4.5. Caminos más cortos

Definición. Sea G un digrafo (V, E) con pesos positivos en los arcos $w : E \to \mathbb{R}_{>0}$.
Si $P = v_0, v_1, \ldots, v_k$ es un dicamino, su *largo* o *costo* es $w(P) = \sum_{i=1}^{k} w(v_{i-1}v_i)$.

Definición. Sea $G = (V, E)$ un digrafo con pesos positivos w y sean $u, v \in V$. La *distancia* de u a v, denotada $d(u, v)$, es

$$d(u,v) = \begin{cases} \infty & \text{si no existe un dicamino de } u \text{ a } v, \\ \text{mín}\{w(P) : P \text{ es un dicamino de } u \text{ a } v\} & \text{en otro caso.} \end{cases}$$

Proposición. Sea $G = (V, E)$ un digrafo con pesos positivos w. Si existe un dicamino desde s hasta v, entonces existe uno de costo mínimo que no repite vértices.

Definición. Sea $G = (V, E)$ un digrafo con pesos w y sea $U \subseteq V$. Los *vecinos de salida* de U son

$$N^+(U) = \{v \in V \setminus U : \text{existe } u \in U \text{ tal que } (u, v) \in E\}.$$

Definición. Sea $U \subseteq V$ tal que $s \in U$. Para $v \in N^+(U)$ definimos su *costo candidato desde* U como

$$c_U(v) = \text{mín}\{d(s, u) + w(u, v) : u \in U \text{ y } (u, v) \in E\}.$$

Lema. Sea $G = (V, E)$ un digrafo con pesos positivos w. Sea $U \subseteq V$ tal que $s \in U$. Si $v^\star \in N^+(U)$ minimiza $c_U(v)$ entre todos los vértices de $N^+(U)$, entonces $d(s, v^\star) = c_U(v^\star)$.

Algoritmo 1 Dijkstra, versión de frontera

Entrada: un digrafo $G = (V, E)$ con pesos positivos w y un origen $s \in V$
Salida: distancias desde s hacia los vértices de V
$U \leftarrow \{s\}$
$\delta(s) \leftarrow 0$
$\delta(v) \leftarrow +\infty$ para cada $v \in V \setminus \{s\}$
while $N^+(U) \neq \emptyset$ **do**
    Calcular $c_U(v) \leftarrow \text{mín}_{u \in U,\ (u,v) \in E}\{\delta(u) + w(u, v)\}$ para cada $v \in N^+(U)$.
    Elegir $v^\star \in N^+(U)$ que minimice $c_U(v)$
    $\delta(v^\star) \leftarrow c_U(v^\star)$
    $U \leftarrow U \cup \{v^\star\}$
end while
return δ para los vértices de V

Teorema. Sea $G = (V, E)$ un digrafo con pesos positivos w y sea $s \in V$. El algoritmo de Dijkstra calcula correctamente $d(s, v)$ para todo vértice $v \in V$.

Ejercicios

1. Una de las grandes ventajas del problema de camino de costo mínimo es que modela varios problemas. Indique como resolver los siguientes problemas usando un algoritmo de camino más corto.

   a) Una red está modelada por un digrafo $G = (V, E)$. Cada arco e tiene una probabilidad de supervivencia $p(e) \in (0, 1)$, y la probabilidad de supervivencia de un dicamino P es

   $$\prod_{e \in P} p(e).$$

   Dados dos vértices $s, t \in V$, queremos encontrar un dicamino de s a t que maximice esa probabilidad.

   b) Una persona parte en s en el instante 0 y quiere llegar a t a más tardar en el instante T. Al cruzar un arco $e = (u, v)$ en el instante τ, se llega a v en el instante $\tau + 1$ y se paga un costo positivo $c_e(\tau)$, que puede depender de τ. También está permitido esperar un instante en cualquier vértice pagando costo 0. Diseñe un algoritmo que encuentre un dicamino de costo mínimo (en el sentido de esta pregunta) desde s hasta t que llegue a más tardar en el instante T.

2. Sea $G = (V, E)$ un digrafo con pesos no negativos w y sea $s \in V$. Definimos el digrafo $H = (R, A)$ mediante

   $$A = \{(u, v) \in E : u, v \in R \text{ y } d(s, u) + w(u, v) = d(s, v)\}.$$

   El objetivo de esta pregunta es esbozar un algoritmo que encuentre los dicaminos de costo mínimo desde s que usan **la menor cantidad de arcos** posible.

   a) Explique como construir H a partir de G.

   b) Demuestre que un dicamino desde s hasta $v \in R$ tiene costo mínimo si y sólo si todos sus arcos pertenecen a A.

   c) ¿Cómo podemos usar H para encontrar un dicamino de costo mínimo desde s hasta $v \in R$ que use la menor cantidad de arcos posible?

   d) Que combinación de algoritmos conocidos debe usar en su algoritmo.