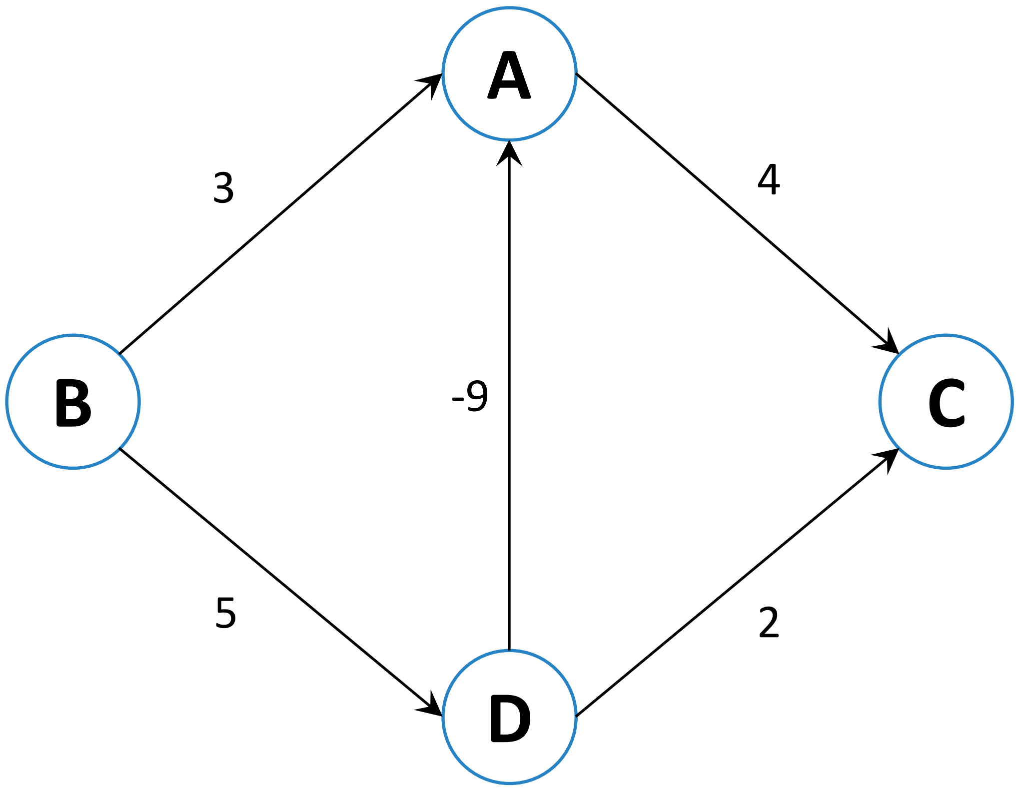
Ruta de menor costo de *B* a *C*



¿Qué pasa si usamos el algoritmo de Dijkstra en este caso?

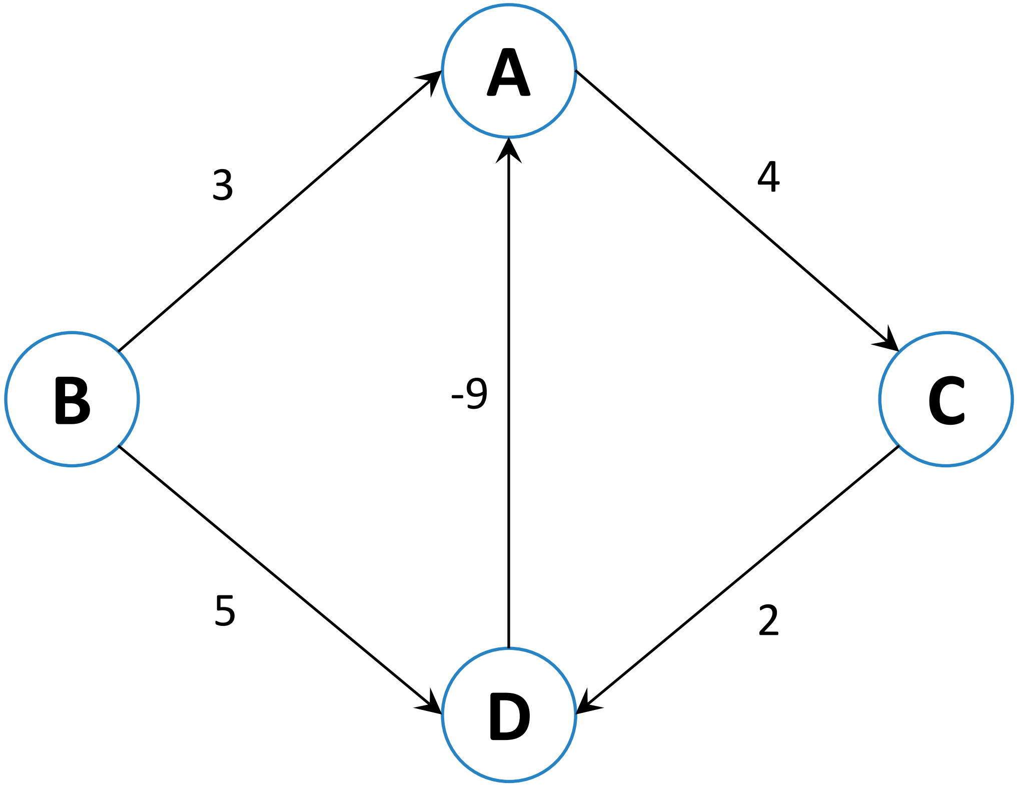
Aristas con costos negativos

Surgen cuando puede haber **ganancias** y **consumos**

Por ejemplo: dinero, energía, materiales, etc.

¿Qué otros casos se les ocurren?

Ruta de menor costo de *B* a *C*



¿Cuál es la ruta de menor costo de *B* a *C* en este caso?

Ciclos (con costos) negativos

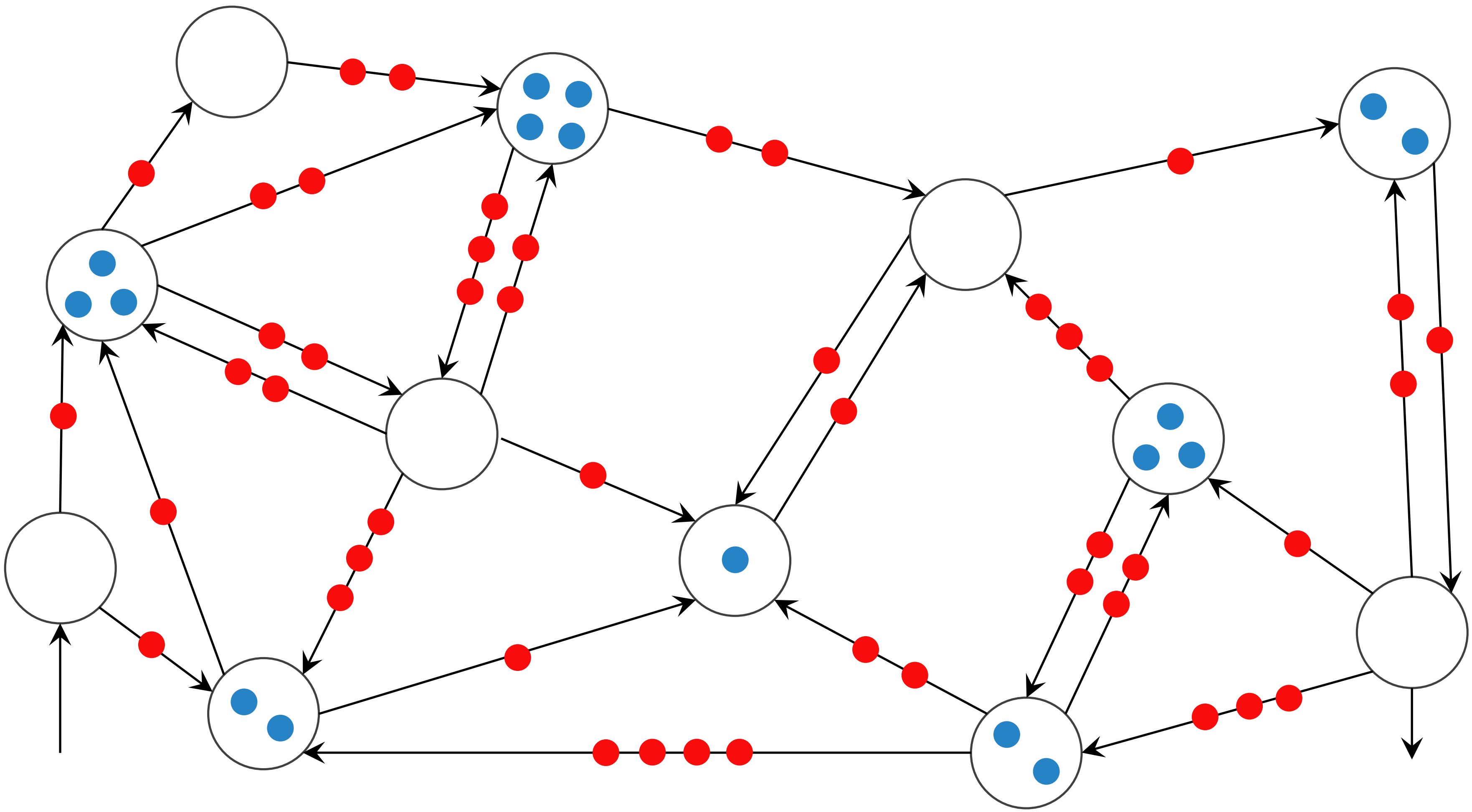
Los ciclos negativos suelen indicar que hay un error:

- ya sea en la **formulación** del problema
- o en la **matrix…**

Los prisioneros de guerra

- El bando **azul** y el bando **rojo** están en guerra
- La base del bando rojo tiene N prisioneros azules
- Los prisioneros están repartidos a lo largo de la base
- La celda i dentro de la instalación tiene p_i prisioneros
- El camino entre las celdas i y j tiene g_{ij} guardias rojos
- Los prisioneros serán ejecutados **mañana**

Mapa de la base



El plan de rescate

- Queremos rescatar a los prisioneros de la base
- El escuadrón azul α se infiltrará esta noche en la base
- Al llegar a una celda, sus prisioneros se unirán al escuadrón
- Cruzarse con m guardias implica que α pierde m miembros
- Los puntos de entrada y de salida han sido designados, S y F

¿Qué ruta debe seguir α para maximizar la supervivencia azul?

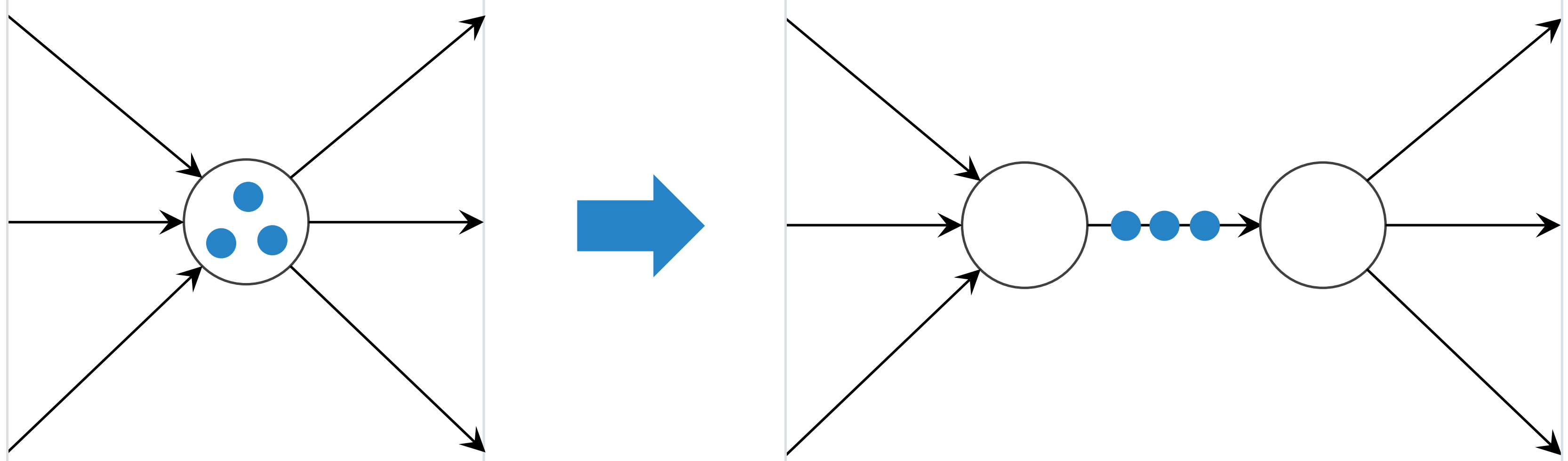
Costos en los nodos

Queremos encontrar la ruta **de menor costo** entre S y F

Pero tenemos algunos costos asociados a los **nodos**

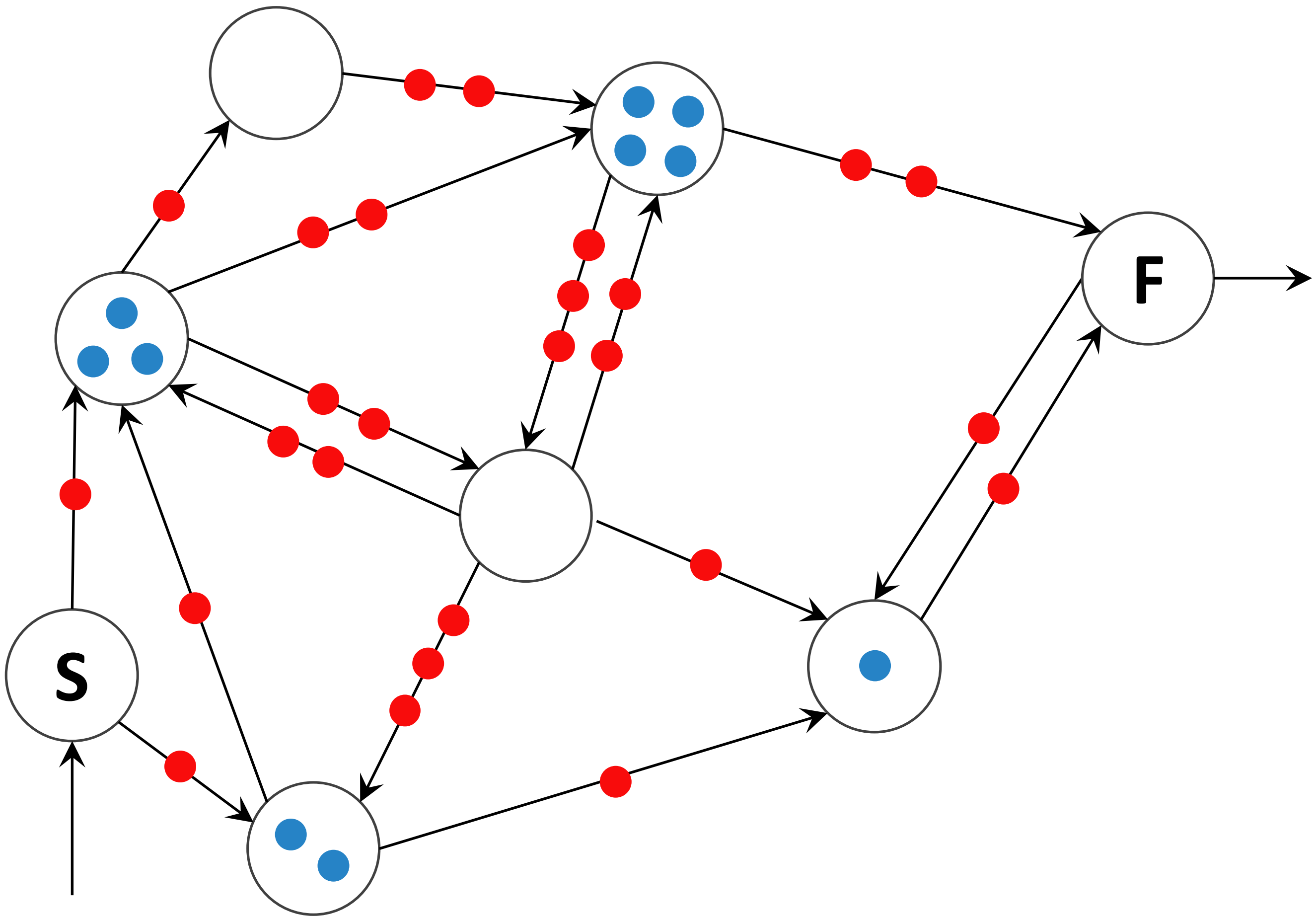
¿Cómo podemos manejar eso?

Costos en los nodos … transformados a costos en aristas adicionales

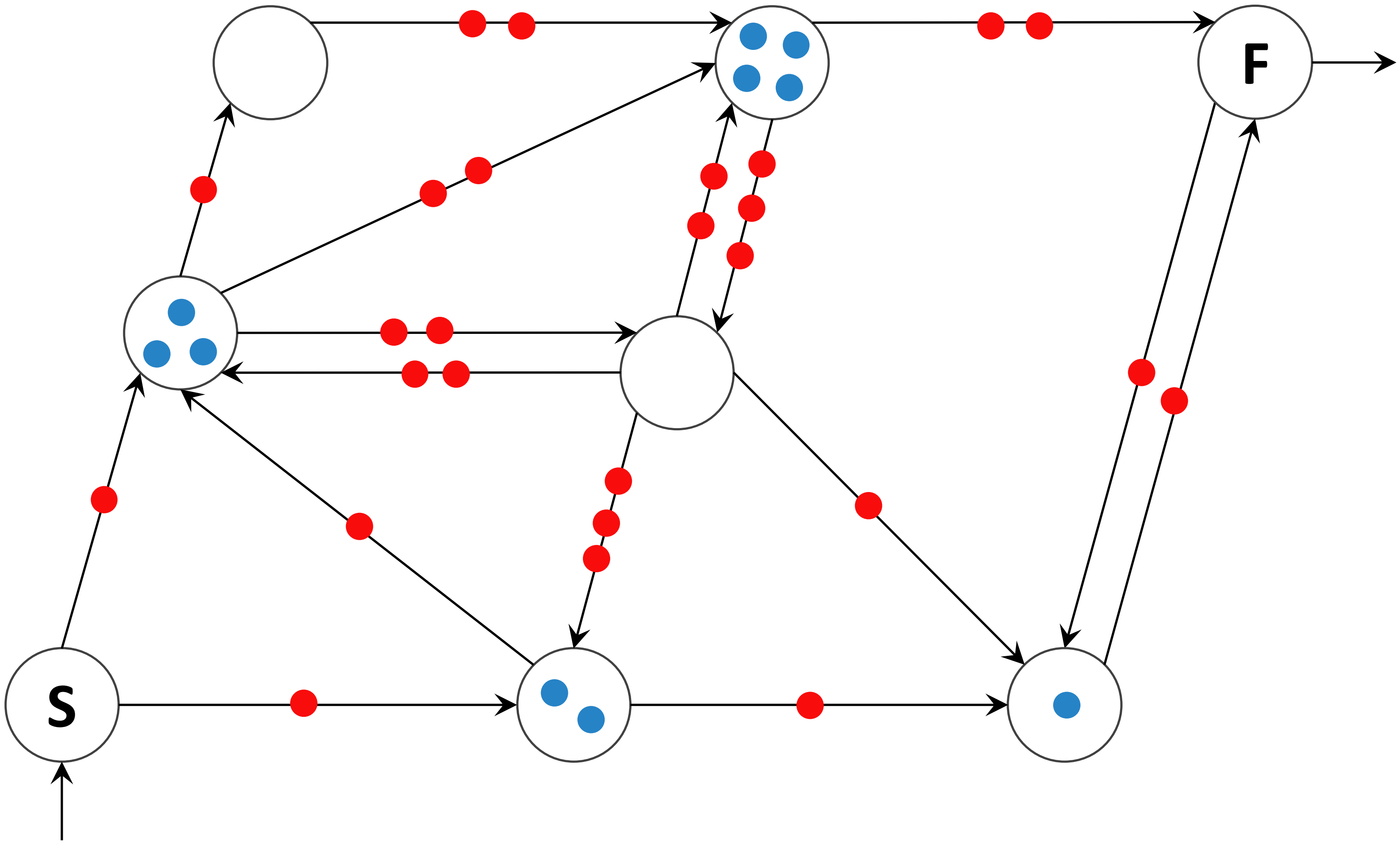


Transformación del mapa de la base

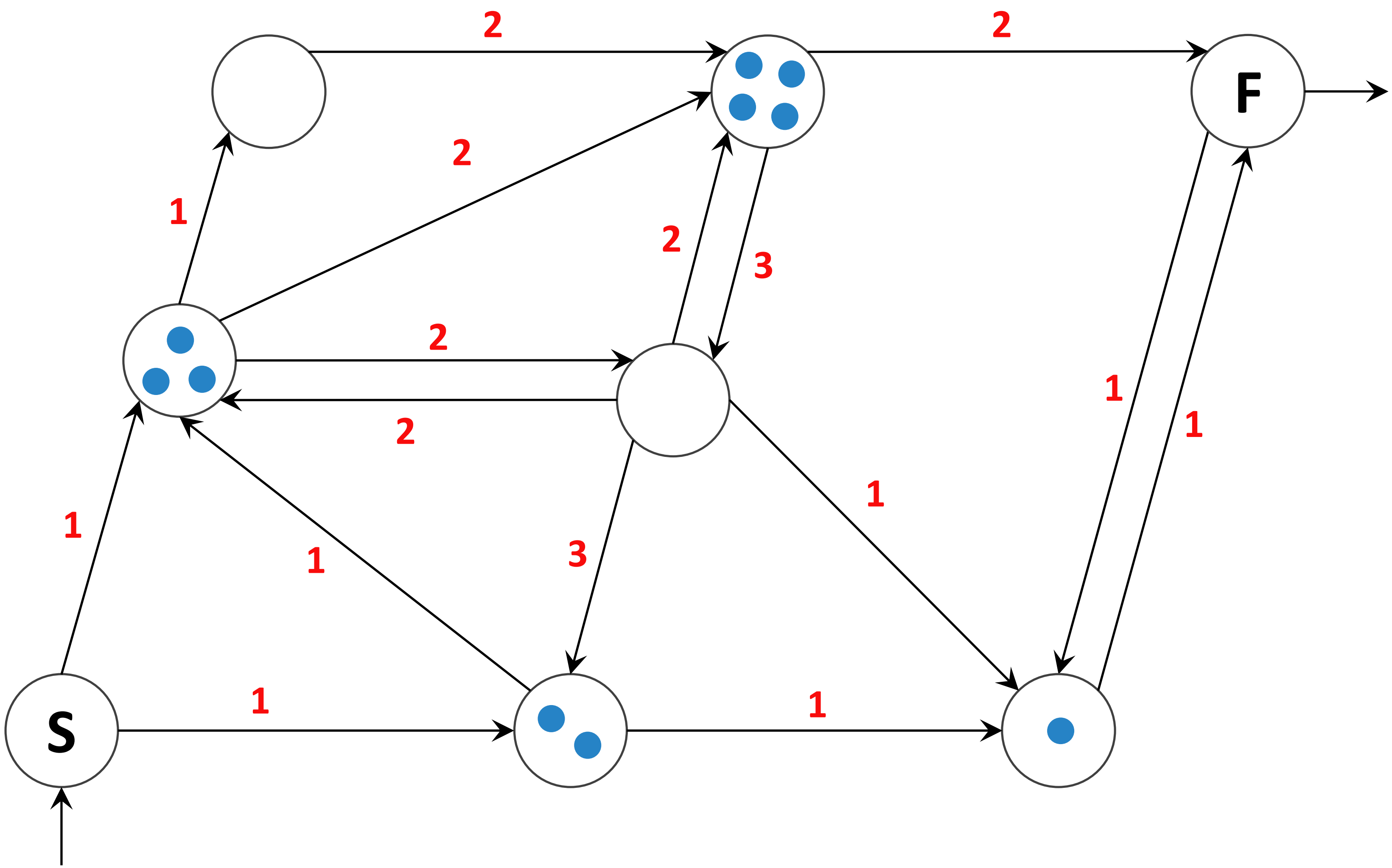
(uno más simple que el original)



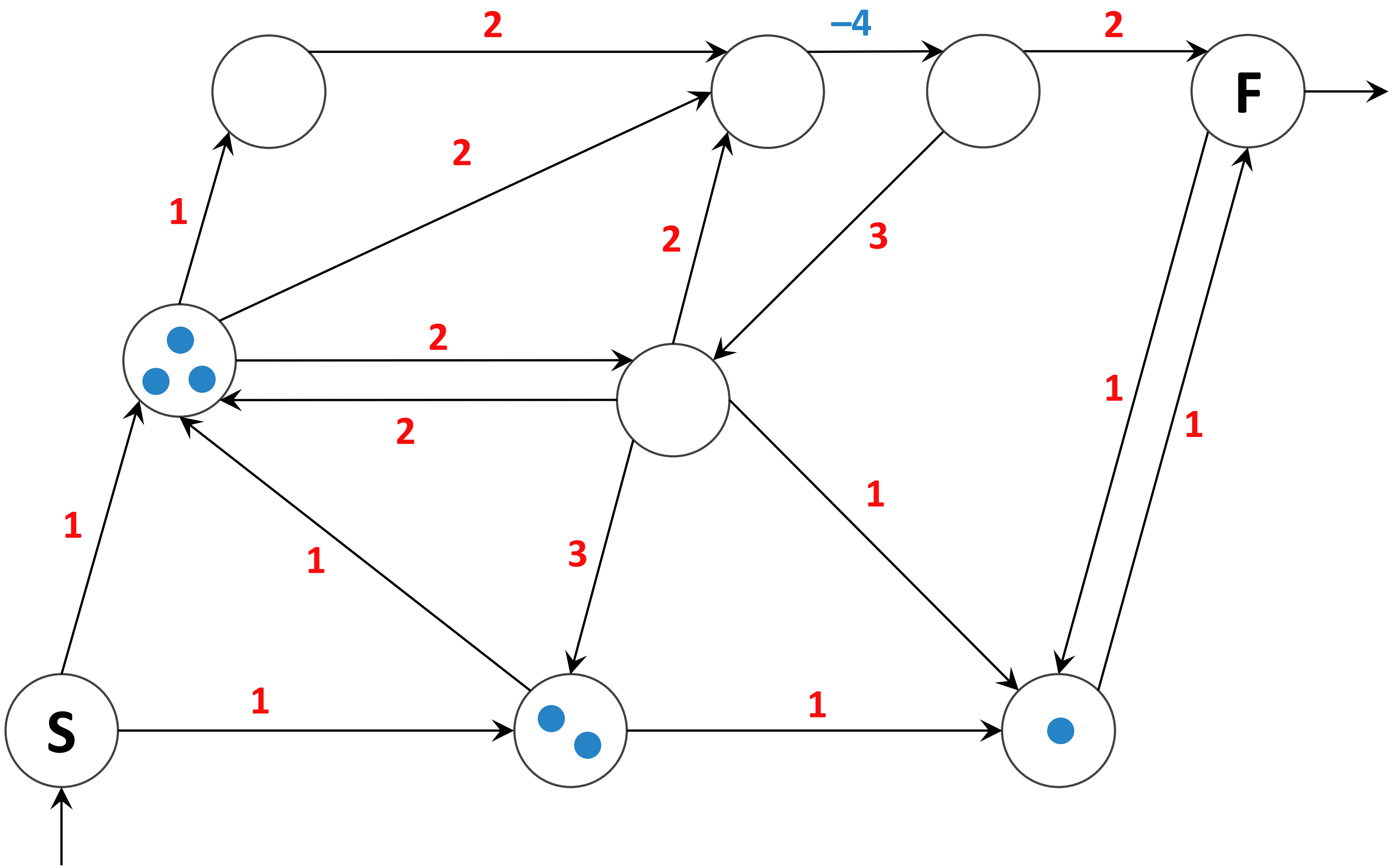
Transformación del mapa de la base



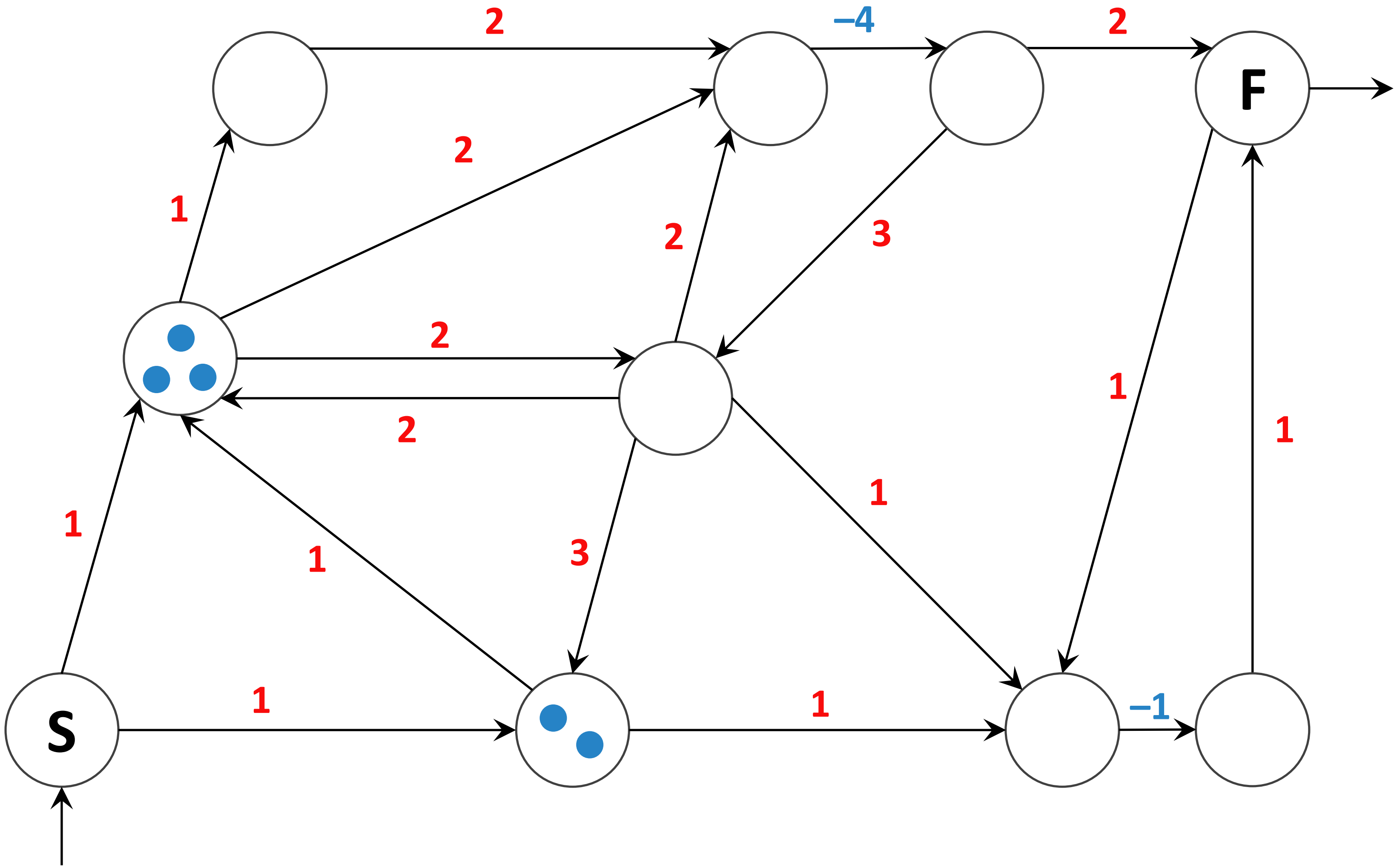
Transformación del mapa de la base



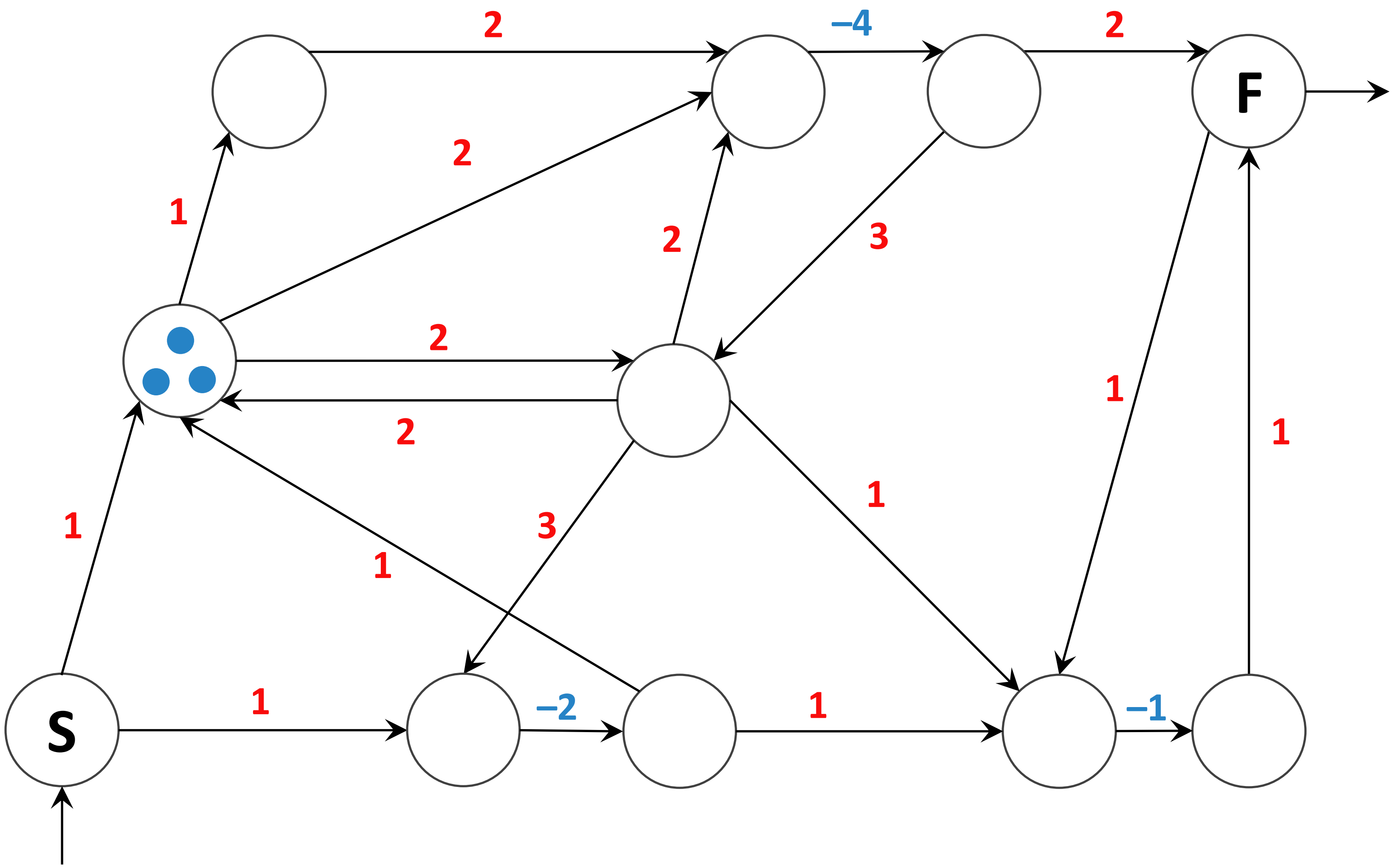
Transformación del mapa de la base



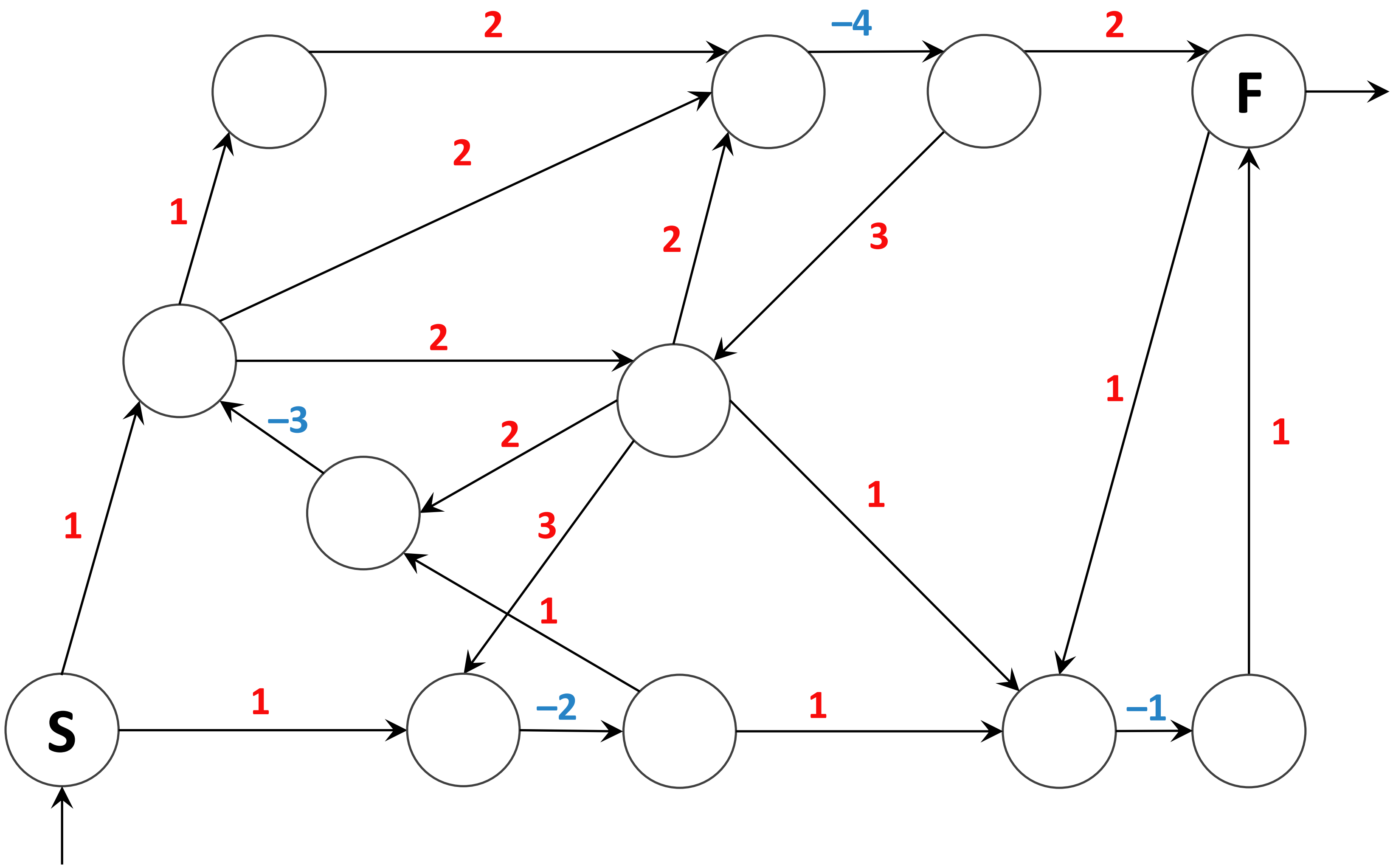
Transformación del mapa de la base



Transformación del mapa de la base



Mapa de la base transformado … finalmente



Ruta de menor costo

El grafo resultante tiene unas aristas con costos **positivos** y otras con costos **negativos**

¿Cómo podemos encontrar la ruta de menor costo de S a F?

Revisemos las propiedades de la ruta más corta

Actualización de aristas

En el algoritmo de Dijkstra, cambiamos el *padre de* y la *distancia a* un nodo cuando encontramos una ruta de menor costo (que la que conocíamos) hacia el nodo:

$$\boldsymbol{if}\ \boldsymbol{u.dist} + \boldsymbol{w}(\boldsymbol{u}, \boldsymbol{v}) < \boldsymbol{v.dist}:$$

$$\boldsymbol{v.parent} \leftarrow \boldsymbol{u}, \qquad \boldsymbol{v.dist} \leftarrow \boldsymbol{u.dist} + \boldsymbol{w}(\boldsymbol{u}, \boldsymbol{v})$$

Esto se conoce como **actualizar** la arista $(\boldsymbol{u}, \boldsymbol{v})$

$$update(u, v, w):$$

$$\quad \textbf{if } u.dist + w(u, v) < v.dist:$$

$$\quad\quad v.parent \leftarrow u$$

$$\quad\quad v.dist \leftarrow u.dist + w(u, v)$$

Subestructura óptima

Las rutas más cortas cumplen la propiedad de **subestructura óptima**:

Si $u \underset{p}{\rightsquigarrow} x \to v$ es una ruta más corta de u a v

… entonces p es una ruta más corta de u a x

… lo que se puede demostrar por contradicción

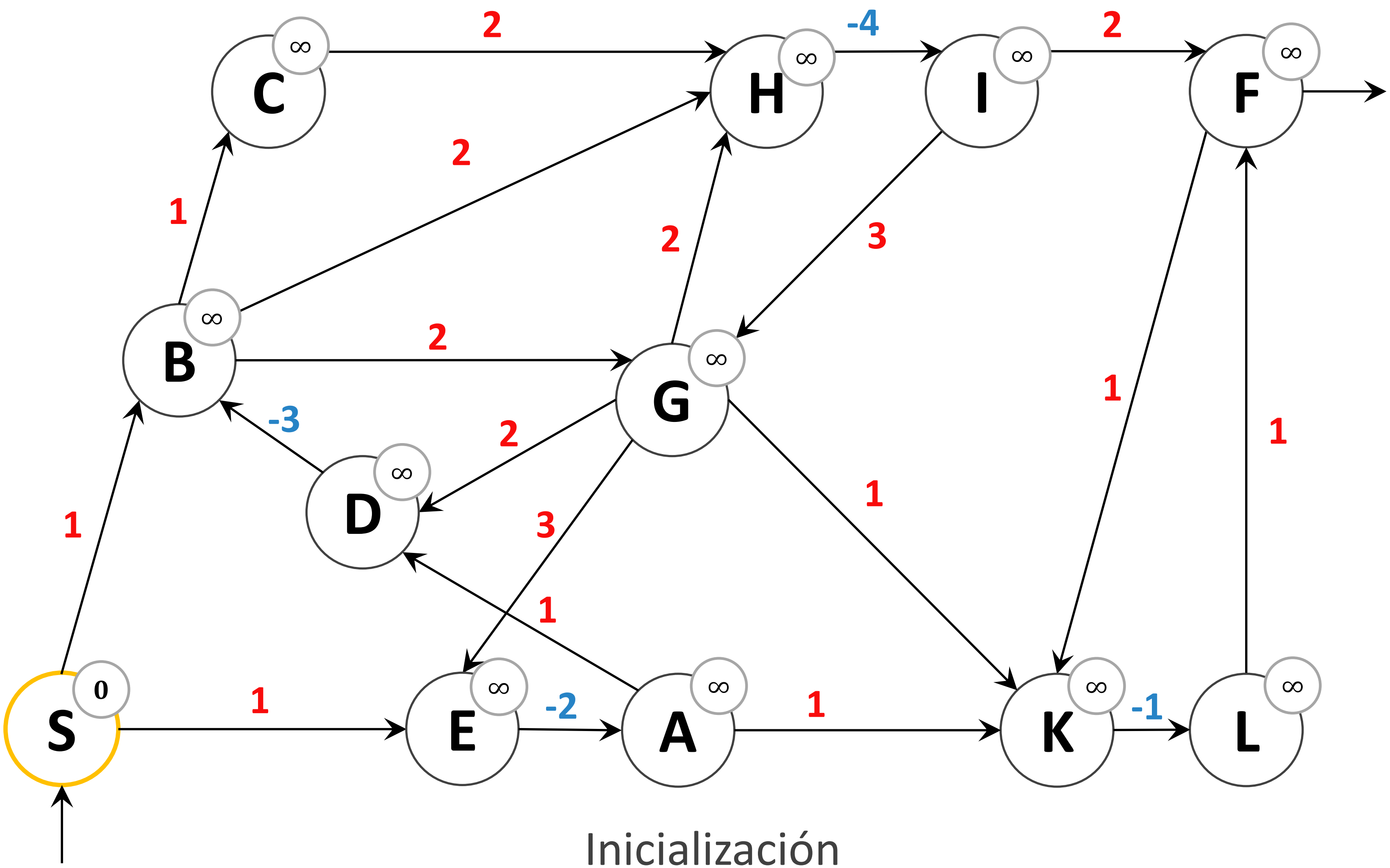
Rutas más cortas desde s con i aristas

Partiendo de s, queremos las rutas más cortas que sólo tengan una arista

¿Cómo las generamos? ¿Y las que tengan dos aristas?

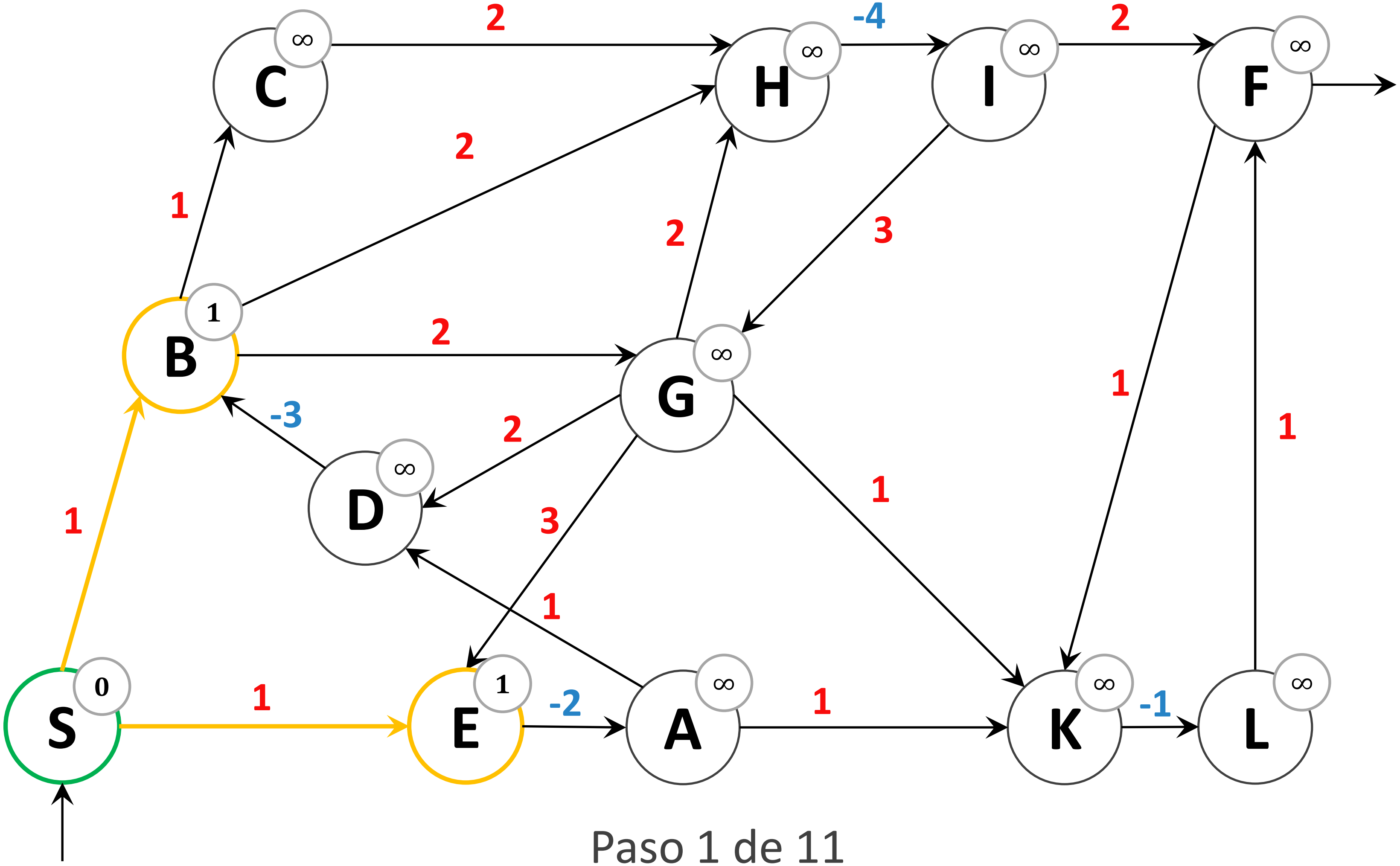
¿Cómo generalizamos esto para las rutas con i aristas?

Bellman Ford en acción



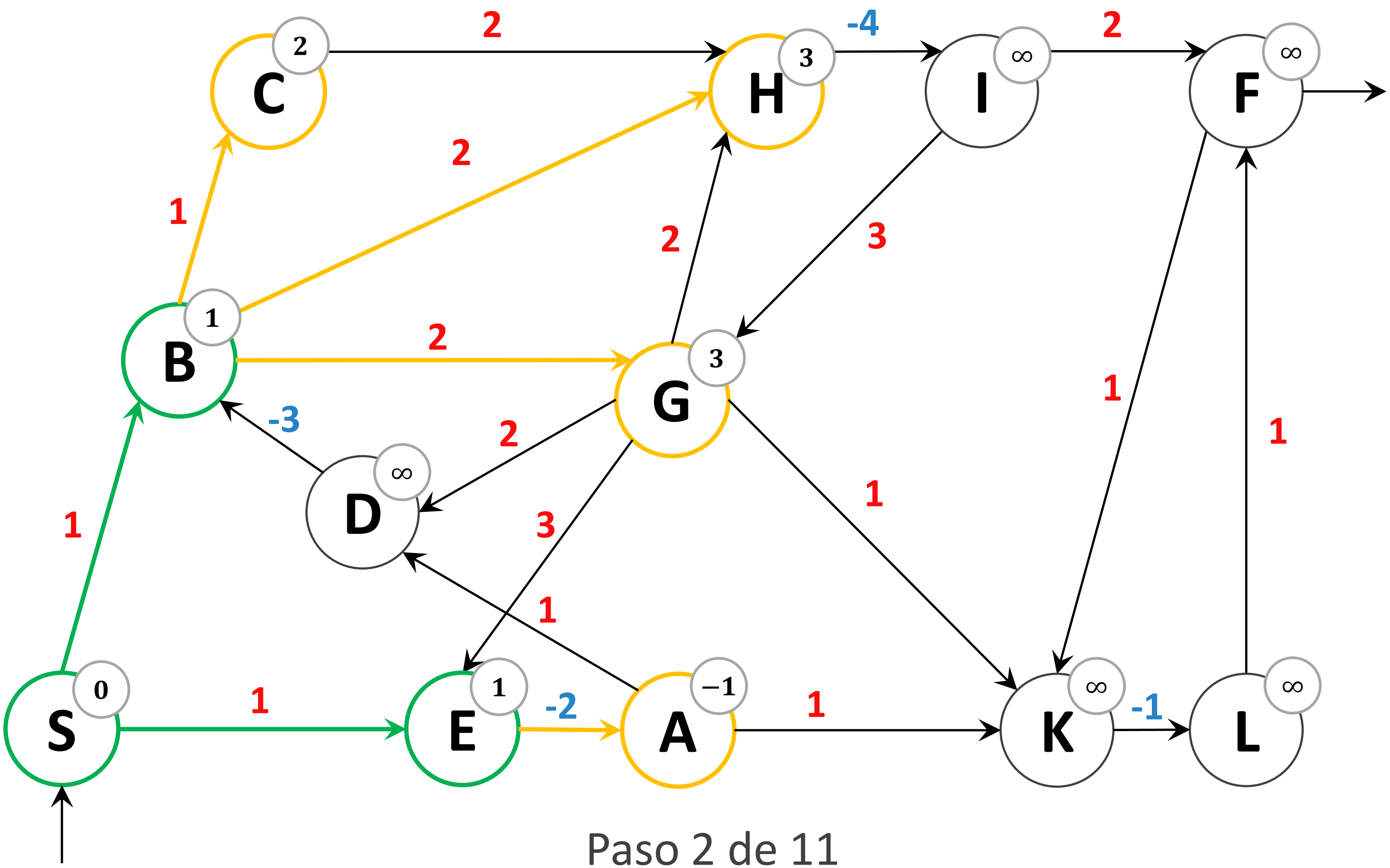
Inicialización

Bellman Ford en acción



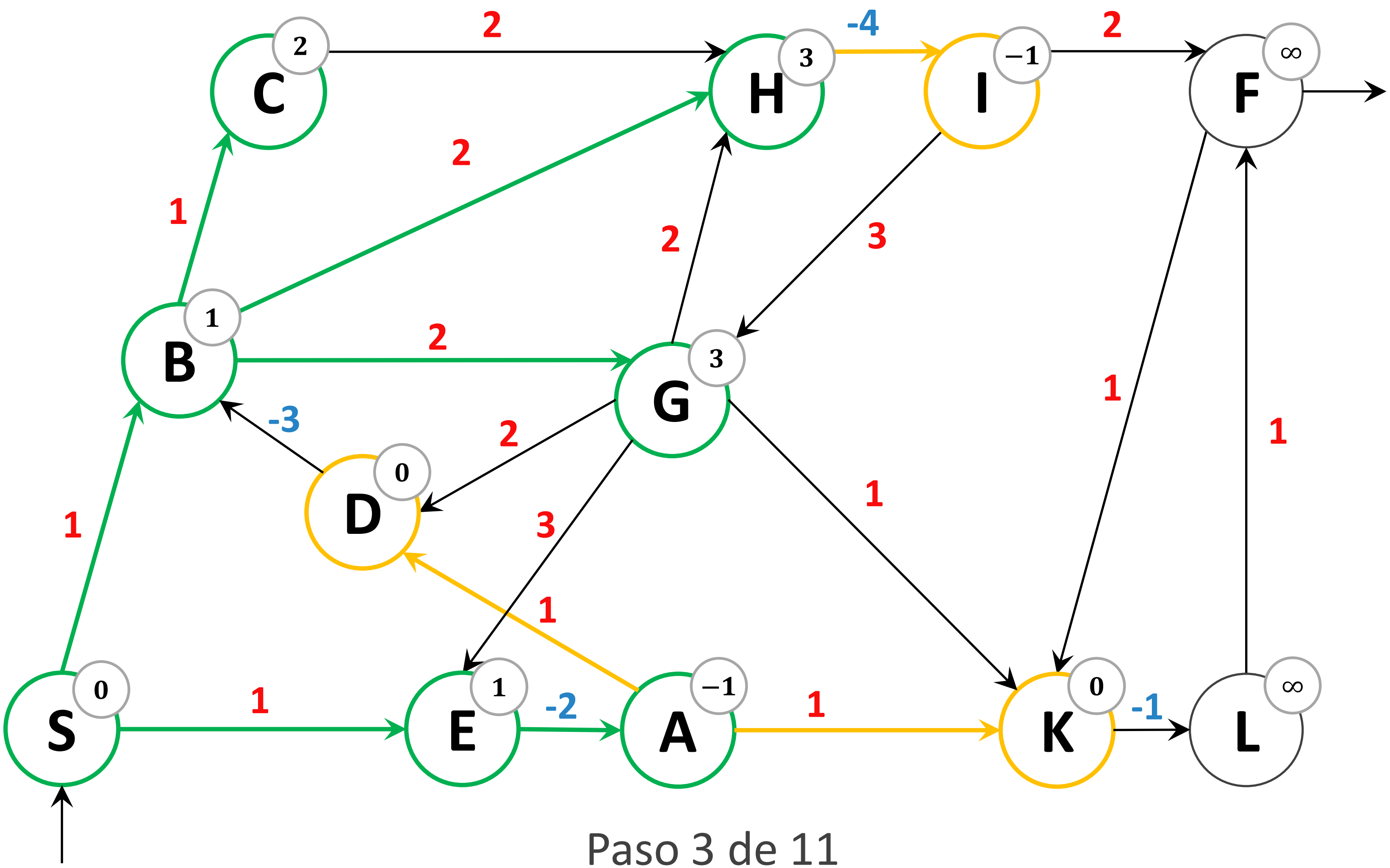
Paso 1 de 11

Bellman Ford en acción



Paso 2 de 11

Bellman Ford en acción



Paso 3 de 11

$$
\begin{array}{l}
\boldsymbol{rutas\ m\acute{a}s\ cortas\ con\ a\ lo\ m\acute{a}s\ i\ aristas}(\boldsymbol{G}(\boldsymbol{V},\boldsymbol{E}),\boldsymbol{s},\boldsymbol{w},\boldsymbol{i}): \\
\qquad \boldsymbol{s}.\boldsymbol{dist} \leftarrow \boldsymbol{0} \\
\qquad \boldsymbol{repeat}\ \boldsymbol{i}\ \boldsymbol{times}: \\
\qquad\qquad \boldsymbol{foreach}\ (\boldsymbol{u},\boldsymbol{v}) \in \boldsymbol{E}: \\
\qquad\qquad\qquad \boldsymbol{update}(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})
\end{array}
$$

Número de aristas de la ruta de menor costo

Sea p una ruta de menor costo de u a v en el grafo $G(V, E)$

¿Qué pasa si p contiene ciclos? ¿Deberíamos permitirlo?

¿Cuántas aristas puede haber **como máximo** en p?

$$\textit{todas las rutas más cortas}(G(V,E),s,w):$$

$$\quad s.dist \leftarrow 0$$

$$\quad \textit{repeat } |V|-1 \textit{ times}:$$

$$\quad\quad \textit{foreach } (u,v) \in E:$$

$$\quad\quad\quad \textit{update}(u,v,w)$$

Corrección del algoritmo

Suponiendo que el grafo es acíclico,

¿cómo demostramos que este algoritmo efectivamente encuentra las rutas más cortas?

¿Cuál es la complejidad del algoritmo?

Ciclos negativos

Si hay ciclos positivos o de costo 0, no hay problema

¿Qué pasa con las distancias de los nodos en cada paso?

¿Cómo podemos detectar si hay un ciclo negativo?

$$\boldsymbol{bellman\ ford}(\boldsymbol{G}(\boldsymbol{V},\boldsymbol{E}),\boldsymbol{s},\boldsymbol{w}):$$

$$\quad \boldsymbol{s.dist} \leftarrow \boldsymbol{0}$$

$$\quad \boldsymbol{repeat}\ |\boldsymbol{V}| - \boldsymbol{1}\ \boldsymbol{times}:$$

$$\quad\quad \boldsymbol{foreach}\ (\boldsymbol{u},\boldsymbol{v}) \in \boldsymbol{E}:$$

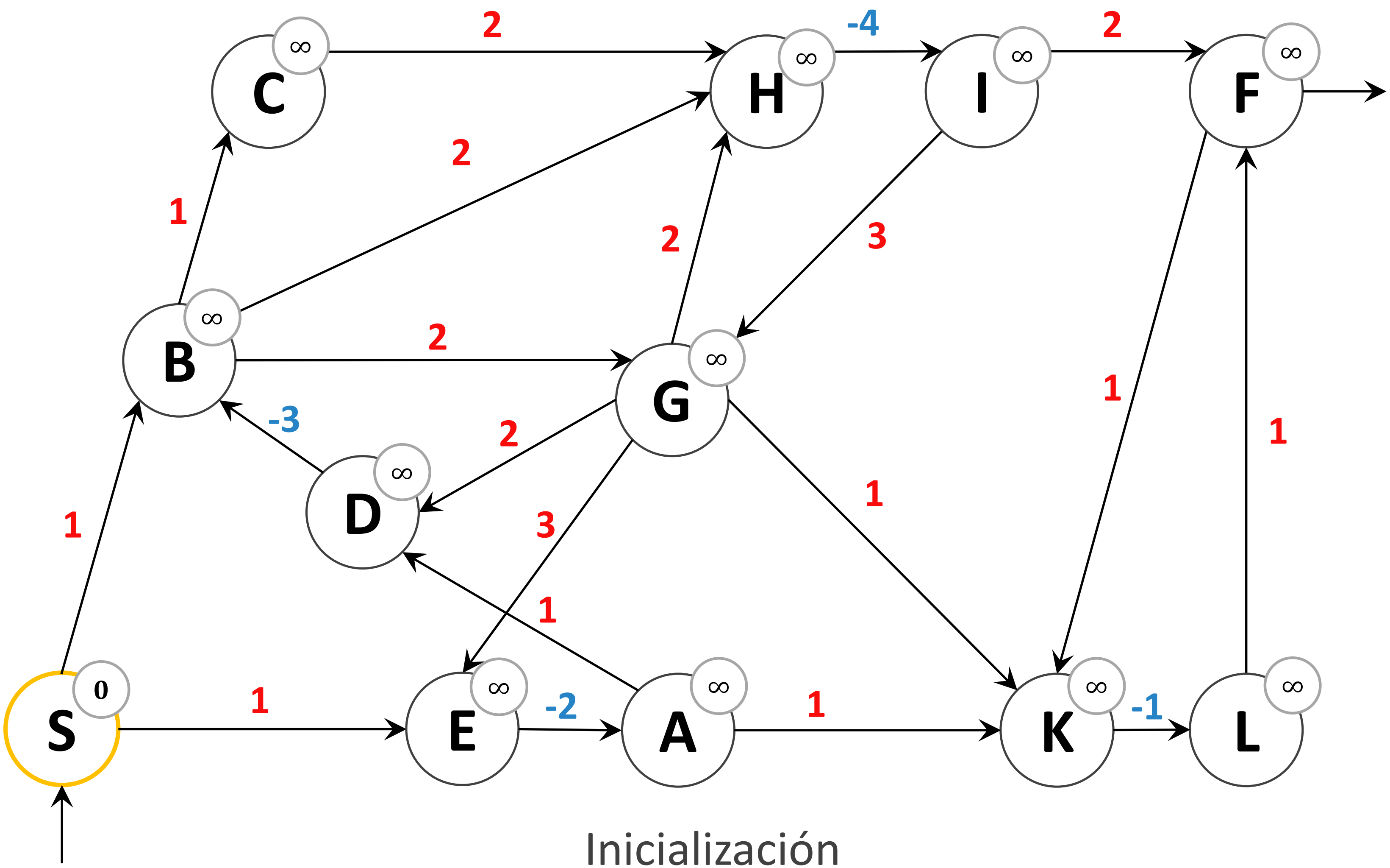
$$\quad\quad\quad \boldsymbol{update}(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})$$

$$\quad \boldsymbol{foreach}\ (\boldsymbol{u},\boldsymbol{v}) \in \boldsymbol{E}:$$

$$\quad\quad \boldsymbol{if}\ \boldsymbol{u.d} + \boldsymbol{w}(\boldsymbol{u},\boldsymbol{v}) < \boldsymbol{v.d},\ \boldsymbol{return\ false}$$

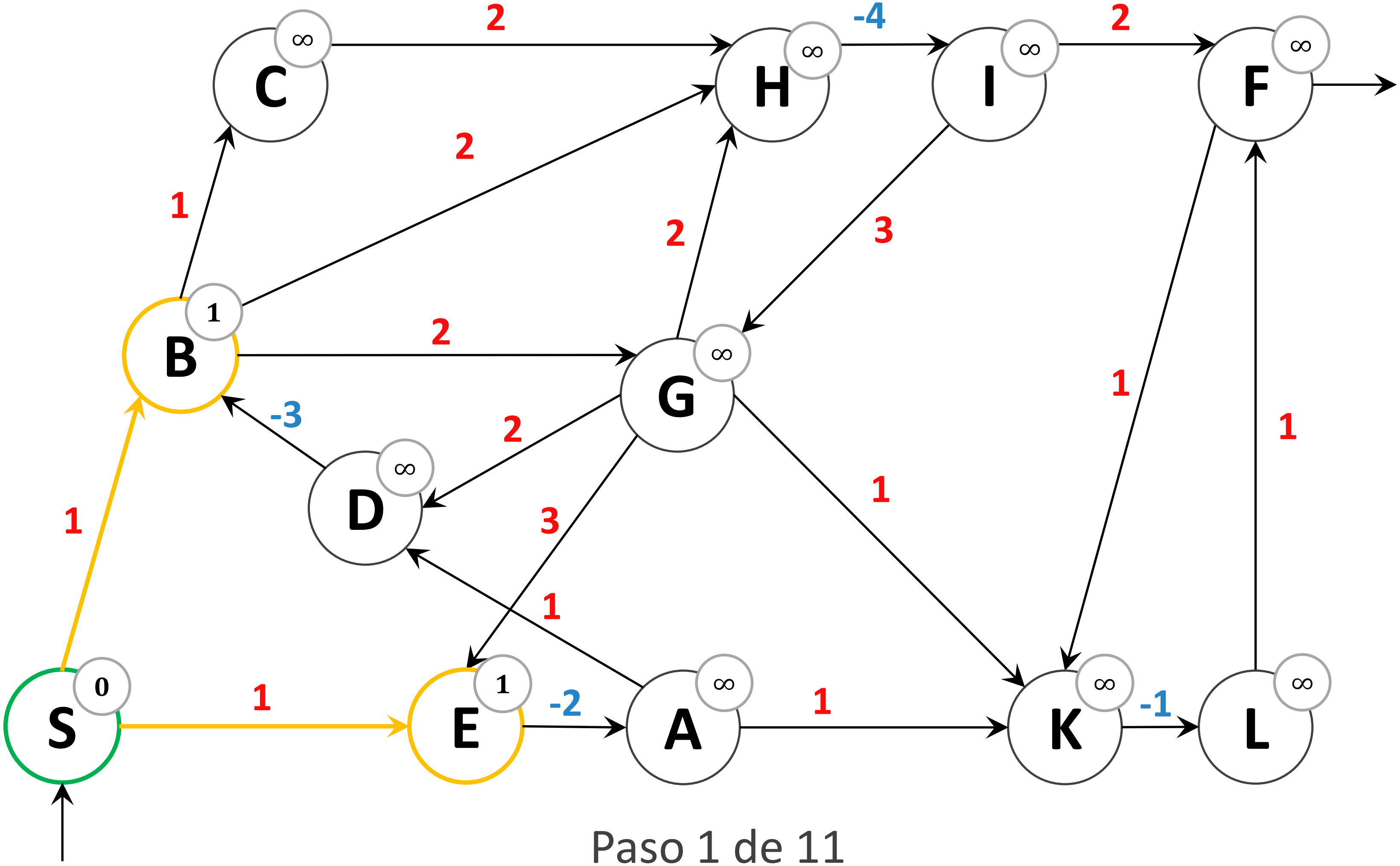
$$\quad \boldsymbol{return\ true}$$

Bellman Ford en acción



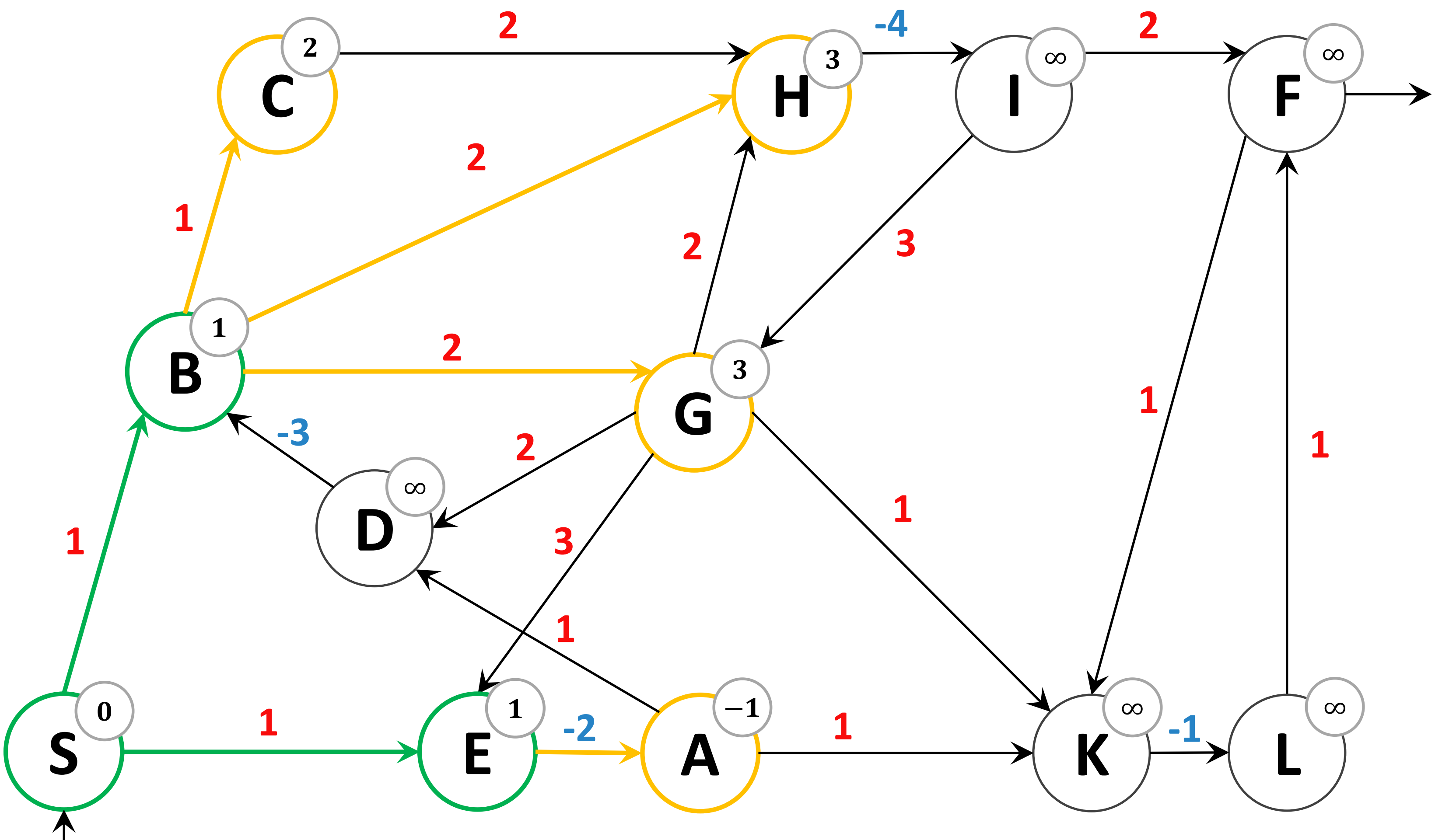
Inicialización

Bellman Ford en acción



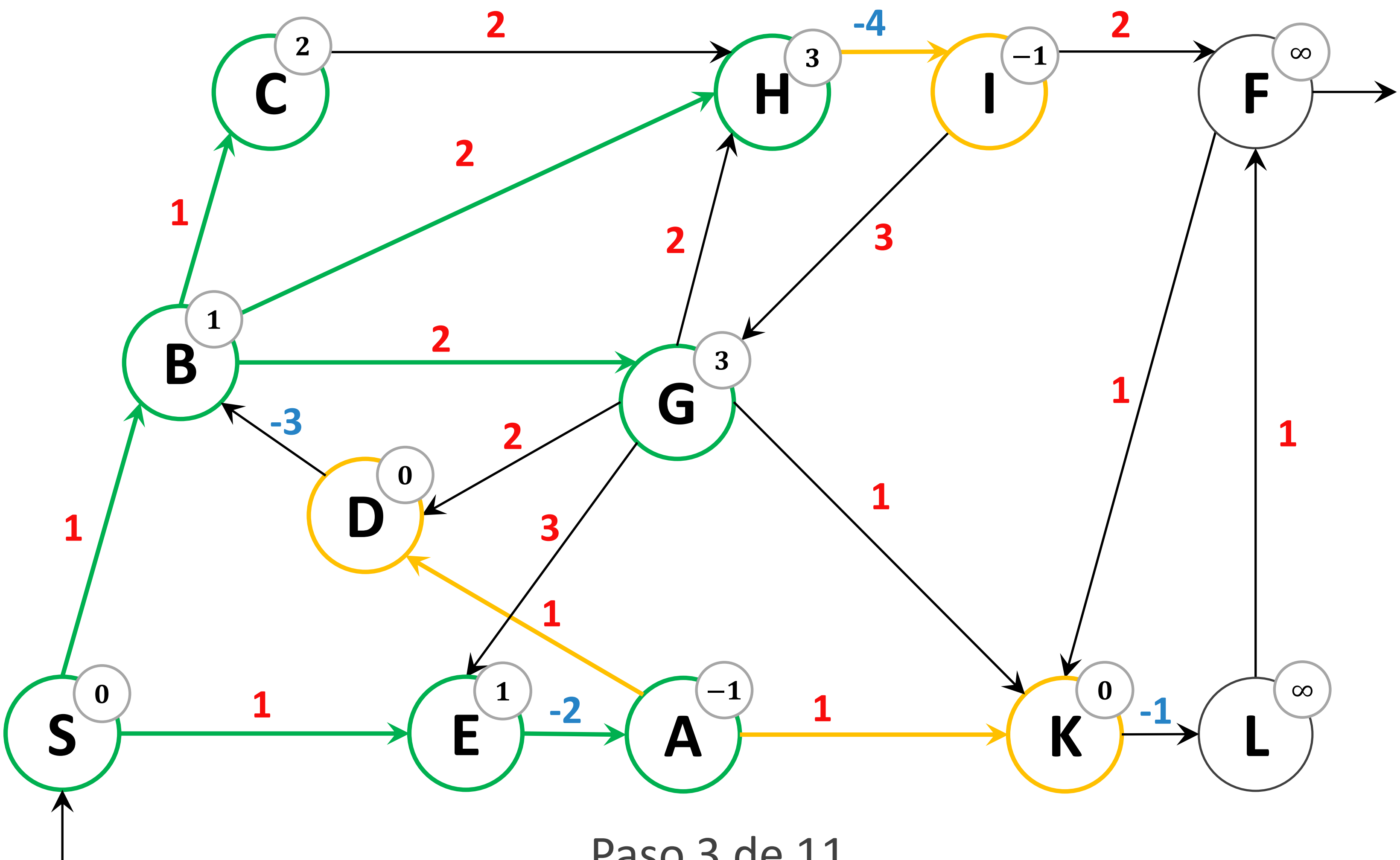
Paso 1 de 11

Bellman Ford en acción



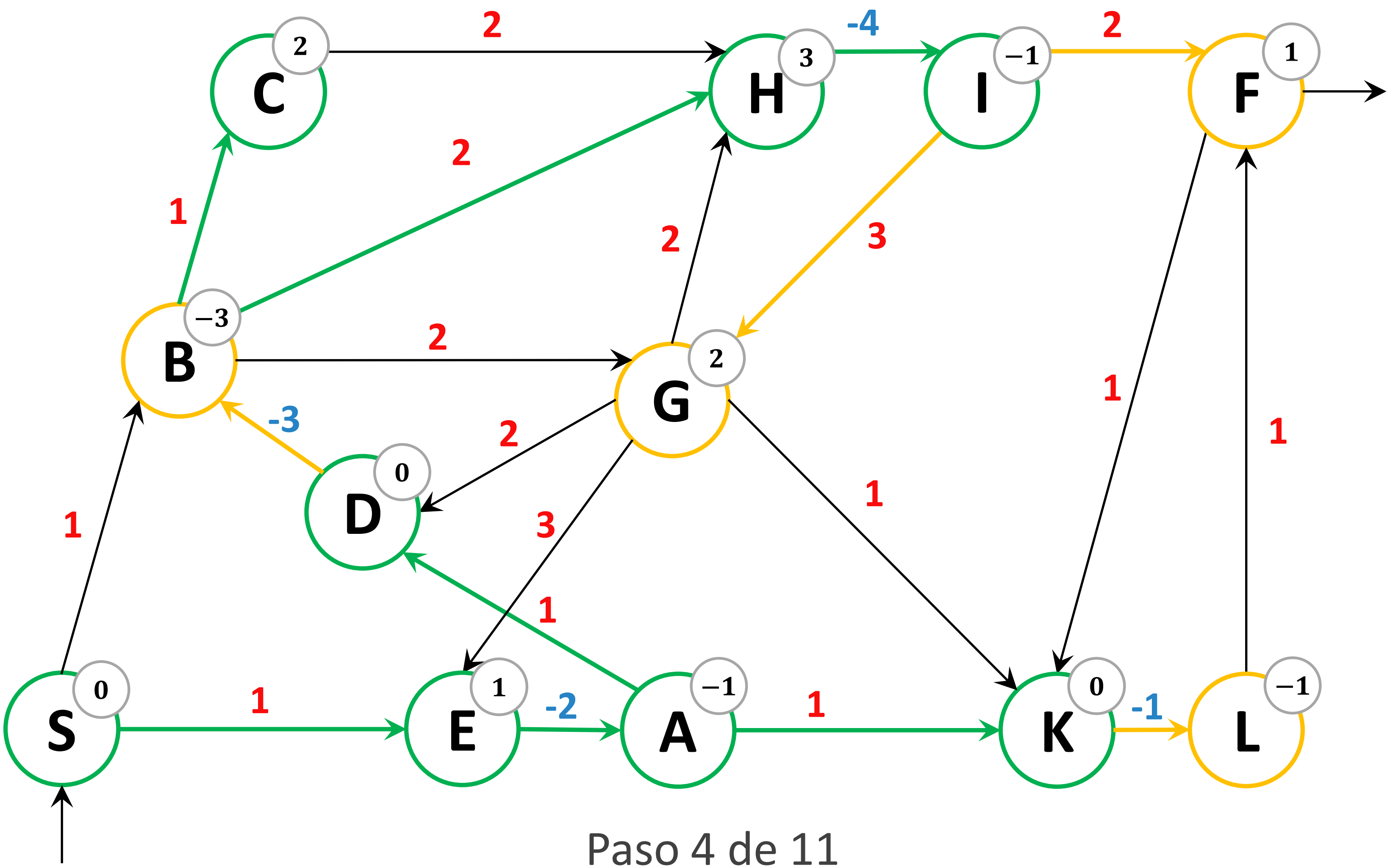
Paso 2 de 11

Bellman Ford en acción



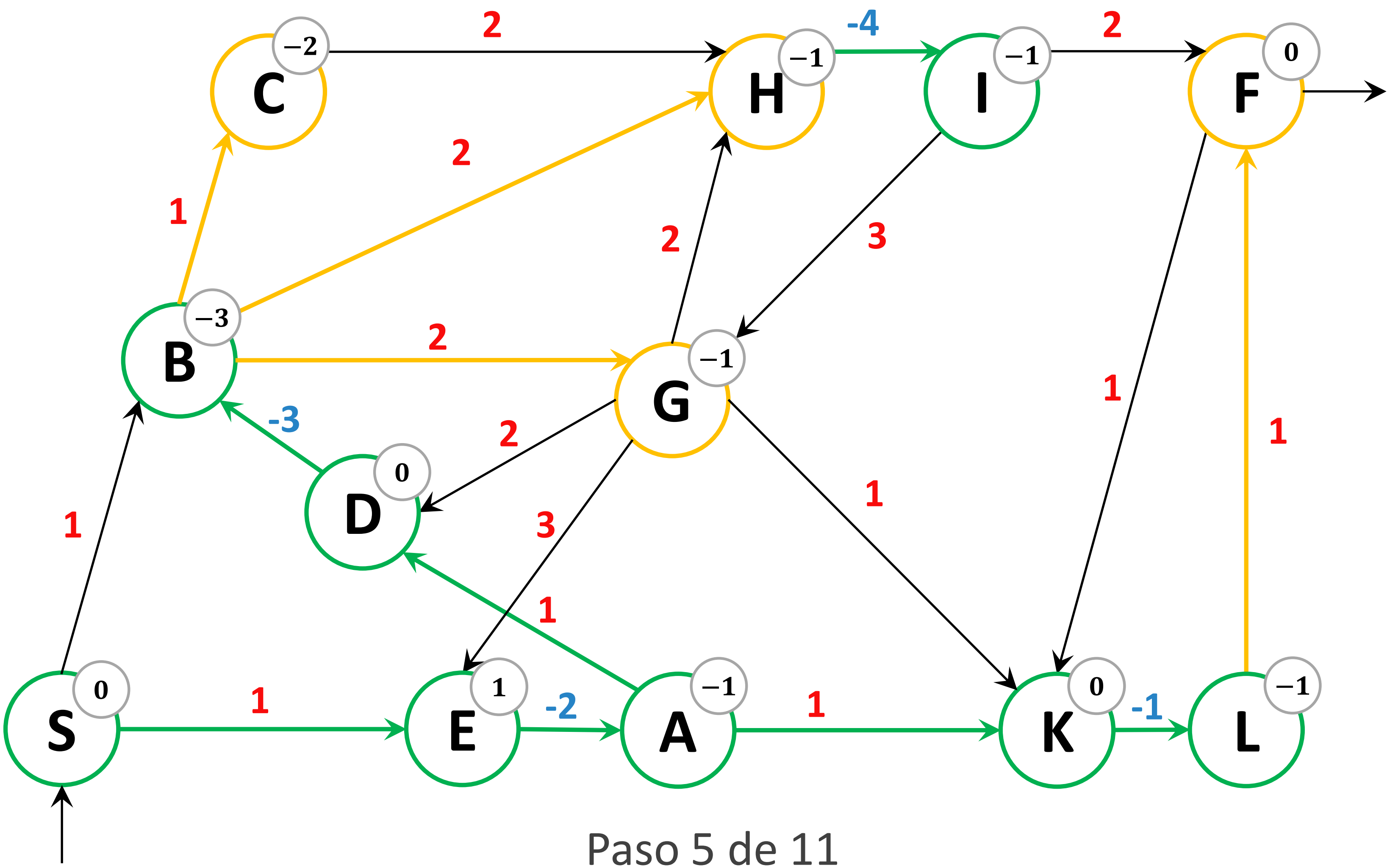
Paso 3 de 11

Bellman Ford en acción



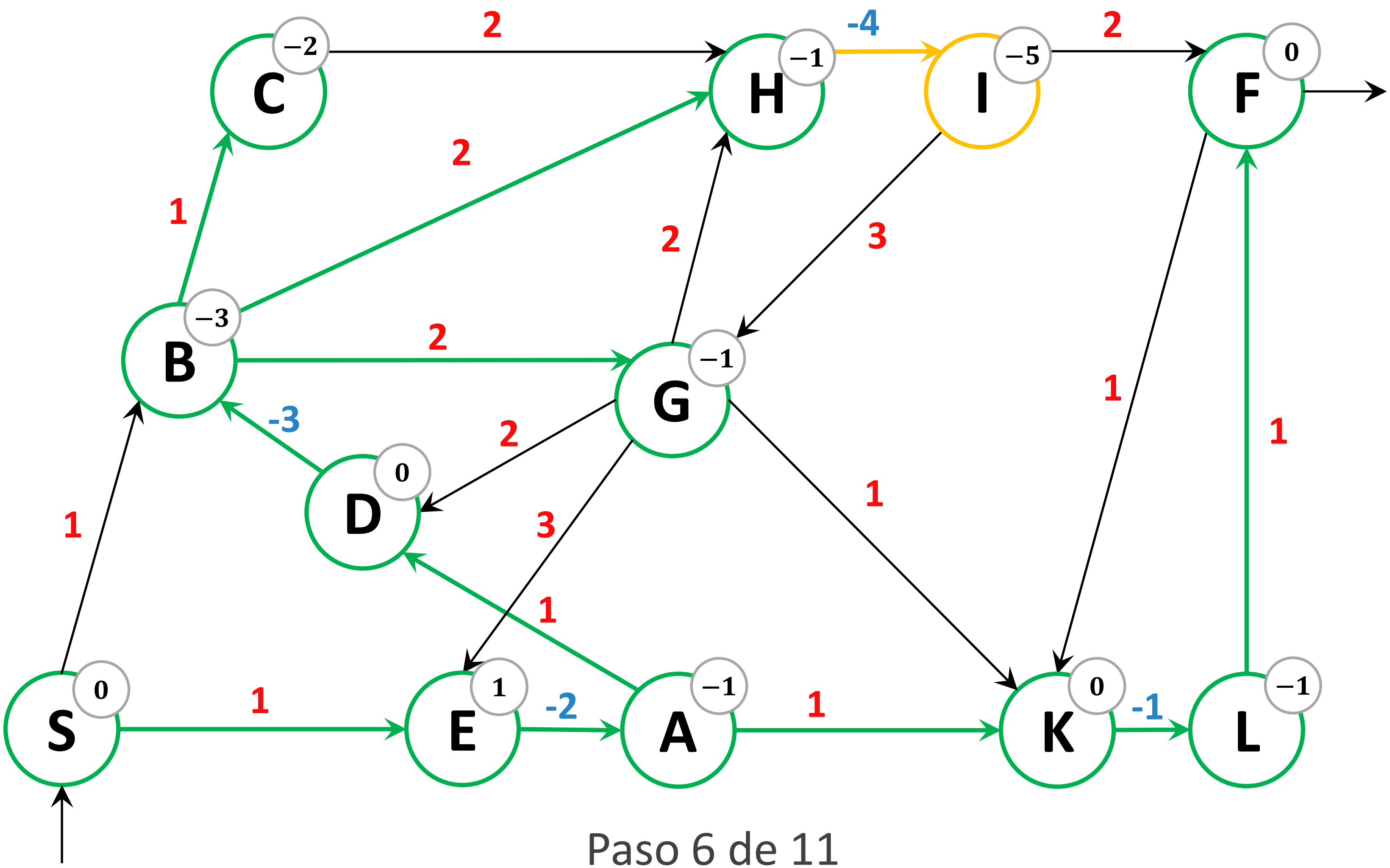
Paso 4 de 11

Bellman Ford en acción



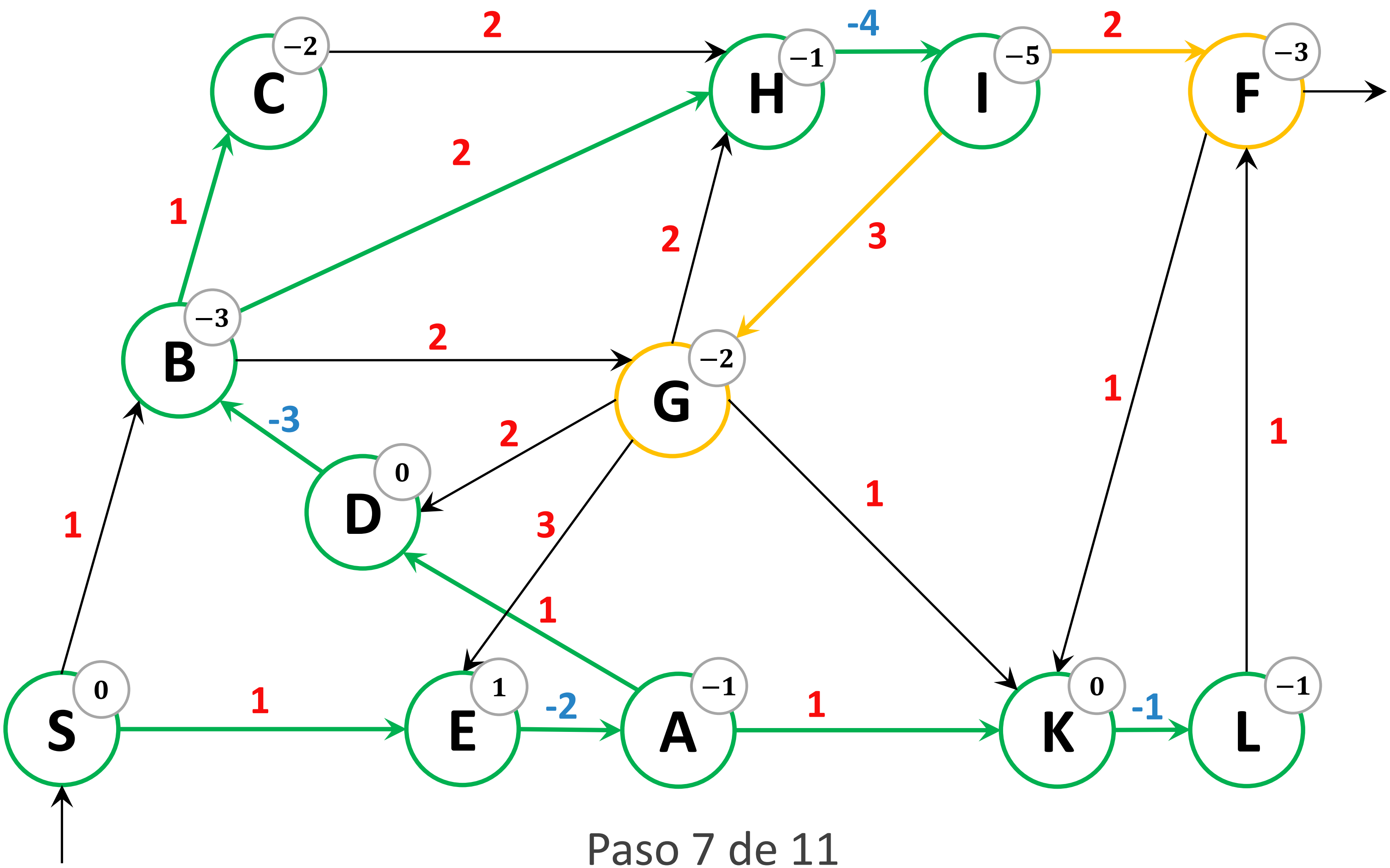
Paso 5 de 11

Bellman Ford en acción



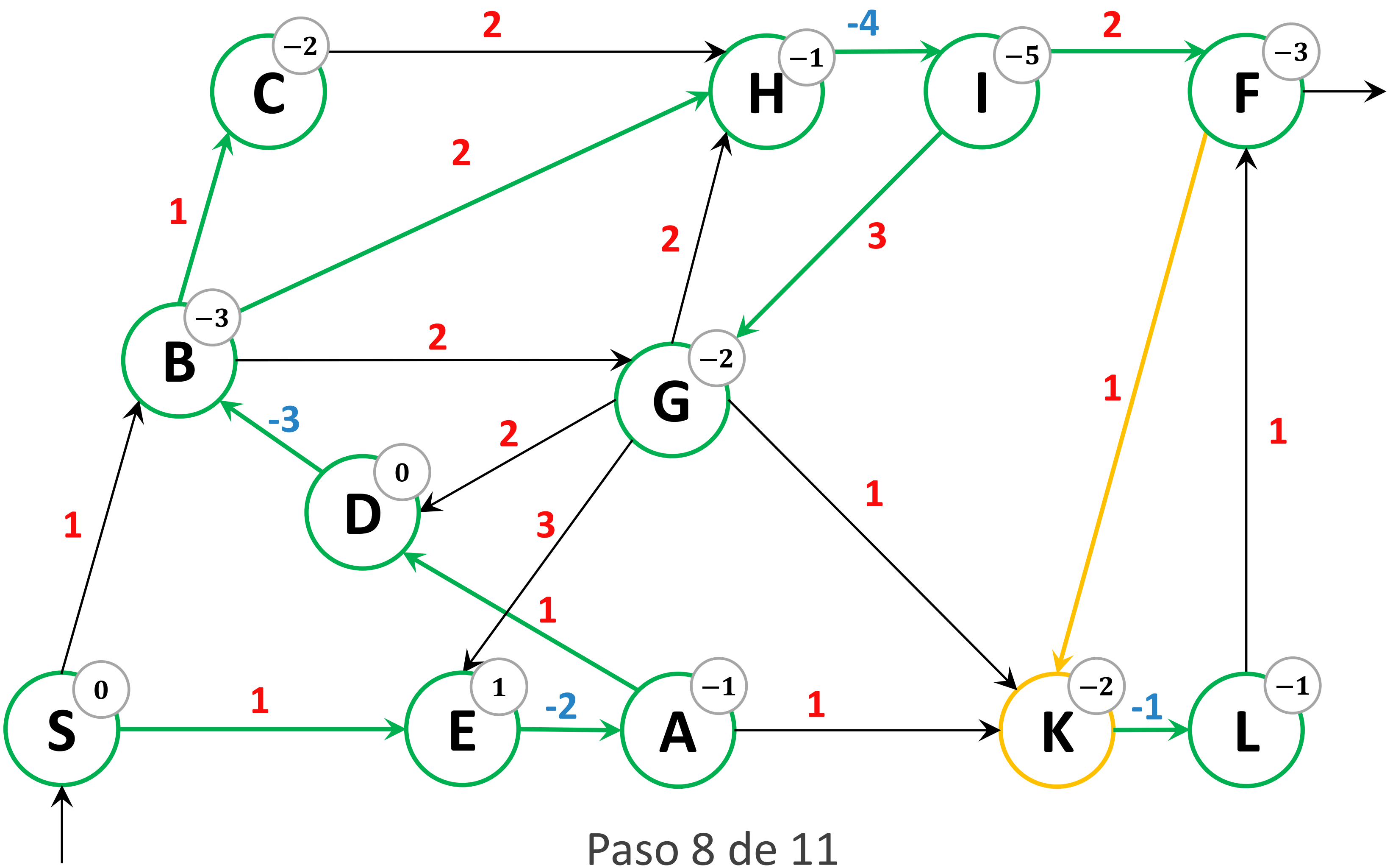
Paso 6 de 11

Bellman Ford en acción



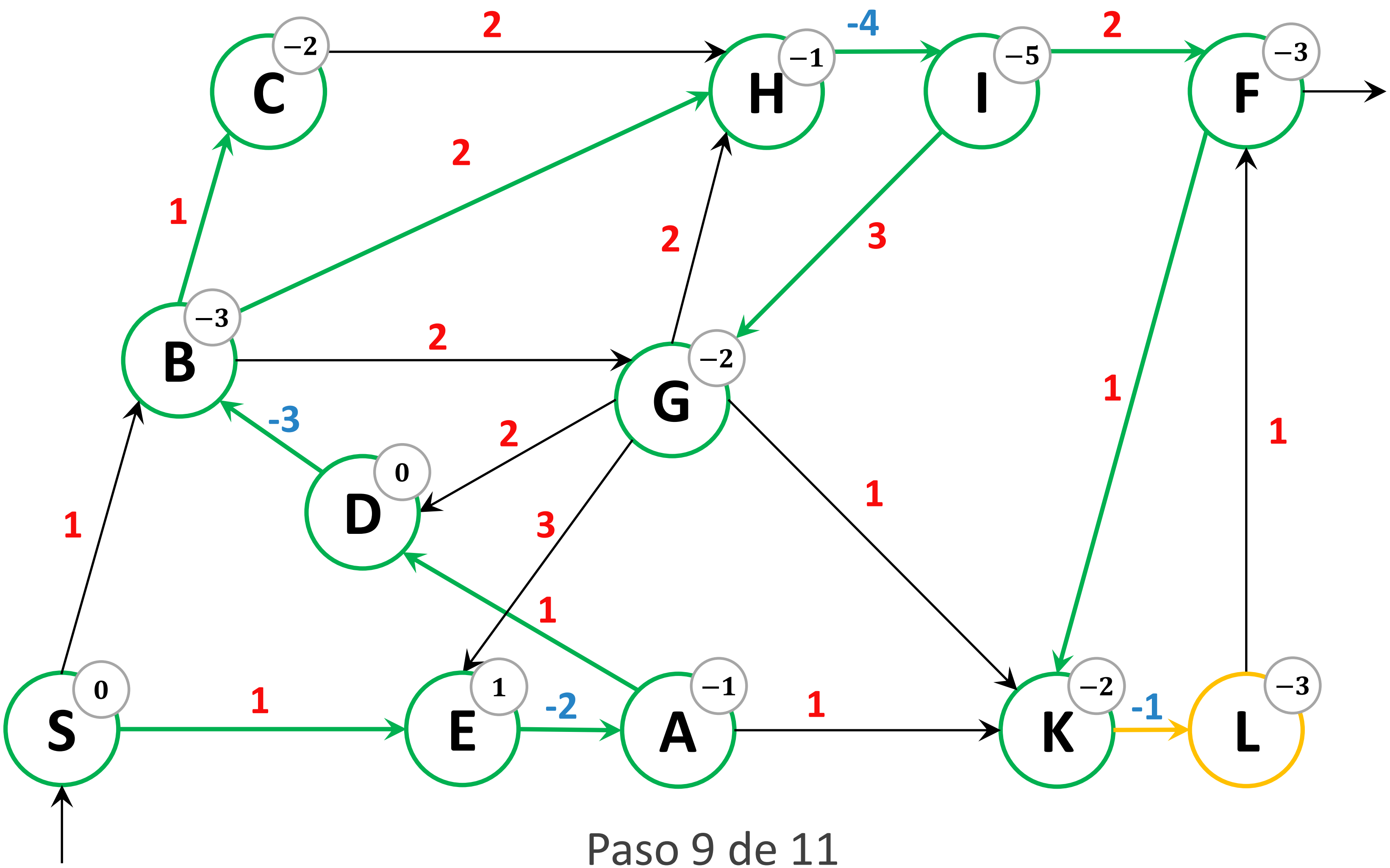
Paso 7 de 11

Bellman Ford en acción



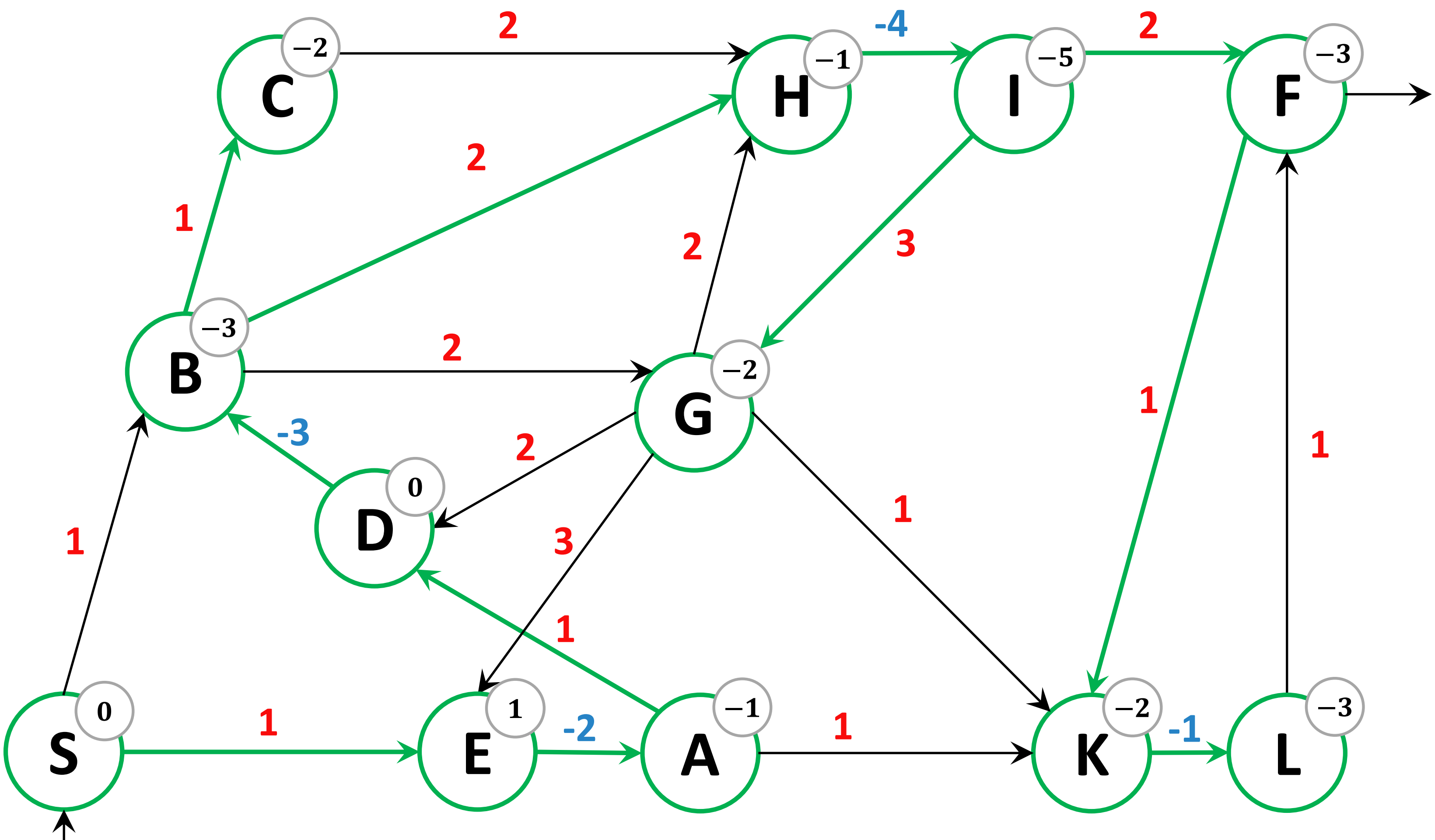
Paso 8 de 11

Bellman Ford en acción



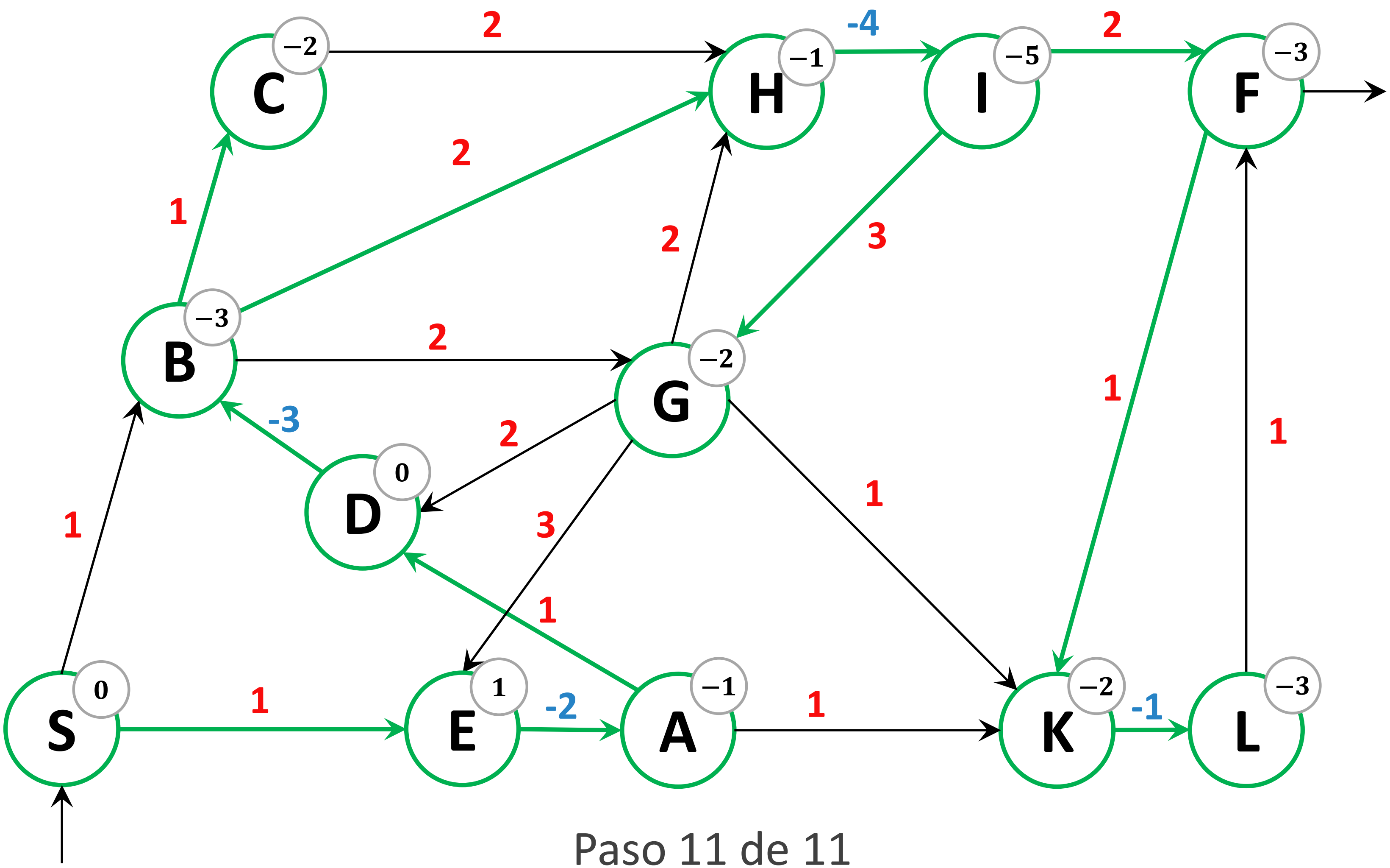
Paso 9 de 11

Bellman Ford en acción



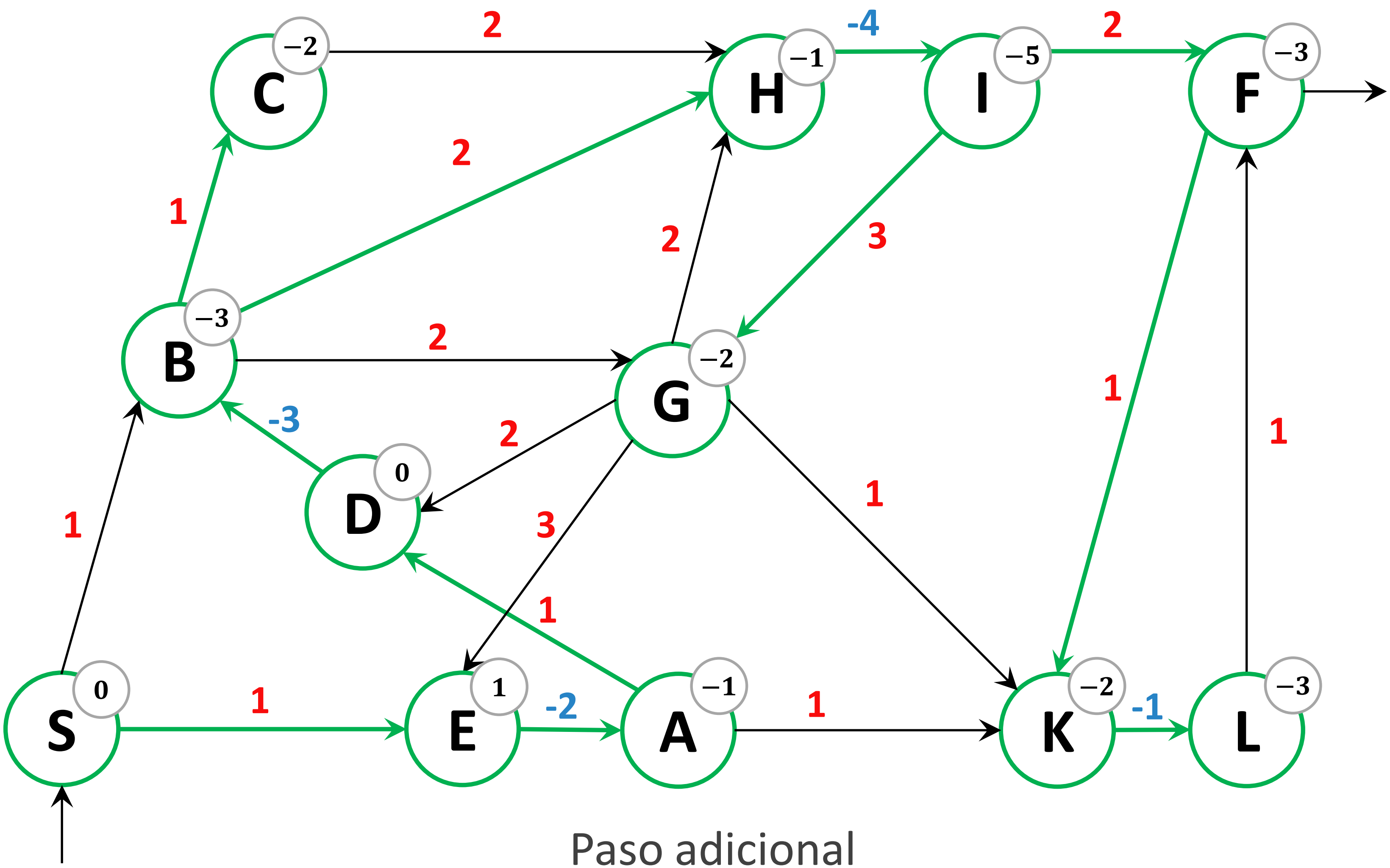
Paso 10 de 11

Bellman Ford en acción



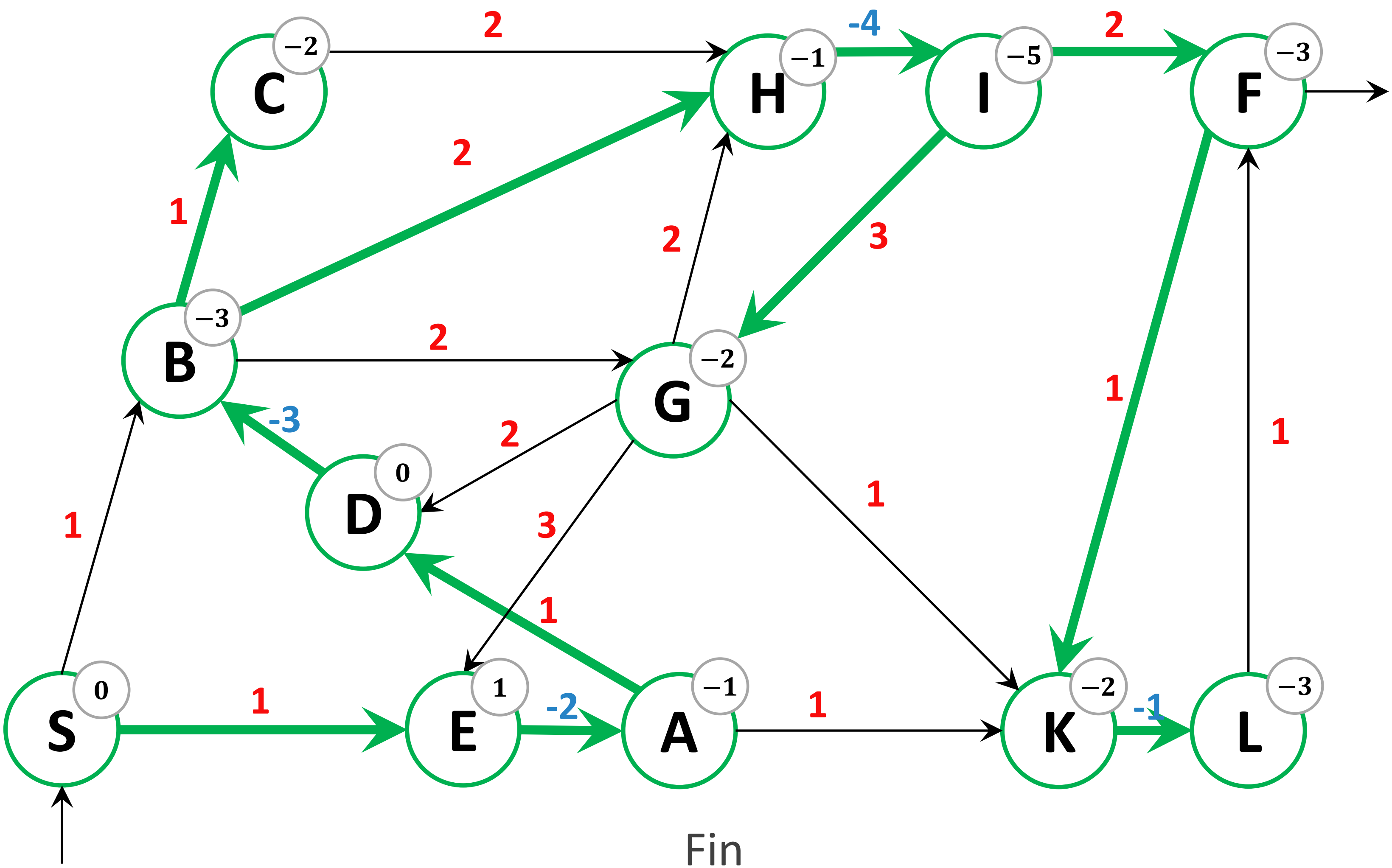
Paso 11 de 11

Bellman Ford en acción



Paso adicional

Bellman Ford en acción



Fin

Complejidad

¿Es necesario iterar sobre **todas** las aristas en cada paso?

¿Cómo podríamos mejorar este algoritmo?

Nótese que el peor caso siempre será $O(VE)$