Problema de los caminos de costo mínimo partiendo desde un solo origen: Algortimo de Bellman - Ford

Bibliografía: “Introduction to Algorithms”. Second Edition.

The MIT Press. Massachusetts Institute of Technology. Cambridge, Massachusetts 02142.

http://mitpress.mit.edu

Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein

Facultad de Matemática y Computación
Universidad de La Habana

generalidades

PROBLEMA que resuelve:

> Determina **caminos de costo mínimo desde un vértice origen hacia los restantes vértices de un grafo** G=(V, E) **dirigido y ponderado**

- A partir de la funcionalidad del Algoritmo **se permite que en el Grafo hayan arcos de costo negativo**

- Inicializaciones hechas a partir de **INITIALIZE SINGLE SOURCE**

- Se aplica la técnica de **RELAX**, **varias veces**, **sobre cada arco del grafo**

Principio de funcionamiento

Sea *G = (V,E),* **dirigido**, **ponderado**, con función de costo *w: E* → $\Re$ definida sobre *G*, *s* origen

ALGORITMO DE *BELLMAN-FORD*:

- Determina si desde el origen se alcanza algún ciclo de costo negativo: Si esto sucede, retorna ***FALSE***
- En otro caso, ***TRUE***. Además, halla el costo y determina, el camino de costo mínimo entre ***s*** y los restantes vértices de Grafo

FUNCIONAMIENTO:

A partir de la inicialización hecha, según **INITIALIZE SINGLE SOURCE**, el algoritmo va reduciendo, progresivamente y a través de sucesivas aplicaciones de **RELAX** sobre los arcos de ***G***, el costo estimado del camino de costo mínimo de ***s*** a ***v***, hasta que éste alcanza su valor real

Recordando las Propiedades

G=(V, E) dirigido, ***s*** origen. **INITIALIZE-SINGLE-SOURCE(*G*, *s*).** La única forma en que varían los **d[v]** y los **π[v]** es a través del **RELAX.** $\forall$**v**$\in$**V**

- **Lema 24.12**

Inmediatamente después de hacer **RELAX(u, v, w),** se cumple, $d[v] \leq d[u] + w(u, v)$

- **Desigualdad Triangular (Lema 24.10)**

Para todo arco (u, v) $\in$ E se cumple $\delta(s, v) \leq \delta(s, u) + w(u, v)$

- **Propiedad de la Cota Superior (Lema 24.11)**

Para todo $v \in V$ se cumple $\boldsymbol{d[v] \geq \delta(s, v)}$ y una vez que d[v] alcanza el valor $\delta(s, v)$, este no varía nunca más.

- **Propiedad de la no existencia de camino (Corolario 24.12)**

Si no existe camino de s a v, entonces el valor de d[v] se mantendrá invariante y se cumplirá **d[v] = δ(s, v) = ∞**

Recordando las Propiedades

- **Propiedad de la convergencia (Lema 24.14)**

Si $s \leadsto u \rightarrow v$ es un camino de costo mínimo en G, $u,v \in V$. Si se alcanza la igualdad d[u] = $\delta(s, u)$ en cualquier momento antes de hacer *RELAX* sobre el arco (u, v), entonces, después de haberlo hecho, d[v] = $\delta(s, v)$ y dicha igualdad se mantiene en lo sucesivo

- **Propiedad del RELAX (Lema 24.15)**

Si $p = <v_0, v_1, \ldots, v_k>$ es un **camino de costo mínimo** de $s=v_0$ a v_k y supongamos que a los arcos de *p* se les aplica *RELAX* en el siguiente orden: $(v_0, v_1), (v_1, v_2), \ldots, (v_{k-1}, v_k)$, entonces $d[v_k] = \delta(s, v_k)$ después de estas "relajaciones". Esta propiedad se cumple incluso cuando se mezcle el RELAX sobre otros arcos, con los arcos de *p*. (Note que k puede ser = 0, 1, …, |V|-1)

BELLMAN-FORD(G, w, s)

```
INITIALIZE-SINGLE-SOURCE(G, s)
for i ← 1 to |V[G]| - 1
    do for cada arco (u, v) ∈ E[G]
        do RELAX(u, v, w)
for cada arco (u, v) ∈ E[G]
    do if d[v] > d[u] + w(u, v)          **
        then return FALSE
return TRUE
```

O(|E||V|)

En el caso peor, cuando el grafo alcanza el máximo de arcos posibles, o sea cuando $|E|$ es $O(|V|^2)$, entonces el algoritmo de BF puede ser $O(|V|^3)$

Detección de ciclos de costo negativo

Si en el ciclo ** se detectan vértices *v* de *V* que cumplen la desigualdad expresada en el cuerpo del mismo esto $\Rightarrow$ en el grafo existen ciclos de costo negativo alcanzables desde *s*, y por tanto, después de acabar el proceso de ***RELAX****s* quedarán vértices suceptibles a que el costo del origen a ellos pueda seguir “bajando”

OBSERVACIONES:

- Los vértices suceptibles a este problema NO SON SOLO los comprendidos en el ciclo de costo negativo. Pueden ser también vértices que no estén propiamente dentro del ciclo pero que sean alcanzbles desde dicho ciclo

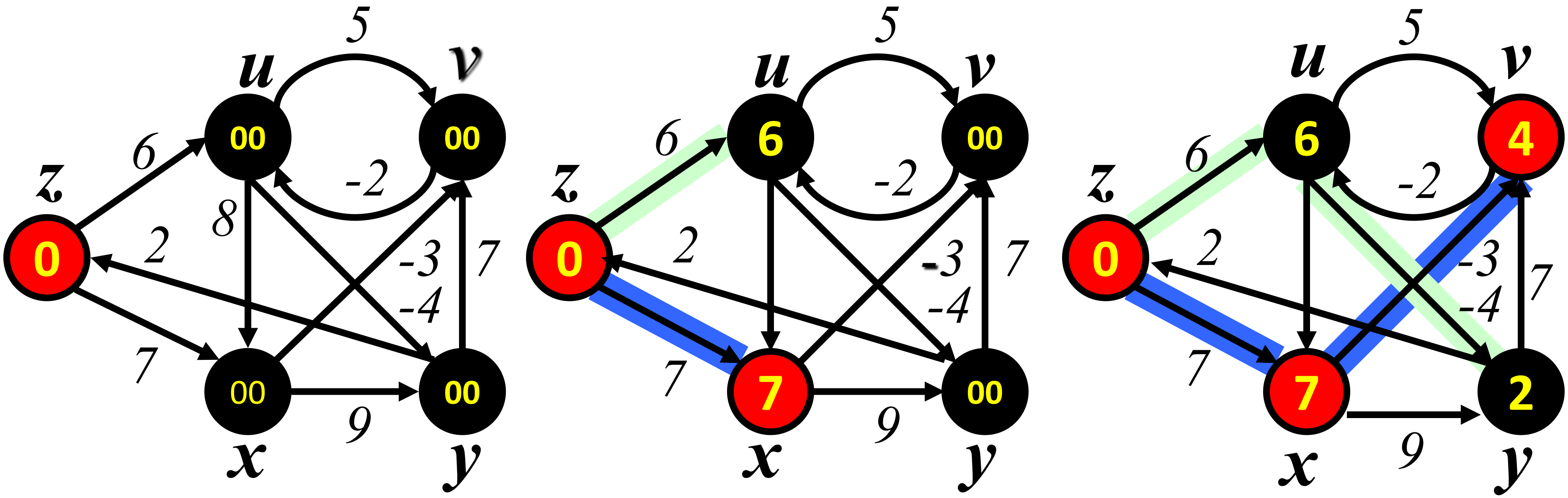
Puede parecer que **los vértices que NO son alcanzables desde el origen** también podrían cumplir la desigualdad expresada en el cuerpo de dicho ciclo, pero veamos que no es así:

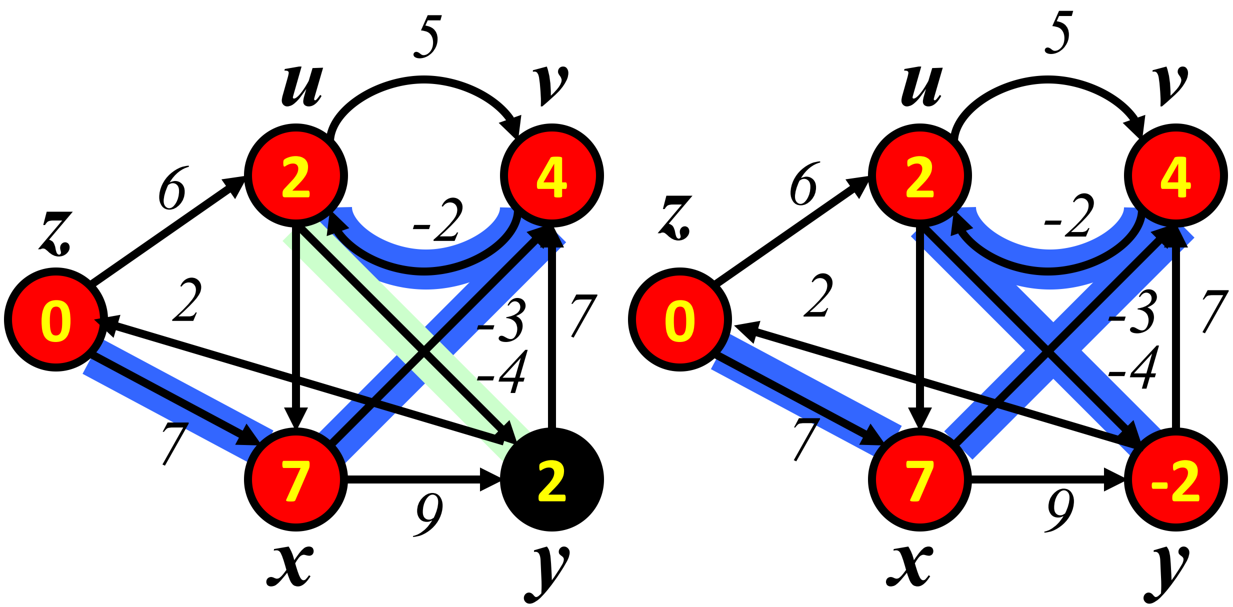
1) Si ***d[v] = ∞***, y ***v*** está en otra componente diferente que ***u***, es trivial que no se cumple, pues **no existe arco** de ***u*** a ***v*** y por tanto, en el ciclo, dicho arco no será analizado, pues no existe

2) Si existe el arco ***(u, v)*** dentro de una componente dada y se cumple:

- ***d[v] = ∞*** y ***d[u] = ∞***, tampoco se cumple la desigualdad pues quedará, en este caso, una igualdad al testar la condición dentro del ciclo. (**∞ = ∞**)
- Si $d[v] \neq \infty$ y $d[u] = \infty$, entonces $d[v] < d[u] + w(u, v)$, o sea, $d[v] < \infty$ y tampoco se cumplirá la desigualdad

Otra posibilidad: no puede existir (pues si ***d[u] ≠ ∞***, o sea, se llega a ***u*** desde ***s*** y existe el arco ***(u, v),*** entonces también se alcanzará ***v*** desde ***s***, por tanto ***d[v] ≠ ∞*** también)

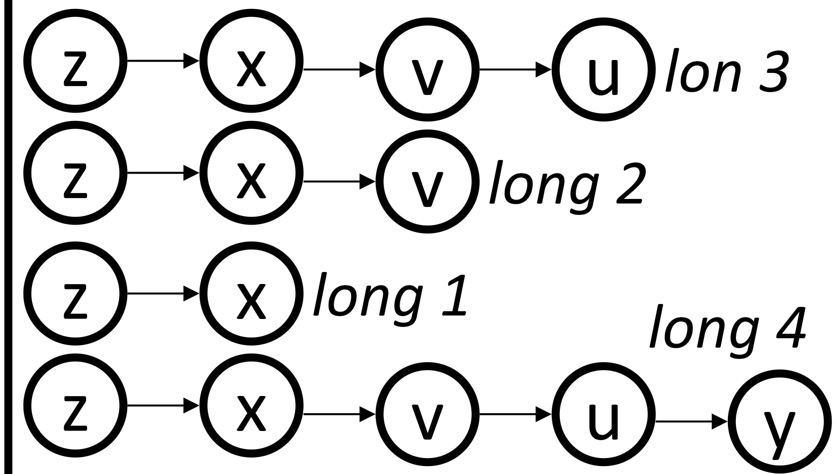




(u, v)
(u, x)
(u, y)
(v, u)
(x, v)
(x, y)
(y, v)
(y, z)
(z, u)
(z, x)

Orden en que se aplica **RELAX** a los arcos de G, en cada pasada

	Ini	1.	2.	3.	4.
d[z]	0	0	0	0	0
d[u]	∞	6	6	2	2
d[v]	∞	∞	4	4	4
d[x]	∞	7	7	7	7
d[y]	∞	∞	2	2	-2
π[z]	-	-	-	-	-
π[u]	-	z	z	v	v
π[v]	-	-	x	x	x
π[x]	-	z	z	z	z
π[y]	-	-	u	u	u



En la 1a. se halla el costo min para todos los caminos de costo min con long 1

En la 2a. se halla el costo min para todos los caminos de costo min con long 2

.... Y así sucesivamente

El algoritmo se basa, fundamentalmente, en la aplicación consecuente y progresiva de la **Propiedad de la Convergencia**

Tras las inicializaciones, ***d[s]* = *0*** = $\delta(s, s)$, por tanto, según la propiedad, en la primera pasada, todos los vértices ***v*** $\in$ ***V*** cuyo camino de costo mínimo entre ***s*** y ellos, tiene ***longitud 1*,** alcanzarán en ***d[v]*** el valor de $\delta(s, v)$, por tanto, tras las segunda pasada, lo alcanzarán los de ***longitud 2***, y así, sucesivamente. Como la mayor longitud que puede tener un camino de ***s*** a ***v*** es ***|V| - 1***, es por eso que se hacen ***|V|-1*** aplicaciones de **RELAX** sobre los arcos de ***G***

caso base: garantiza que en la primera pasada ya haya un vértice que en el paso anterior (el paso de las **inicializaciones**) alcanzó su costo minimo. Es el propio origen ***d[origen]=0***. Con esto, se garantiza que en la primera pasada, a partir de este antecedente, al menos, para un vértice con longitud de camino del origen a él = 1, se determina su camino de costo mínimo

Prueba de correctitud – Algoritmo BF

LEMA 1:

Sea ***G=(V,E),*** dirigido, ponderado, con función de costo $w: E \rightarrow R$, definida sobre *G*, sea ***s*** el vértice origen. Si *G* **no tiene ciclos de costo negativo alcanzables desde s**, entonces, al concluir la ejecución del algoritmo del algoritmo de **Bellman-Ford**, se cumplirá

d[v] = $\delta(s,v), \forall v \in V$ alcanzable desde ***s***

Prueba de correctitud – Algoritmo BF

Demostración LEMA 1:

Sea ***p***={$v_0 = s$, $v_1, v_2, \ldots, v_k = v$} un **camino de costo mínimo** de ***s*** a ***v***. ***p*** ***es simple***

los ciclos (que en este caso solo podrían haber sido de costo positivo o de costo cero) que hubieran habido en *p*, por ser este un camino de costo mínimo, fueron eliminados por el propio algoritmo. Estudio individual: comprobar con ejemplos concretos esta afirmación

de arcos implicados en ***p*** **≤ |V|-1**

Probemos por inducción que para ***i=1,2, …,k***
d(v_i)=δ(s, v_i) después de la *i-ésima* pasada del algoritmo sobre los arcos de ***G*** y que está igualdad se mantendrá posteriormente

Prueba de correctitud – Algoritmo BF

Demostración LEMA 1 *(continuación)* :

Caso Base:

Tras las inicializaciones, $d[\ s=v_0\ ]=\delta(s, s)=0$,
por la ***propiedad de la cota superior*** y esto se mantendrá así hasta el final

Paso de inducción:

Supongamos $d[\ v_{i-1}] = \delta[s, v_{i-1}]$ después de la *i-1 ésima* pasada

Como a todos los arcos del grafo, al arco (v_{i-1}, v_i), en particular, se le hará RELAX en la **i-ésima** pasada, pero como supusimos que en ese momento $d[\ v_{i-1}] = \delta[s, v_{i-1}]$, entonces, por la **propiedad de la convergencia**, tras el ***RELAX*** hecho sobre (v_{i-1}, v_i) en dicha pasada, se alcanzará $d[\ v_i] = \delta[s, v_i]$ y esta igualdad se mantiene hasta que concluya la ejecución del algoritmo

Prueba de correctitud – Algoritmo BF

Corolario 1:

Sea $G = (V,E)$, dirigido, ponderado, con función de costo $w: E \rightarrow R$, definida sobre G, sea $\boldsymbol{s}$ el vértice origen, entonces, $\forall v \in V$, existe un camino de s a v, si y solo si, $d[v] < \infty$ después de concluir la ejecución del algoritmo de BELLMAN-FORD sobre G

Demostración

Similar a la del Lema 1 **(Ejercicio propuesto)**

Prueba de correctitud – Algoritmo BF

TEOREMA: Prueba de correctitud de BF

Sea *G=(V,E),* dirigido, ponderado, con función de costo *w: E* $\rightarrow$ *R*, definida sobre *G*, sea *s* el vértice origen. Desde *s* no se alcanzan ciclos de costo negativos en *G*, entonces,

(I) Si el algoritmo retorna **TRUE**:

a) $d[v] = \delta(s, v)$, $\forall v \in V$

b) La secuencia de arcos, en orden inverso, establecida por $\pi[v]$, expresa el camino de costo mínimo de *s* a *v*

(II) Si el algoritmo retorna **FALSE**:

G tiene un ciclo de costo negativo que se alcanza desde *s*

Prueba de correctitud – Algoritmo BF

Demostración (I)

(a)

Probemos que al terminar algoritmo $d[v] = \delta(s, v)$, $\forall v \in V$

- Si *v* se alcanza desde *s*, entonces por el **Lema 1** se prueba esta igualdad
- Si *v* no se alcanza desde *s*, entonces por la **propiedad de la no existencia de camino** se prueba la igualdad

(b)

Es evidente que la secuencia de arcos en orden inverso establecida por $\pi[v]$, $\forall v \in V$, establece un camino de costo mínimo de *s* a *v*. (Ver **Lema 25.9 del "Introduction to Algorithms")**

Finalmente, con la certeza de que se cumplen **a)** y **b)** demostremos que **Bellman-Ford** retorna **TRUE**

Prueba de correctitud – Algoritmo BF

Al concluir el algoritmo se tiene:

$d[v] = \delta(s, v) \leq \delta(s, u) + w(u, v)$ por **propiedad desigualdad triangular**

$= d(u) + w(u, v)$

Por tanto, ningún arco cumplirá la condición

$$\boxed{6 \quad \textbf{do if } d[v] > d[u] + w(u, v)}$$

de la línea 6 del algoritmo que es la que provoca que el algoritmo retorne **FALSE**

Prueba de correctitud – Algoritmo BF

Demostración (II)

Supongamos que *G* contiene un ciclo de costo negativo que se alcanza desde *s*, sea *c* dicho ciclo

$c=<v_0, v_1, v_2, \ldots, v_k>$ $\quad v_0 = v_k$

entonces se cumplirá

$\sum_{i=1}^{k} w(v_{i-1}, v_i) < 0$

Por reducción al absurdo, supongamos que bajo estas condiciones ***Bellaman-Ford*** retornara TRUE, en tal caso se cumplirá

$d[v_i] \leq d[v_{i-1}] + w(v_{i-1}, v_i)$ para todo $i=1, 2, \ldots, k$

Sumemos las desigualdades alrededor del ciclo *c*

Prueba de correctitud – Algoritmo BF

Demostración (II) (continuación)

$$\boxed{\sum_{i=1}^{k} d[v_i]} \leq \boxed{\sum_{i=1}^{k} d[v_{i-1}]} + \sum_{i=1}^{k} w(v_{i-1}, v_i)$$

$$\boxed{v_1 + v_2 + v_3 + \ldots + v_k} \qquad \boxed{v_0 + v_1 + v_2 + \ldots + v_{k-1}}$$

como $v_0 = v_k$,

$$\cancel{\sum_{i=1}^{k} d[v_i]} \leq \cancel{\sum_{i=1}^{k} d[v_{i-1}]} + \sum_{i=1}^{k} w(v_{i-1}, v_i)$$

$$0 \leq \sum_{i=1}^{k} w(v_{i-1}, v_i)$$

Y esto contradice lo supuesto, o sea,

$$\sum_{i=1}^{k} w(v_{i-1},\ v_i) < 0$$