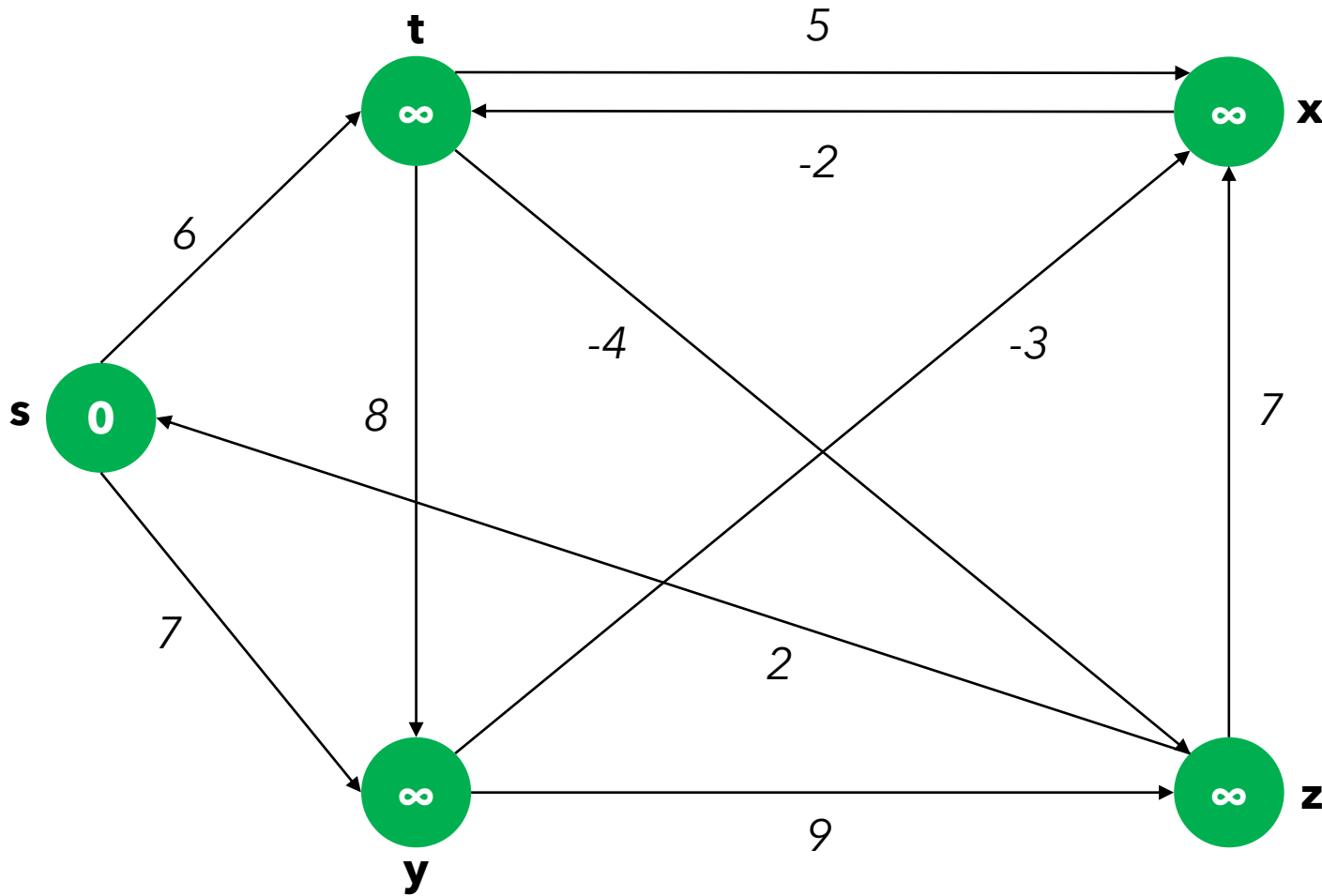


L'algoritmo di Bellman-Ford **(*single-source shortest-path algorithm*)**

Liceo G.B. Brocchi – Bassano del Grappa (VI)
Liceo Scientifico – opzione scienze applicate
Giovanni Mazzocchin

L'algoritmo di Bellman-Ford



vogliamo trovare i cammini di costo minimo da **s** verso tutti gli altri nodi

le chiavi dei nodi sono valorizzate con il costo dello shortest path da **s** noto all'iterazione corrente (inizialmente infinito)

a differenza dell'algoritmo di Dijkstra, sono ammessi costi negativi

L'algoritmo di Bellman-Ford

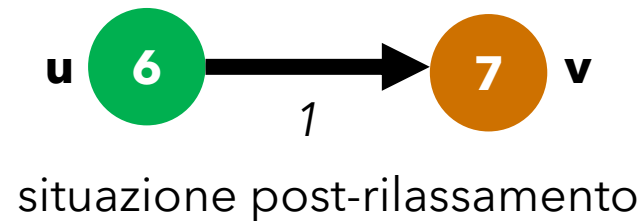
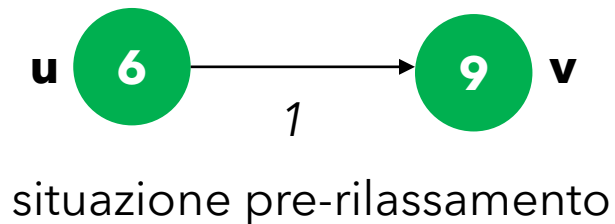
```
bellman_ford_part_1(G):  
    repeat |G.V| - 1 times:  
        for each edge (u, v) in G:  
            if u.cost + cost(u, v) < v.cost:  
                v.cost = u.cost + cost(u, v)  
                v.predecessor = u
```

- s : sorgente da cui calcolare i cammini minimi verso tutti gli altri nodi
- $|G.V|$: numero di nodi del grafo G
- $u.cost$: costo minimo noto all'iterazione corrente del path $s \rightarrow u$
- $cost(u, v)$: costo dell'arco (u, v)

L'algoritmo di Bellman-Ford

```
bellman_ford_part_1(G):  
  repeat |G.V| - 1 times:  
    for each edge (u, v) in G:  
      if u.cost + cost(u, v) < v.cost:  
        v.cost = u.cost + cost(u, v)  
        v.predecessor = u
```

le istruzioni interne al ciclo interno effettuano il **rilassamento**:

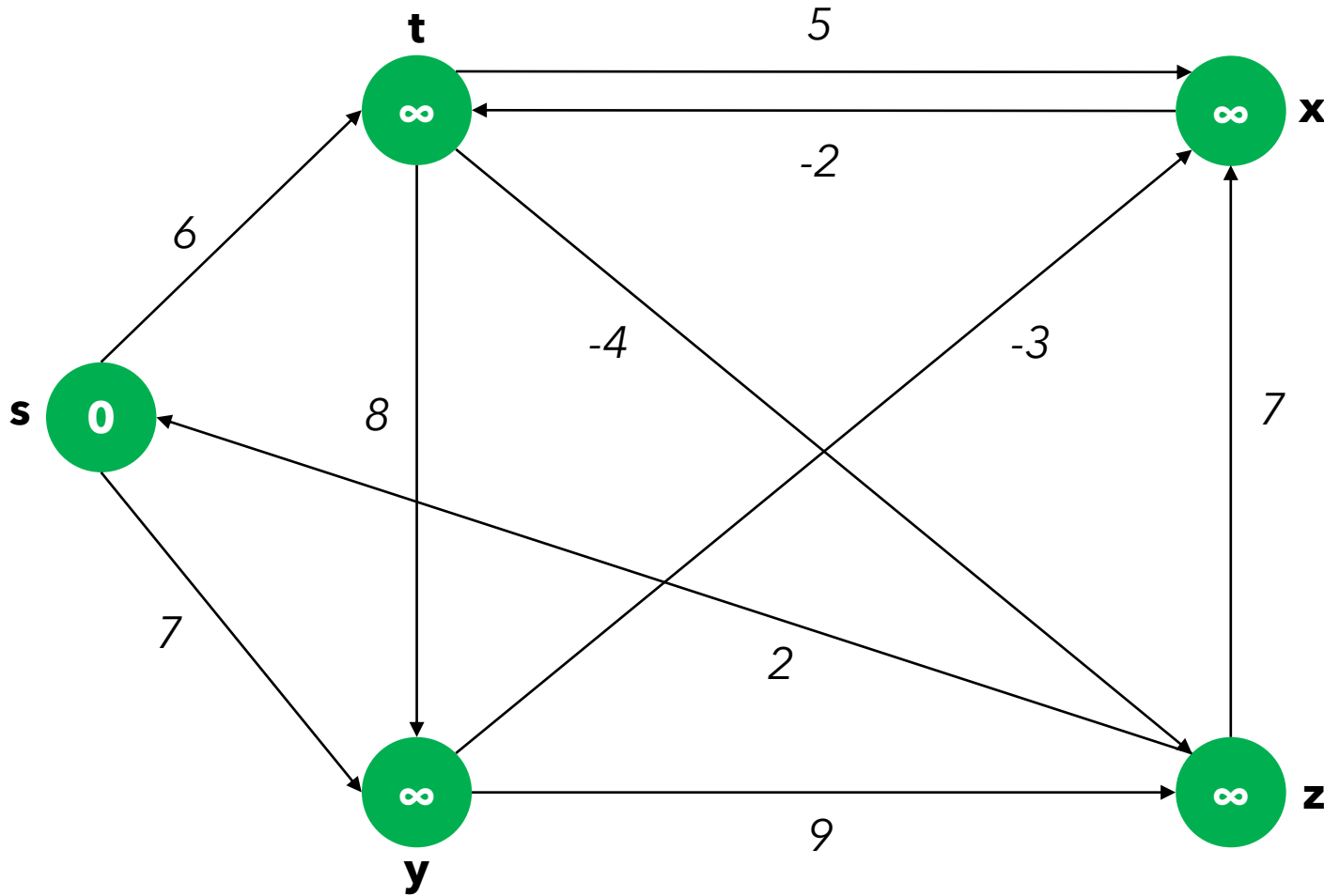


L'algoritmo di Bellman-Ford

```
bellman_ford_part_1(G):  
    repeat |G.V| - 1 times:  
        for each edge (u, v) in G:  
            if u.cost + cost(u, v) < v.cost:  
                v.cost = u.cost + cost(u, v)  
                v.predecessor = u
```

- il nostro grafo di esempio ha 5 nodi, quindi il ciclo esterno verrà eseguito 4 volte
- quindi si tratterà di eseguire 4 passate su tutti gli archi

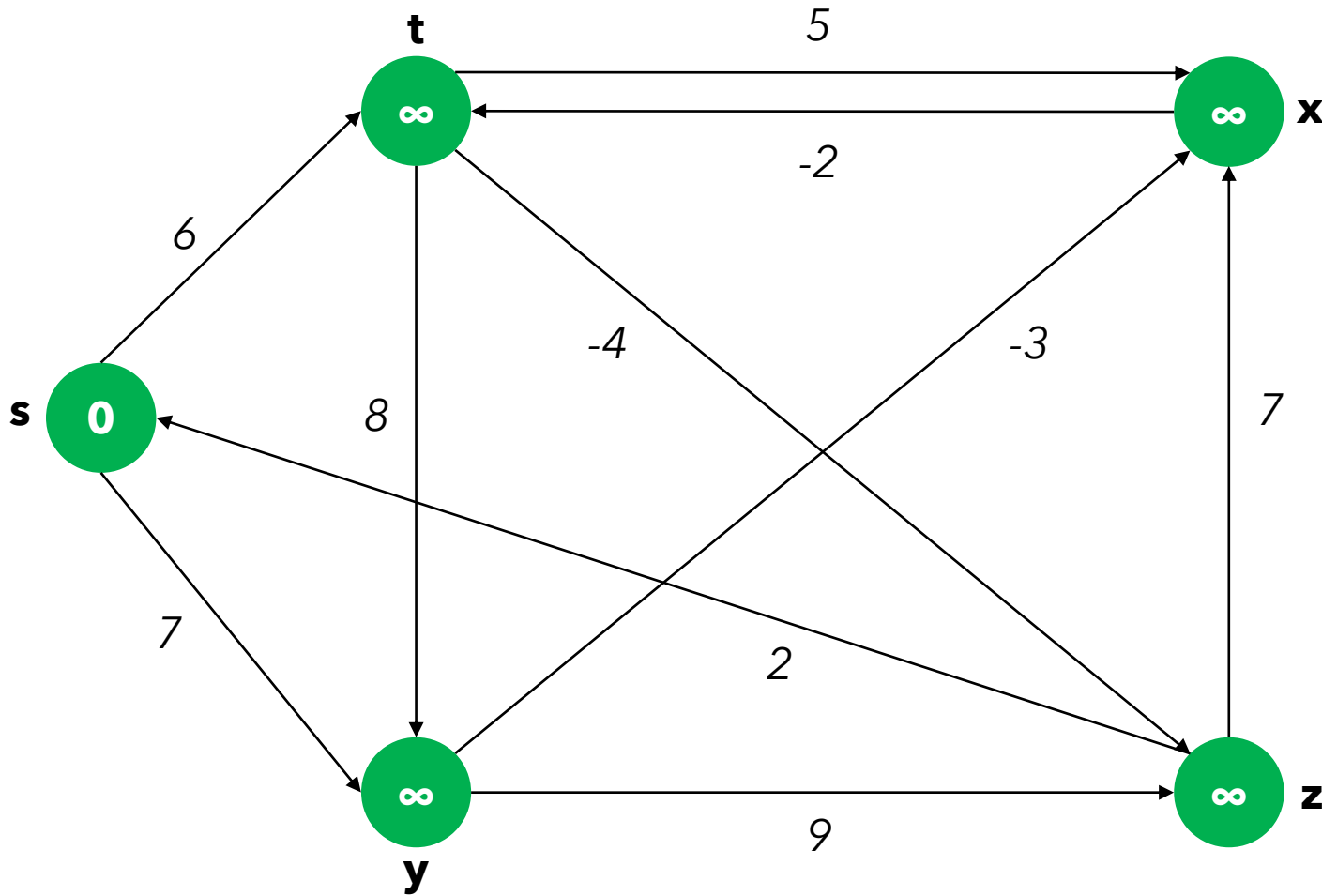
L'algoritmo di Bellman-Ford



iteriamo sugli archi del grafo
ordinati (arbitrariamente) così:

t -> x
t -> y
t -> z
x -> t
y -> x
y -> z
z -> x
z -> s
s -> t
s -> y

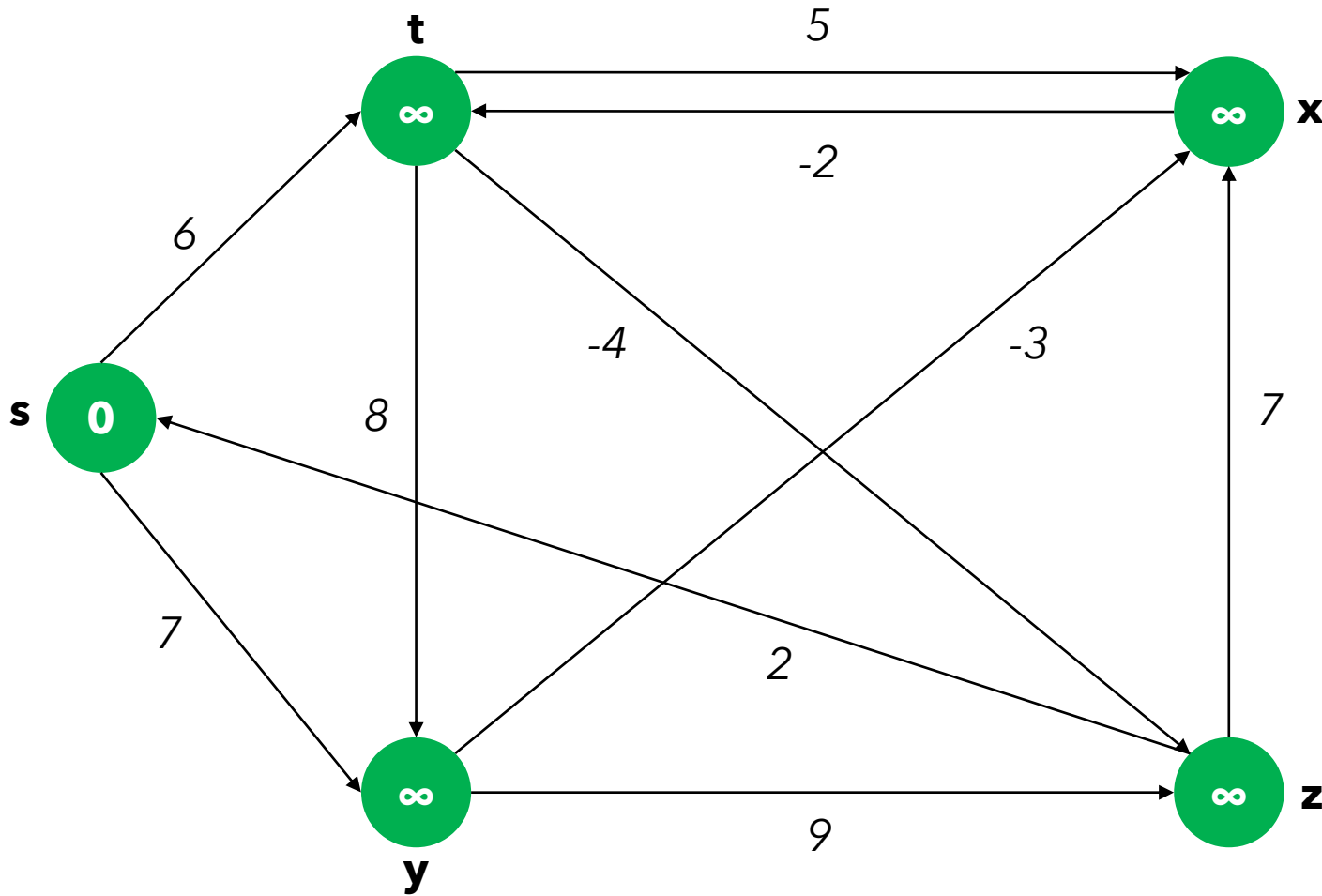
Bellman-Ford – 1st pass



t → **x**
t → y
t → z
x → t
y → x
y → z
z → x
z → s
s → t
s → y

**rilassamento di t → x
(nessun effetto)**

Bellman-Ford – 1st pass



t \rightarrow x

t \rightarrow y

t \rightarrow z

x \rightarrow t

y \rightarrow x

y \rightarrow z

z \rightarrow x

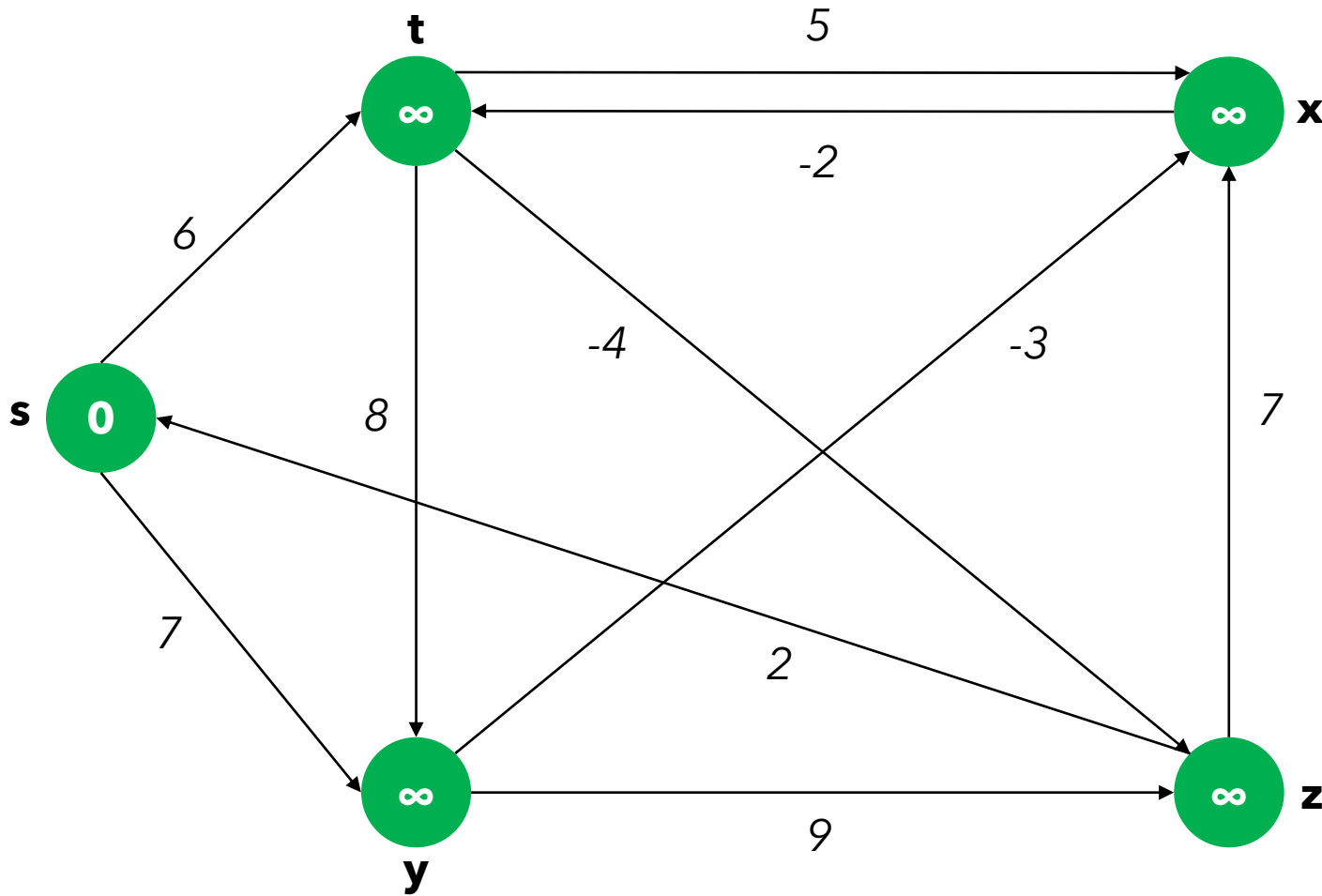
z \rightarrow s

s \rightarrow t

s \rightarrow y

**rilassamento di t \rightarrow y
(nessun effetto)**

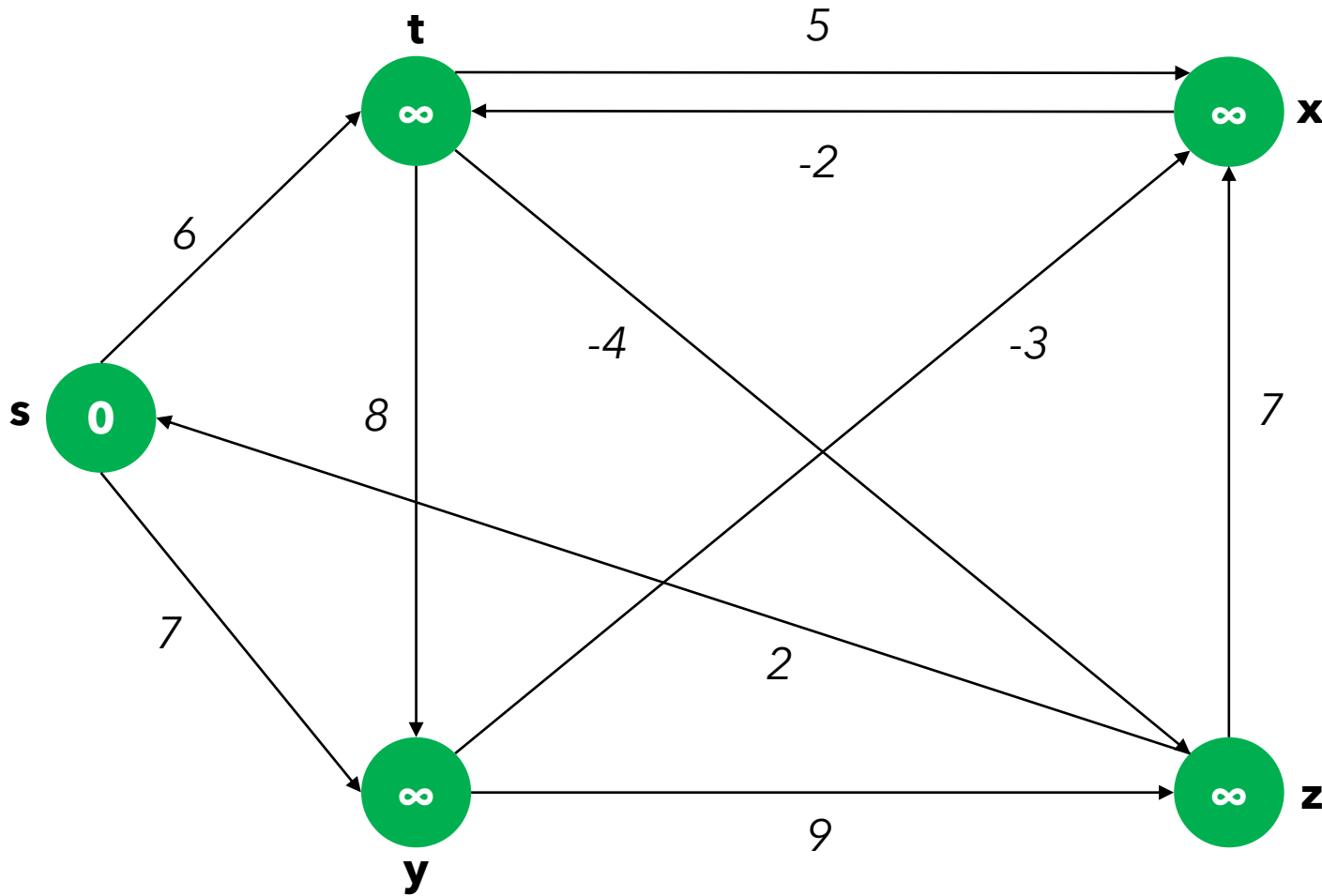
Bellman-Ford – 1st pass



t -> x
t -> y
t -> z
x -> t
y -> x
y -> z
z -> x
z -> s
s -> t
s -> y

**rilassamento di t -> z
(nessun effetto)**

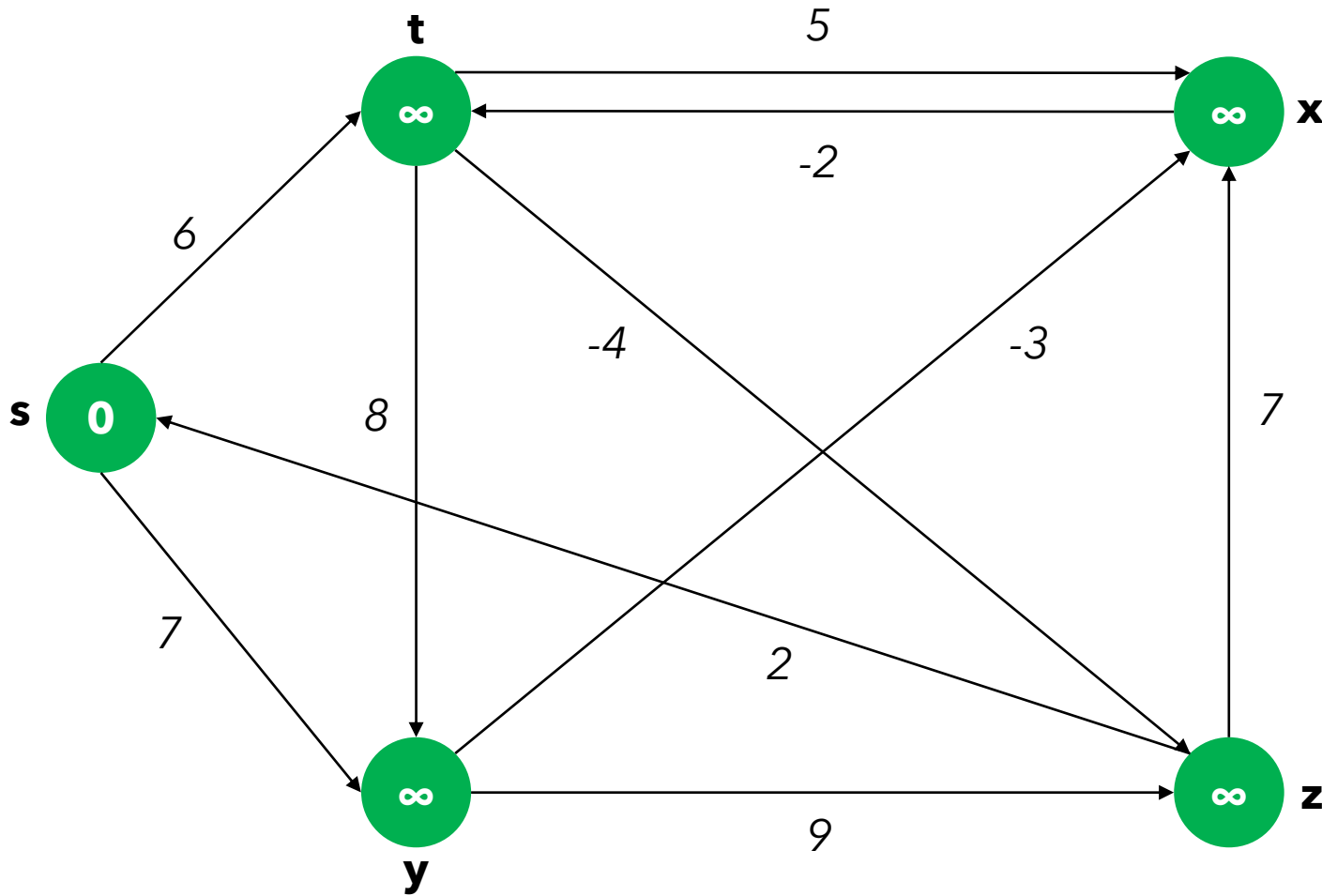
Bellman-Ford – 1st pass



t \rightarrow x
t \rightarrow y
t \rightarrow z
x \rightarrow t
y \rightarrow x
y \rightarrow z
z \rightarrow x
z \rightarrow s
s \rightarrow t
s \rightarrow y

**rilassamento di x \rightarrow t
(nessun effetto)**

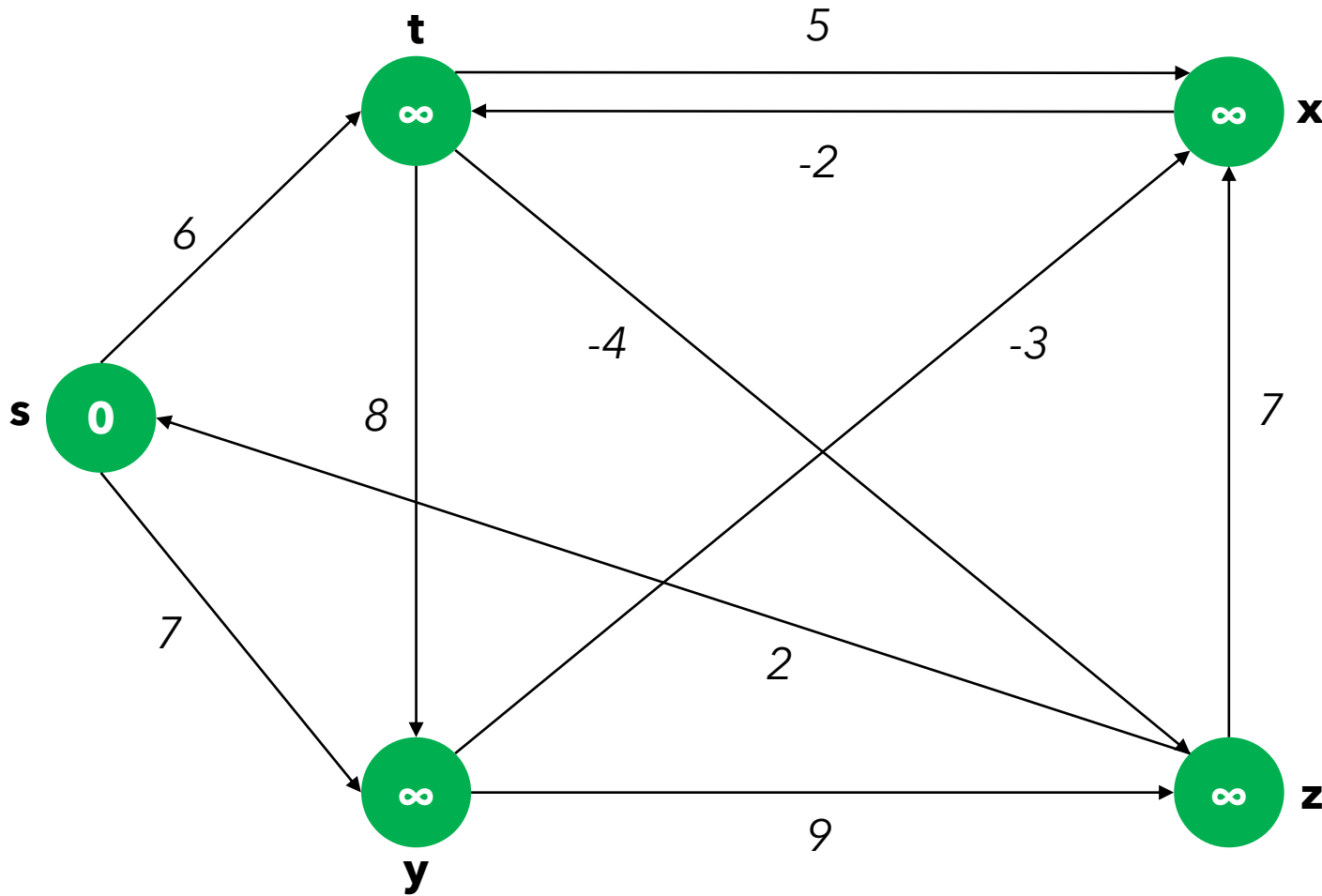
Bellman-Ford – 1st pass



t \rightarrow x
t \rightarrow y
t \rightarrow z
x \rightarrow t
y \rightarrow x
y \rightarrow z
z \rightarrow x
z \rightarrow s
s \rightarrow t
s \rightarrow y

**rilassamento di y \rightarrow x
(nessun effetto)**

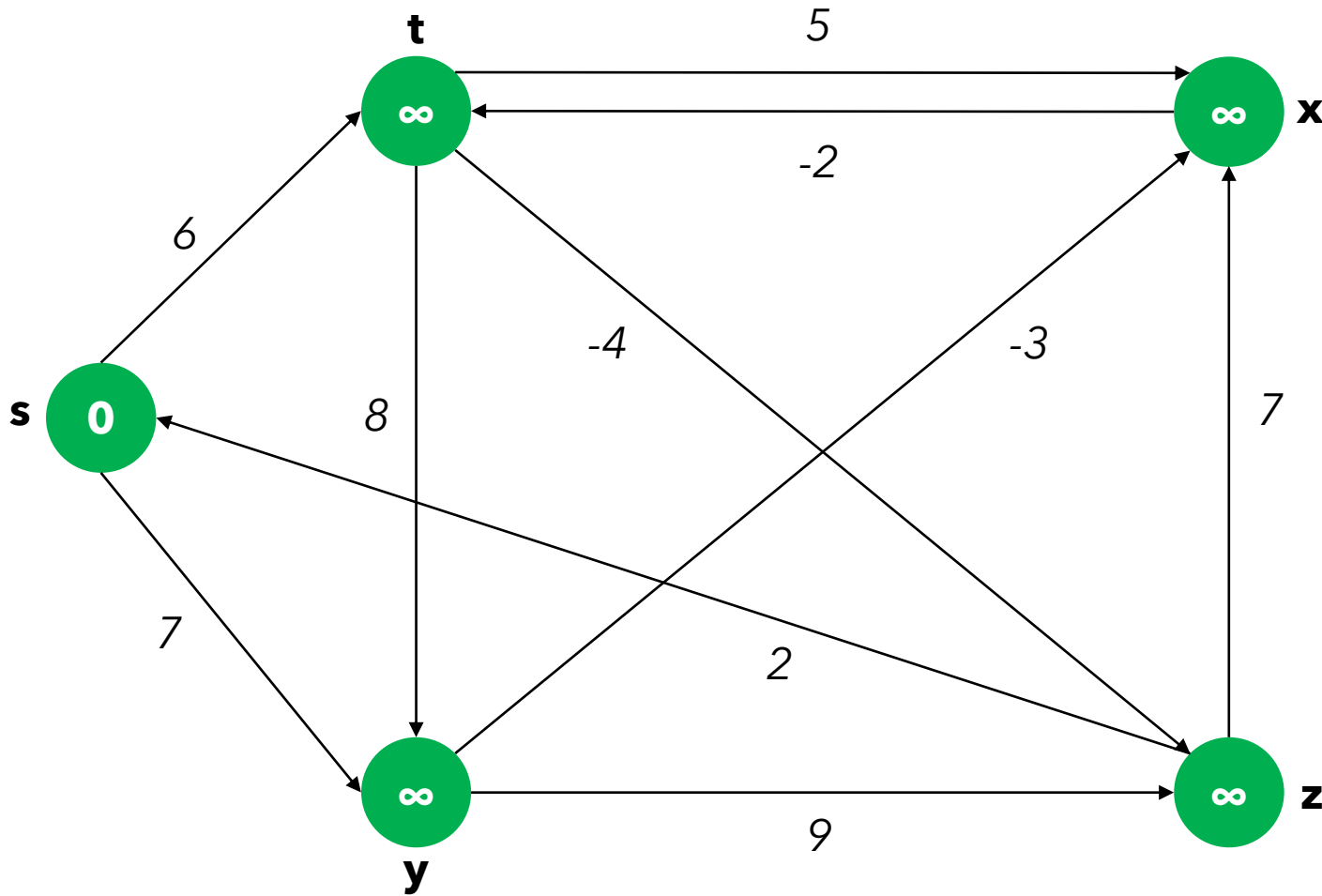
Bellman-Ford – 1st pass



t → x
t → y
t → z
x → t
y → x
y → z
z → x
z → s
s → t
s → y

**rilassamento di y → z
(nessun effetto)**

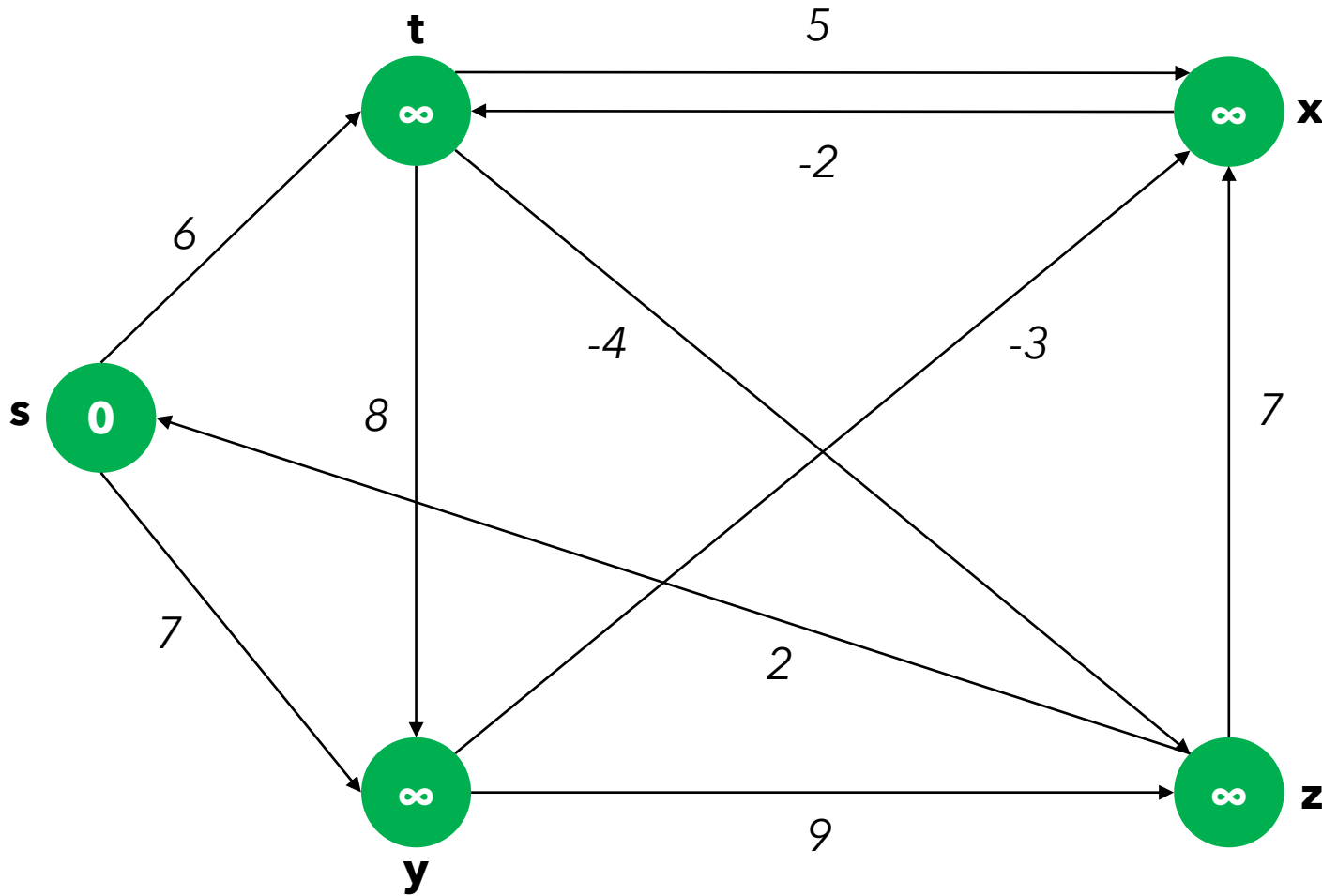
Bellman-Ford – 1st pass



t \rightarrow x
t \rightarrow y
t \rightarrow z
x \rightarrow t
y \rightarrow x
y \rightarrow z
z \rightarrow x
z \rightarrow s
s \rightarrow t
s \rightarrow y

rilassamento di z \rightarrow x
(nessun effetto)

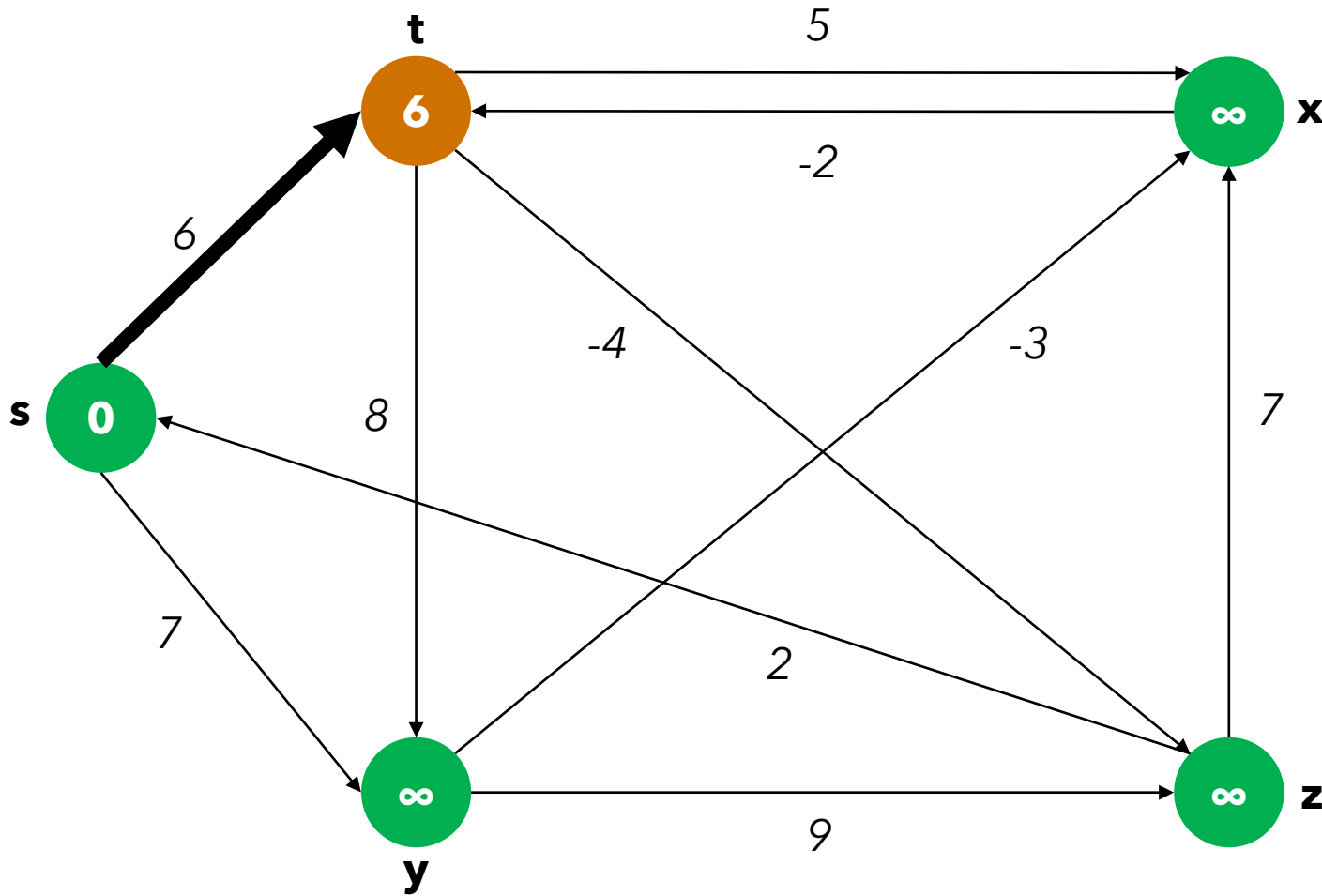
Bellman-Ford – 1st pass



t \rightarrow x
t \rightarrow y
t \rightarrow z
x \rightarrow t
y \rightarrow x
y \rightarrow z
z \rightarrow x
z \rightarrow s
s \rightarrow t
s \rightarrow y

**rilassamento di z \rightarrow s
(nessun effetto)**

Bellman-Ford – 1st pass

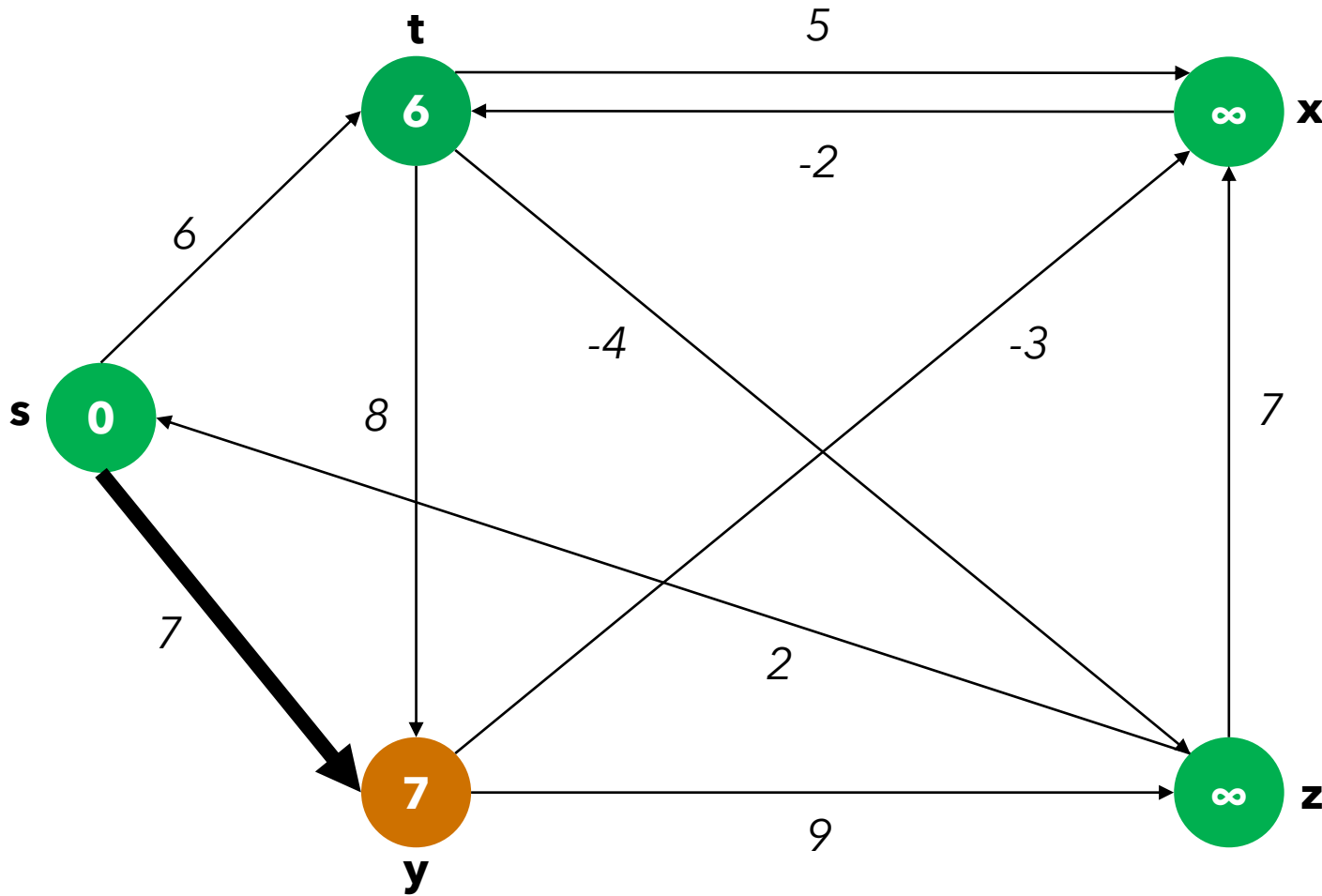


t → x
t → y
t → z
x → t
y → x
y → z
z → x
z → s
s → t
s → y

rilassamento di s → t

t.predecessor = s

Bellman-Ford – 1st pass

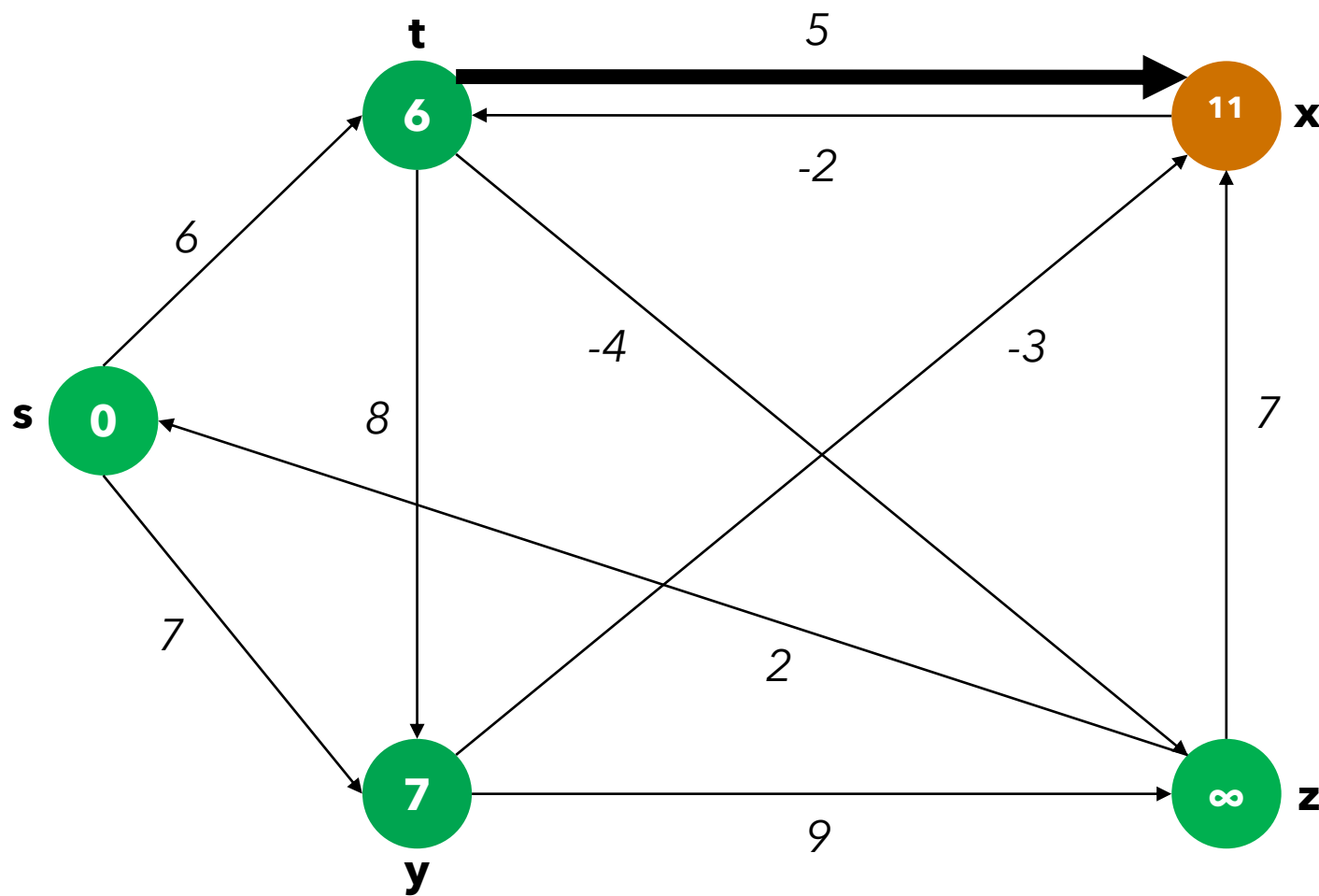


t → x
t → y
t → z
x → t
y → x
y → z
z → x
z → s
s → t
s → y

rilassamento di $s \rightarrow y$

y.predecessor = s

Bellman-Ford – 2nd pass



t → **x**

t → y

t → z

x → t

y → x

y → z

z → x

z → s

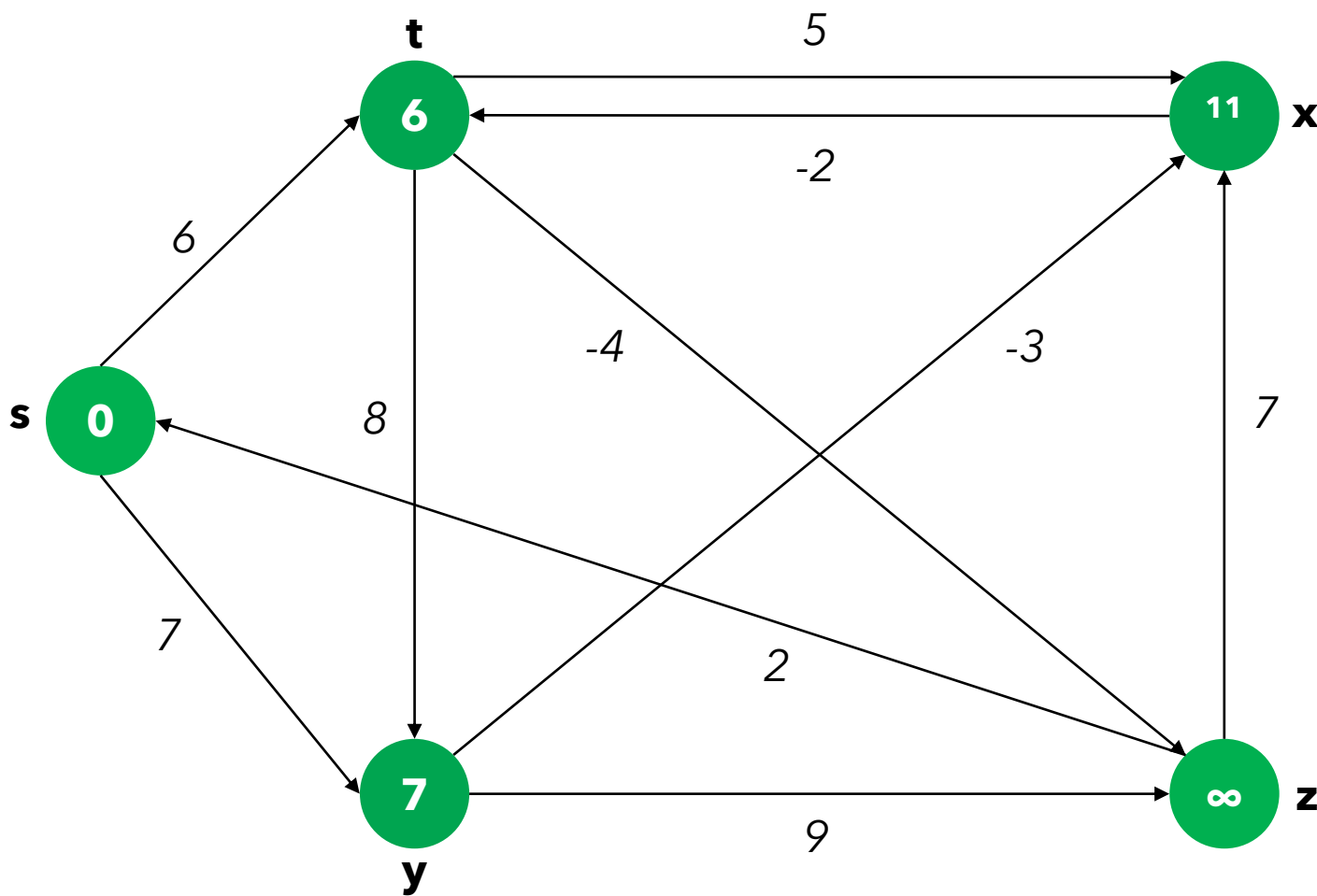
s → t

s → y

rilassamento di t → x

x.predecessor = t

Bellman-Ford – 2nd pass



t \rightarrow x

t \rightarrow y

t \rightarrow z

x \rightarrow t

y \rightarrow x

y \rightarrow z

z \rightarrow x

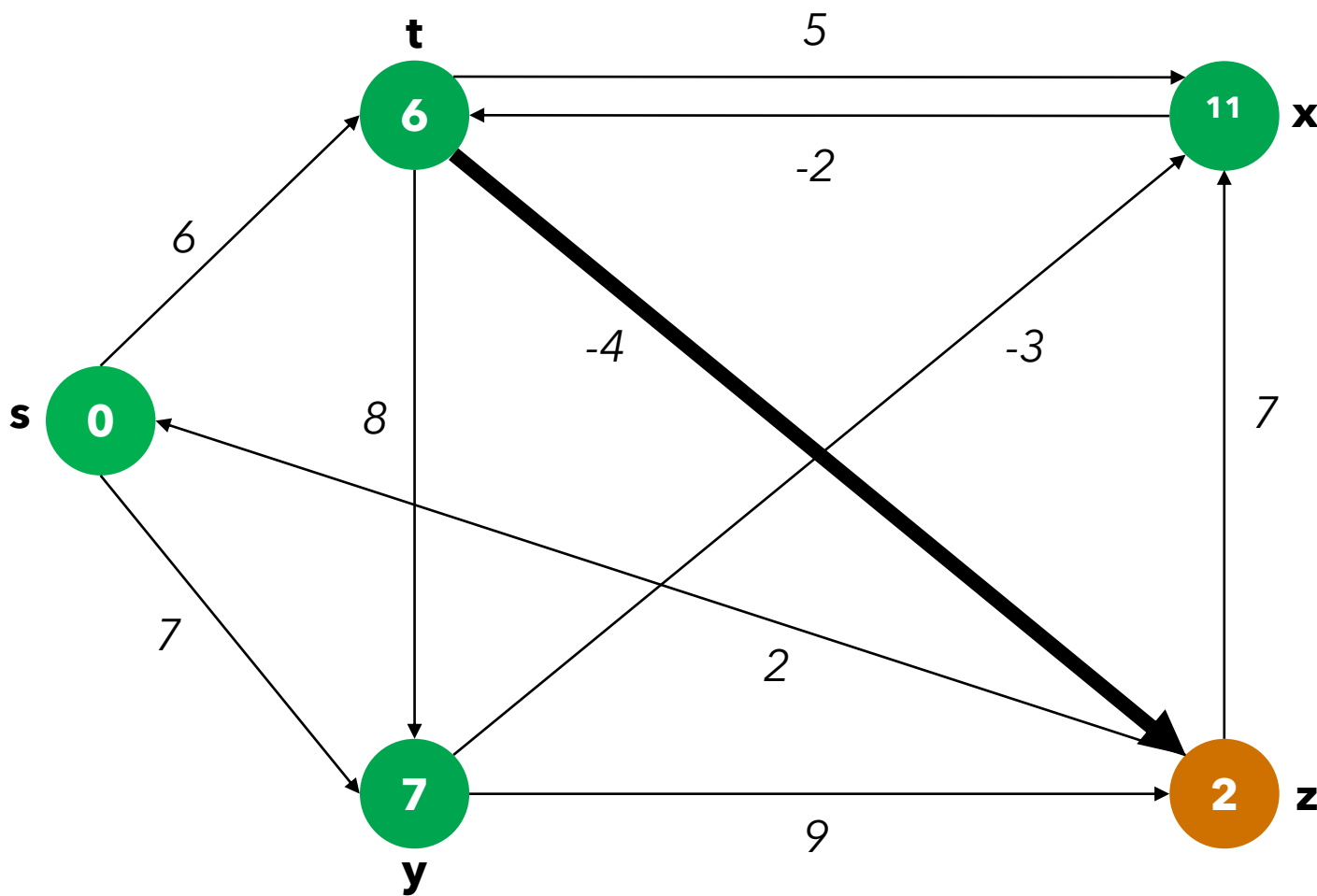
z \rightarrow s

s \rightarrow t

s \rightarrow y

**rilassamento di t \rightarrow y
(nessun effetto)**

Bellman-Ford – 2nd pass



t → x

t → y

t → z

x → t

y → x

y → z

z → x

z → s

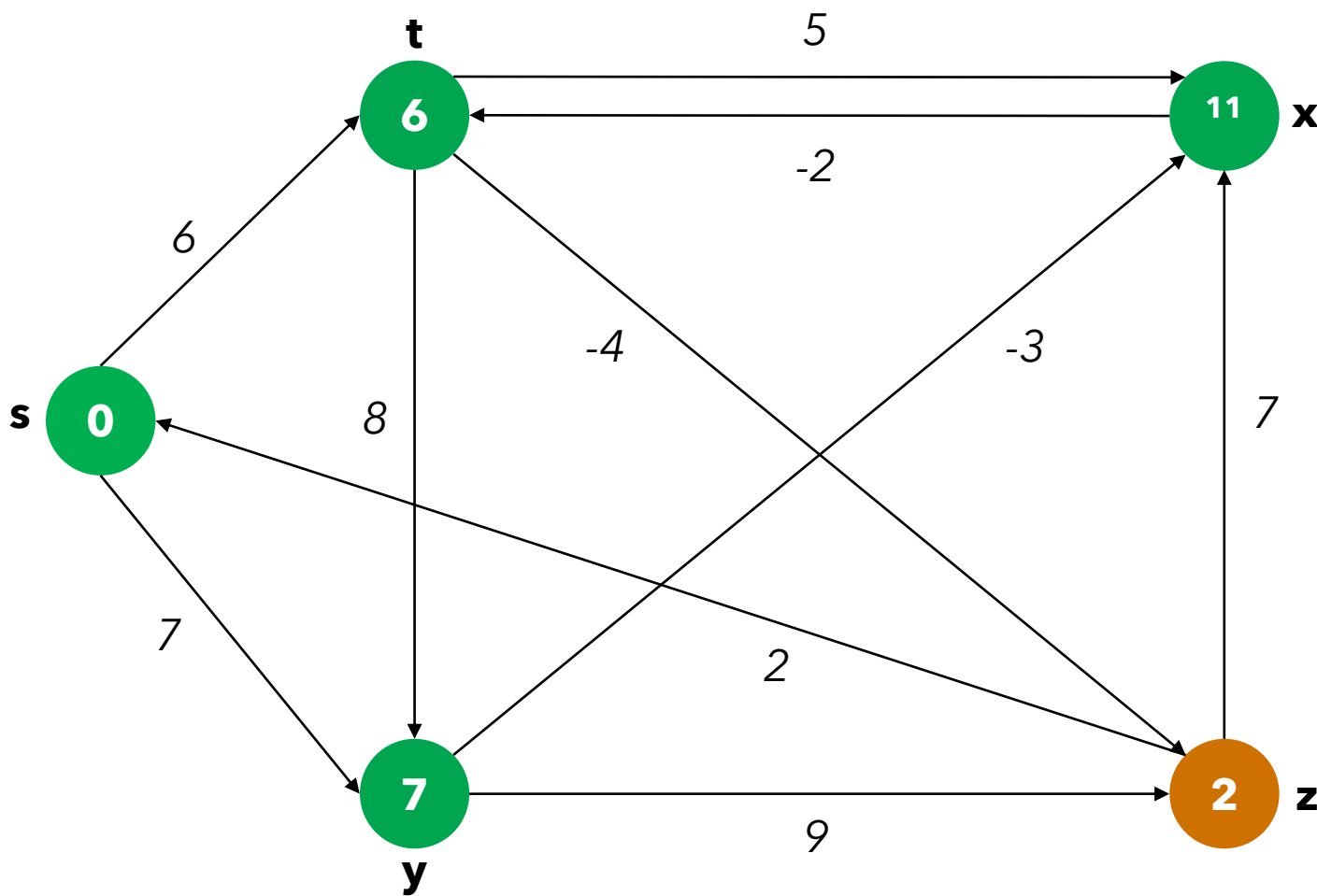
s → t

s → y

rilassamento di t → z

z.predecessor = t

Bellman-Ford – 2nd pass



t -> x

t -> y

t -> z

x -> t

y -> x

y -> z

z -> x

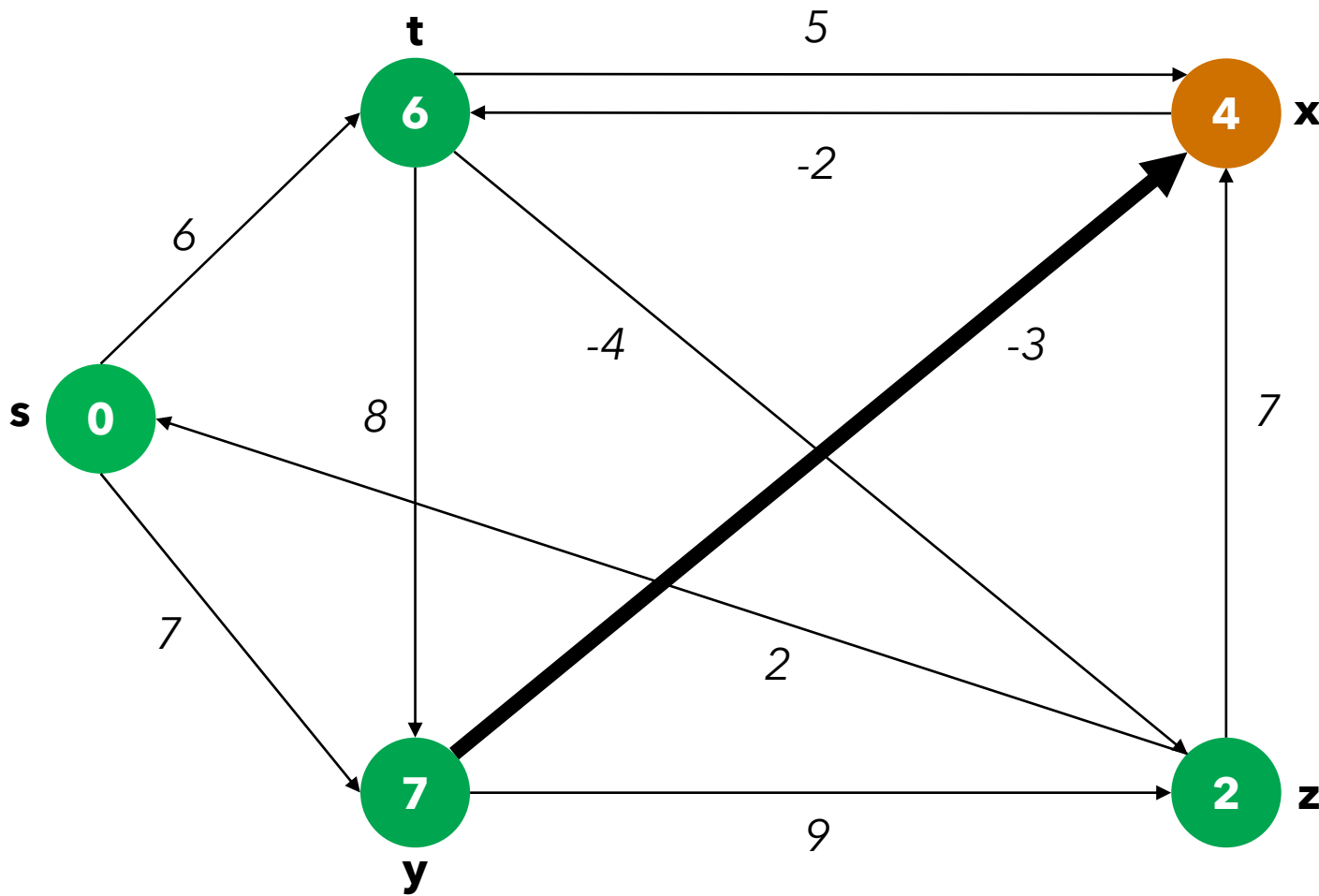
z -> s

s -> t

s -> y

**rilassamento di x -> t
(nessun effetto)**

Bellman-Ford – 2nd pass

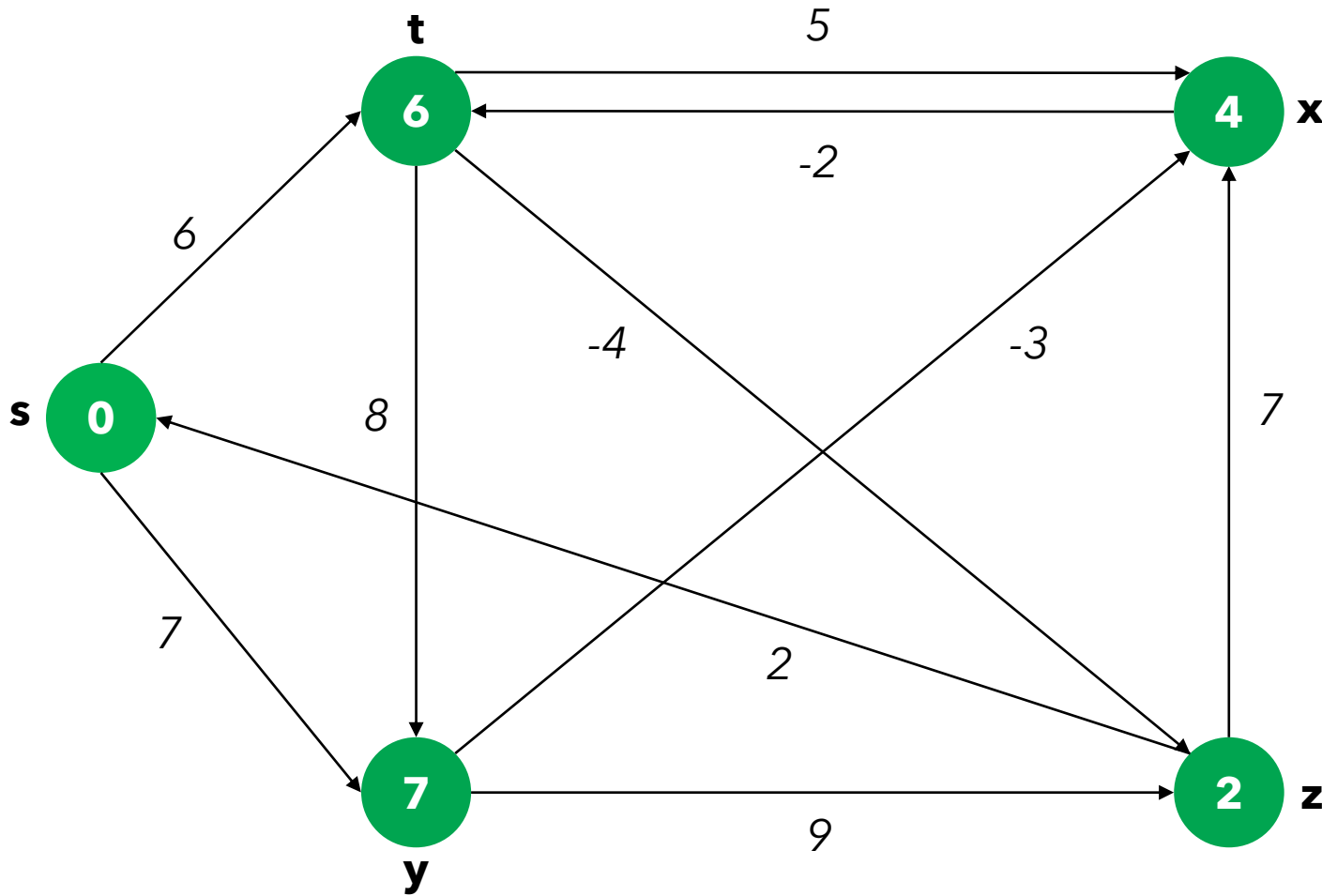


t → x
t → y
t → z
x → t
y → x
y → z
z → x
z → s
s → t
s → y

rilassamento di y → x

x.predecessor = y

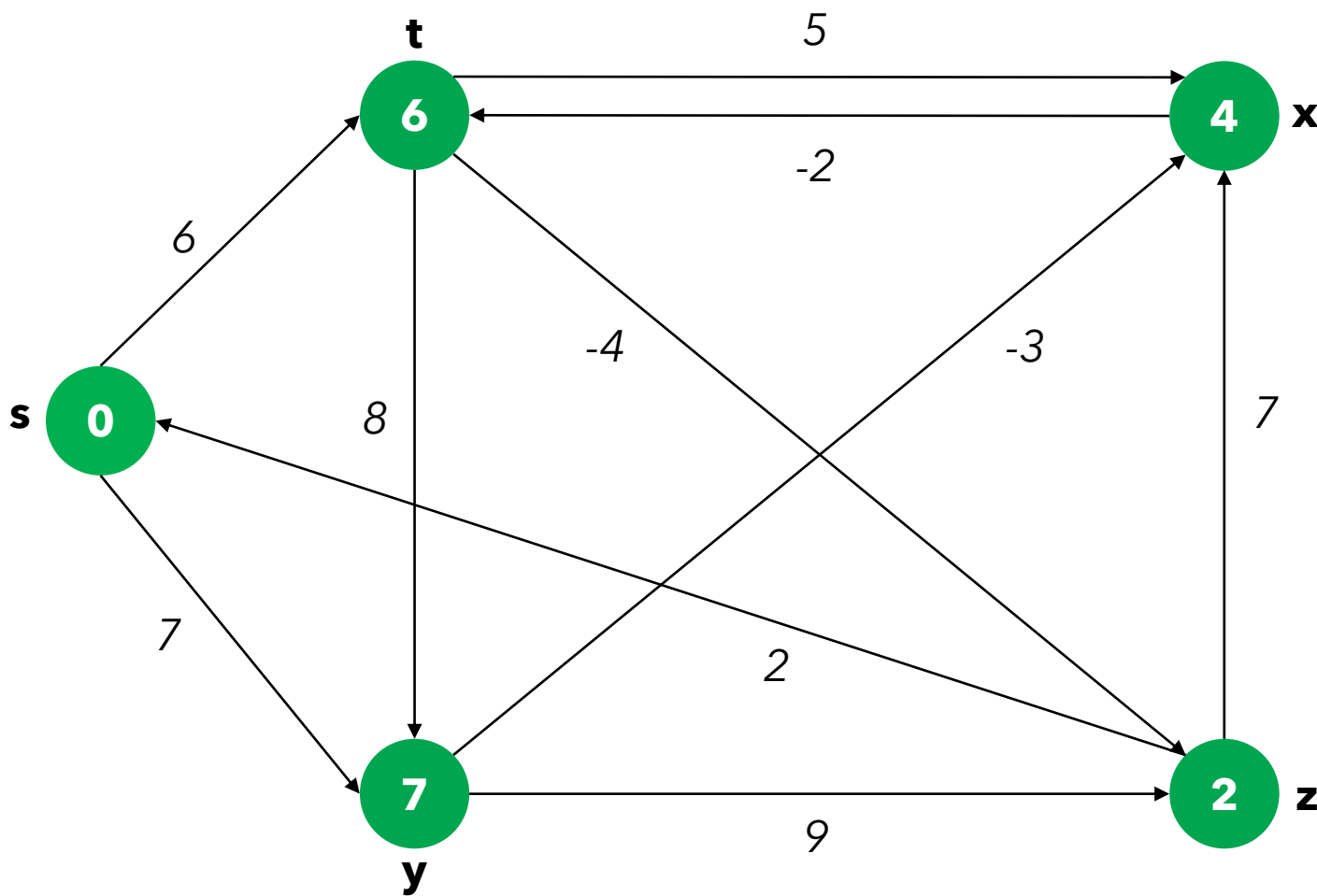
Bellman-Ford – 2nd pass



t -> x
t -> y
t -> z
x -> t
y -> x
y -> z
z -> x
z -> s
s -> t
s -> y

**rilassamento di y -> z
(nessun effetto)**

Bellman-Ford – 2nd pass



t \rightarrow x

t \rightarrow y

t \rightarrow z

x \rightarrow t

y \rightarrow x

y \rightarrow z

z \rightarrow x

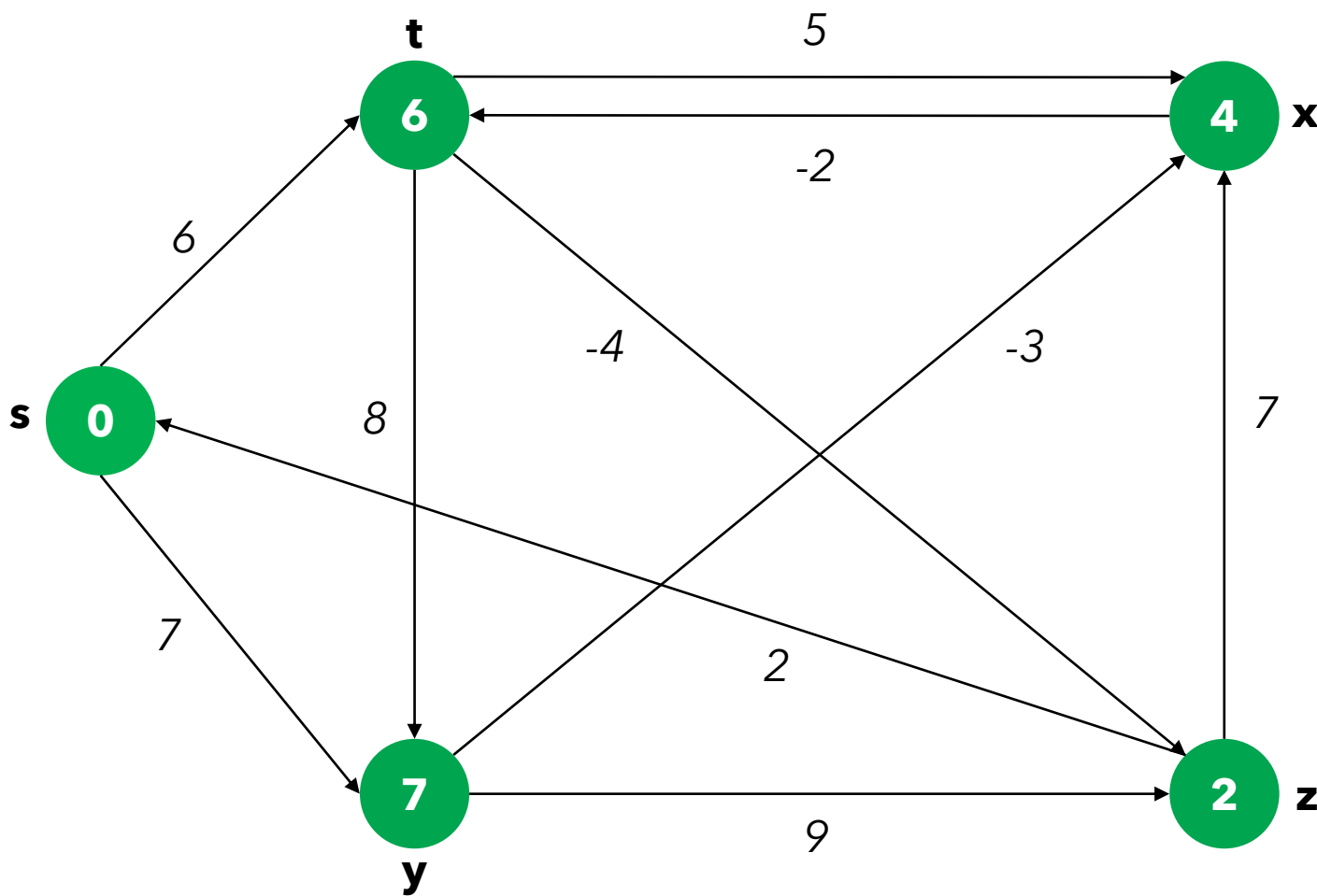
z \rightarrow s

s \rightarrow t

s \rightarrow y

**rilassamento di z \rightarrow x
(nessun effetto)**

Bellman-Ford – 2nd pass



t -> x

t -> y

t -> z

x -> t

y -> x

y -> z

z -> x

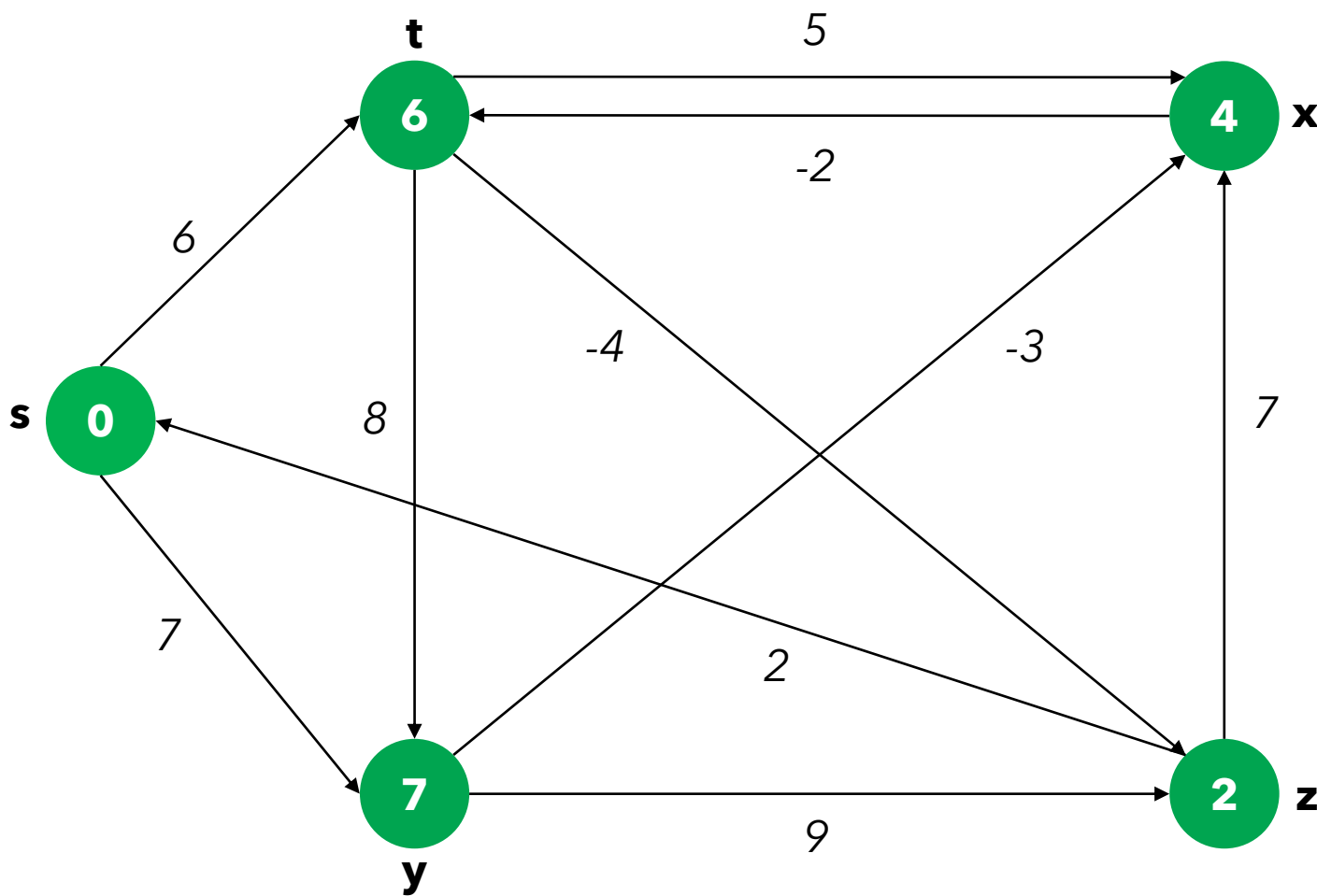
z -> s

s -> t

s -> y

**rilassamento di z -> s
(nessun effetto)**

Bellman-Ford – 2nd pass



t -> x

t -> y

t -> z

x -> t

y -> x

y -> z

z -> x

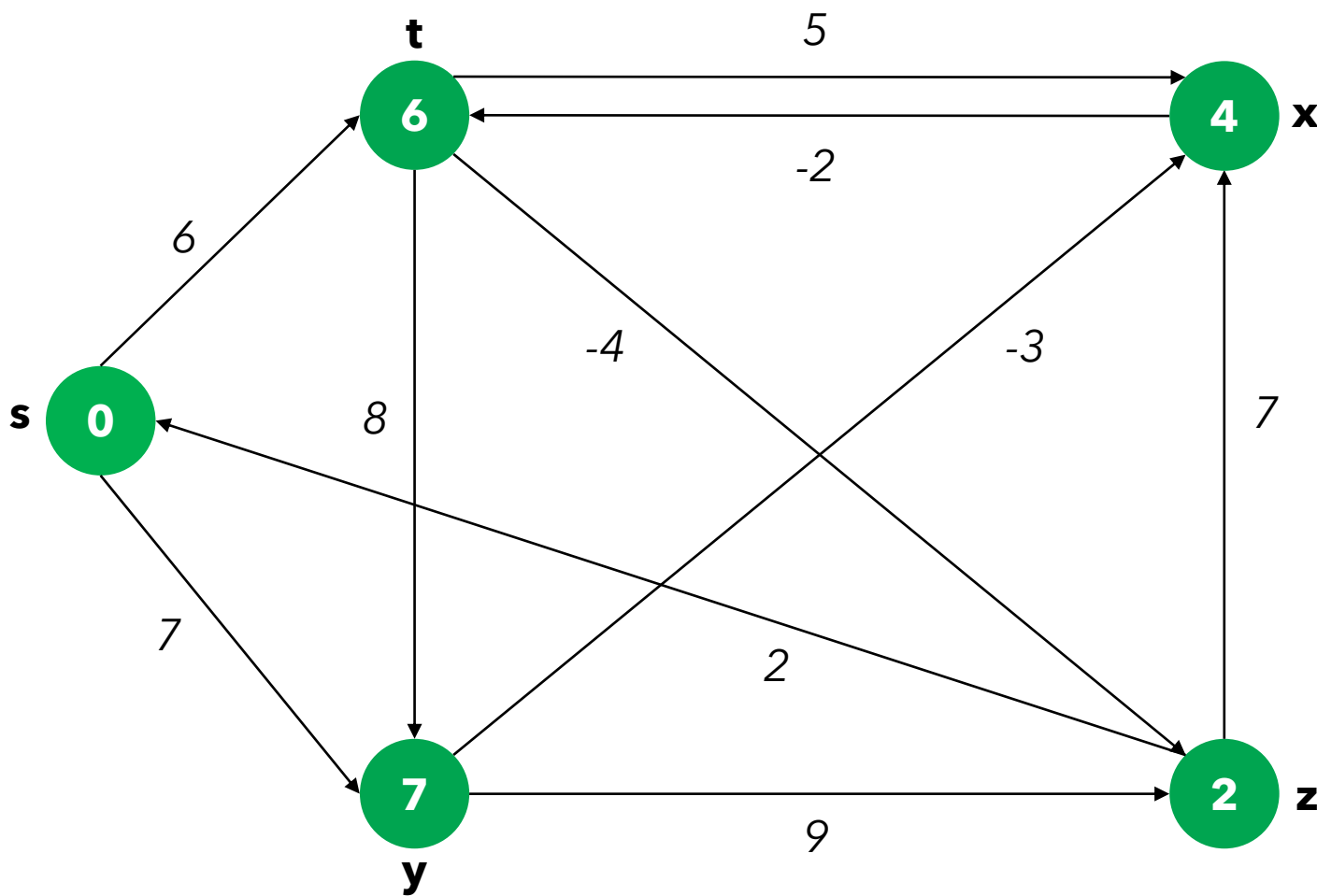
z -> s

s -> t

s -> y

**rilassamento di s -> t
(nessun effetto)**

Bellman-Ford – 2nd pass



t \rightarrow x

t \rightarrow y

t \rightarrow z

x \rightarrow t

y \rightarrow x

y \rightarrow z

z \rightarrow x

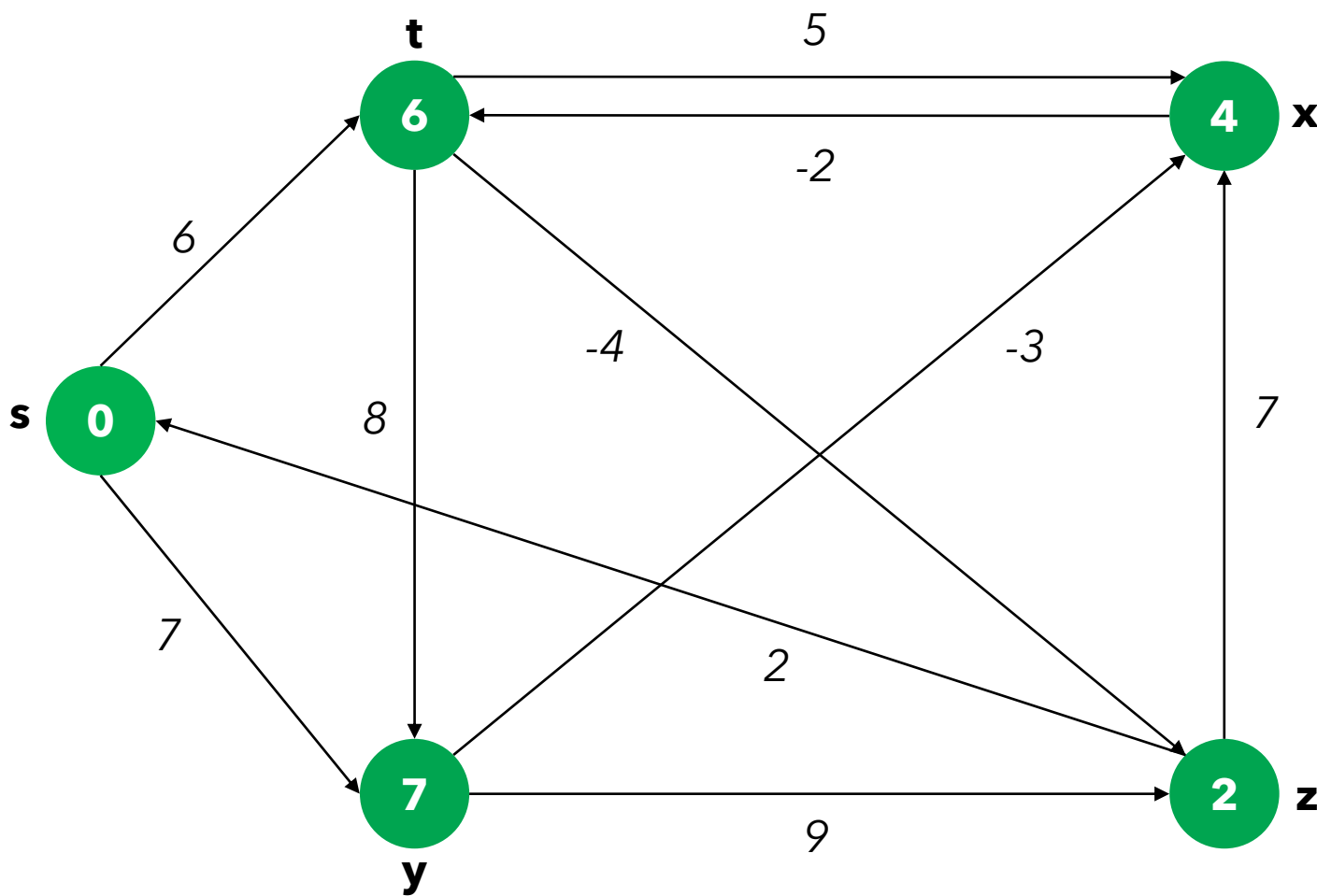
z \rightarrow s

s \rightarrow t

s \rightarrow y

**rilassamento di s \rightarrow y
(nessun effetto)**

Bellman-Ford – 3rd pass



t -> **x**

t -> y

t -> z

x -> t

y -> x

y -> z

z -> x

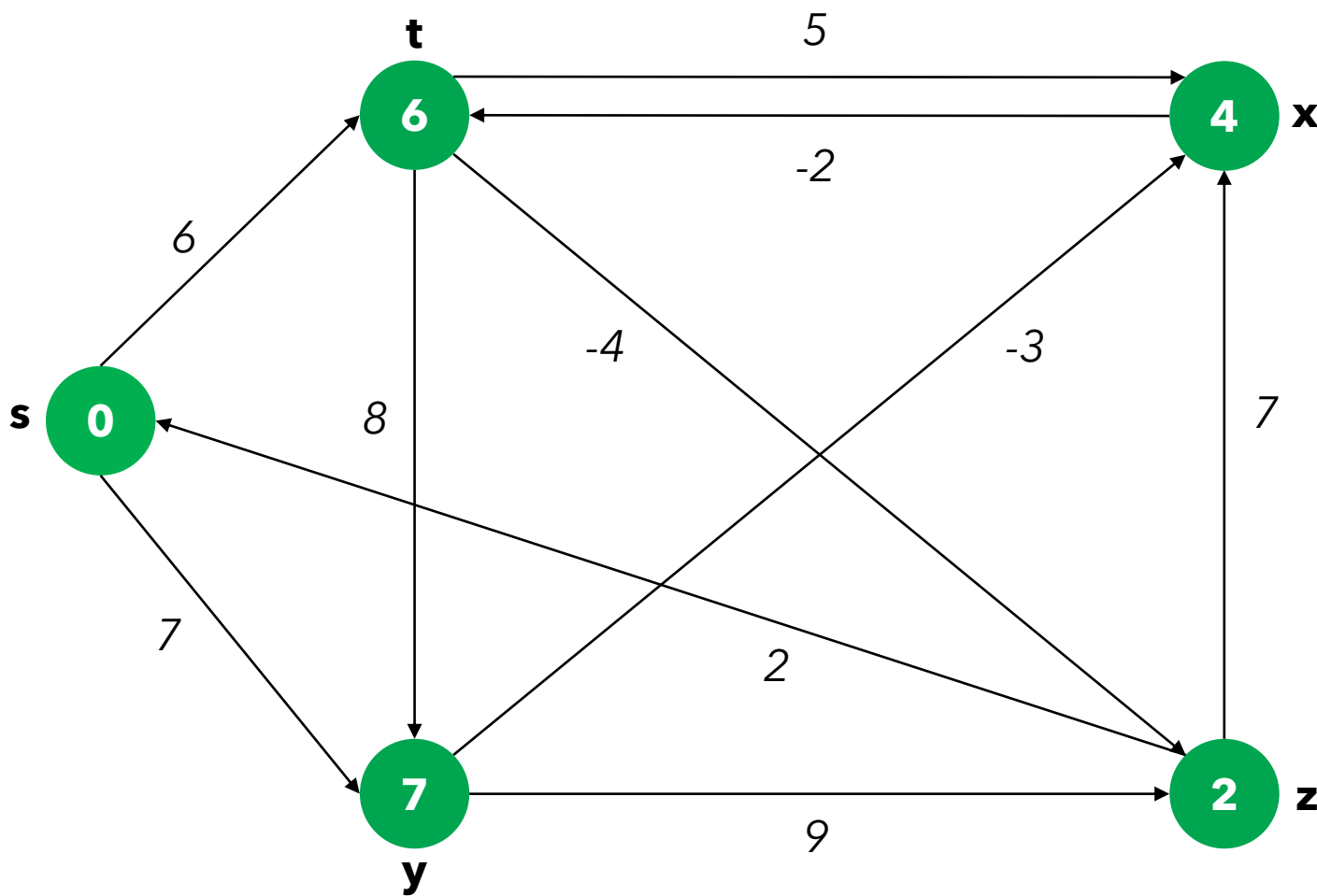
z -> s

s -> t

s -> y

rilassamento di t -> x
(nessun effetto)

Bellman-Ford – 3rd pass



t -> x

t -> y

t -> z

x -> t

y -> x

y -> z

z -> x

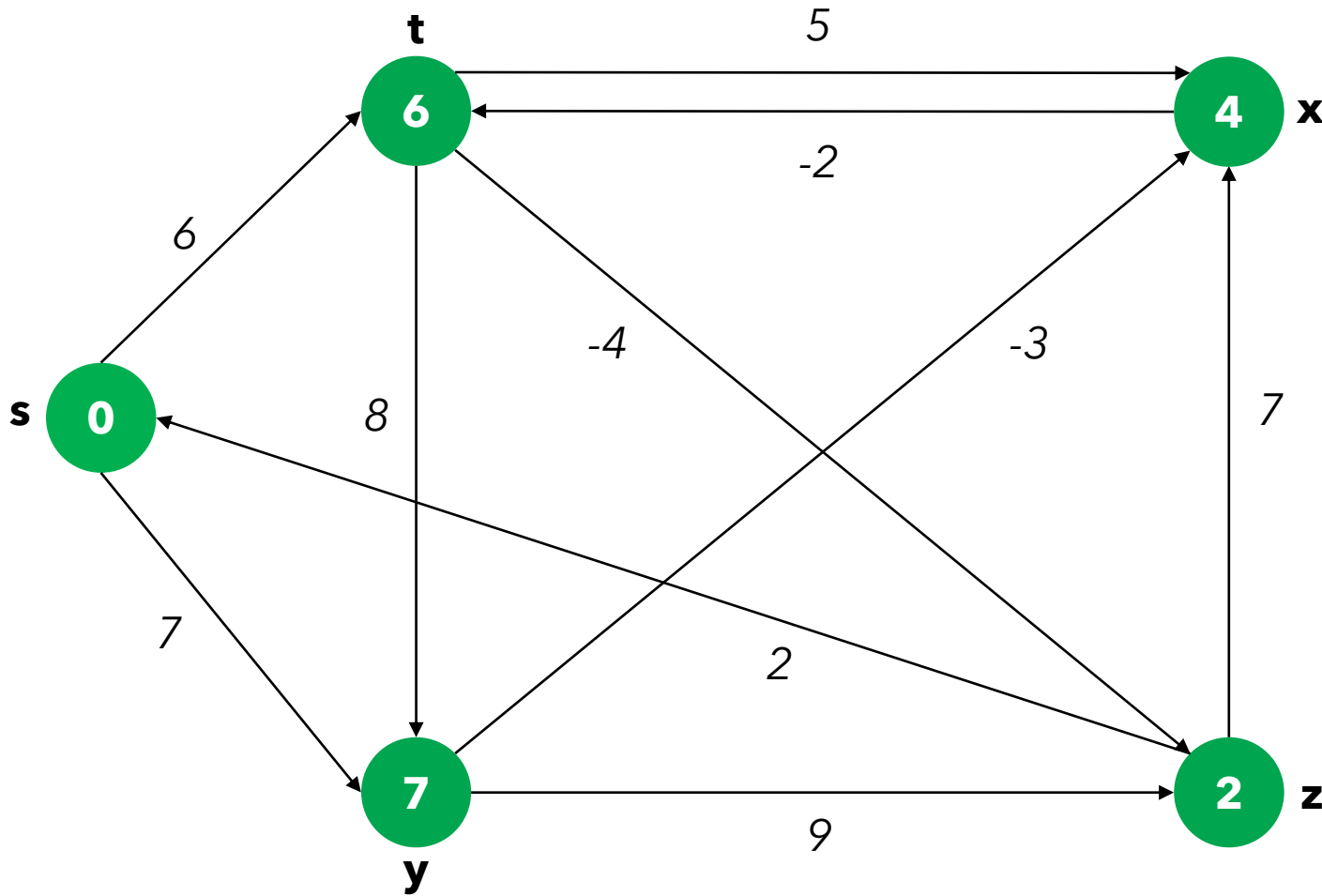
z -> s

s -> t

s -> y

**rilassamento di t -> y
(nessun effetto)**

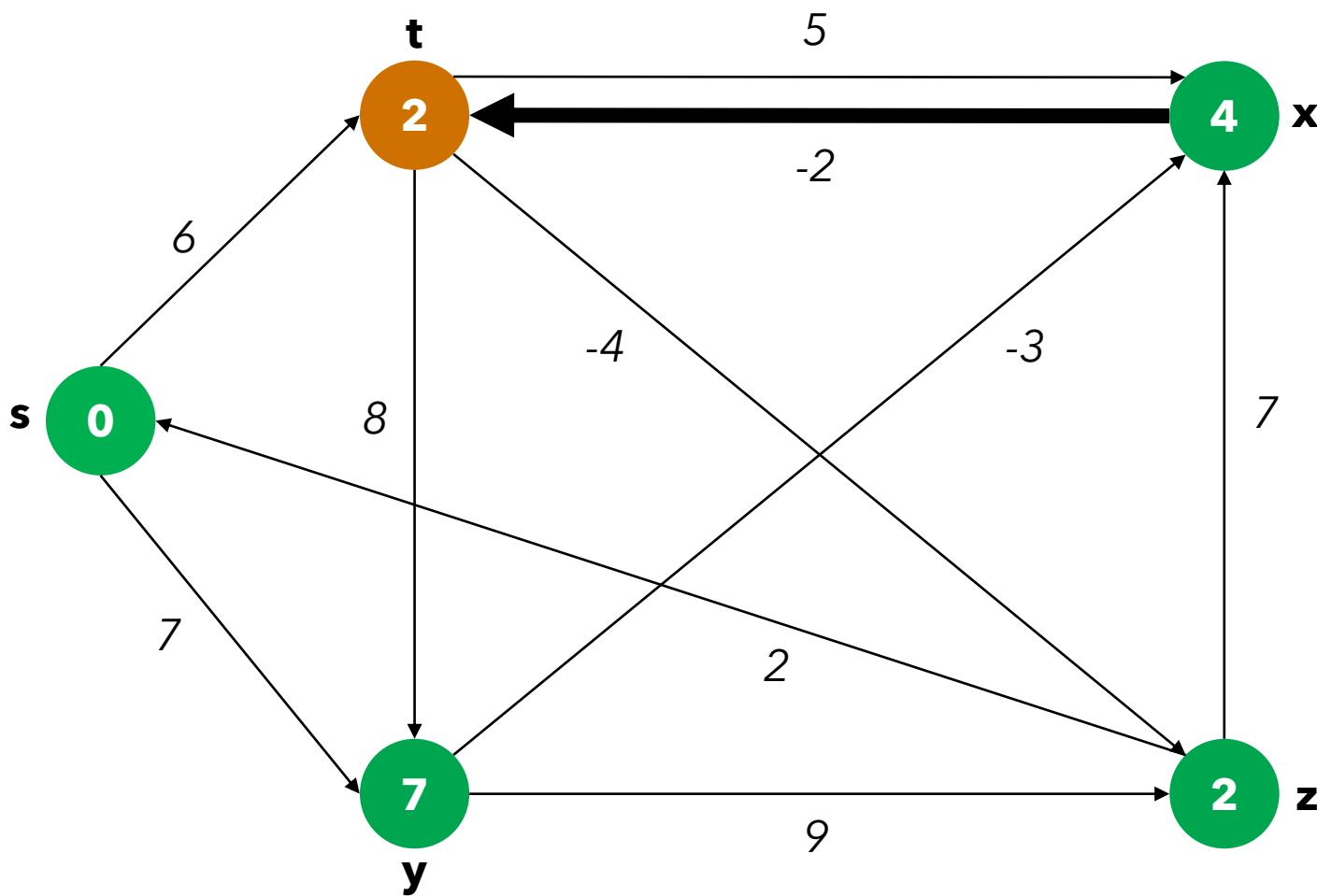
Bellman-Ford – 3rd pass



t -> x
t -> y
t -> z
x -> t
y -> x
y -> z
z -> x
z -> s
s -> t
s -> y

**rilassamento di t -> z
(nessun effetto)**

Bellman-Ford – 3rd pass



t → x

t → y

t → z

x → t

y → x

y → z

z → x

z → s

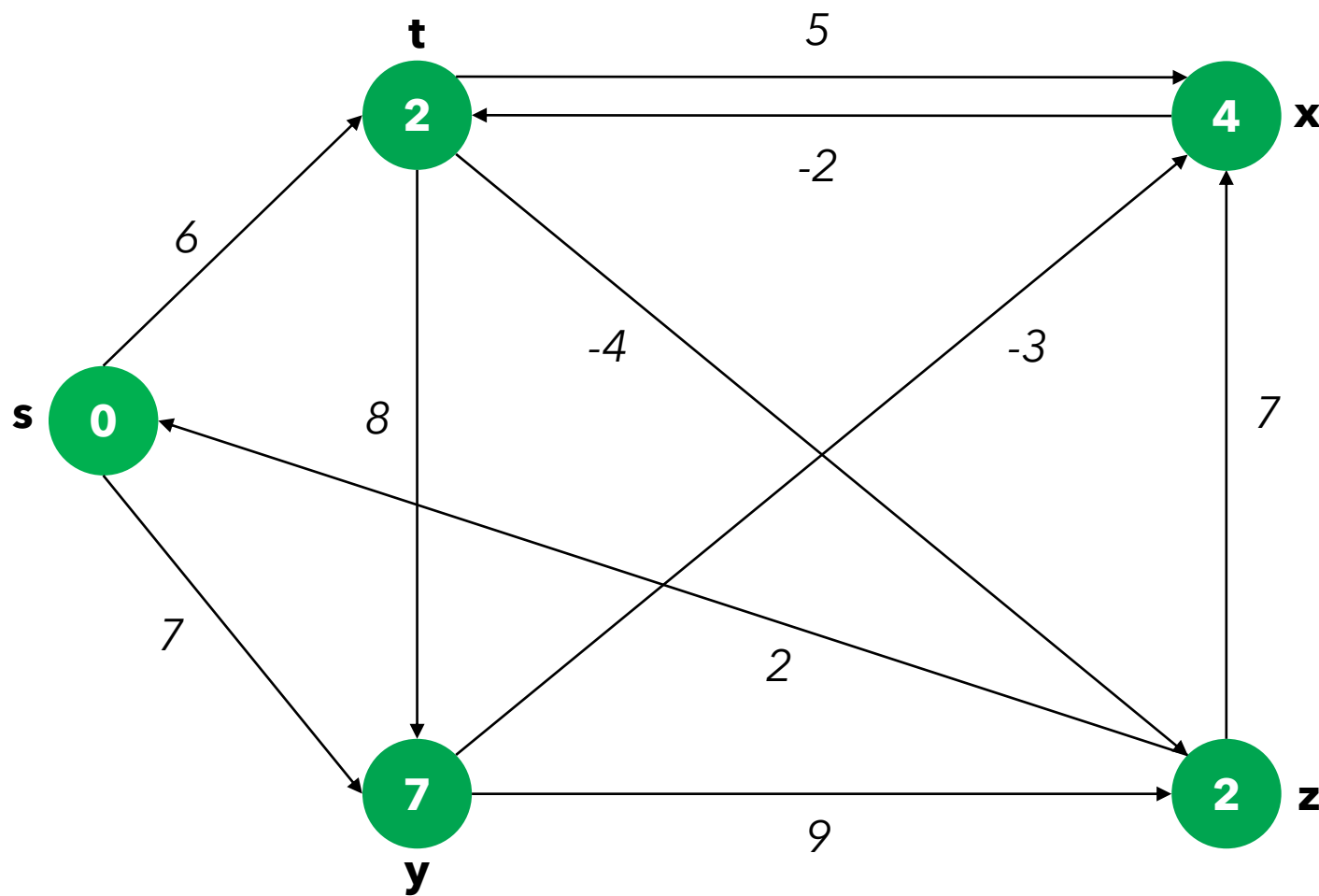
s → t

s → y

rilassamento di t → z

t.predecessor = x

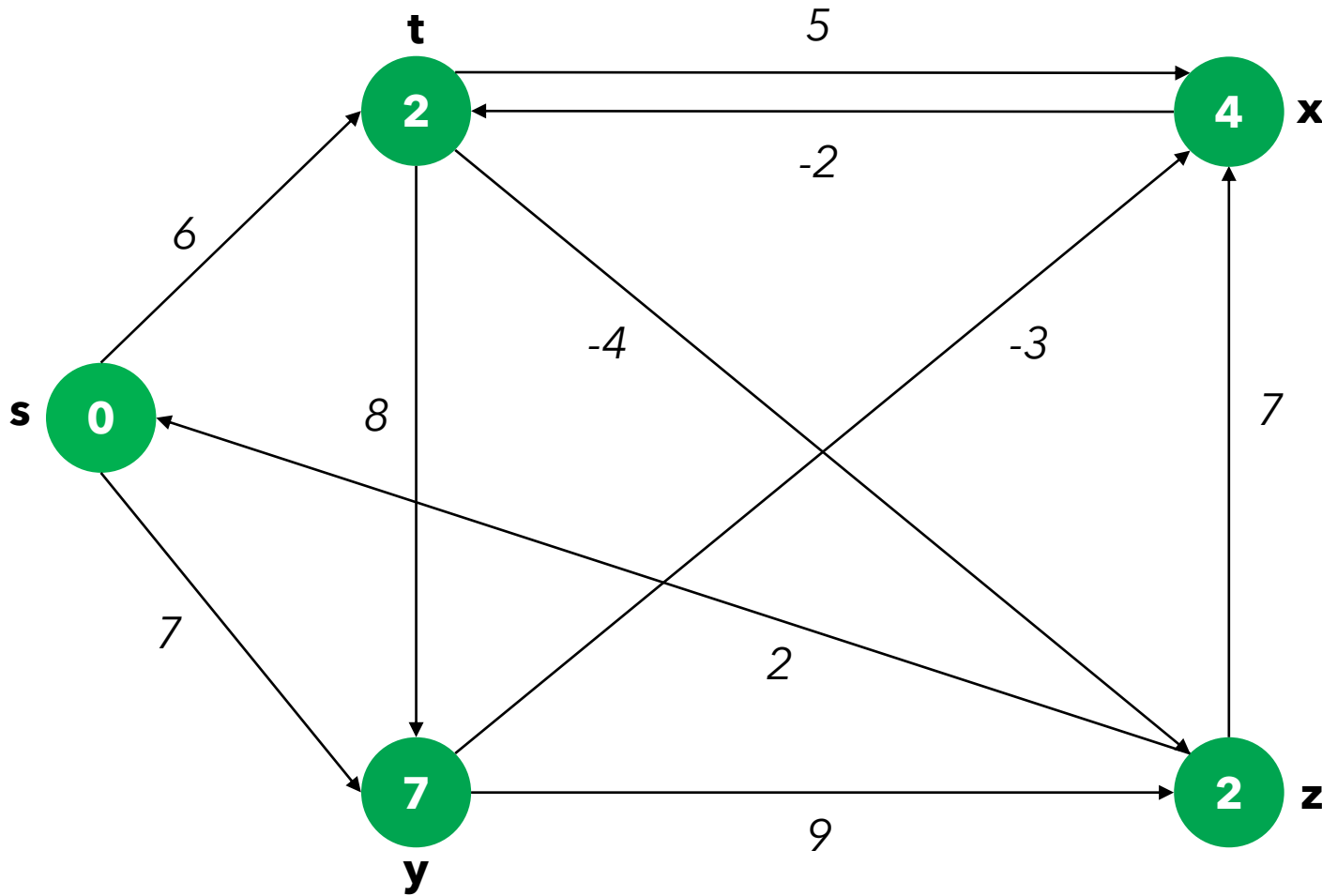
Bellman-Ford – 3rd pass



t -> x
t -> y
t -> z
x -> t
y -> x
y -> z
z -> x
z -> s
s -> t
s -> y

**rilassamento di y -> x
(nessun effetto)**

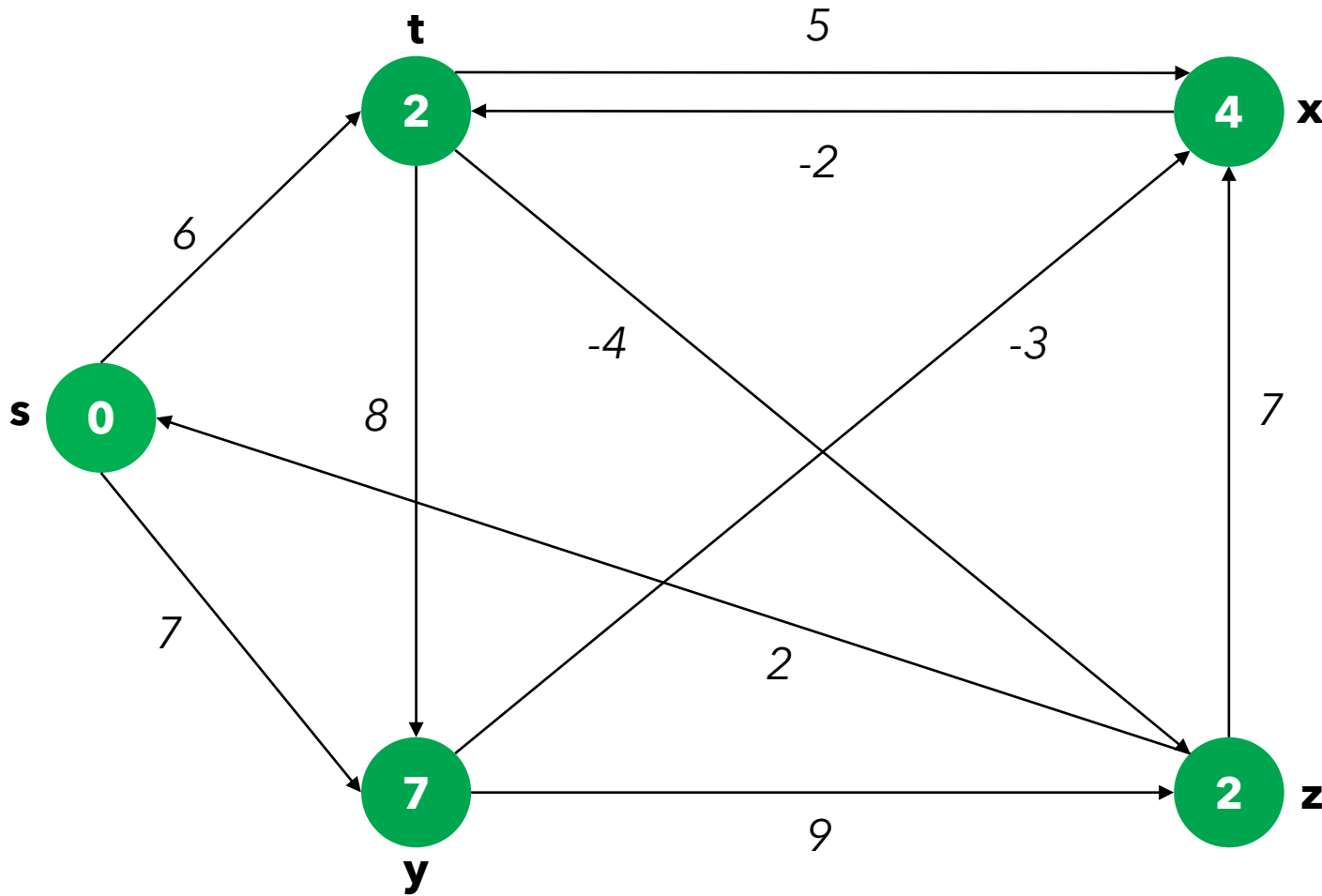
Bellman-Ford – 3rd pass



t -> x
t -> y
t -> z
x -> t
y -> x
y -> z
z -> x
z -> s
s -> t
s -> y

**rilassamento di y -> z
(nessun effetto)**

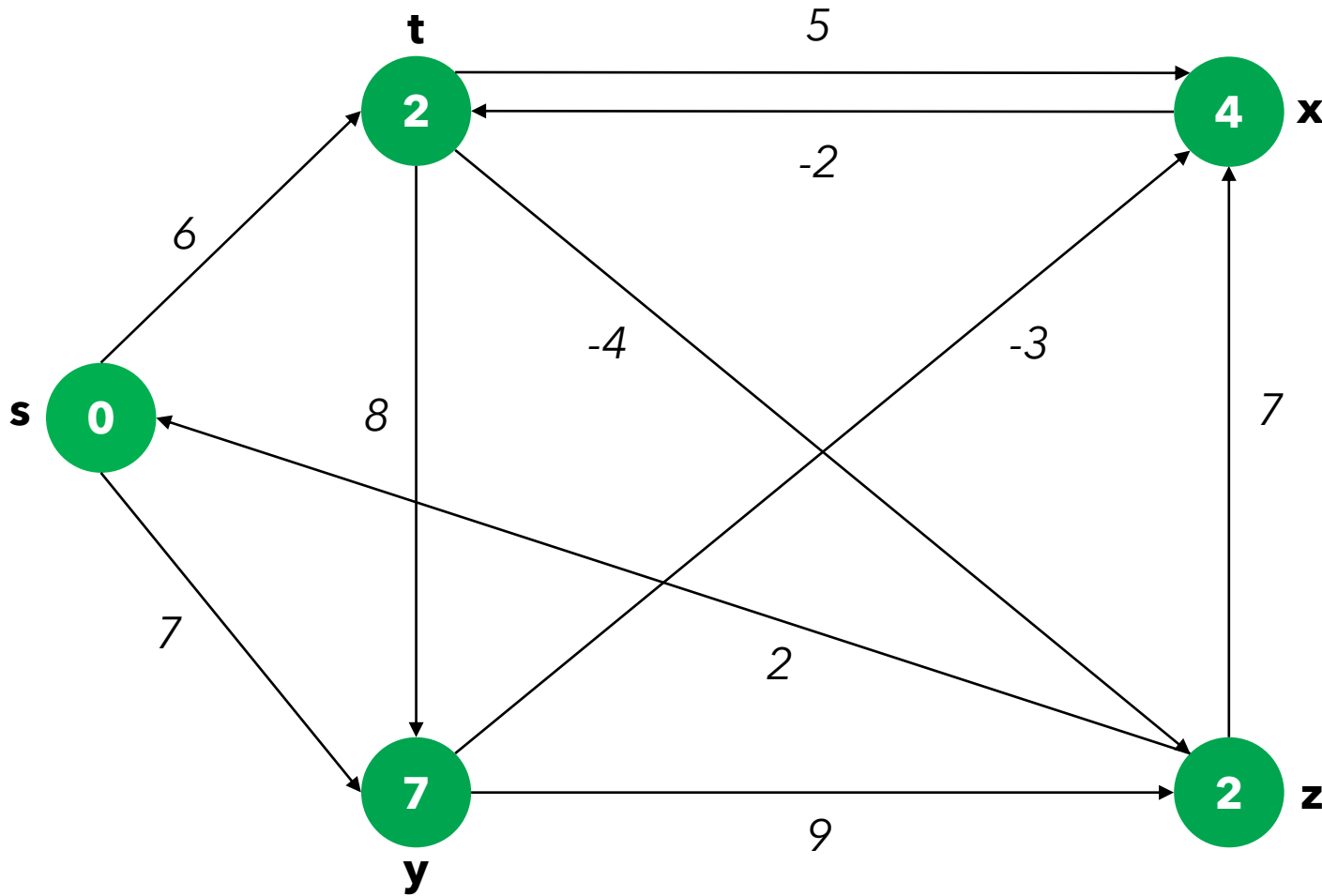
Bellman-Ford – 3rd pass



t -> x
t -> y
t -> z
x -> t
y -> x
y -> z
z -> x
z -> s
s -> t
s -> y

**rilassamento di z -> x
(nessun effetto)**

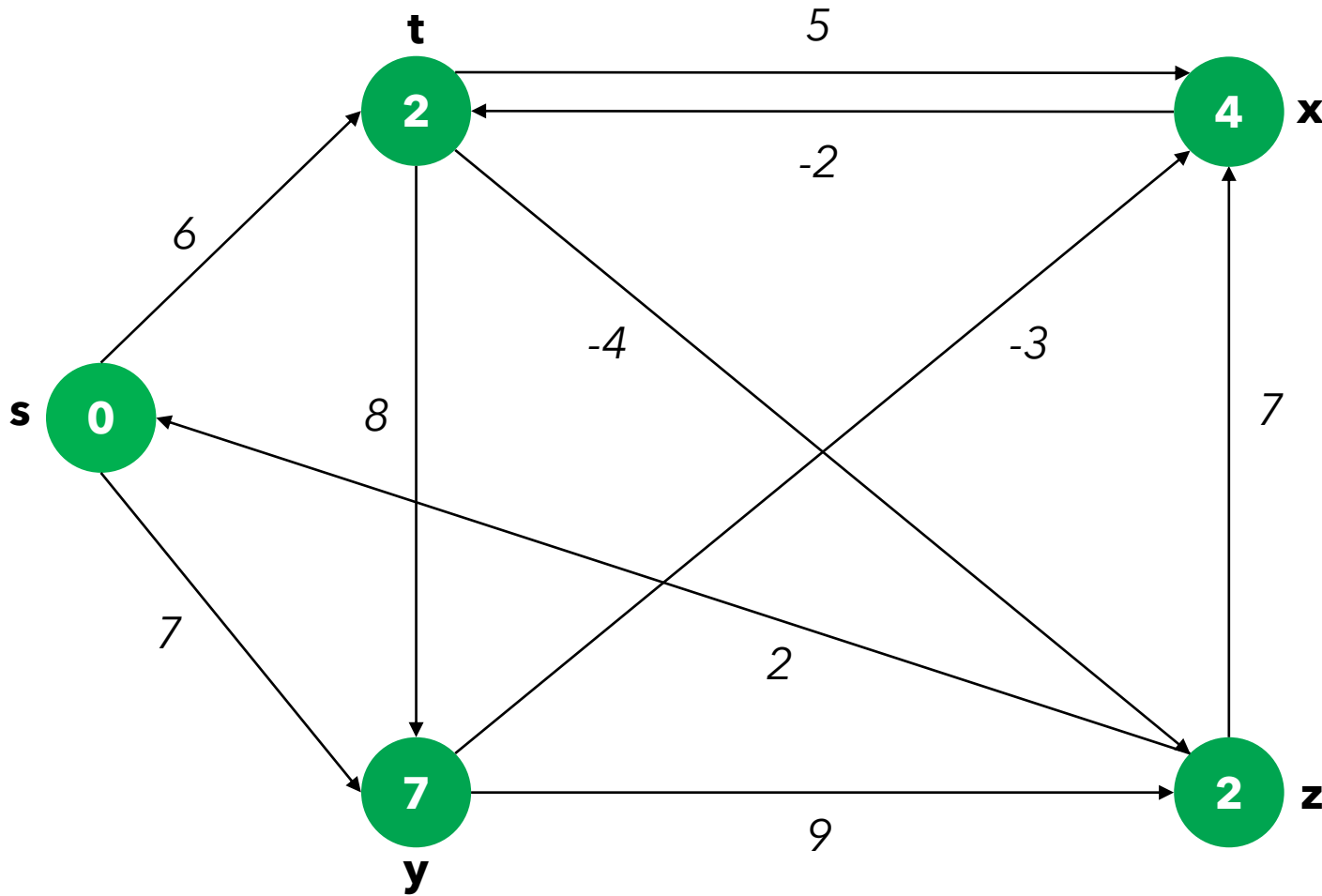
Bellman-Ford – 3rd pass



t → x
t → y
t → z
x → t
y → x
y → z
z → x
z → s
s → t
s → y

**rilassamento di z → s
(nessun effetto)**

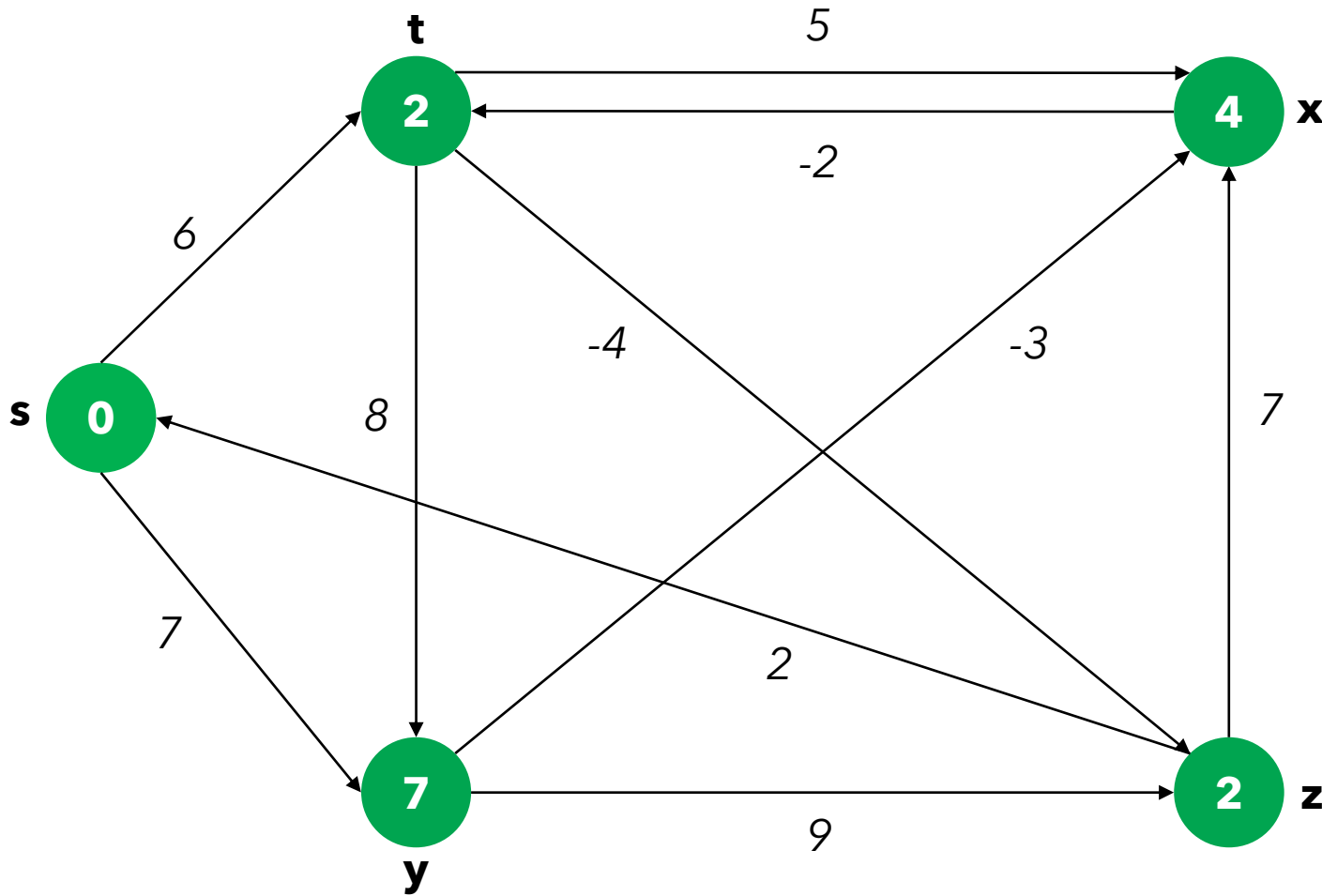
Bellman-Ford – 3rd pass



t -> x
t -> y
t -> z
x -> t
y -> x
y -> z
z -> x
z -> s
s -> t
s -> y

**rilassamento di s -> t
(nessun effetto)**

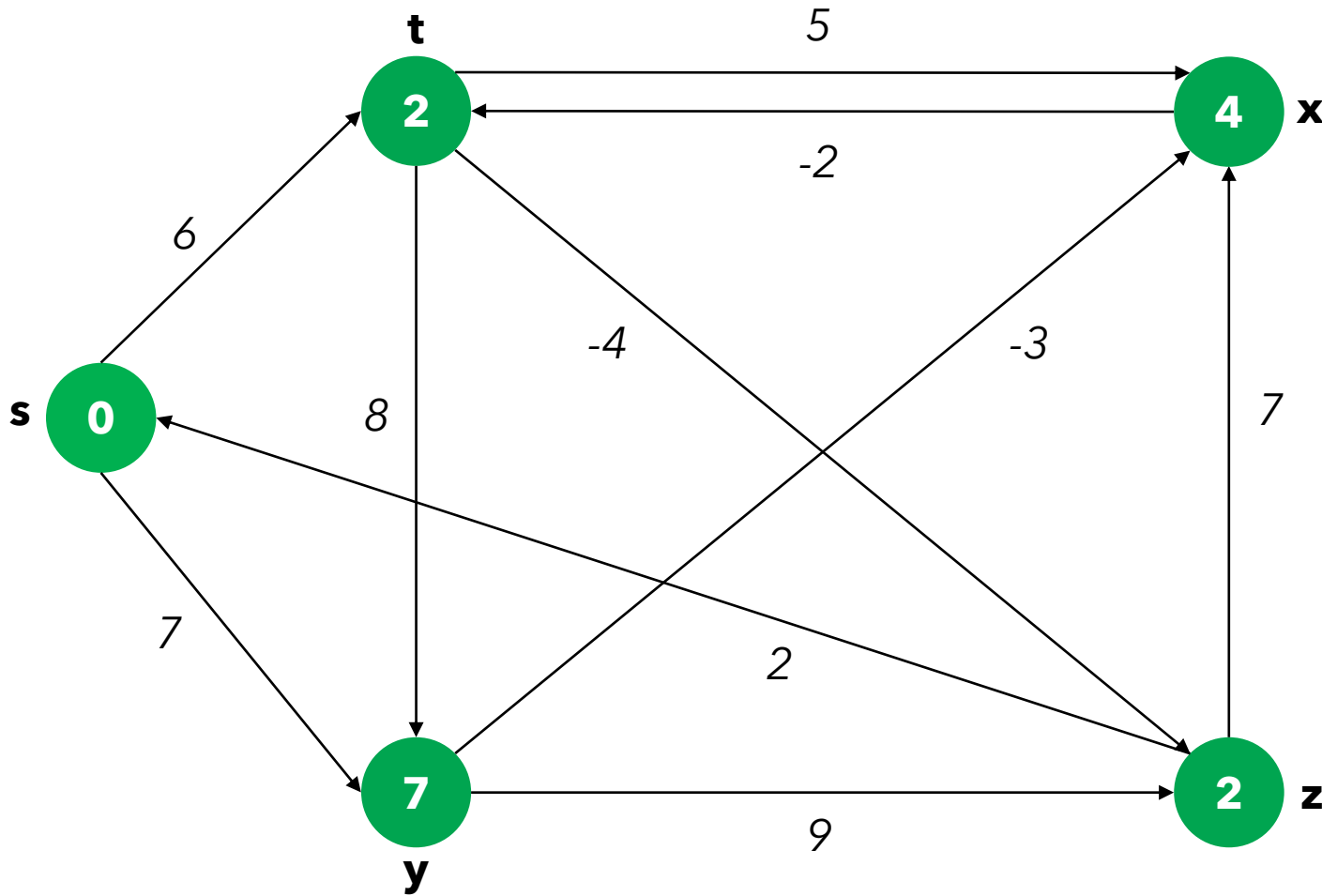
Bellman-Ford – 3rd pass



t → x
t → y
t → z
x → t
y → x
y → z
z → x
z → s
s → t
s → y

**rilassamento di s → y
(nessun effetto)**

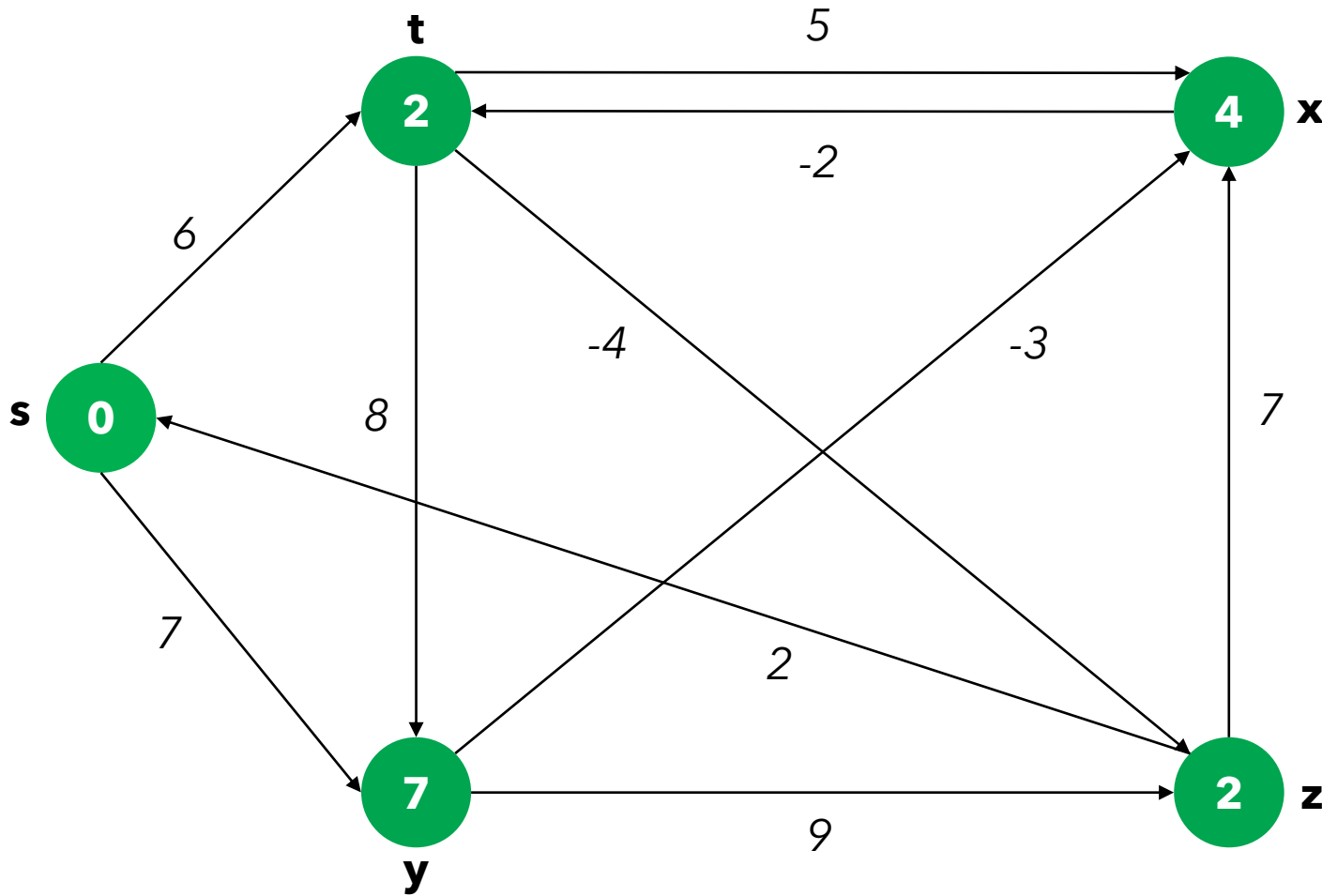
Bellman-Ford – 4th pass



t -> **x**
t -> y
t -> z
x -> t
y -> x
y -> z
z -> x
z -> s
s -> t
s -> y

**rilassamento di t -> x
(nessun effetto)**

Bellman-Ford – 4th pass



t -> x

t -> y

t -> z

x -> t

y -> x

y -> z

z -> x

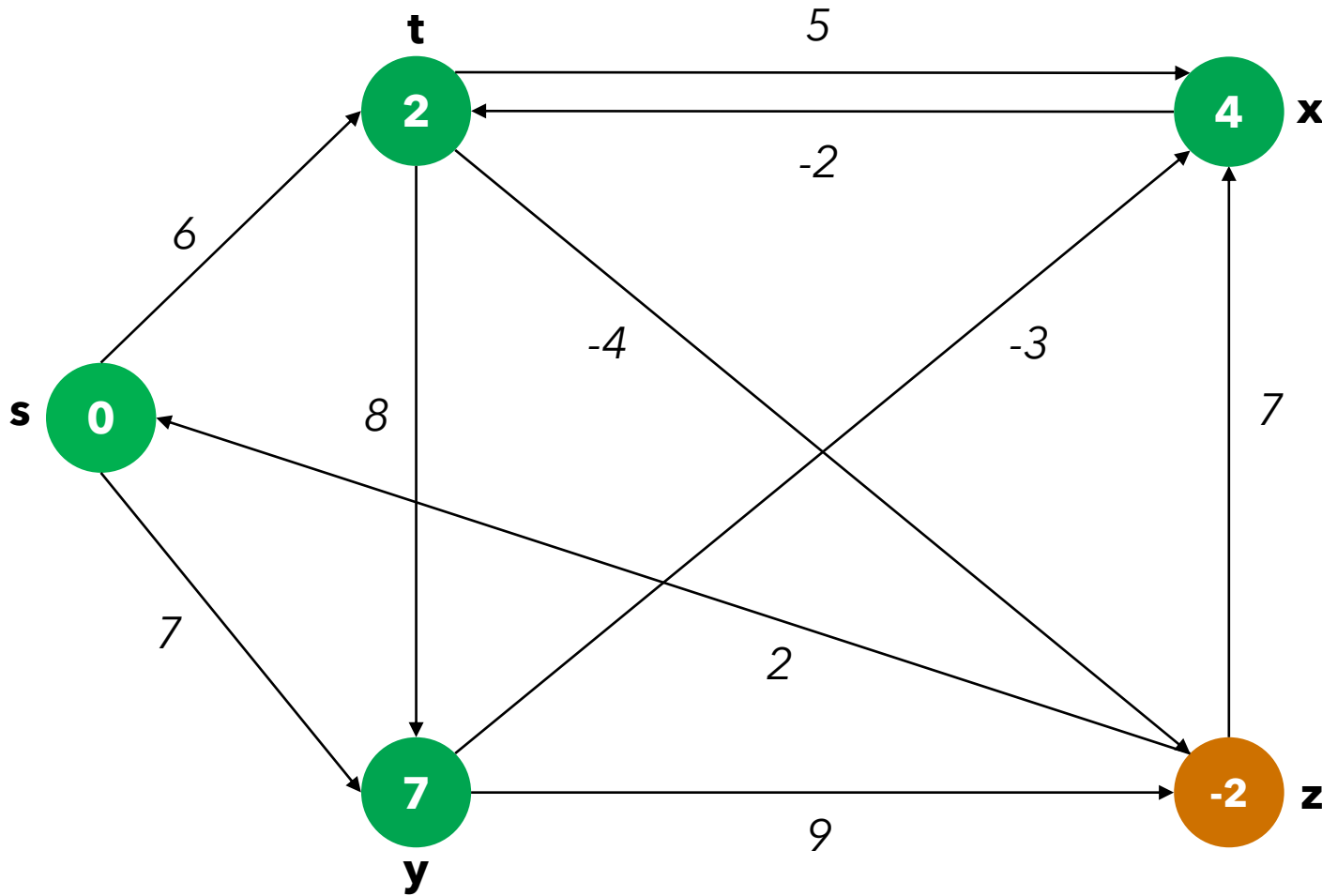
z -> s

s -> t

s -> y

**rilassamento di t -> y
(nessun effetto)**

Bellman-Ford – 4th pass

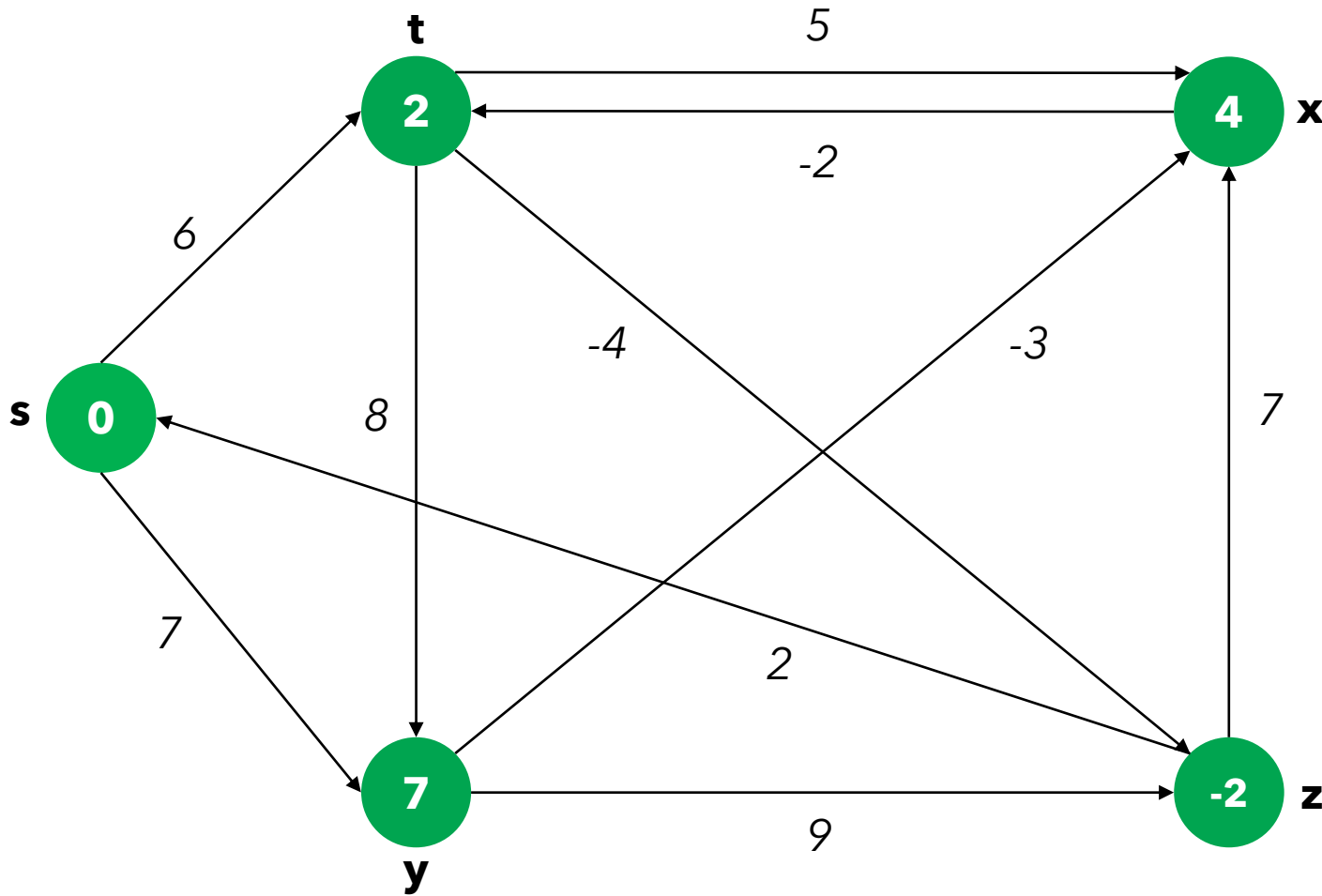


t -> x
t -> y
t -> z
x -> t
y -> x
y -> z
z -> x
z -> s
s -> t
s -> y

rilassamento di t -> z

z.predecessor = t

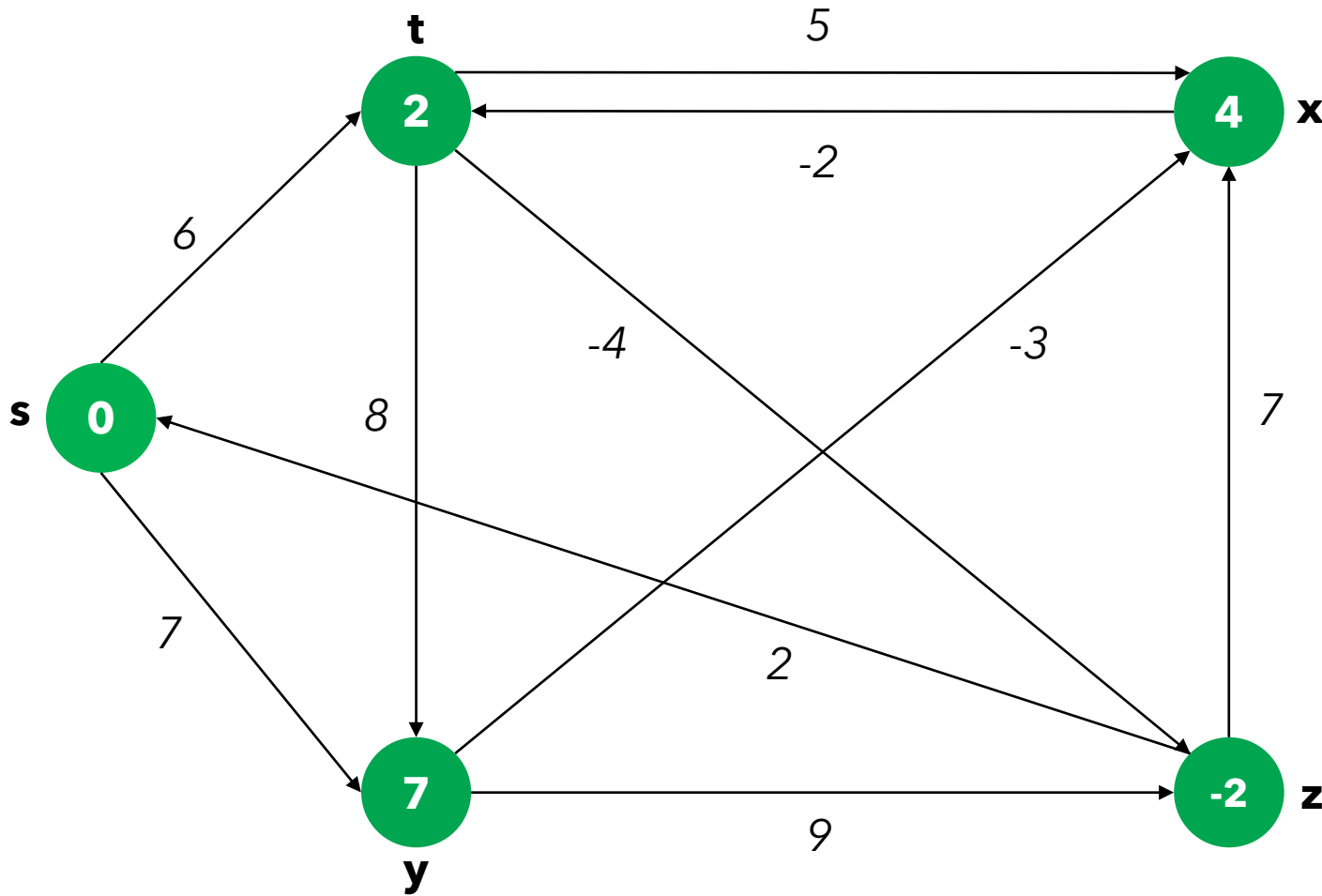
Bellman-Ford – 4th pass



t -> x
t -> y
t -> z
x -> t
y -> x
y -> z
z -> x
z -> s
s -> t
s -> y

**rilassamento di x -> t
(nessun effetto)**

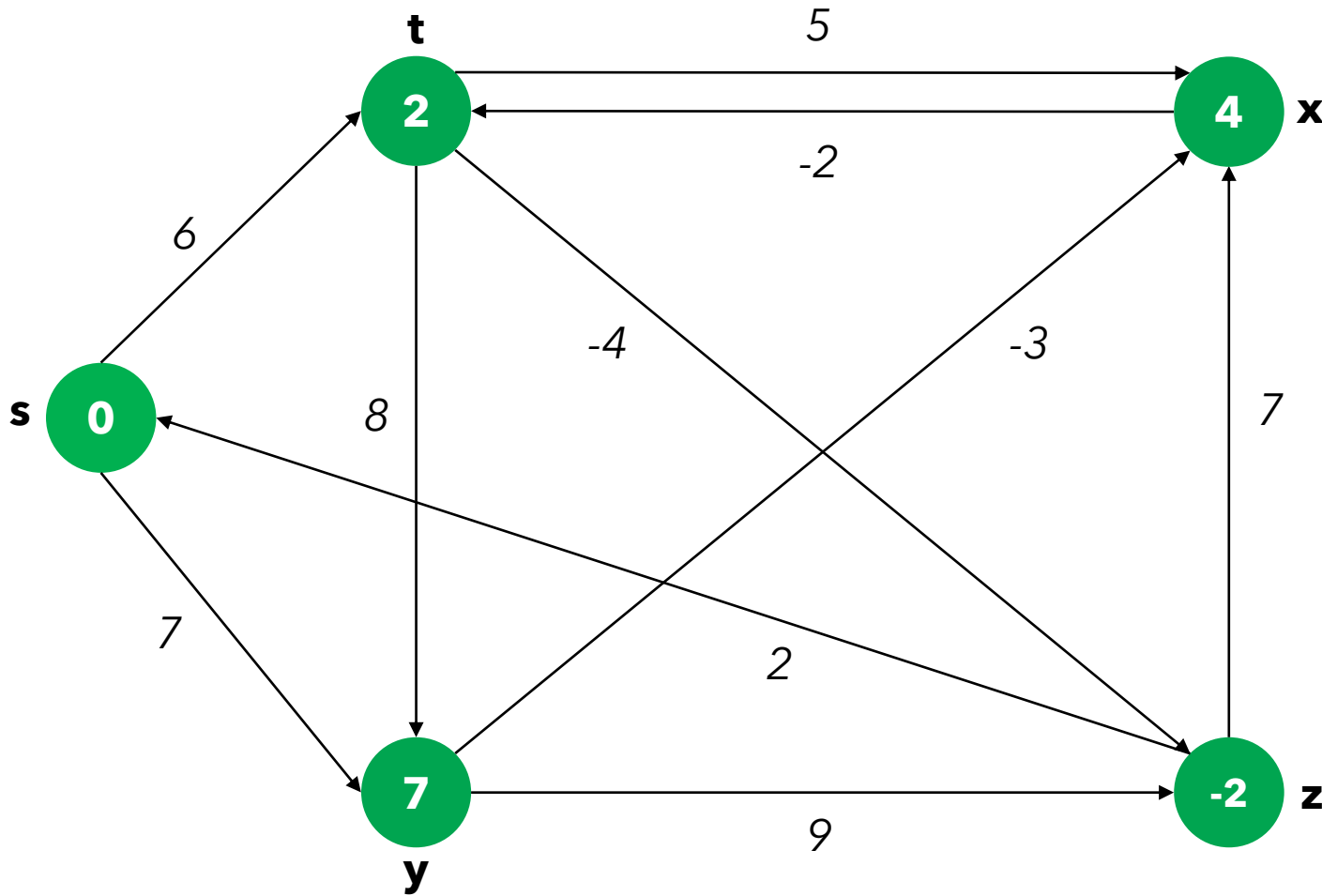
Bellman-Ford – 4th pass



t \rightarrow x
t \rightarrow y
t \rightarrow z
x \rightarrow t
y \rightarrow x
y \rightarrow z
z \rightarrow x
z \rightarrow s
s \rightarrow t
s \rightarrow y

**rilassamento di y \rightarrow x
(nessun effetto)**

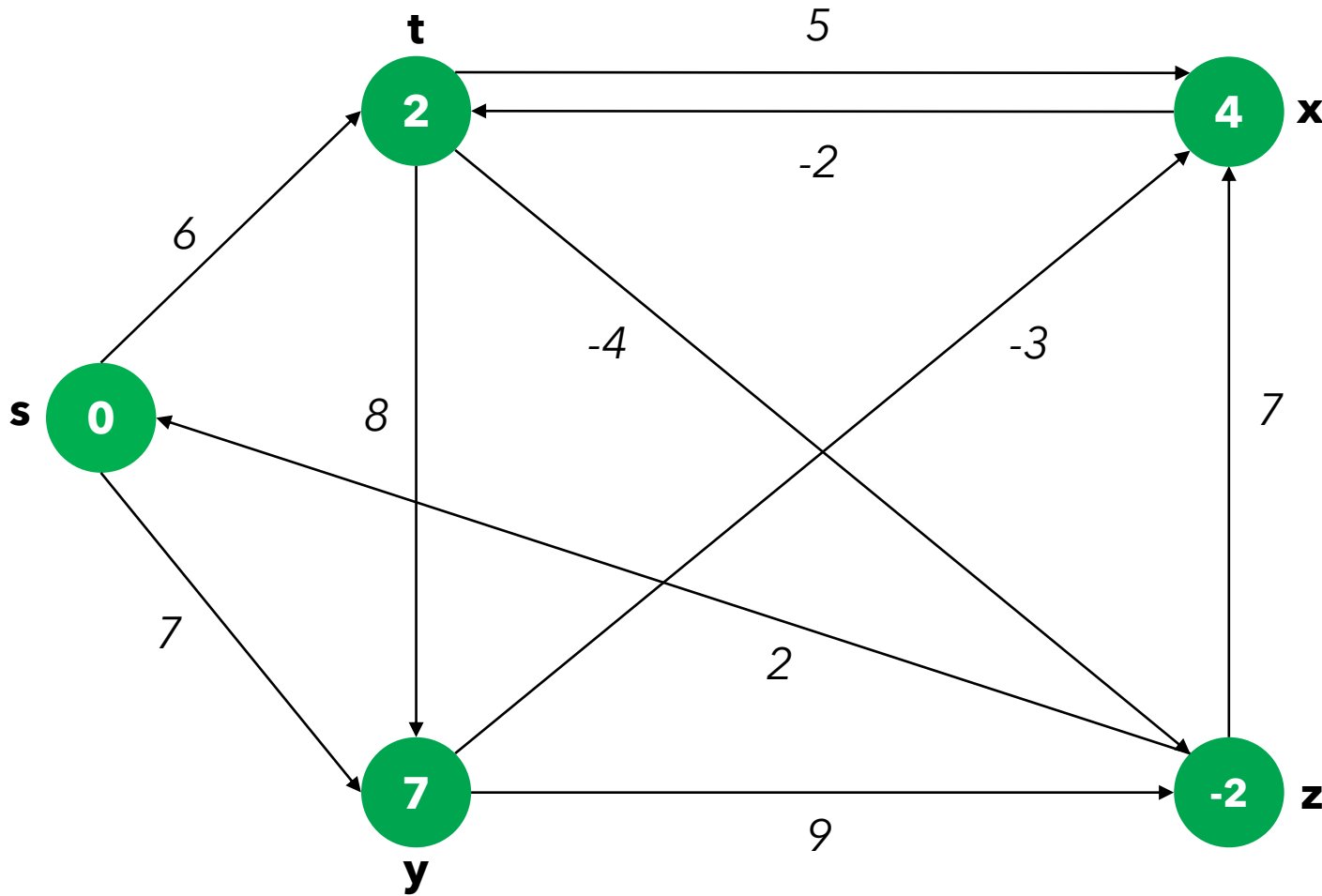
Bellman-Ford – 4th pass



t -> x
t -> y
t -> z
x -> t
y -> x
y -> z
z -> x
z -> s
s -> t
s -> y

**rilassamento di y -> z
(nessun effetto)**

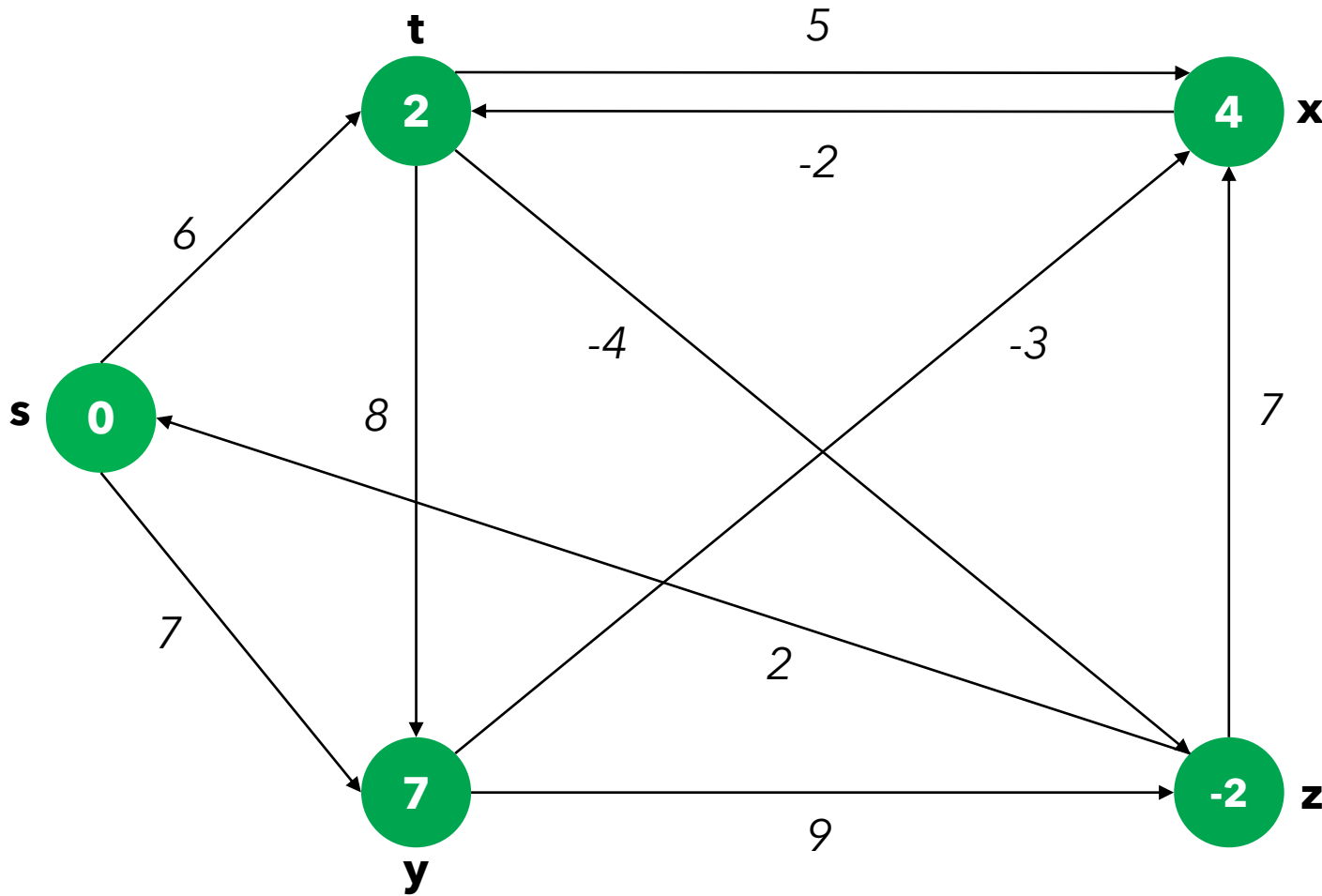
Bellman-Ford – 4th pass



t -> x
t -> y
t -> z
x -> t
y -> x
y -> z
z -> x
z -> s
s -> t
s -> y

**rilassamento di z -> x
(nessun effetto)**

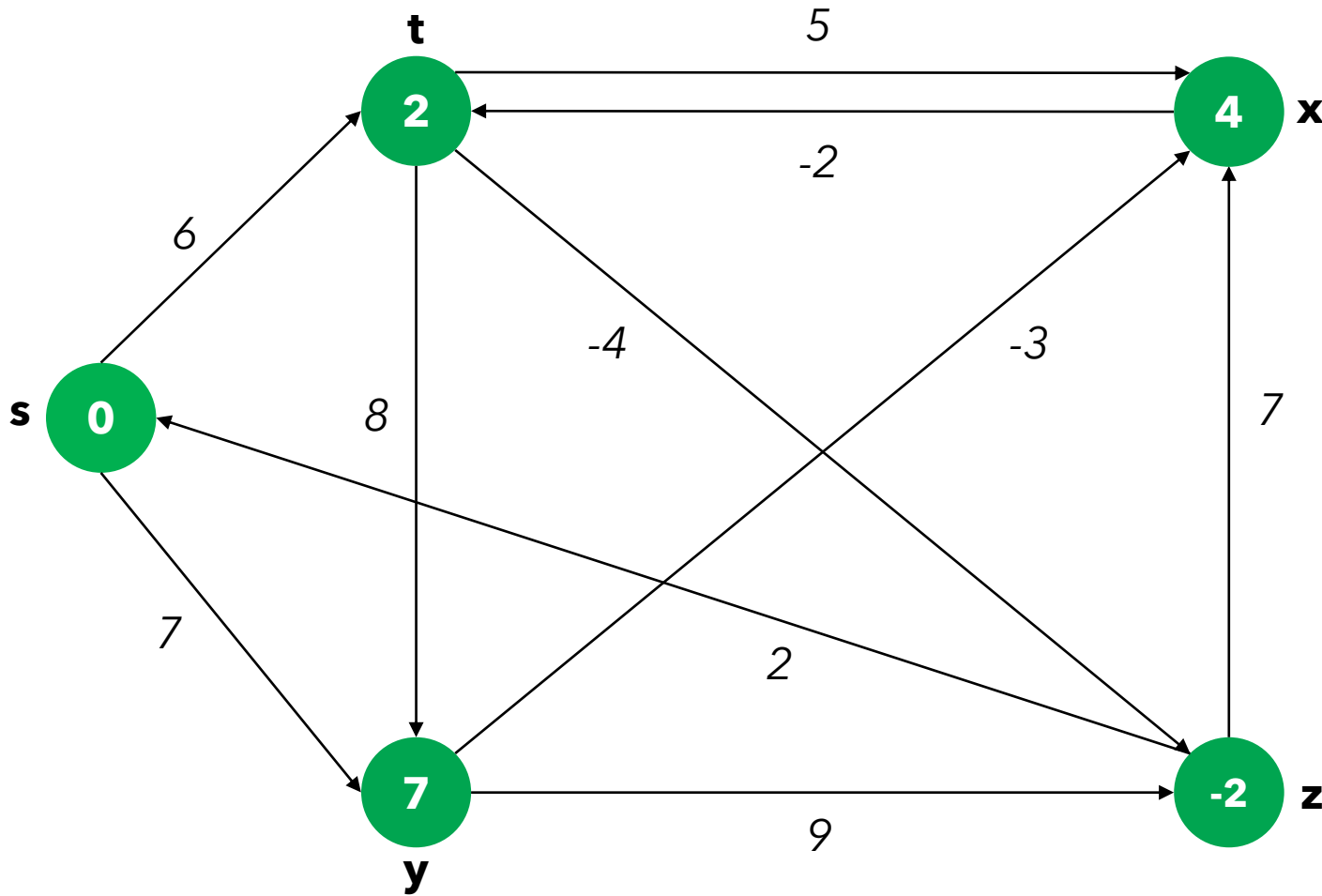
Bellman-Ford – 4th pass



t -> x
t -> y
t -> z
x -> t
y -> x
y -> z
z -> x
z -> s
s -> t
s -> y

**rilassamento di z -> s
(nessun effetto)**

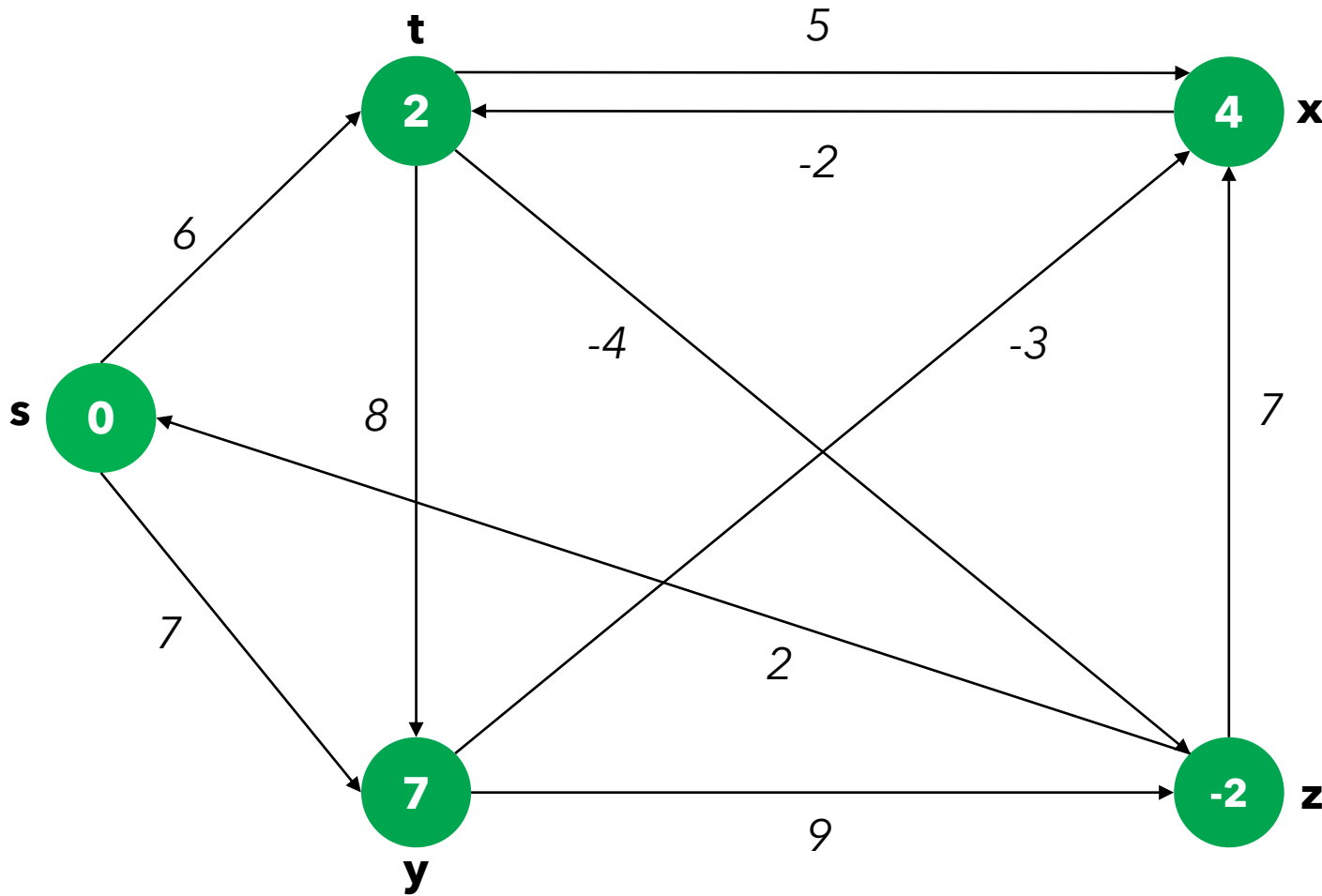
Bellman-Ford – 4th pass



t -> x
t -> y
t -> z
x -> t
y -> x
y -> z
z -> x
z -> s
s -> t
s -> y

**rilassamento di s -> t
(nessun effetto)**

Bellman-Ford – 4th pass



t -> x
t -> y
t -> z
x -> t
y -> x
y -> z
z -> x
z -> s
s -> t
s -> y

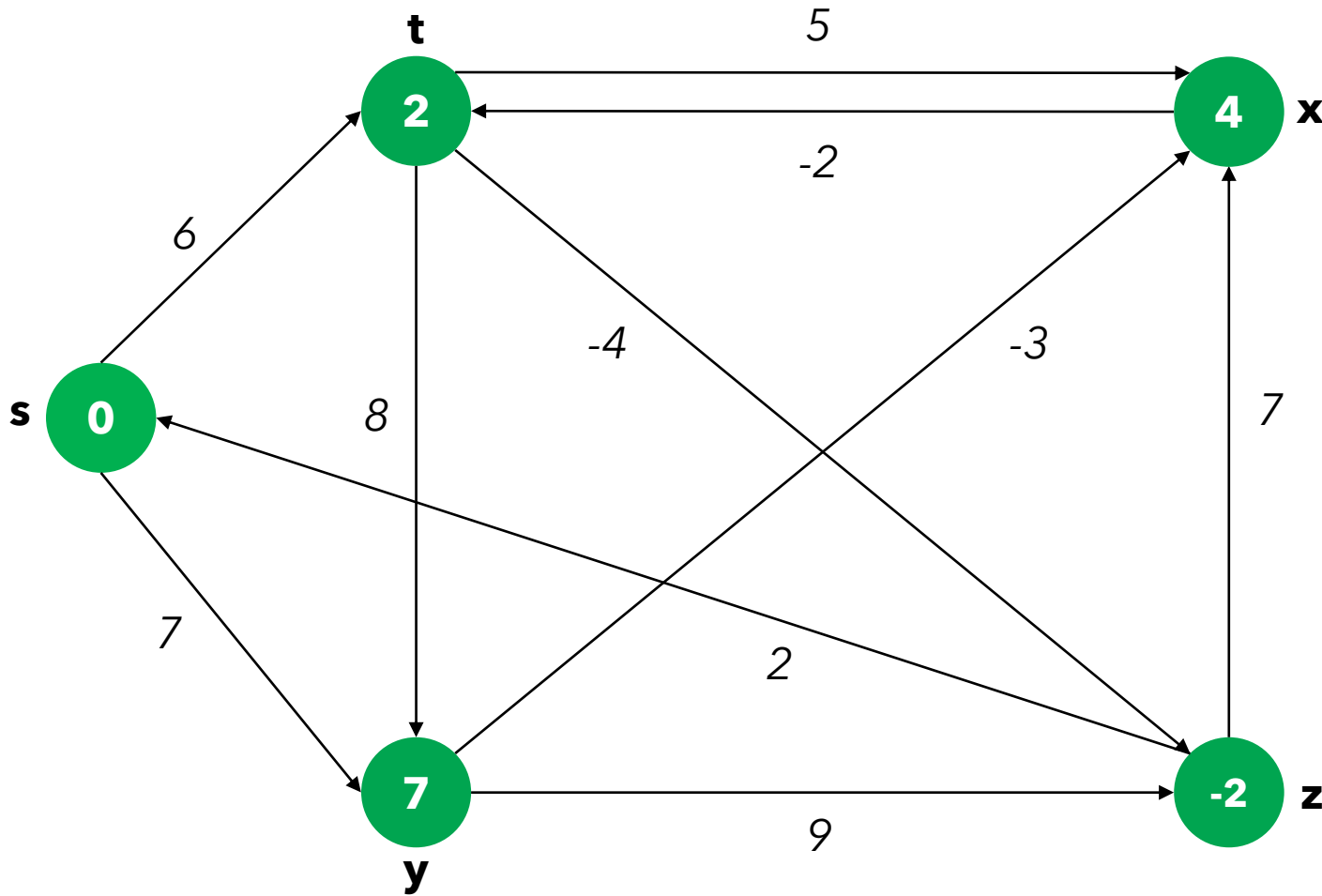
**rilassamento di s -> y
(nessun effetto)**

Bellman-Ford – verifica cicli di costo negativo

```
bellman_ford_part_2(G):  
    for each edge (u, v) in G:  
        if v.cost > u.cost + cost(u, v):  
            return TRUE  
    return FALSE
```

- se c'è un ciclo di costo negativo, la procedura restituisce TRUE
- altrimenti, restituisce FALSE

Bellman-Ford – verifica cicli di costo negativo



t -> **x**

t -> y

t -> z

x -> t

y -> x

y -> z

z -> x

z -> s

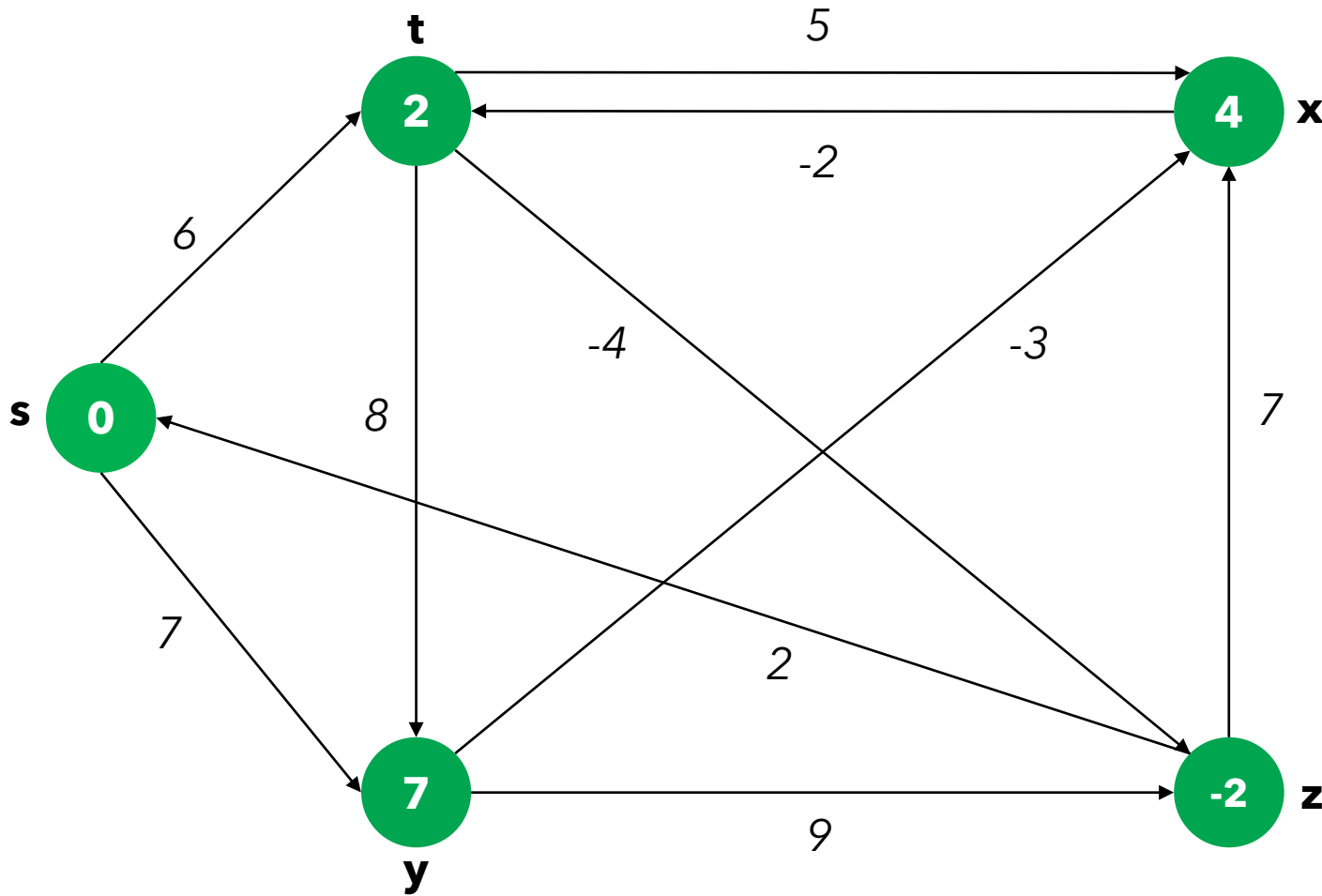
s -> t

s -> y

$x.cost > t.cost + cost(t, x)$

-> FALSE

Bellman-Ford – verifica cicli di costo negativo



t -> x

t -> **y**

t -> z

x -> t

y -> x

y -> z

z -> x

z -> s

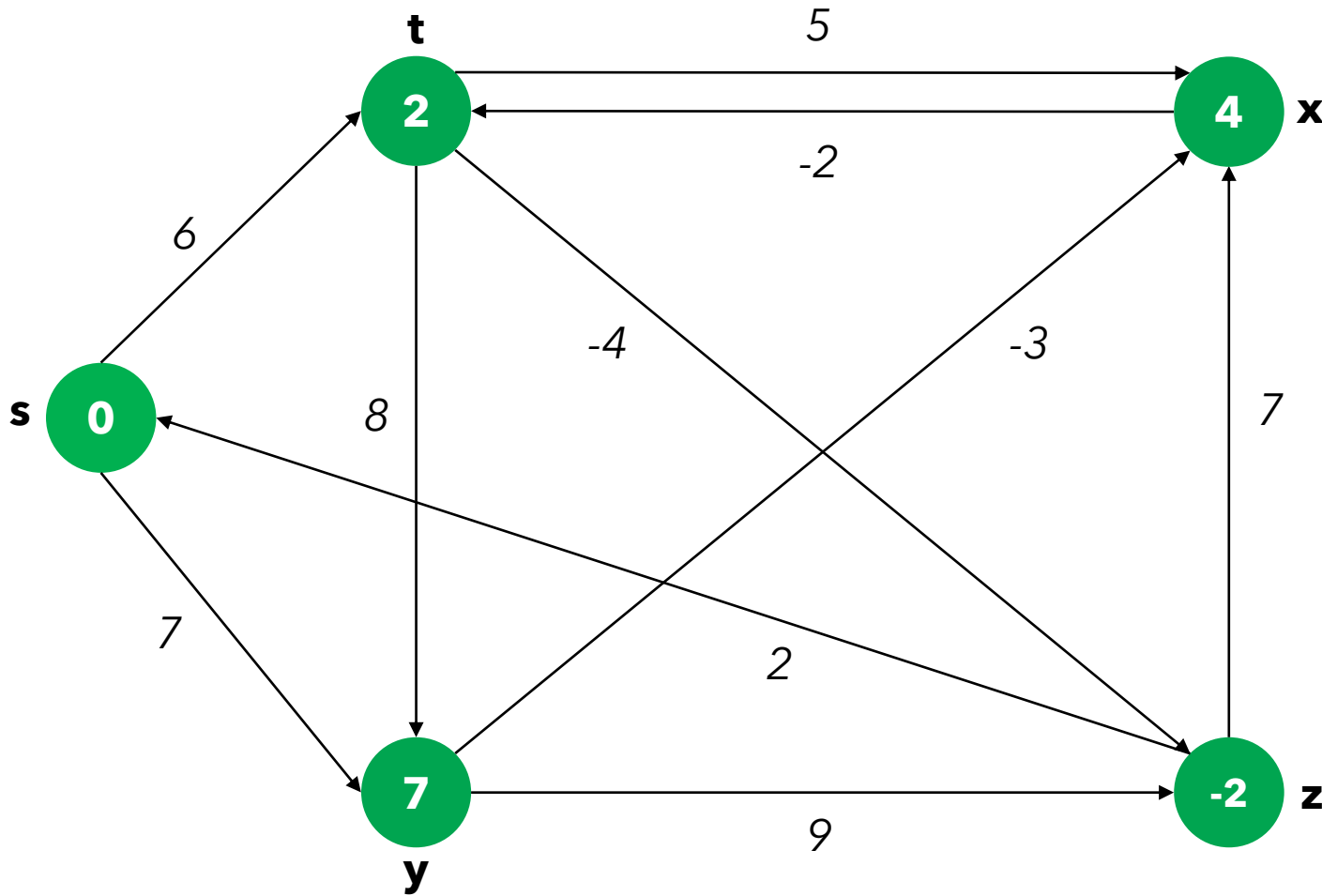
s -> t

s -> y

$y.cost > t.cost + cost(t, y)$

-> FALSE

Bellman-Ford – verifica cicli di costo negativo



t -> x

t -> y

t -> z

x -> t

y -> x

y -> z

z -> x

z -> s

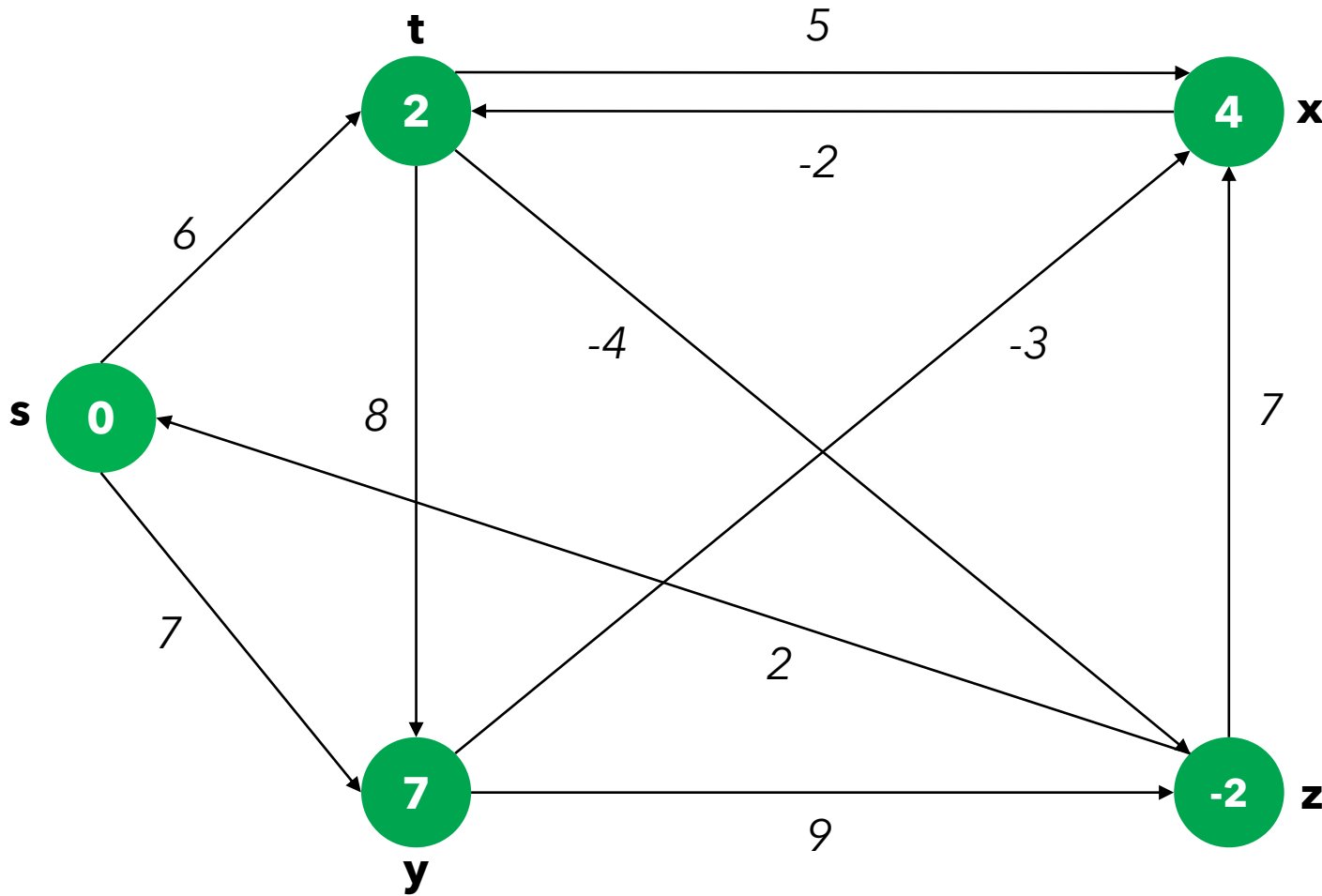
s -> t

s -> y

$z.cost > t.cost + cost(t, z)$

-> FALSE

Bellman-Ford – verifica cicli di costo negativo



t -> x

t -> y

t -> z

x -> t

y -> x

y -> z

z -> x

z -> s

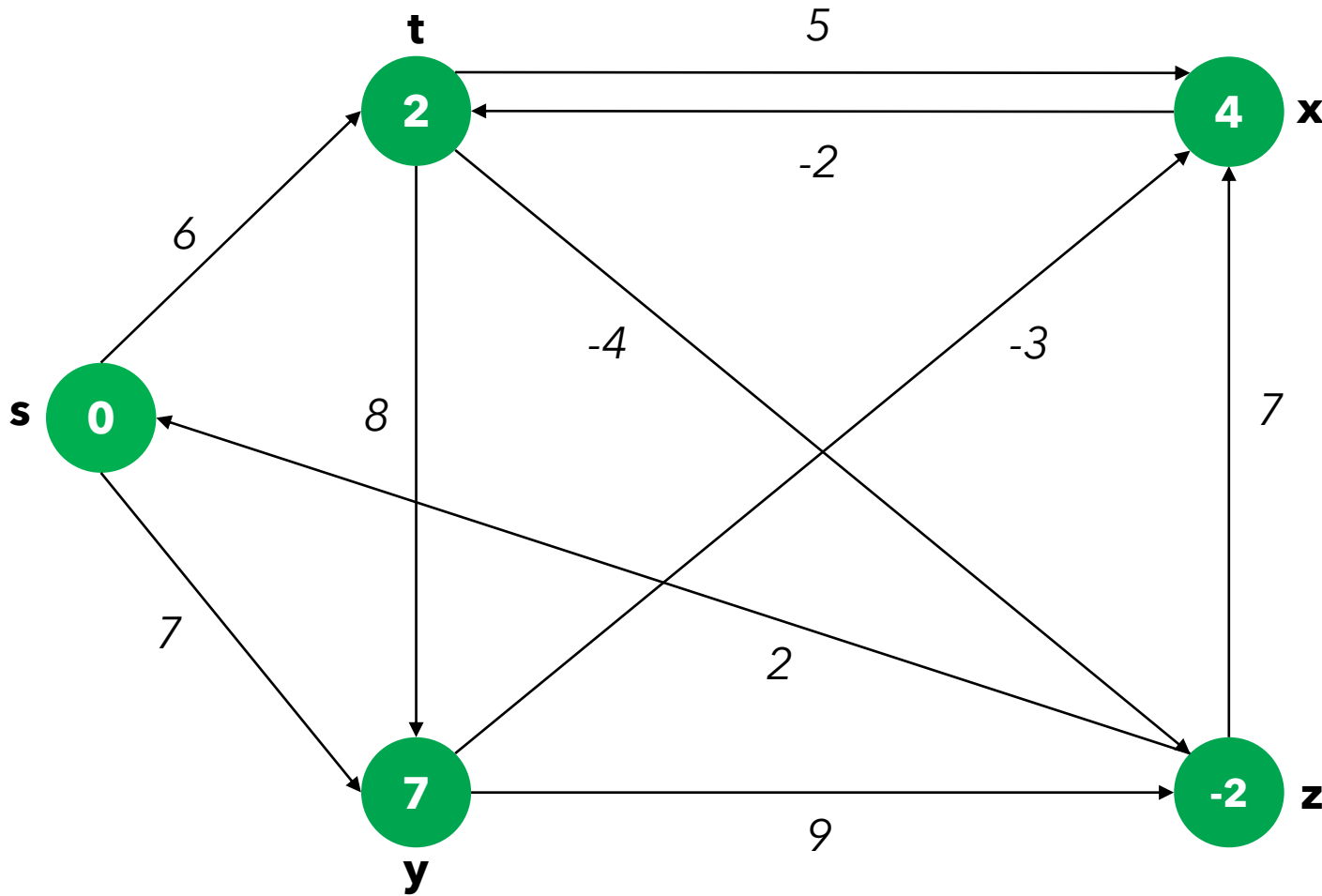
s -> t

s -> y

t.cost > x.cost + cost(x, t)

-> FALSE

Bellman-Ford – verifica cicli di costo negativo



t -> x

t -> y

t -> z

x -> t

y -> x

y -> z

z -> x

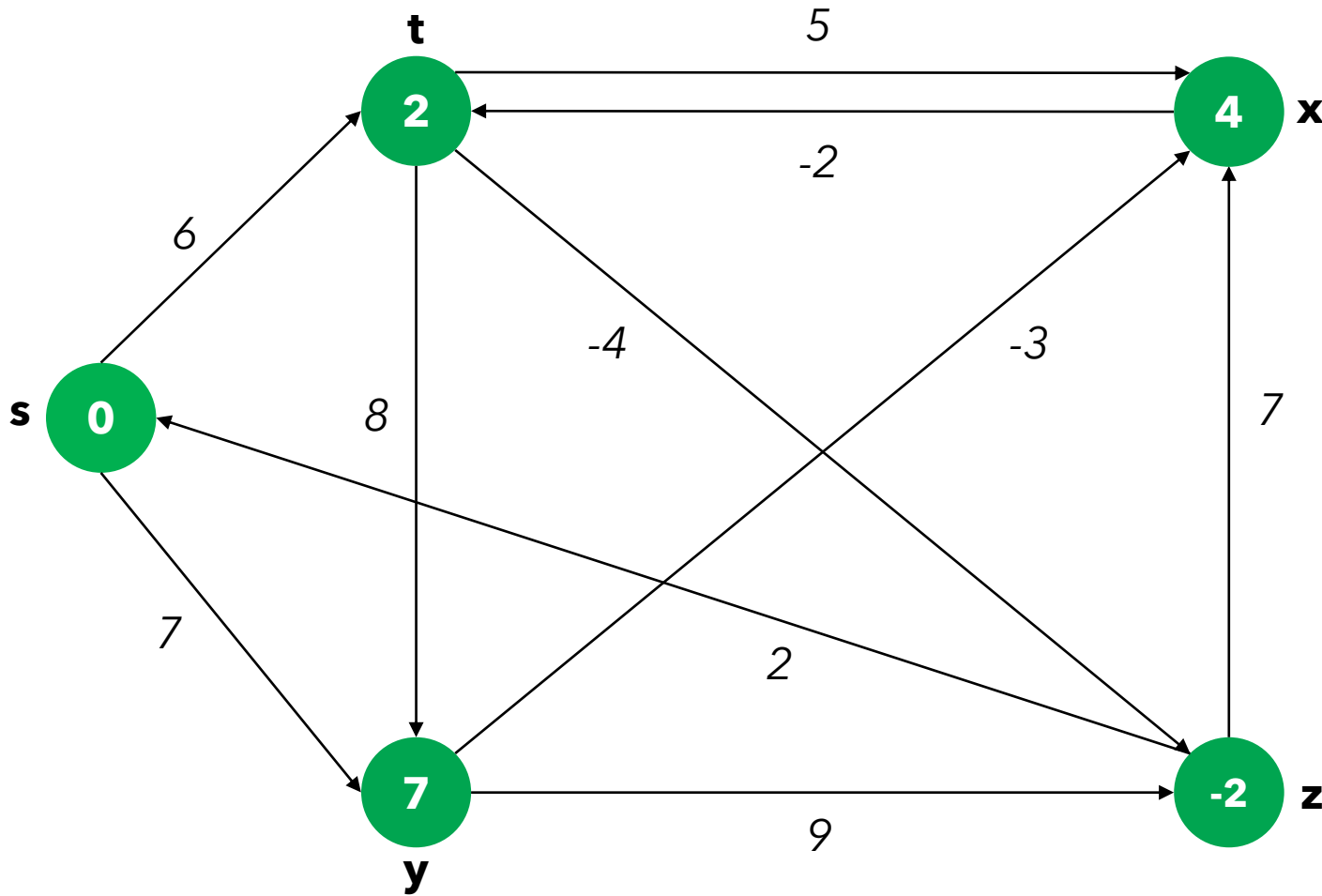
z -> s

s -> t

s -> y

$x.cost > y.cost + cost(y, x)$
-> FALSE

Bellman-Ford – verifica cicli di costo negativo



t -> x

t -> y

t -> z

x -> t

y -> x

y -> z

z -> x

z -> s

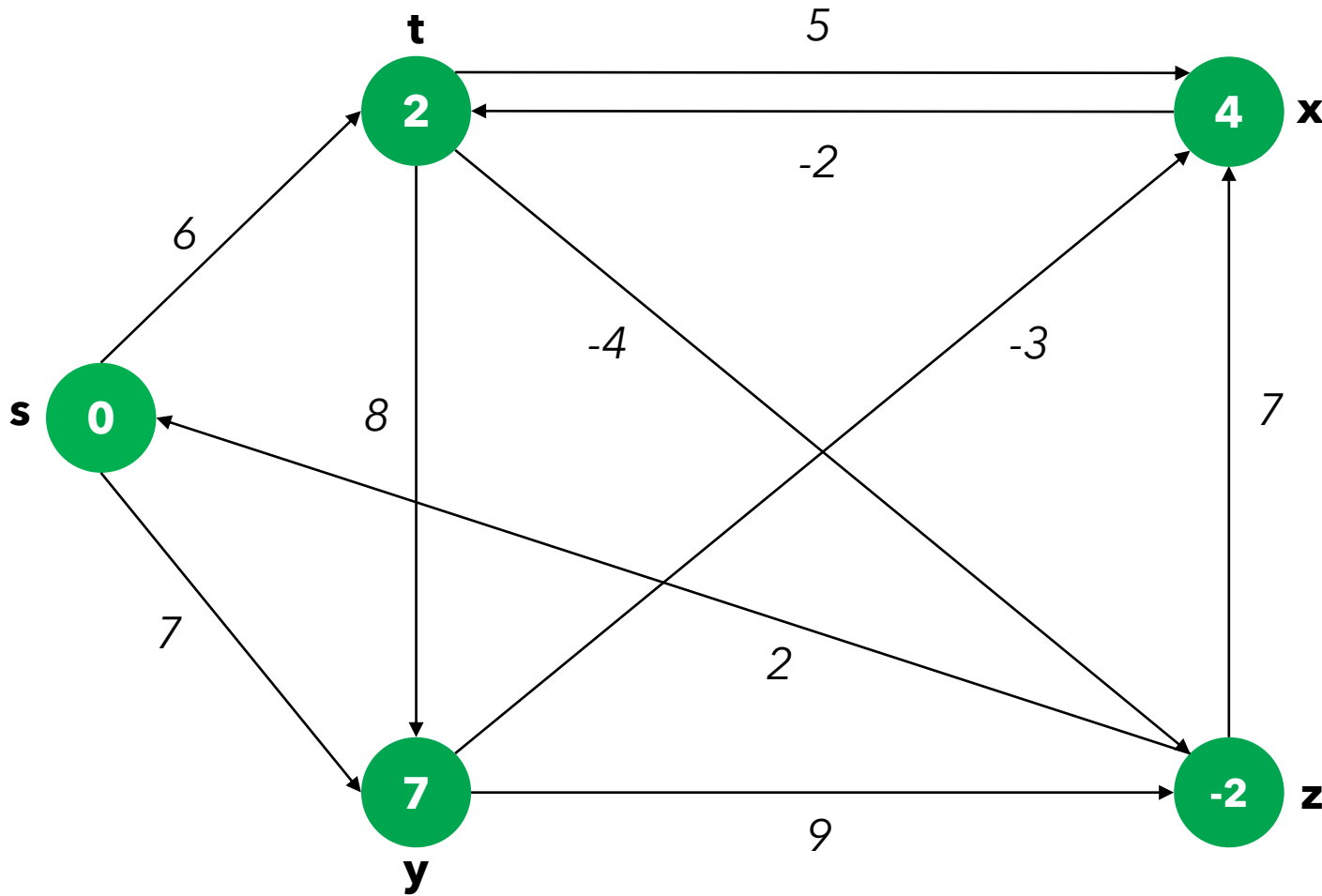
s -> t

s -> y

z.cost > y.cost + cost(y, z)

-> FALSE

Bellman-Ford – verifica cicli di costo negativo



t -> x

t -> y

t -> z

x -> t

y -> x

y -> z

z -> x

z -> s

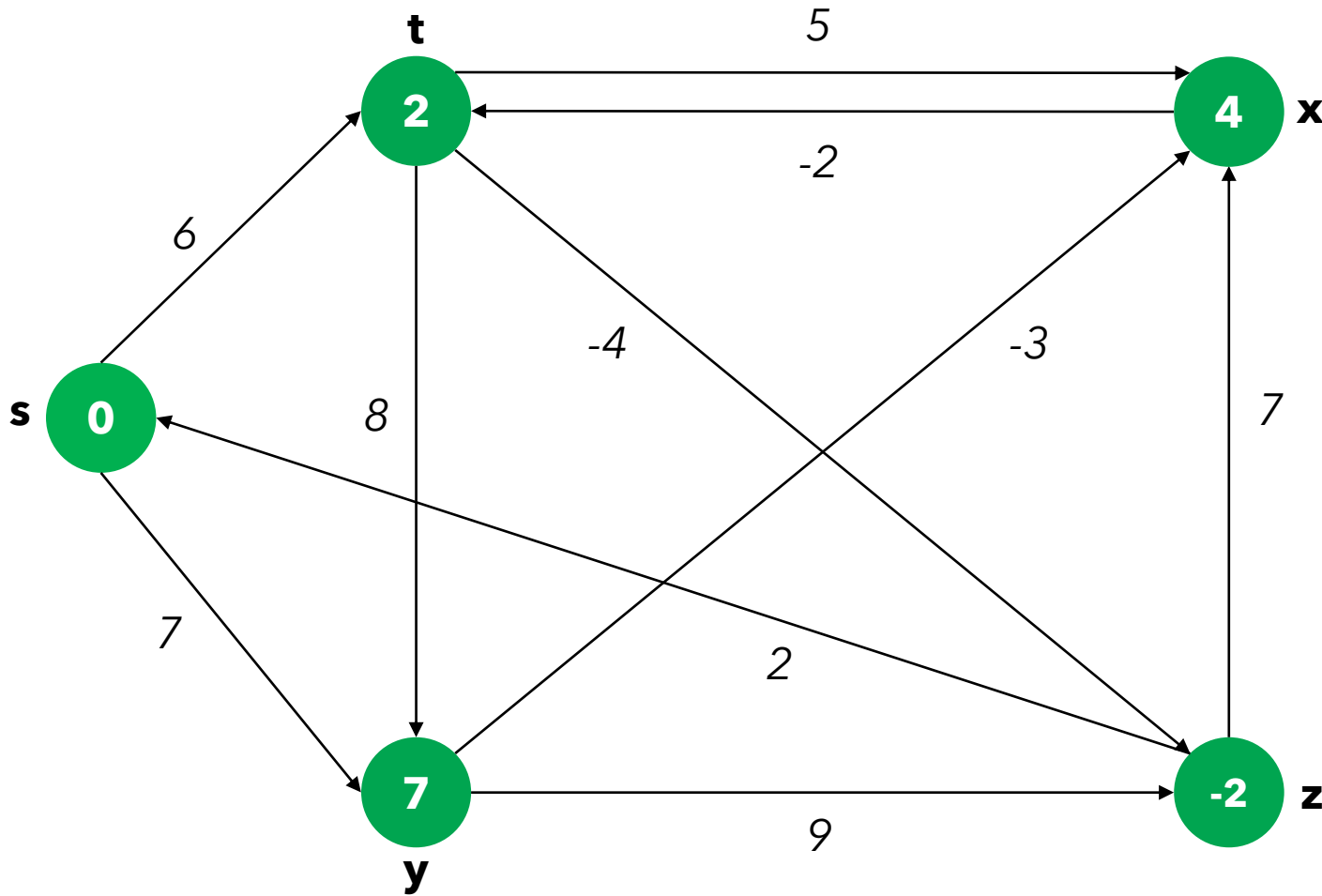
s -> t

s -> y

$x.cost > z.cost + cost(z, x)$

-> FALSE

Bellman-Ford – verifica cicli di costo negativo



t -> x

t -> y

t -> z

x -> t

y -> x

y -> z

z -> x

z -> s

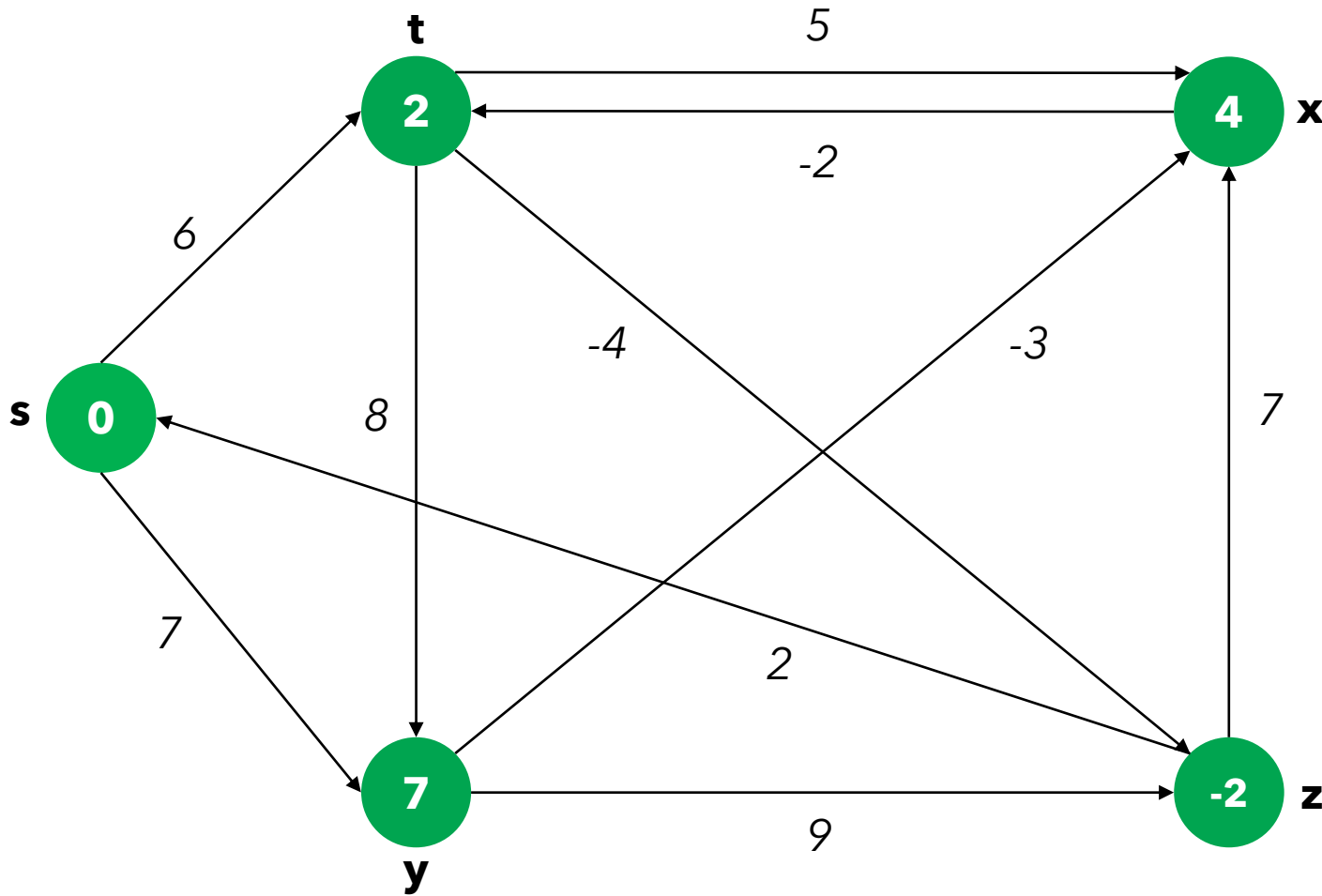
s -> t

s -> y

s.cost > z.cost + cost(z, s)

-> FALSE

Bellman-Ford – verifica cicli di costo negativo



t -> x

t -> y

t -> z

x -> t

y -> x

y -> z

z -> x

z -> s

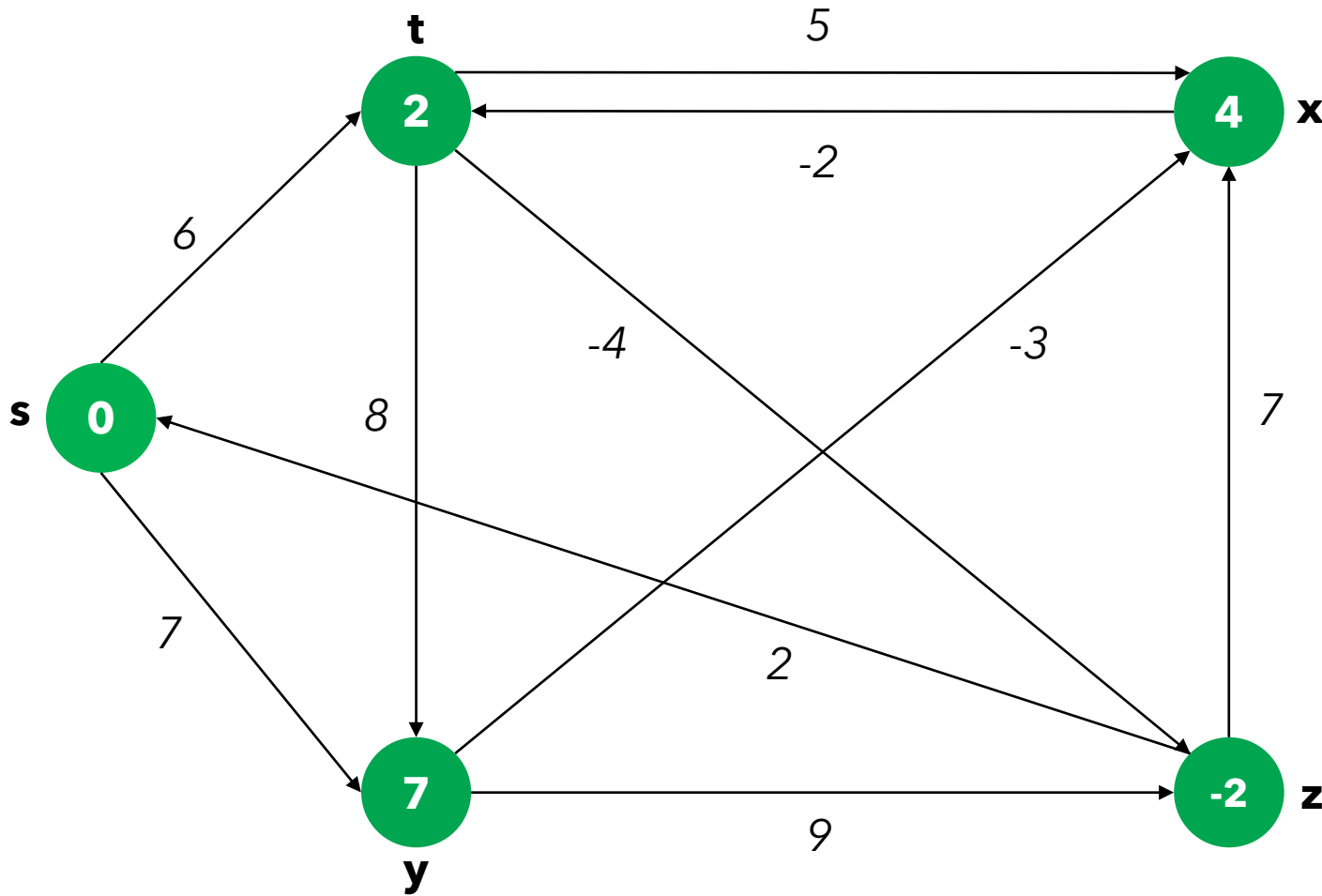
s -> t

s -> y

t.cost > s.cost + cost(s, t)

-> FALSE

Bellman-Ford – verifica cicli di costo negativo



t -> x

t -> y

t -> z

x -> t

y -> x

y -> z

z -> x

z -> s

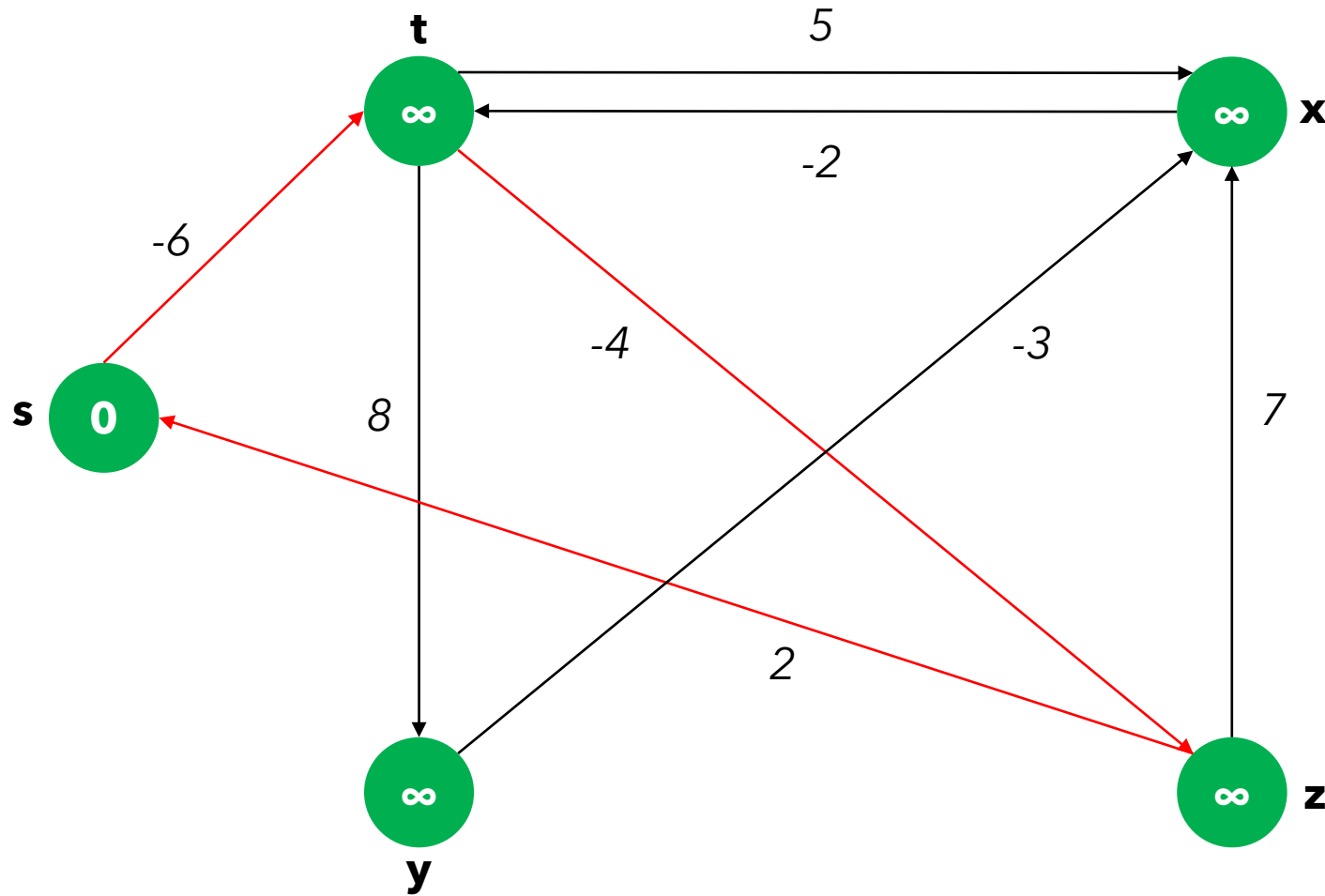
s -> t

s -> y

$y.cost > s.cost + cost(s, y)$
-> FALSE

no negative weight cycle

Bellman-Ford



eseguire le 2 sottoprocedure dell'algoritmo e verificare che il ciclo di costo negativo (indicato in rosso) venga rilevato correttamente