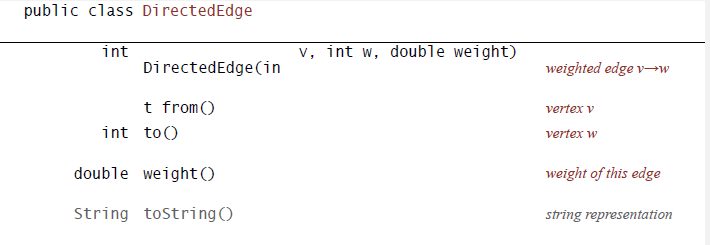
**Shortest Paths**

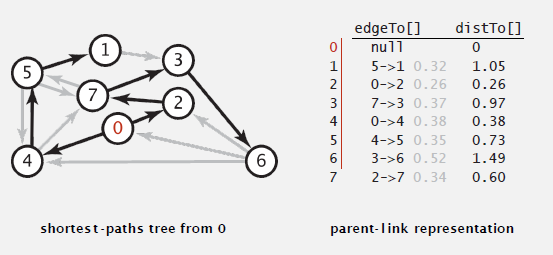
Të gjitha algoritmet për të gjetur shortest paths bazohen në një rregull dhe ajo është relaksimi i segmenteve. Dallimi midis **shortes path tress** dhe **minimum spanning tree** është se tek shortest path kemi kahje , pra e dim se prej ku deri ku dëshirojm të shkojmë, për shembull prej 0 deri tek të gjitha kulmet e tjera tek të cilat ka rrugë në graf . Aplikimi kryesor i shortest path algoritmeve që ka ndikim të madh në jetën ton është **Network routing protocols (OSPF, BGP, RIP).**

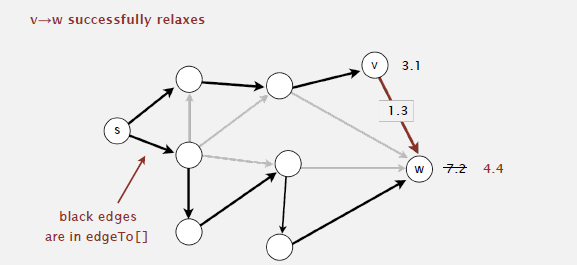
Ekzistojnë disa **variante** të shortest path ndër to janë : **Single source :** Gjen rrugët më të shkurta prej një kulmi **s** tek çdo kulm tjetër. (s-source, t-target), **Single sink :** Gjen rrugët më të shkurtra prej çdo kulmi tek një kulm **t**. **Source sink :** Gjen rrugën më të shkurtër prej një kulmi **s** në një kulm **t**. **All pairs :** Gjen rrugët më të shkurtëra nga secili kulm tek secili kulm.

Për të gjetur shortest path duhet të kemi : **segmente të peshuara dhe segmente të orientuara**. Nuk mund të kemi shortest path në një graf ku ka cikle negative (cikle çimi i të cilit është numër negativ).

**Weighted directed edge API** :



Shortest path definohet në kompjuter me anë të dy array, **edgeTo[v]-** segmenti i fundit në shortest path prej s tek v, **distTo[v]-** gjatësia e shortest path prej s tek v.

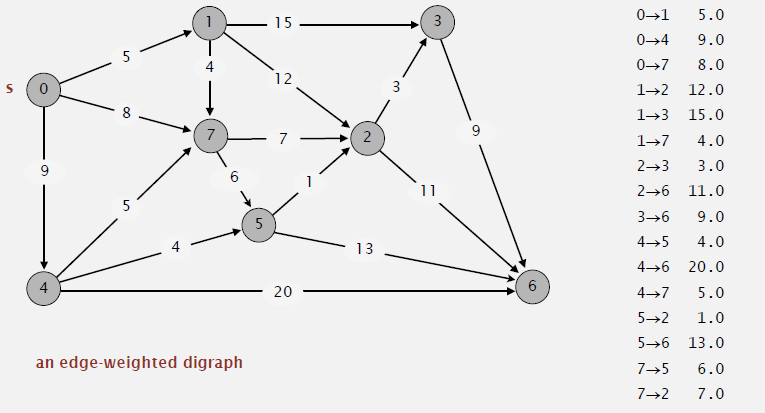
**Relaksimi(lehtësimi) i segmenteve :** procesi gjat të cilit testohen të gjitha peshat e segmenteve dhe kërkohen rrug të reja të cilat janë më të shkurtra.

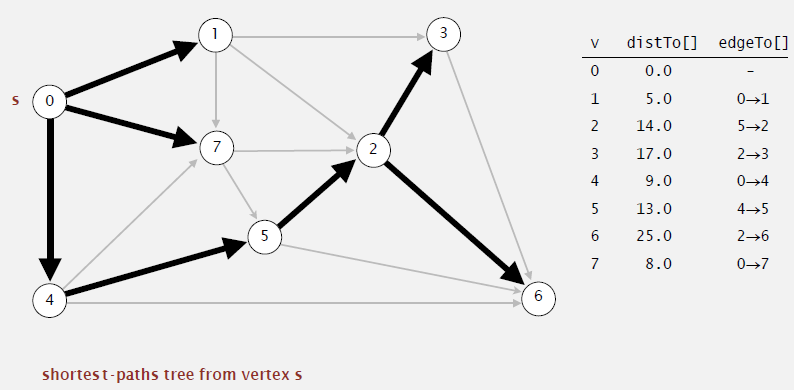
Kemi tre algoritme për gjetjen e shortest path tree : **Dijkstra’s , Topological sort(Acyclic) ,Bellman-Ford.**

**Dijkstra :** Smund të përdoret në grafe ku ka pesha negative, **Topological sort :** Nuk mund të përdoret në grafe ku ka directed cycles ( cikle direkte), **Bellman-Ford :** Nuk mund të përdoret në grafe ku ka cikle negative.

**Dijkstra’s Algorithm :** Përdoret për të gjetur shortest path tree në një graf.

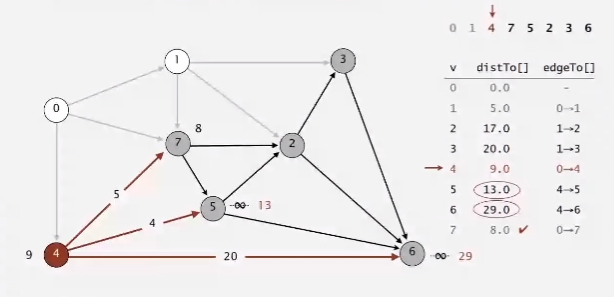
Hapi i parë është relaksimi i segmenteve duke filluar nga ato të sourcit, në rastin tonë nga 0 , pastaj vazhdojm me relaksimin e segmenteve tjera dhe shtimin ose zëvendsimin e segmenteve në array edgeTo dhe peshat e tyre në array distTo.



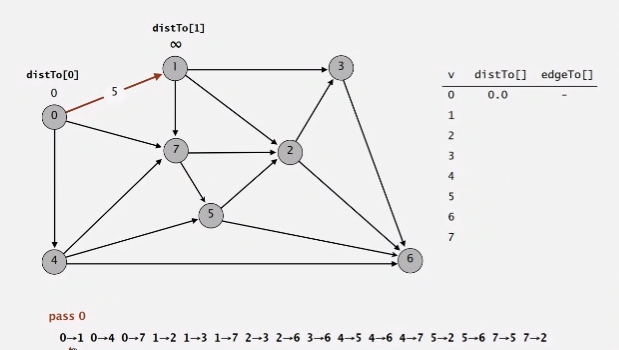


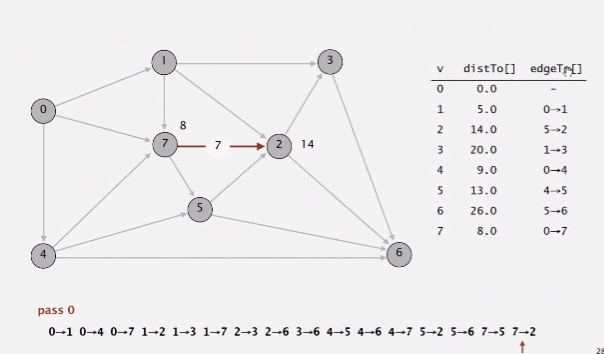
1.Dijkstra nuk pranon pesha negative , 2. Lexon kulmet bazuar në peshën minimale tek distTo[].

**Topological order algorithm (**AcyclisSP**) :** Algoritëm për gjetjen e shortest path tree në graf.

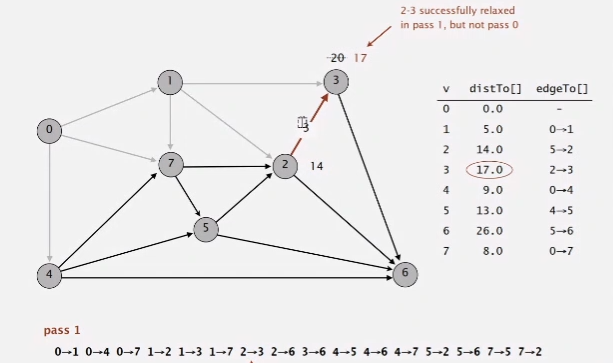
Fillimisht e nxjerrin topological order nga grafi, pastaj i relaksojmë të gjitha segmentet duke u bazuar në topolgical order. 

**Acyclic** : 1.Nuk punon nëse ka cikle , 2. Lexon kulmet nga topological order, 3.Për një graf mund të ket topolgical order të ndryshme zgjidhja del njësoj.

**Bellman-Ford Algorithm :** Algortiëm që gjen shorest path tree. Kur inicializohet grafi në fillim i kemi dy arrayt e zbrazta edgeTo dhe distTo , përshkak se ky algoritëm nuk starton nga kulmet , por duke i lexuar segmentet për relaksim, i vendos në një rresht të gjitha segmentet dhe vazhdon duke i relaksuar një nga një. 



Këtë proces duhet ta bëjmë për V herë dmth sa kemi kulme , aq pass duhet të bëjmë. Ne duhet të bëjm pass deri kur e shohim që nuk ka më ndryshime, dmth nëse në passin 2 nuk ka më ndryshime nuk vazhdojm më tej.



**Longest path** mund të na hyjn në pun atëher kur kemi shum pun për një her ,i gjejm në këtë rast punët që marin koh (rrug) më gjat i kryejm, pastaj paralelisht i kryejm punët me koh më të shkurtë, ndryshe quhet optimizim.

**Order of growth**: Në përgjëgjsi Dijkstra është më e ngadalta.

