Algoritmo de Johnson

Guillermo Palma

Universidad Simón Bolívar
Departamento de Computación y Tecnología de la Información

Sobre el algoritmo de Johnson

- Obtiene los caminos de costo mínimo entre todos los pares de vértices en un digrafo.
- Detecta si hay ciclos negativos.
- Es buena opción si es un grafo disperso.
- Si no hay lados negativos, ejecuta el algoritmo de Dijkstra para todos los pares de vértices.
- Si hay lados negativos, se produce un grafo equivalente donde todos los costos sean no negativos y luego se ejecuta Dijkstra para todos los pares de vértices.
- Si en el algoritmo de Dijkstra se usa como cola de prioridad un Fibonacci heap, el tiempo es de $O(|V|^2 \log |V| + |V||E|)$.
- Si en el algoritmo de Dijkstra se usa como cola de prioridad un Min heap, el tiempo es de $O(|V||E| \log |V|)$.
- Es más rápido que Floyd-Warshall, que es $O(|V|^3)$, si $E = o(|V^2|)$.

Sobre el grafo equivalente de costos no negativos

Si el digrafo $G = (V, E)$ tiene lados negativos pero no ciclos negativos, se hacen los lados no negativos con una función de costo $\hat{w}$, que debe satisfacer:

1. Para todo $u, v \in V$, p es un camino de costo mínimo $u \rightsquigarrow v$ usando w, si solo si p es un camino de costo mínimo $u \rightsquigarrow v$ usando $\hat{w}$.
2. Para todo $(u, v) \in E$, se cumple que $\hat{w}(u, v) \geq 0$.

Sobre la función de costo del grafo equivalente

Lema 1 (El nuevo costo que no cambia los caminos de costo mínimo)

Dado un digrafo $G(V, E)$, con costos en los lados dados por una función de costo $w : E \to \mathbb{R}$, y sea $h : V \to \mathbb{R}$ una función cualquiera que mapea los vértices a números reales. Para cada $(u, v) \in E$, se define:

$$\hat{w}(u, v) = w(u, v) + h(u) - h(v) \tag{1}$$

Sea $p = \langle v_0, v_1, \ldots, v_k \rangle$ un camino $v_0 \rightsquigarrow v_k$. Entonces p es un camino de costo mínimo $v_0 \rightsquigarrow v_k$ con w, si solo si p es camino de costo mínimo $v_0 \rightsquigarrow v_k$ con $\hat{w}$. Esto es, $w(p) = \delta(v_0, v_k)$ si solo si $\hat{w}(p) = \hat{\delta}(v_0, v_k)$. Además, G tiene un ciclo negativo usando w, si solo si G tiene un ciclo negativo usando $\hat{w}$.

Prueba Lema 1

- **Parte 1**: probamos que $\hat{w}(p) = w(p) + h(v_0) - h(v_k)$.

$$\begin{aligned}
\hat{w}(p) &= \sum_{i=1}^{k} \hat{w}(v_{i-1}, v_i) \\
&= \sum_{i=1}^{k} (w(v_{i-1}, v_i) + h(v_{i-1}) - h(v_i)) \\
&= \sum_{i=1}^{k} w(v_{i-1}, v_i) + h(v_o) - h(v_k) \text{ (suma telescopica)} \\
&= w(p) + h(v_o) - h(v_k)
\end{aligned}$$

- Por lo tanto, para cualquier camino $v_0 \rightsquigarrow v_k$, se tiene que $\hat{w}(p) = w(p) + h(v_0) - h(v_k)$.
- En consecuencia, como $h(v_0)$ y $h(v_k)$ no dependen del camino que se escoja desde v_0 hasta v_k, entonces si un camino $v_0 \rightsquigarrow v_k$ es de menor costo usando w, entonces también lo es usando $\hat{w}$.

Prueba Lema 1, cont.

- **Parte 2**: probamos que existe un ciclo negativo usando w, si solo si existe un ciclo negativo usando $\hat{w}$.
- Sea el ciclo $c = \langle v_0, v_1, \ldots, v_k \rangle$, donde $v_0 = v_k$.
- Se tiene que:

$$\begin{aligned}
\hat{w}(c) &= w(c) + h(v_0) - h(v_k) \\
&= w(c) \text{ (debido a que } v_0 = v_k)
\end{aligned}$$

- Por lo tanto, c tiene un ciclo negativo usando w, si solo si c tiene un ciclo negativo usando $\hat{w}$. $\square$

Reponderación de los costos de los lados

- Queremos $h : V \to \mathbb{R}$, tal que $\hat{w}(u,v) = w(u,v) + h(u) - h(v)$ y $\hat{w}(u,v) \geq 0$
- Se crea un grafo equivalente $G' = (V', E')$.
- $V' = V \cup \{s\}$ donde s es un vértice nuevo.
- $E' = E \cup \{(s,v) : v \in V\}$.
- $w(s,v) = 0$ para todo $v \in V$.
- Como no hay ciclo entrando en s, entonces G' tiene los mismos ciclos que G.
- G' tiene ciclos negativos, si solo si G tiene ciclos negativos.
- Se define $h(v) = \delta(s,v)$ para todo $v \in V$.

Reponderación de los costos de los lados, cont.

Afirmación

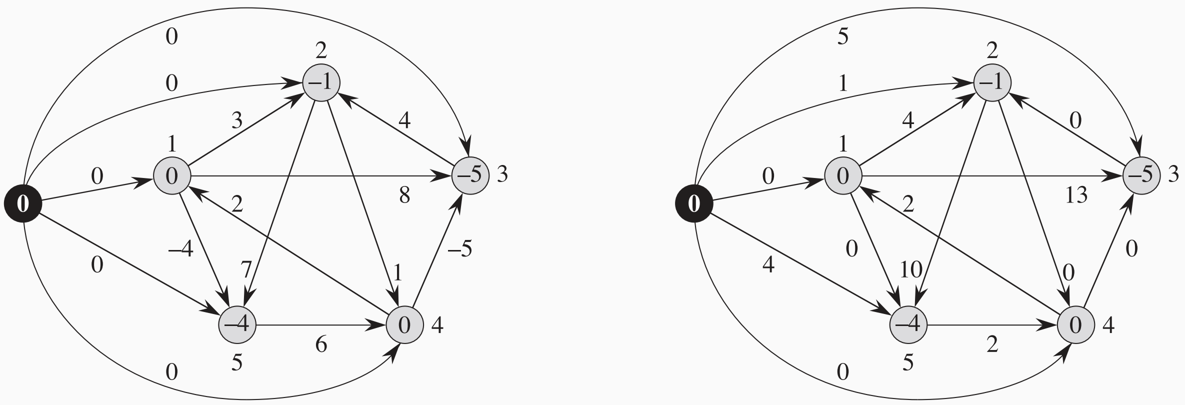
$\hat{w}(u,v) = w(u,v) + h(u) - h(v) \geq 0$

Prueba

$$\delta(s,v) \leq \delta(s,u) + w(u,v)$$
$$h(v) \leq h(u) + w(u,v).$$

Por lo tanto, $\hat{w}(u,v) = w(u,v) + h(u) - h(v) \geq 0.$ $\square$

Ejemplo de la reponderación de los costos de los lados



(a) (b)

Figura 1: En (a) se muestra el grafo equivalente con los valores de h calculados para cada vértice y con los costos en los lados dados por w. En (b) se muestra los costos de los lados computados con $\hat{w}$. Fuente [1]

Algoritmo de Johnson

Función Johnson($G = (V,\ E),\ w$)

inicio
- Crear grafo $G' = (V', E')$ a partir de G ;
- **para cada** $v \in V$ **hacer** $w(s, v) \leftarrow 0$ en G' ;
- **si** `Bellman-Ford(`G', s, w`)` $=$ *False* **entonces**
  - `Imprimir(`*Ciclo negativo*`)` ;
  - **devolver** NIL ;
- **para cada** $v \in V'$ **hacer**
  - $h(v) \leftarrow \delta(s, v)$ usando el resultado de `Bellman-Ford(`G', s, w`)` anterior ;
- **para cada** $(u, v) \in E'$ **hacer**
  - $\hat{w}(u, v) \leftarrow w(u, v) + h(u) - h(v)$
- Sea $D = (d_{u,v})$ una nueva matriz $|V| \times |V|$;
- **para cada** $u \in V$ **hacer**
  - Computar $\hat{\delta}(u, v) \quad \forall v \in V$ usando `Dijkstra(`G, u, $\hat{w}$`)` ;
  - **para cada** $v \in V$ **hacer**
    - $d_{u,v} \leftarrow \hat{\delta}(u, v) + h(v) - h(u)$;
- **devolver** D ;

Análisis del algoritmo de Johnson

- Computar G' es tiempo $\Theta(|V| + |E|)$.
- Computar Bellman-Ford es tiempo $O(|V||E|)$.
- Computar costos $\hat{w}$ es tiempo $\Theta(|E|)$.
- Ejecutar Dijkstra $|V|$ veces es tiempo $O(|V|^2 \log |V| + |V||E|)$ usando Fibonacci heaps como cola de prioridad.
- Computar la matriz D es tiempo $\Theta(|V^2|)$.
- Por lo tanto, el algoritmo de Johnson es $O(|V|^2 \log |V| + |V||E|)$.

Referencias

[1] T. Cormen, C. Leirserson, R. Rivest, and C. Stein.
Introduction to Algorithms.
McGraw Hill, 3ra edition, 2009.