar caminos mínimos en un grafo con pesos.

Definición 1.17. *Sea $G = (V,E)$ un grafo y sea $\omega : E \longrightarrow \mathbb{N}$, una función sobre las aristas de G. Entonces a la triada $G_\omega = (V,E,\omega)$ se le llama* **grafo con pesos** *o* **grafo ponderado**.

En general el codominio de ω puede ser cualquier conjunto numérico, pero tomarlo como $\mathbb{N}$ es suficiente. Dicho lo anterior reformulemos un poco algunos conceptos ya definidos.

Definición 1.18. *Sea $G_\omega = (V,E,\omega)$ un grafo ponderado y α un camino en G_ω. Entonces definimos $l_\omega(\alpha)$, la longitud de α como la suma de los pesos de cada arista en α, esto es*

$$l_\omega(\alpha) \sum_{e \in l} \omega(e).$$

De esta manera l_ω define una métrica nueva en G_ω, que es básicamente el camino de longitud mínima (peso mínimo) que una a dos vértices. Escrito formalmente, si

$$C_{uv} = \{\alpha | \phi(\alpha) = \{u,v\}\}$$

entonces

$$d_\omega(u,v) = min_{\alpha \in C_{uv}}\{l_\omega(\alpha)\}.$$

Definición 1.19. *Sea $G_\omega = (V,E,\omega)$ un grafo ponderado y $S \subsetneq V$, esto es, un subconjunto propio de los vértices. Si tomamos $u \in V - S$, entonces definimos la distancia de u a S como*

$$d_\omega(u,S) = min_{v \in S}\{d_\omega(u,v)\}.$$

Un camino mínimo de u a S es $\alpha = u \ldots v$, con $v \in S$, tal que

$$l_\omega(\alpha) = d_\omega(u,S).$$

El problema del camino mínimo es encontrar, dados $u, v \in V$, el camino α con l_w mínimo tal que $\alpha = u \dots v$. O dicho de otra manera, encontrar α tal que

$$\phi(\alpha) = \{u, v\}$$

y

$$l_\omega(\alpha) = d_\omega(u, v).$$

La idea del algoritmo es la siguiente. Supongamos que tenemos un conjunto $S \subsetneq V$, tomemos $u_0 \in S$ y denotemos $\overline{S} = V - S$. Entonces si $P = u_0 \dots u\overline{v}$ es un camino mínimo de u_0 a $\overline{S}$, es obvio que $u \in S$ y el camino $u_0 \dots u$ es un camino mínimo de u_0 a u (por la definición todos los subcaminos de un camino mínimo son mínimos). Entonces es claro que

$$d_\omega(u_0, \overline{v}) = d(u_0, u) + \omega(u\overline{v})$$

por lo que la distancia de u_0 a $\overline{S}$ se puede escribir como

$$d_\omega(u_0, \overline{S}) = min_{u \in S, v \in \overline{S}} \{d(u_0, u) + \omega(u\overline{v})\}.$$

El algoritmo de Dijkstra se basa en la ecuación de arriba para $d_\omega(u_0, \overline{S})$ y hace uso de un proceso iterativo para calcular el camino mínimo de u_0 a todos los vértices.

Empezamos con un conjunto $S_0 = \{u_0\}$ y calculamos $d_\omega(u_0, \overline{S}_0)$, a través de un camino $P_1 = u_0 \dots u_1$, con $u_1 \in \overline{S}_0$. En este punto vale la pena notar que $u_0 u_1$ es un camino mínimo de u_0 a u_1, es decir, u_1 es el otro extremo de la arista de menor peso adyacente a u_0. Dicho de otra forma

$$d_\omega(u_0, \overline{S}_0) = min_{u \in S_0, u_1 \in \overline{S}_0} \{d(u_0, u) + \omega(u u_1)\}$$

pero $\forall u \in S_0$, $d_\omega(u_0, u) = 0$, entonces

$$d_\omega(u_0, \overline{S}_0) = min_{u_1 \in \overline{S}_0} \{\omega(u_0 u_1)\}.$$

Una vez que tengamos tal u_1, hacemos $S_1 = \{u_0, u_1\}$, tomamos P_1 y eso completa el primer paso del proceso iterativo.

En general tendremos en el paso k, conjuntos $S_0 \subset \dots \subset S_k$ y caminos $P_1, \dots, P_k$, con $S_k = \{u_0, \dots, u_k\}$. Para calcular $d_\omega(u_0, \overline{S}_k)$, escribimos

$$d(u_0, \overline{S}_k) = min_{u \in S_k, u_{k+1} \in \overline{S}_k} \{d(u_0, u) + \omega(u u_{k+1})\}$$

lo que nos da un procedimiento para encontrar u_{k+1}. Más aún, vemos en la ecuación de arriba que $u \in S_k$, es decir, $u = u_j$ para algún $0 \leq j \leq k$. Entonces, como ya tenemos $P_j = u_0 \dots u_j$ que es camino mínimo, hacemos $P_{k+1} = u_0 \dots u_j u_{k+1} = P_j + (u_j, u_{k+1})$ y $S_{k+1} = \{u_0, \dots, u_k, u_{k+1}\}$.

Computacionalmente hablando, algunos de los cálculos anteriores pueden ser repetidos, el algoritmo es básicamente una forma eficiente de hacer el procedimiento anterior. Para esto se le asignan etiquetas $l(v)$ para cada $v \in V$, donde cada una de estas es una cota superior de la longitud del camino mínimo de u_0 a v. A medida que el computo de los S_i avanza estas etiquetas se modifican de forma que en el paso k, se cumple que $\forall v \in S_k$, $l(v) = d\omega(u_0, v)$ y

$$l(v) = min_{u \in S_{k-1}} \{d(u_0, u) + \omega(uv)\}$$

para $v \in \overline{S}_k$. Como eventualmente $S_k = V$ para algún k, entonces todas las etiquetas $l(v)$, con $v \in V$, tendrán los calores correctos. Veamos la descripción del algoritmo en pseudo código. Haremos uso de dos elementos importantes, un número M que debe ser suficientemente grande, para esto podemos tomar $M = \sum_{e \in E} \omega(e)$, y de el hecho de que si $(u,v) \notin E$, entonces $\omega(u,v) = M$.

Algoritmo 1 (de Dijkstra para el camino mínimo)

Require: $i = 0, S_0 = \{u_0\}, l(u_0) = 0$ y $l(v) = M, \forall v \in V - \{u_0\}$
  while $i \leq |V| - 1$ **do**
    for $v \in \overline{S}_i$ **do**
      $l(v) \leftarrow min_{l(v), l(u_i)+\omega(u_i v)}$
      $u_{i+1} \leftarrow u \in \{v | l(v) = min_{v \in \overline{S}_i}\{l(v)\}\}$
      $S_{i+1} \leftarrow S_i \cup \{u_{i+1}\}$
    end for
    $i \leftarrow i + 1$
  end while

Cuando el algoritmo se detiene, las etiquetas $l(v)$ contienen la longitud del camino mínimo de u_0 al resto de los vértices. Uno podría ir almacenando los P_j en cada iteración si quisiera tener un registro de los vértices que forman el camino.

1.2. Grafos dirigidos

1.2.1. Matrices de adyacencia e inicidencia

1.2.2. Caminos dirigidos

1.2.3. Ciclos dirigidos

1.2.4. Aplicaciones

1.3. Conectividad

1.3.1. Componentes

1.3.2. Nodos y aristas de corte

1.3.3. Bloques

1.3.4. K-Cores

1.3.5. Otras propiedades

1.3.6. Aplicaciones

1.4. Redes y flujos

1.4.1. Redes

1.4.2. Flujos

1.4.3. Cortes

1.4.4. Aplicaciones

2. Teoría Aleatoria de Grafos

2.1. El modelo de grafos aleatorios

2.1.1. Número de links

2.1.2. Distribución de grados

2.1.3. Aplicaciones

2.2. Propiedades de los grafos aleatorios

2.2.1. Mundo pequeño

2.2.2. Clustering

2.2.3. Redes aleatorias reales

2.2.4. Aplicaciones

3. Teoría Algebraica de Grafos

3.1. Teoría espectral de grafos

3.1.1. Valores propios

3.1.2. Polinomio característico

3.1.3. Aplicaciones

3.2. Grafos regulares

3.2.1. Teoría

3.2.2. Aplicaciones

3.3. Grafos de distancia transitiva

3.3.1. Teoría

3.3.2. Aplicaciones

4. Teoría topológica de Grafos

4.1. Grafos embedidos

4.1.1. Triangulaciones

4.1.2. Simplejos

4.1.3. Aplicaciones: Ejemplo

4.2. Reconstruccion de variedades

4.2.1. Teoría

4.2.2. Aplicaciones: Ejemplo