Analysis of Algorithms - Notes

Philip Warton

November 10, 2020

1 Graph Search Algorithms

We begin with this simple example of the "whatever first search". This is an algorithm that will brute force find some path from $v \to s$ for any s-reachble vertex v.

Whatever-first-search algorithm with path rememberance:

```
WFS(G, s) {
    Parent(s) = ∅
    Bag = {(s, ∅)}
    while Bag ≠ ∅ {
        (v, p) = any vertex from Bag
            (remove v from Bag)
        if v is not marked {
            mark v
            Parent(v) = p
            for all (v, w) ∈ E {
              ''  add (w, v) to Bag
            }
        }
    }
}
```

Once the bag is eventually empty, we can find the path from v to s by

$$v \to \text{parent}(v) \to \text{parent}(\text{parent}(v)) \to \cdots \to s$$

The WFS algorithm marks all vertices reachable from s.

Proof. Let s be our initial point. We use induction that is based on the shortest path s to v.

Base Case: Our vertex $v = s$, and ShortestPathLength$(v \to s) = 0$, and WFS marks it on the first iteration.

Inductive Step: For any point v for which the shortest path $s \leftarrow v$ is smaller than $k \in \mathbb{N}$, we assume that WFS has already marked v. Let v be a point for which its minimum distance from s is k. Then let u be the neighbor of v that lies on a shortest path from v to s. Then the length of $u \to s$ is $k - 1$, and by assumption u is marked. Since v is a neighbor of u, v will be marked as well. □

What kind of data structures would be good to use for Bag? If we use a stack, then this algorithm becomes a depth first search algorithm (DFS). If we use a queue, then we have a breadth first search (BFS) algorithm. If there is a weighted graph, one can use a priority queue based on edge weight, resulting in Dijkstra's shortest path algorithm.

```
Serach(G, s):
    Insert s into Bag
    while Bag ≠ ∅:
        v: any vertex from Bag
        if v is not marked:
            mark v
            ∀v → w ∈ E
                insert w into Bag
```

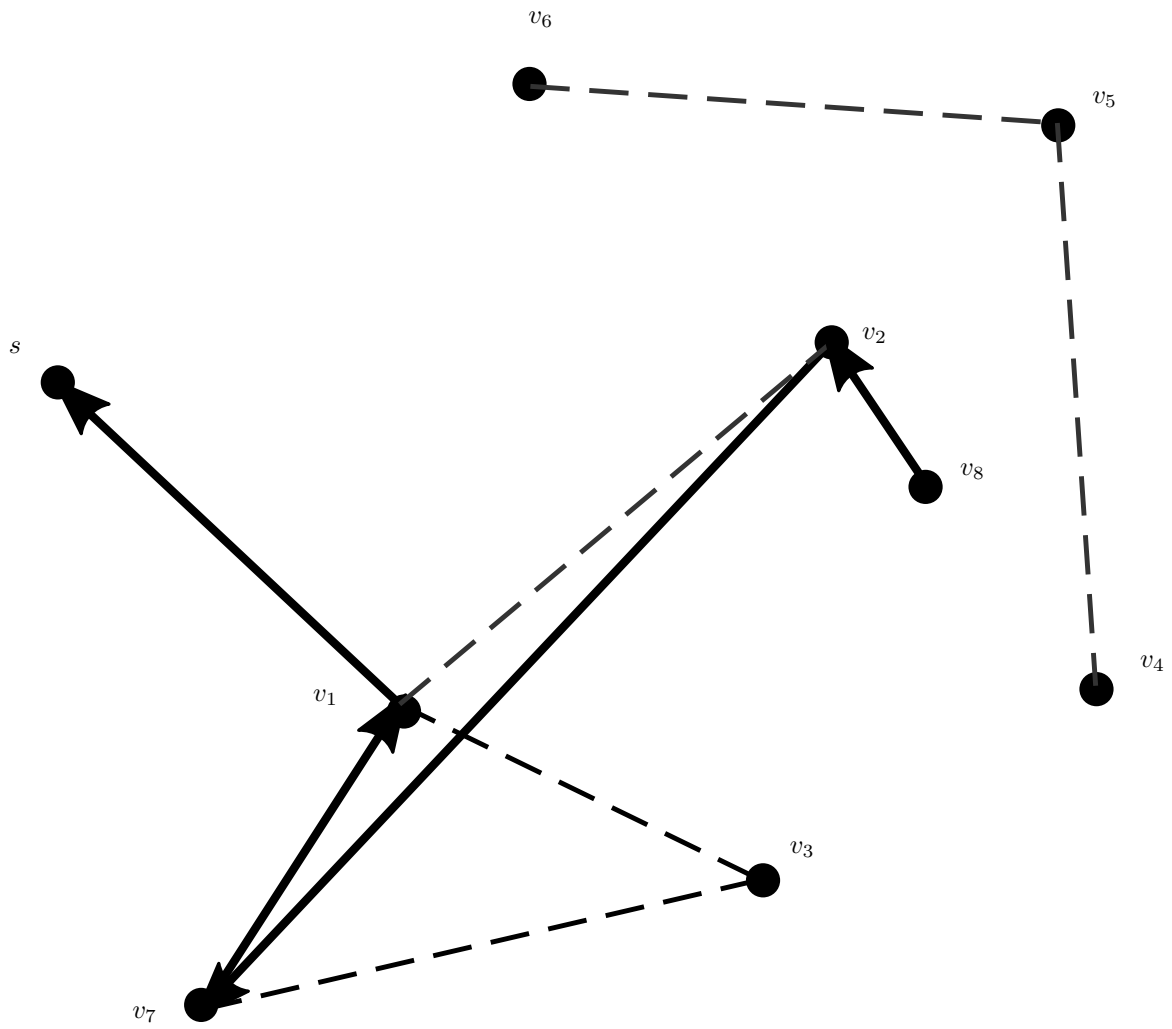

Figure 1: Whatever First Search For $v_8 \to s$

If we use a stack, we have DFS, if we use queue we have BFS, if we have priority queue/heap, we have either Dijkstra's or Prim's algorithm.

Dijkstra's Algorithm

This algorithm utilizes a priority queue in order to find the shortest path from s to any vertex v.

```
Dijkstra(G, s):
    priQ = {(s, 0)}
    while priQ ≠ ∅:
        (v, d) ← priQ.ExtractMin
        if v is not marked:
            mark v
            ∀v → w ∈ E:
                priQ.Insert(w, d + l(v → w))
```

The shortes path problem takes some directed graph $G = (V, E)$ where we have a length function $l : E \to \mathbb{R}^+$, $s \in V$. Then we output the shortest path from s to v for every vertex $v \in V$. The running time of this algorithm is $O(E \log V)$ since our priority queue insert function runs in $O(\log V)$ time. A graph with negative weights does not work, especially so if it is cyclic, because the shortest Dijkstra path may not be the shortest actual path, and if it is cyclic, there may not even exist a shortest path.

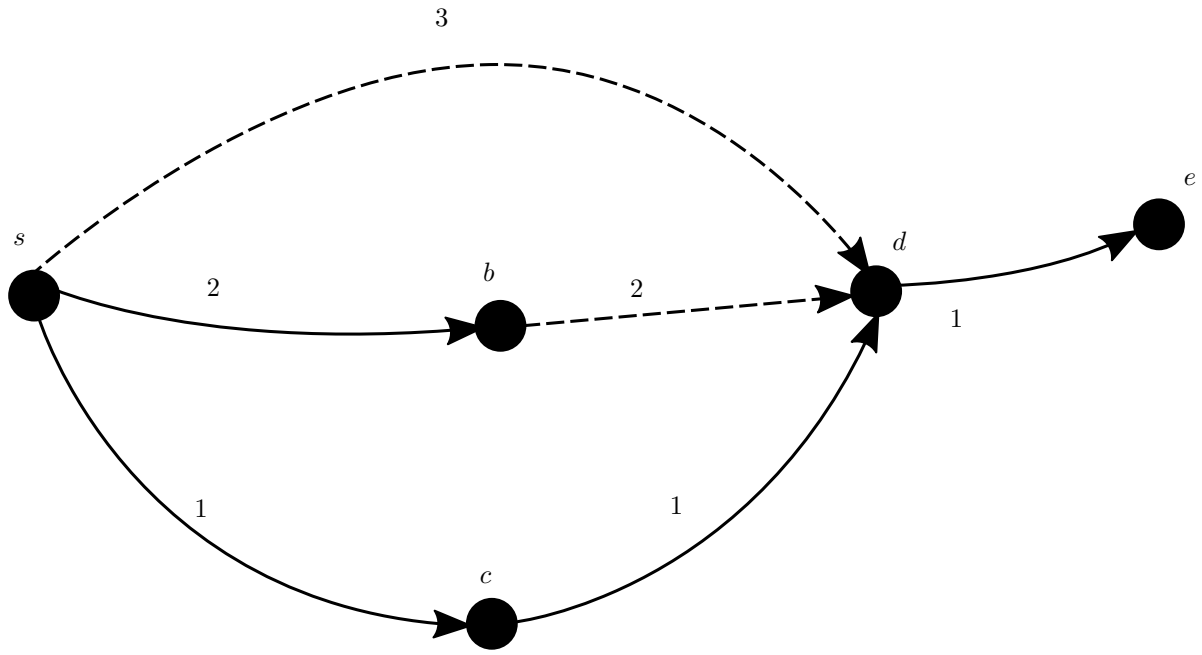

Figure 2: Dijkstra's Algorithm on $G = (V, E)$

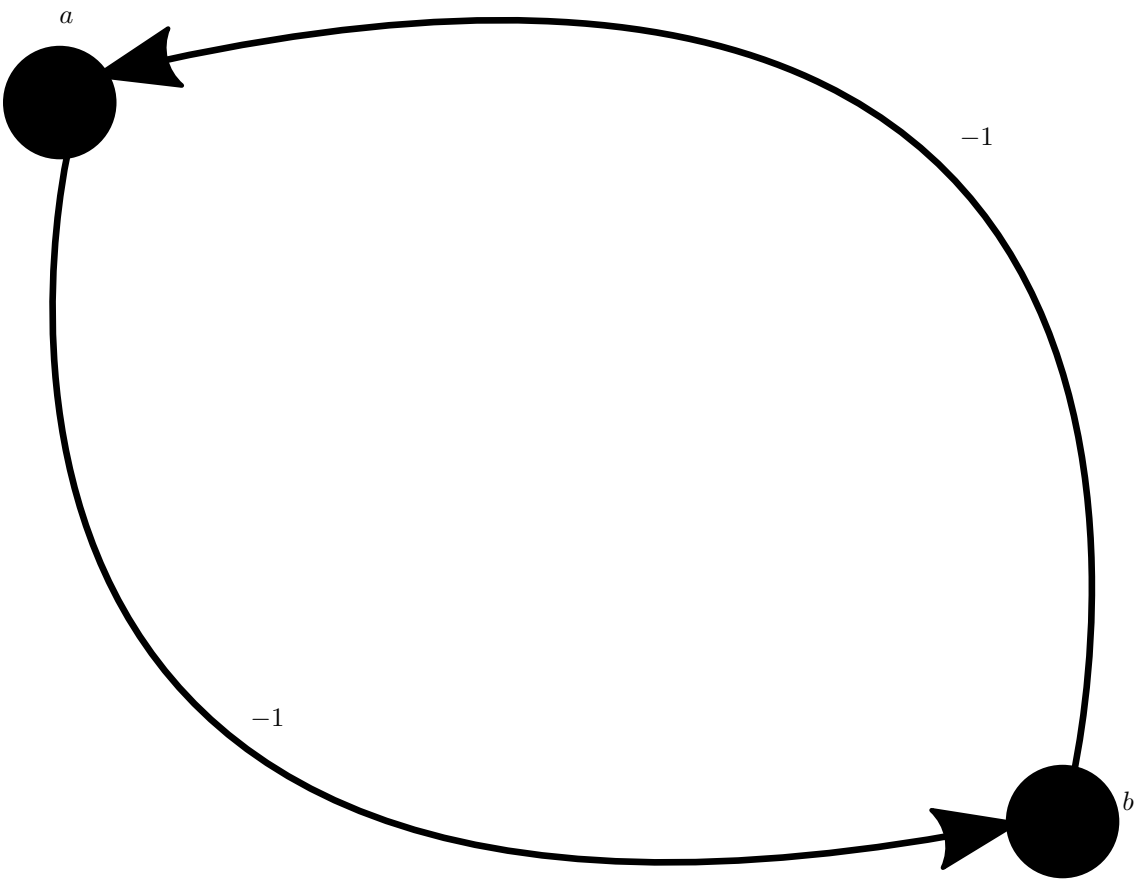

Figure 3: A Negative Cyclic Graph