Breadth First Search

Data Structures and Algorithms (094224)

Yuval Emek

Winter 2022/23

The breadth first search algorithm

breadth first search (BFS) ⟨סריקה לרוחב⟩

- One of the simplest graph algorithms
- Input:
  - (Di)graph $G = (V, E)$
  - A designated *source vertex* ⟨צומת מקור⟩ $s \in V$
- Explores G, discovering every vertex reachable from s
- Vertices discovered in order of their *distance* ⟨מרחק⟩ from s
  - Distance from $x \in V$ to $y \in V$ = length of a shortest (x, y)-path
    - Shortest path is not necessarily unique
- Constructs a *shortest path tree* ⟨עץ מסלולים קצרים ביותר⟩ rooted at s
  - Contains all vertices reachable from s
  - Unique simple (s, v)-path in the tree is a shortest (s, v)-path in G
  - a.k.a. *BFS tree*

Data structures

- $G = (V, E)$ represented using adjacency lists
- Each vertex $v \in V$ maintains additional attributes:
  - $v.color$ = the discovery status of v
    - white = v is still undiscovered
    - gray = v has been discovered but it may still have undiscovered neighbors
    - black = v and all its neighbors have been discovered
  - $v.\pi$ = the parent of v in the BFS tree
  - $v.d$ = the distance from s to v in G
- Queue Q

the v.pi of the root is null

Pseudocode

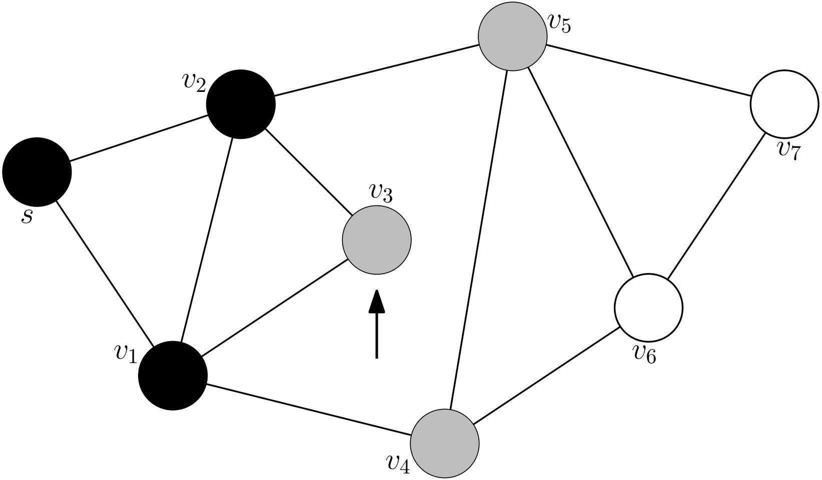
BFS(G, s)

```
BFS_Initialization(G, s, Q)
while Q ≠ ∅ do
    u = Dequeue(Q)
    for all v ∈ G.Adj[u] do
        if v.color == white then
            v.color = gray
            v.d = u.d + 1
            v.π = u
            Enqueue(Q, v)
    u.color = black
```

BFS_Initialization(G, s, Q)

```
for all v ∈ G.V − {s} do
    v.color = white
    v.d = ∞
    v.π = NIL
s.color = gray
s.d = 0
s.π = NIL
Q = ∅
Enqueue(Q, s)
```

Example



Run-time analysis

- Initialization takes $O(n)$ time
- After initialization, BFS never colors a vertex white
- If vertex $v \in V$ passes the test in line 5, then it is colored gray in line 6
- $\Longrightarrow$ every vertex v is enqueued (line 9) at most once
- $\Longrightarrow$ every vertex v is dequeued (line 3) at most once
  - $O(n)$ time for the queue operations
- $\Longrightarrow$ Every adjacency list is traversed at most once
- Total size of all adjacency lists is $O(m)$
- Each entry of each adjacency list accounts for $O(1)$ time
- $\Longrightarrow$ In total, the run-time of BFS is $O(n + m)$

1. Correctness
   - BFS tree

The goal

- (Di)graph $G = (V, E)$ with designated source vertex $s \in V$
- $\delta(x, y)$ = the distance from $x \in V$ to $y \in V$
  - ∞ if y is not reachable from x
- Goal: show that when BFS terminates:
  - $v.d = \delta(s, v)$ for every vertex $v \in V$
  - If $v \neq s$ is reachable from s, then $v.\pi$ is the vertex preceding v along some shortest (s, v)-path in G
- Don't we have to show that every vertex reachable from s is discovered by BFS? because in Goal we checked v.d and v.pi so if goal is correct then each vertex already discovered

Subpath of a shortest path

Lemma

If $P = \langle v_0, v_1, \ldots, v_k \rangle$ is a shortest (v_0, v_k)-path in G, then $P_{i,j} = \langle v_i, v_{i+1}, \ldots, v_j \rangle$ is a shortest (v_i, v_j)-path in G for every $0 \leq i \leq j \leq k$.

- Assume by contradiction that (v_i, v_j)-path Q is shorter than $P_{i,j}$
- Define $P' = \langle v_0, v_1, \ldots, v_i \rangle \cdot Q \cdot \langle v_j, v_{j+1}, \ldots, v_k \rangle$
- P' is a (v_0, v_k)-path in G
- $\mathrm{length}(P') = \mathrm{length}(P) - \mathrm{length}(P_{i,j}) + \mathrm{length}(Q)$
- $\mathrm{length}(Q) < \mathrm{length}(P_{i,j}) \implies \mathrm{length}(P') < \mathrm{length}(P)$ $_{(\rightarrow\leftarrow)}$
- $\implies$ Assumption must be wrong $\blacksquare$

Distance to adjacent vertices

Lemma

If $(u, v) \in E$, then $\delta(s, v) \leq \delta(s, u) + 1$. true if u,v is as in the bfs code.

- Clearly holds if u is not reachable from s and $\delta(s, u) = \infty$
- Let P be a shortest (s, u)-path in G
- $\text{length}(P \cdot v) = \text{length}(P) + 1$
- $\Longrightarrow$ there exists an (s, v)-path of length $\delta(s, u) + 1$
- $\Longrightarrow \delta(s, v) \leq \delta(s, u) + 1$ ■

Upper bound on the distance

Lemma

Upon termination of BFS, $v.d \geq \delta(s, v)$ for all $v \in V$.

- We show that the assertion holds immediately after each enqueue operation (line 9) by induction on #enqueue operations
- Base: Immediately after enqueuing s, $s.d = 0 = \delta(s, s)$ and $v.d = \infty \geq \delta(s, v)$ for all $v \in V - \{s\}$
- Step: Consider a white vertex v discovered from vertex u
  - $v.d$ is the only d attribute that changes until next enqueue operation
  - $\Longrightarrow$ by ind. hyp., $w.d \geq \delta(s, w)$ for all $w \in V - \{v\}$ upon next enqueue operation
- In line 7, we set $v.d = u.d + 1 \geq \delta(s, u) + 1$
  - $\geq$ holds due to the ind. hyp.
  - This remains the situation when v is enqueued in line 9
- By previous lemma, $\delta(s, v) \leq \delta(s, u) + 1$ since $(u, v) \in E$ ∎

The queue

Lemma

If during the execution $Q = \langle v_1, \ldots, v_r \rangle$*, where* v_1 *is the head and* v_r *is the tail, then* $v_1.d \leq v_2.d \leq \cdots \leq v_r.d \leq v_1.d + 1$.

- We show that the assertion holds immediately after each queue operation (lines 3 and 9) by induction on #queue operations
- Base: Clearly holds immediately after enqueuing s when $Q = \langle s \rangle$
- Step, dequeue operations: If v_1 is dequeued, then $Q = \langle v_2, \ldots, v_r \rangle$ and the assertion holds by the ind. hyp.

Proof cont.

- Step, enqueue operations: Consider a white vertex v discovered from vertex u
  - v will soon be enqueued (line 9) and become the tail v_{r+1}
- u was the last dequeued vertex
- We argue that $u.d \leq v_1.d$
  - If v_1 was discovered from u, then $v_1.d = u.d + 1 > u.d$
  - Else, u and v_1 co-existed in Q, thus $u.d \leq v_1.d$ by the ind. hyp.
- In line 7, we set $v.d = u.d + 1$
  - This remains the situation when v is enqueued
- $\Longrightarrow v_{r+1}.d = v.d = u.d + 1 \leq v_1.d + 1$
- We argue that $v_r.d \leq u.d + 1$
  - If v_r was discovered from u, then $v_r.d = u.d + 1$
  - Else, u and v_r co-existed in Q (right before u was dequeued), thus $v_r.d \leq u.d + 1$ by the ind. hyp.
- $\Longrightarrow v_r.d \leq v.d = v_{r+1}.d$ ■

Enqueuing order

מסקנה

Corollary

If u is enqueued before v, then at all times, $u.d \leq v.d$.

- Follows directly from the last lemma and the fact that the d attribute of a vertex is set once

Lower bound on the distance

Lemma

Upon termination of BFS, $v.d \leq \delta(s, v)$ for every vertex $v \in V$ reachable from s.

- Assume by contradiction that upon termination of BFS, $v.d > \delta(s, v)$ for some vertex v reachable from s
  - Clearly, $v \neq s$
- Take v to be such a vertex that minimizes $\delta(s, v)$
- Let u be the vertex preceding v along some shortest (s, v)-path in G
  - $\delta(s, v) = \delta(s, u) + 1$

מקב

- By the choice of v, $u.d \leq \delta(s, u)$ since $\delta(s, u) < \delta(s, v)$
- $\implies v.d > \delta(s, v) = \delta(s, u) + 1 \geq u.d + 1$

Proof cont.

- Consider time t when u was dequeued (line 3)
- If v is white at time t, then it is discovered from u, hence $v.d = u.d + 1$ $_{(\rightarrow\leftarrow)}$
- If v is gray at time t, then it was discovered from vertex w that was dequeued before time t and $v.d = w.d + 1$
- By the corollary, $w.d \leq u.d$, hence $v.d \leq u.d + 1$ $_{(\rightarrow\leftarrow)}$
- If v is black at time t, then it was dequeued before u
- By the corollary, $v.d \leq u.d$ $_{(\rightarrow\leftarrow)}$
- $\Longrightarrow$ Assumption must be wrong $\blacksquare$

Establish correctness

Theorem

Upon termination of BFS, $v.d = \delta(s, v)$ for every vertex $v \in V$. Moreover, if $v \neq s$ is reachable from s, then $v.\pi$ is the vertex preceding v along some shortest (s, v)-path in G.

- $v.d = \delta(s, v)$ follows from two earlier lemmas 4, 15
- If $v.\pi = u$, then $v.d = u.d + 1$ (see lines 7 and 8)
  - $\Longrightarrow \delta(s, v) = \delta(s, u) + 1$
- A shortest (s, v)-path in G can be constructed by augmenting a shortest (s, u)-path in G with the edge (u, v) ■

1. Correctness
   - BFS tree

Shortest path tree

Run BFS on some (di)graph $G = (V, E)$ with source vertex $s \in V$

Lemma

The (undirected) graph $T = (U, F)$ defined by taking

$$U = \{v \in V \mid v.\pi \neq NIL\} \cup \{s\} \quad \textit{and} \quad F = \{(v, v.\pi) \mid v \in U - \{s\}\}$$

is a shortest path tree for s in G.

- T is a tree:
  - $|F| = |U| - 1$ by the construction of F
  - By the theorem, $P_v = \langle v, v.\pi, v.\pi.\pi, \ldots, s \rangle$ is well defined for every $v \in U$
    - Forms a (v, s)-path in T
  - $\Longrightarrow$ T is connected
- $(v.\pi, v) \in E$ for every $v \in V$, $v.\pi \neq NIL$

הפכנו כיוונים
שוויון נכון גם בגרף מכוון

- $\Longrightarrow$ $\overleftarrow{P}_v$ is an (s, v)-path in G
  - The theorem guarantees that this is a shortest path ∎