.dest

LP3 ideas: s.

$u \longrightarrow v$

Graph $G=(V,E)$
source $s \in V$

Q: Does (u,v) appear in any shortest path from s (to anywhere)?

Check if $\delta(s,u) + w(u,v) = \delta(s,v)$?

Yes $\longrightarrow$ ✓

step 1: Find shortest paths from s — BFS / DAG-SP / Dijkstra / B-F ?

for each edge $e=(u,v) \in G$:
  if $\delta(s,u) + w(u,v) = \delta(s,v)$
  (u.distance + e.weight == v.distance) $\to$
    Add $e=(u,v)$ to graph D.

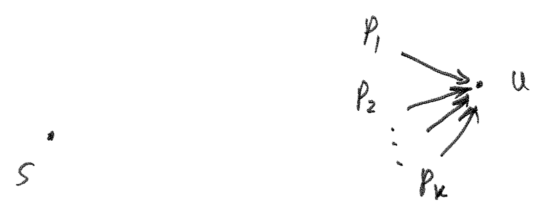
if D has a cycle: $\longrightarrow$ No solution.

D is acyclic (i.e. D is a DAG)

Every path in D from s is a shortest path in G.

\# of shortest paths from s to u in G $=$ \# of paths from s to u in D.

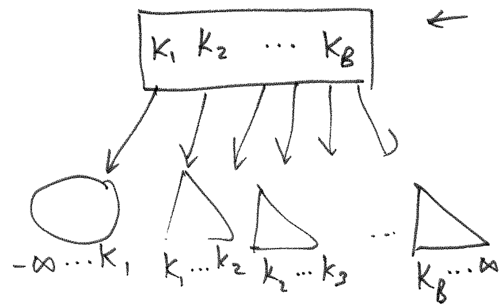
$\uparrow$ Ex ... in CLRS.



\# of path from s to u $=$ \# of paths from s to p_1
$+$ . . . . . . p_2
$+$ . . . . : p_k

B-trees — balanced trees that generalize BST with trees of higher degree.

← root of B-tree.



← B children?

Given x: Find i: $K_{i-1} \leq x < K_i$ → Follow child i

Insert / Delete: Each internal node has $B/2 \cdots B$ children.

Rotations to rebalance trees.

Rough calculation: Find x: Height of tree $\approx \log_B n$

Time to find i at each level: $\log B$ (with binary search)

Total time: $\log_B n \cdot \log B = \log n$.

For in-memory dictionaries, B-trees are no better than AVL or Red-Black trees