Lecture 17

Friday, 17 September 2021 8:17 AM

COL 351: Analysis and Design of Algorithms

Lecture 17

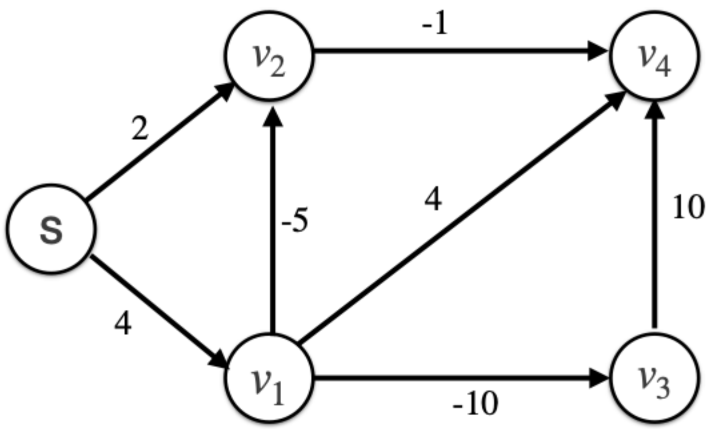
Single Source Distances in graph with negative edge-weights

Single Source Distance Problem

Given: A directed weighted graph $G = (V, E)$ with possibly negative edge weights, and a source s.

Output: Either a Shortest-path-tree rooted at s, or report that G contains a negative cycle reachable from s.

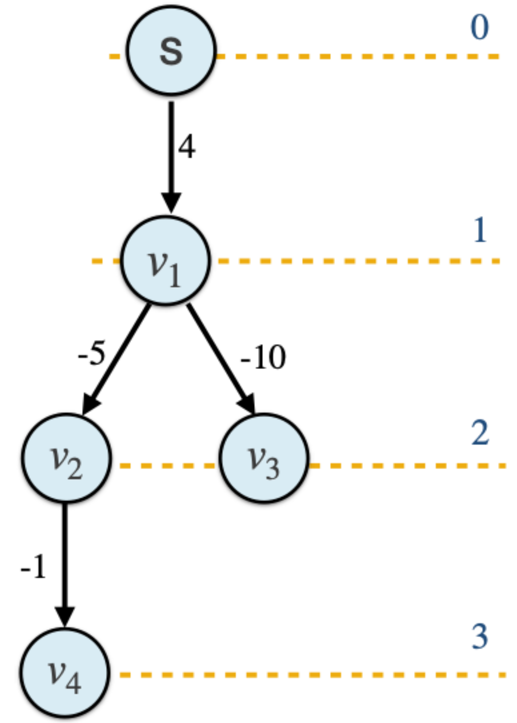
Example:



Notation: "Level" with respect to tree T

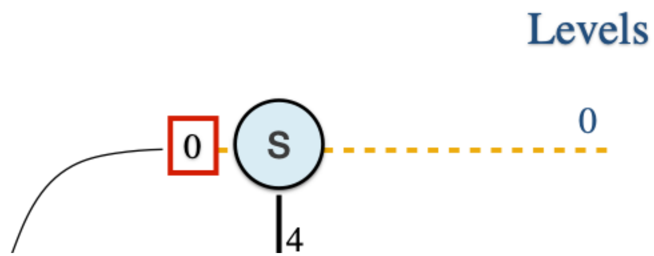
Levels

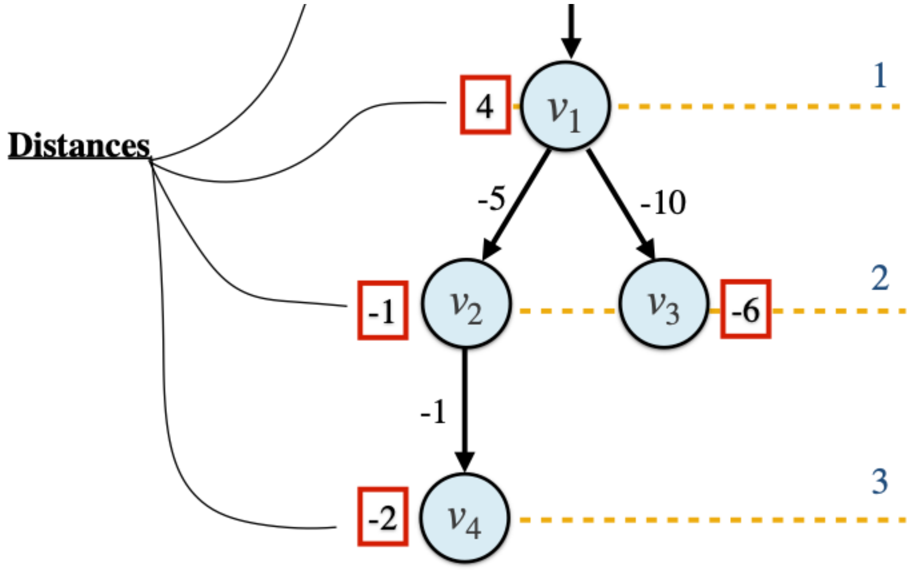
Level (v): Number of ancestors of "v" in T



Shortest path tree, T

Example: Level and Distances



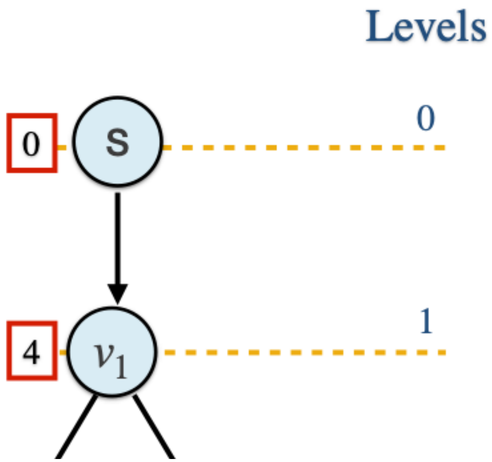


Shortest path tree, T

Definition

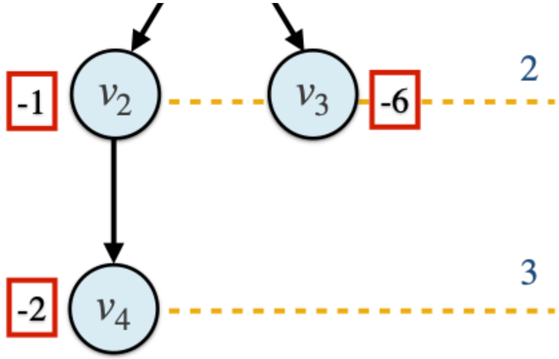
An n sized array D is said to be valid up to level i if:

- For $v \in V$ with level at most i, $D[v] = distance(s, v, G)$.
- For $v \in V$ with level greater than i, $D[v] \geq distance(s, v, G)$.



valid up to level $i = 0,\ 1,\ 2$

D	0	4	-1	-6	99
Levels	0	1	2	2	3



valid up to level $i = 0,\ 1$

D	0	4	-1	66	99
Levels	0	1	2	2	3

Lemma (assuming no negative weight cycle)

Lemma: Let D be an n sized array that is valid upto level i. Then executing the following step ensures that D is valid upto level $i + 1$.

> **For Each** $(x, y) \in E$:
>   **If** (D[y] > D[x] + weight(x, y)) **then**
>   D[y] = D[x] + weight(x, y)

Proof Sketch:

Consider a vertex y in level $i + 1$. Let x be parent of y in T. Then, level(x) = i.

Now, distance(s, y) = distance(s, x) + weight(x, y), as x is predecessor of y on a $s - y$ shortest path.

The level of x is i which means $D[x]$ is correct, so executing the above code will ensure $D[y] = distance(s, y)$.

<u>Homework</u>: Argue that for vertices upto level i, there will be no change in D on executing the above code.

Algorithm (assuming no negative weight cycle)

For Each $v \in V$:
    D[v]=∞ and parent[v]=null
D[s]=0

For i = 1 to $n - 1$:
    For Each $(x, y) \in E$:
        If (D[y] > D[x] + weight(x, y)) **then**
            D[y] = D[x] + weight(x, y)
            parent[y] = x

Return D, parent.

time = $O(mn)$

What if G has negative weight

time = O(mn)

cycle reachable from s?

What if G has negative weight cycle reachable from s?

Lemma: G has 'negative weight cycle' if and only if we can make improvement in vector D even in n^{th} round by using the following procedure.

```
For Each (x, y) ∈ E :
    If (D[y] > D[x] + weight(x, y)) then
        D[y] = D[x] + weight(x, y)
```

Proof: Homework

Bellman Ford algorithm

For Each $v \in V$:
    D[v]=∞ and parent[v]=null
D[s]=0

For i = 1 to n-1:
    For Each $(x, y) \in E$:
        If (D[y] > D[x] + weight(x, y)) **then**
            D[y] = D[x] + weight(x, y)
            parent[y] = x

For Each $(x, y) \in E$:
    If (D[y] > D[x] + weight(x, y)) **then**
        Return "Negative-weight cycle found"

Return D, parent.

O(mn) time algorithm for graphs with negative weights