2014

|  |
| --- |
| Algoritmo de la trayectoria más corta de Edsger Dijkstra |

Grupo 6

Curso K1001

Integrantes:  
Luis Cheng, Legajo: 146.614-8, E-Mail: chengluis24@hotmail.com  
ViIlalba Caceres, Vicente Javier (Curso - K1021), Legajo: 151.683-8,   
E-Mail: acela798@hotmail.com  
Ignacio Nicolás Brandariz, Legajo: 147.124-7, E-Mail: nicobrandariz@hotmail.com  
Frank Bari Davinson Egoavil, Legajo: 149.580-0, E-Mail: baridavinson@yahoo.com

|  |  |
| --- | --- |
| Universidad Tecnológica Nacional Facultad Regional Buenos Aires | Trabajo Práctico – Matemática Discreta |

Algoritmo de la trayectoria más corta de Dijkstra

*Definiciones:*

* *Un* ***grafo pesado*** *es un grafo en el que cada arista tiene un peso, número real positivo asociado.*
* *Un grafo se llama* ***árbol*** *si y solo si, está libre de circuitos y es conexo.*

En 1959 el pionero informático,   
Edsgar Dijkstra, desarrolló un algoritmo para encontrar la trayectoria más corta entre un vértice inicial y un vértice en un grafo pesado en el que todos los pesos son positivos.

En el inicio del algoritmo, a cada vértice u de G (grafo) se le da una etiqueta L(u), que indica la mejor estimación actual de la longitud de la ruta más corta de a hacia u.

L(a) se hace inicialmente igual a 0 ya que el camino más corto de a hacia a tiene longitud cero, pero, debido a que no existe información previa acerca de las longitudes de las trayectorias más cortas de a hacia cualquier otro vértice de G, la etiqueta L(u) de cada vértice u diferente de a se hace inicialmente igual a un número, que se denota por ∞, que es mayor que la suma de los pesos de todas las aristas de G. Como la ejecución de los progresos del algoritmo, hace cambiar los valores de L(u), convirtiéndose en las longitudes reales de la trayectorias más cortas de a hacia u en G.

Porque T (árbol) se construye hacia el exterior desde a, en cada etapa de ejecución del algoritmo los únicos vértices que son candidatos a formar parte de T son los que están junto al menos un vértice de T. Así, en cada etapa del algoritmo de Dijkstra, el grafo G puede pensarse como dividido en tres partes: El árbol T que se está construyendo en el conjunto de vértices “marginales” que son adyacentes al menos a un vértice del árbol y el resto de los vértices de G. Cada franja de vértices es un candidato a ser el siguiente vértice agregado a T. El elegido es aquel para el que la longitud de la trayectoria más corta de a hacia T es un mínimo entre todos los vértices de la franja.

Una observación fundamental subyacente del algoritmo de Dijkstra es que después de cada adición de un vértice de v a T, solo los vértices de la franja para los que una trayectoria más corta de a se pueda encontrar son aquellas que son adyacentes a v [ya que la longitud de la trayectoria de a hacia v fue un mínimo entre todas las trayectorias de a hacia todos los vértices que estaban entonces en la franja]. Así después de cada adición de un vértice de v a T, cada vértice de u de la franja adyacente a v, se examina y se comparan dos números; el valor actual de L(u) y el valor de L(v)+w(v,u), donde L(v) es la longitud más corta a v (en T) y w(v,u) el peso de la arista que une a v con u. Si L(v) + w(v,u) < L(u), entonces el valor de L(u) se cambia a L(v) + w(v,u).

Al comienzo de la ejecución del algoritmo, el árbol solo consta solo del vértice a y L(a) = 0.  
Cuando se termina la ejecución, L(z) es la longitud de la trayectoria más corta de a hacia z.  
(Siendo a el vértice inicial y z el vértice elegido)

*Algoritmo Dijkstra*

***Entrada:*** *G [un grafo conexo simple con un peso positivo para cada arista], ∞ [un número mayor que la suma de los pesos de todas las aristas del grafo], w(u,v)[el peso de la arista (u,v)], a [el vértice inicial], z [el vértice final].*

***Cuerpo del Algoritmo:***

1. *Inicializa T como el grafo con el vértice a y sin aristas. Sea V(T) el conjunto de los vértices de T y sea E(T) el conjunto de aristas de T.*
2. *Sea L(a)=0 y para todos los vértices en G excepto a, sea L(u)= ∞*

*[El número L(x) se llama la etiqueta de x]*

1. *Inicialice v igual a a y F será {a}*

*[El símbolo v se utiliza para denotar el vértice agregado a T]*

1. *While(z no pertenece a V(T))*
   1. *F:=(F-{v})U{vértices que son adyacentes a v y no están en V(T)}*

*El conjunto F se llama la franja. Cada vez que se agrega un vértice a T, se elimina de la franja y se agregan los vértices adyacentes a la franja si ya no están en la franja o en el árbol T]*

* 1. *Para cada vértice u que es adyacente a v y no está en V(T),*

*If L(u) + w(v,u) < L(u) then*

*L(u):= L(v)+w(v,u)*

*D(u) := v*

*[Observe que agregar v a T no afecta las etiquetas de los vértices en la franja F excepto aquellas adyacentes a v. También, cuando L(v) se cambia a un valor menor, se introduce la notación D(u) para realizar un seguimiento de qué vértice en T dio lugar al menor valor]*

* 1. *Encuentre un vértice x en F con la etiqueta más pequeña  
     Agregue el vértice x a V(T) y se agrega una arista (D(x),x) a E(T)*

*v:=x [Este enunciado estable la notación para la siguiente iteración del bucle.]*

***end while***

***Salida:*** *L(z), [L(z) un entero no negativo, es la longitud de la trayectoria más corta de a hacia z ]*

***Nota: La única trayectoria en el árbol T de a hacia z es la trayectoria más corta en G de a hacia z.***