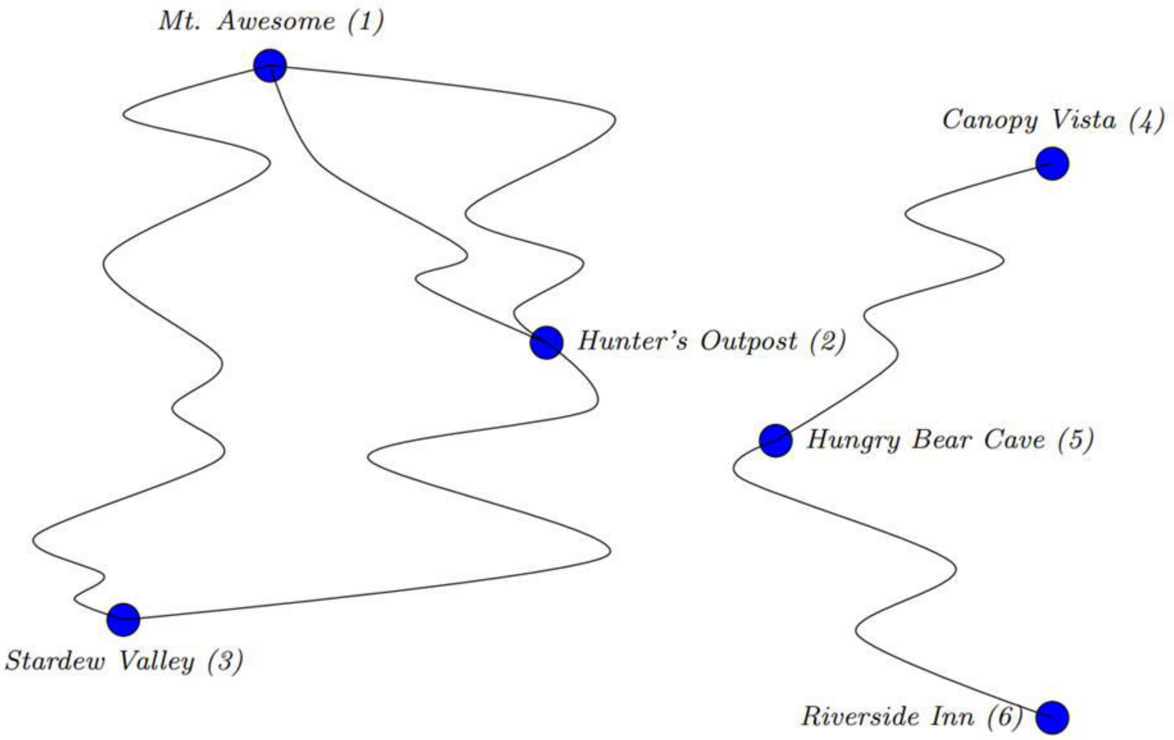
Problem A:
It's All Downhill From Here

Klaus Biehler, Markus Theiner

Problem

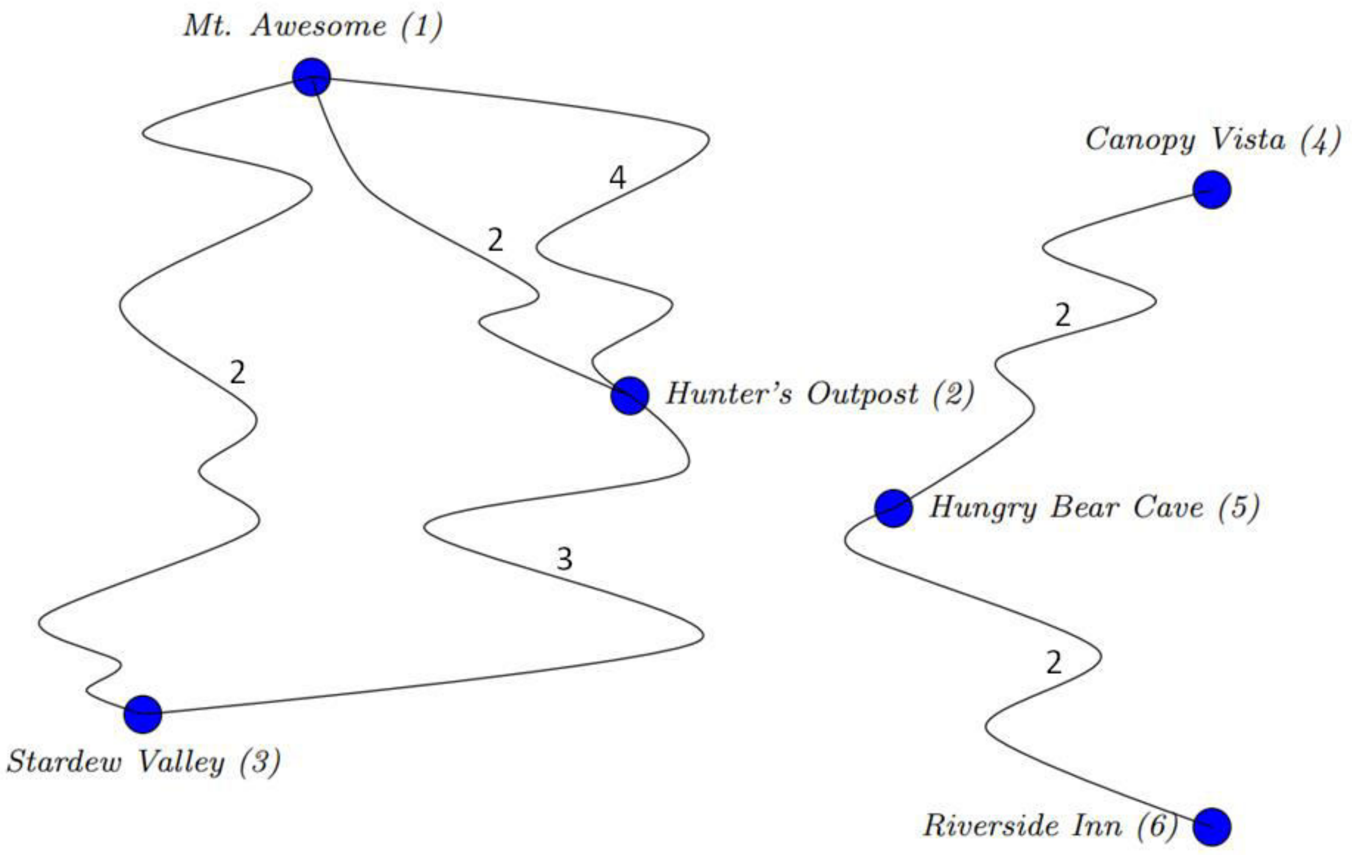
Input + Constraints

- Slopes
  - at least 1 slope
  - at most 5000 slopes
  - slopes go downhill
  - condition measure between 1 and 100
- Points
  - at least 2 points
  - at most 1000 points
  - point without incoming slope = mountain top
  - point without outgoing slope = valley
  - helicopter can land at any point



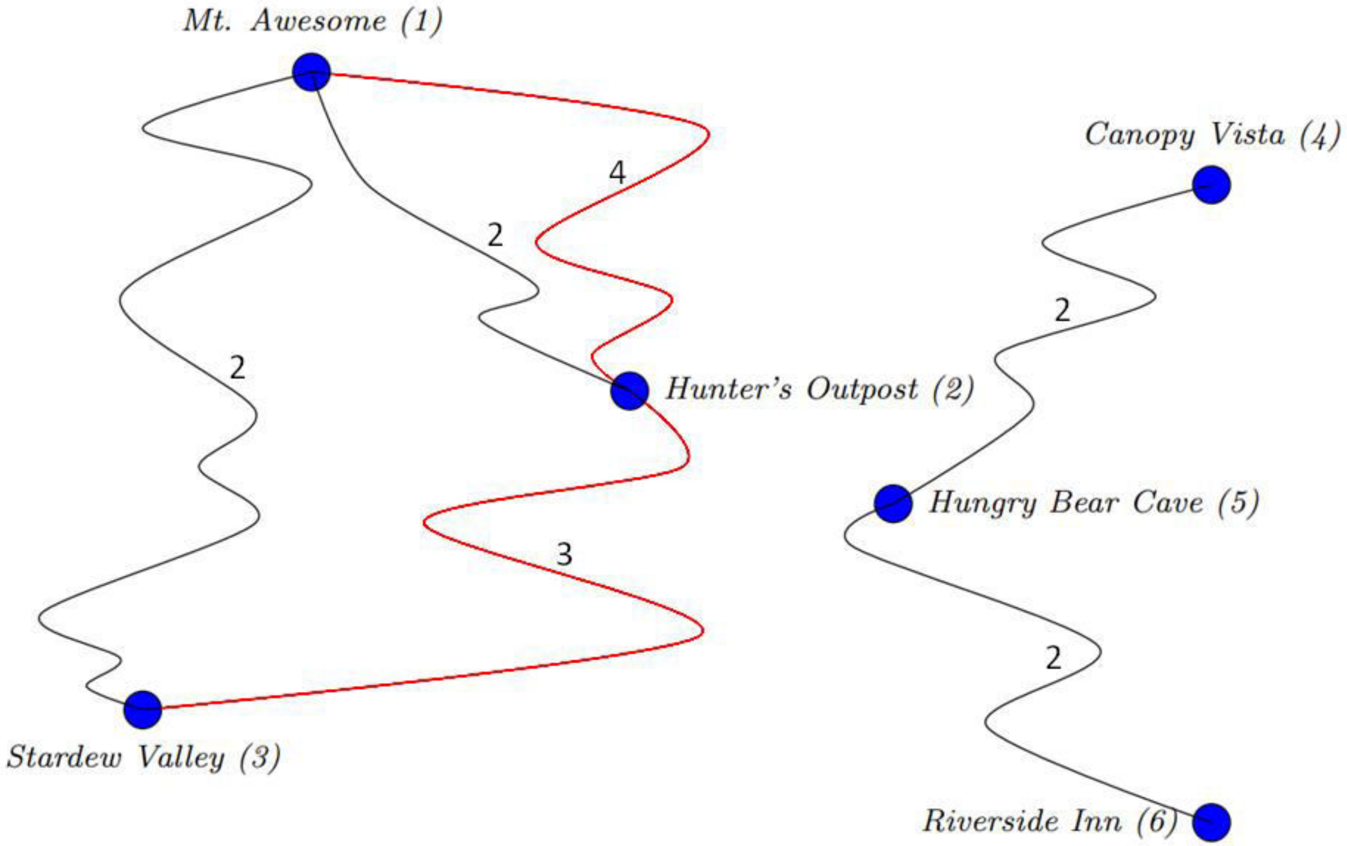
Input

```
6 6
1 2 2
4 5 2
2 3 3
1 3 2
5 6 2
1 2 4
```



Output

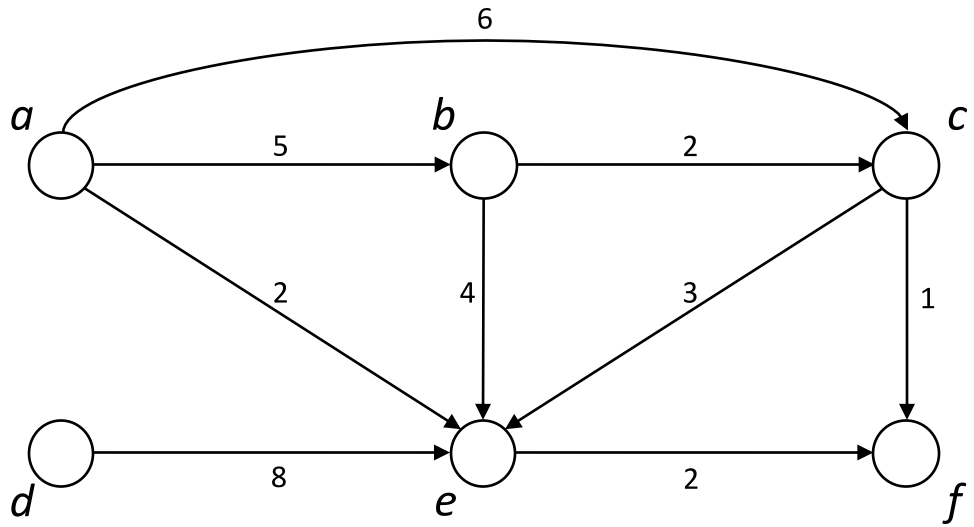
7


Mt. Awesome (1)
Canopy Vista (4)
Hunter's Outpost (2)
Hungry Bear Cave (5)
Stardew Valley (3)
Riverside Inn (6)
4
2
2
2
3
2

Summary

- Input:
  - directed, weighted, acyclic graph
- Output:
  - weight of longest path
- Optimizations:
  - Starting points are always at mountain tops
  - Removal of multiple slopes between 2 points

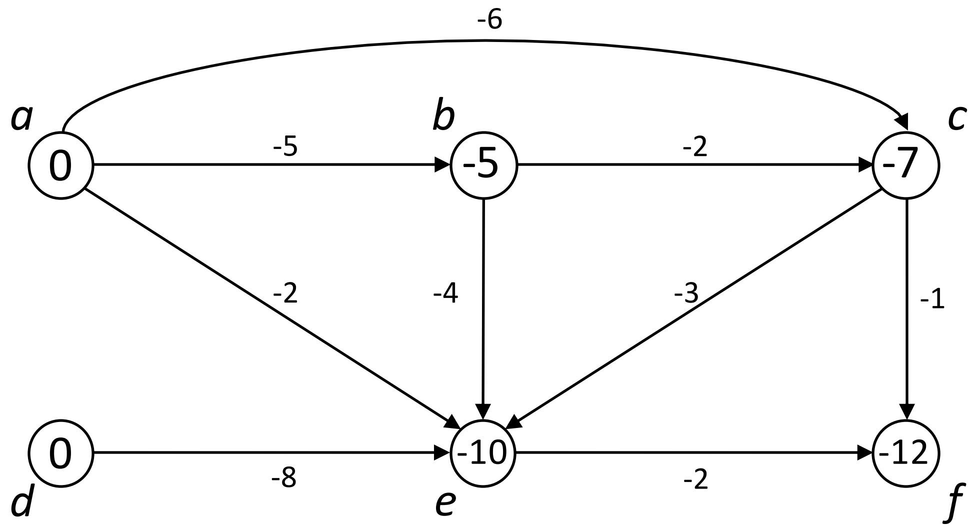
Approach 1: Brute Force



maxWeight=12

- for each $v \in V$ with $indeg(v) = 0$ walk (recursively) every path with starting point v
- return weight of longest path
- runtime? $\mathcal{O}(2^V)$

Approach 2: Bellman-Ford

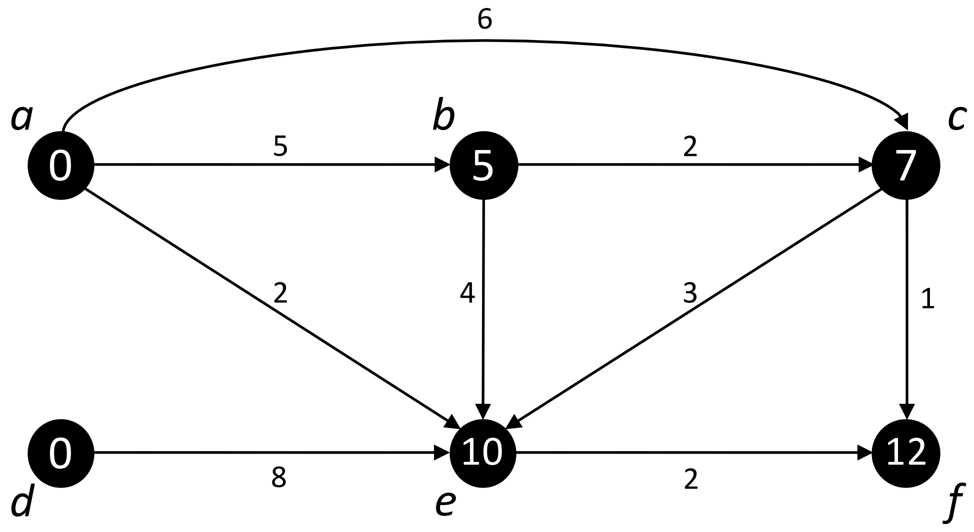


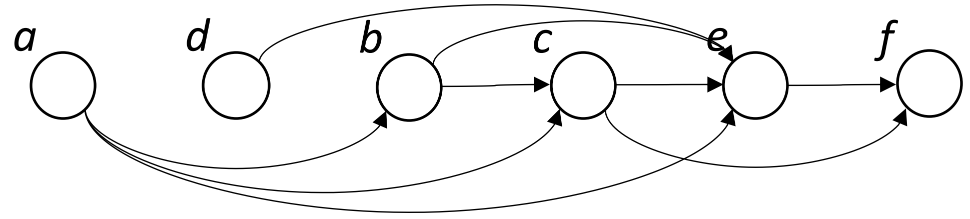
- negate edge weights and search for shortest path
- Bellman-Ford:

  for $i = 1$ *to* $|V| - 1$ **do**
      for $uv \in E$ **do**
          $v.d \leftarrow min\{v.d, u.d + w(u, v)\}$
      end
  end

- runtime? $\mathcal{O}(V \cdot E)$

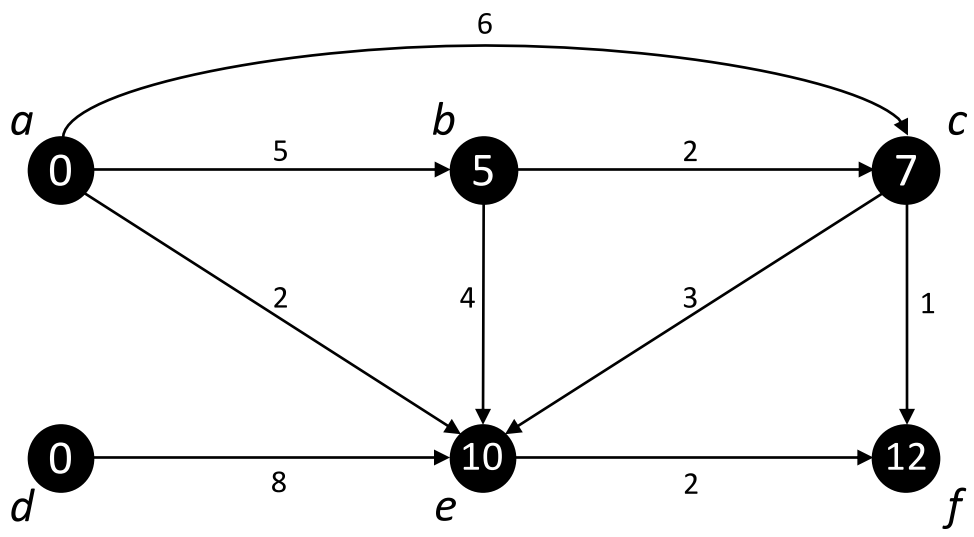
Approach 3: ?

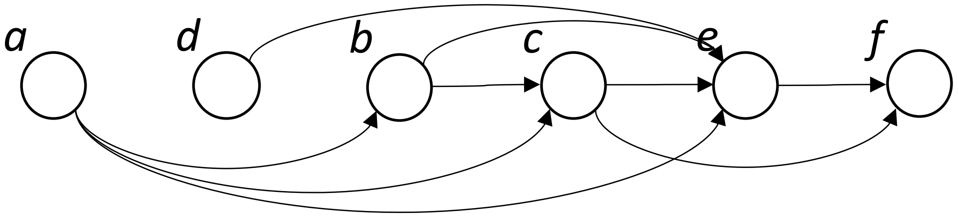




- color each $v \in V$ with indegree $= 0$ black
- while not every vertex is black
  - find vertex v, where all incoming edges are connected to black vertices
  - for each incoming edge uv set $v.d = \max\{v.d,\ u.d + w(uv)\}$
  - color v black
- return $\max\{v.d \mid v \in V\}$
- runtime?

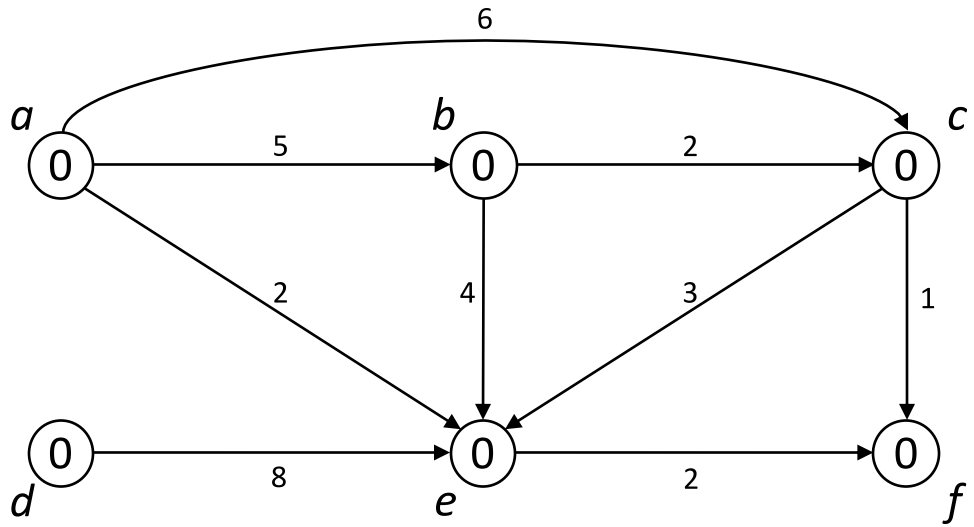
Approach 3: Topo-Sort





- sort vertices in topological order
- color each $v \in V$ with indegree $= 0$ black
- while not every vertex is black
  - ~~find vertex v, where all incoming edges are connected to black vertices~~
    take next vertex v of the topological order
  - for each incoming edge uv set $v.d = \max\{v.d,\ u.d + w(uv)\}$
  - color v black
- return $\max\{v.d \mid v \in V\}$
- runtime? $\mathcal{O}(V + E)$

Implementation topological sort



- for each $v \in V$ set $v.in = indegree(v)$
- append each vertex with $v.in = 0$ to a List L
- for each vertex $v \in L$ (in order)
  - for each outgoing edge vu
    - $u.in = u.in - 1$
    - if u.in == 0 append u to L
- L is now sorted in topological order
- runtime? $\mathcal{O}(V + E)$

L: