Breadth-First Search

Breadth-First Search

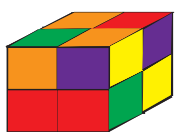
Graph Search

"Explore a graph", e.g.:

- find a path from start vertex s to a desired vertex
- visit all vertices or edges of graph, or only those reachable from s

Pocket Cube:

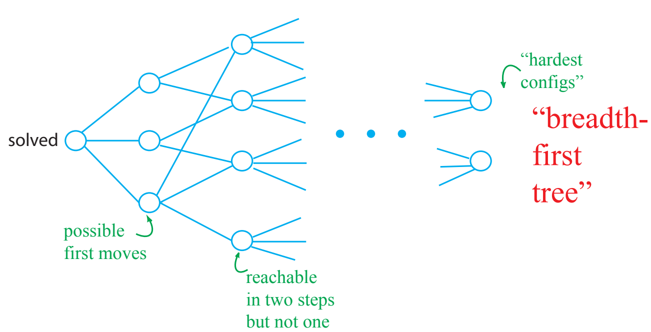
Consider a $2 \times 2 \times 2$ Rubik's cube



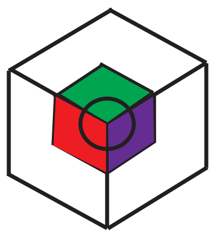
Configuration Graph:

- vertex for each possible state
- edge for each basic move (e.g., 90 degree turn) from one state to another
- undirected: moves are reversible

Diameter ("God's Number")
11 for $2 \times 2 \times 2$, 20 for $3 \times 3 \times 3$, $\Theta(n^2/\lg n)$ for $n \times n \times n$ [Demaine, Demaine, Eisenstat Lubiw Winslow 2011]



vertices $= 8! \cdot 3^8 = 264,539,520$ where 8! comes from having 8 cubelets in arbitrary positions and 3^8 comes as each cubelet has 3 possible twists.



This can be divided by 24 if we remove cube symmetries and further divided by 3 to account for actually reachable configurations (there are 3 connected components).

Breadth-First Search

Explore graph level by level from s

- level $0 = \{s\}$
- level $i =$ vertices reachable by path of i edges but not fewer

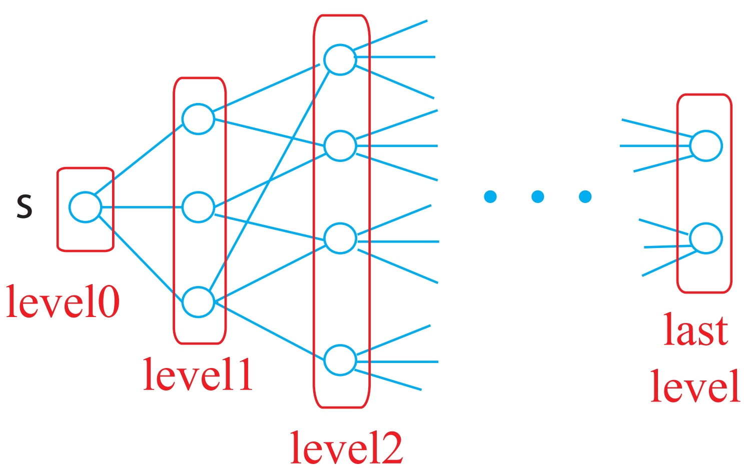


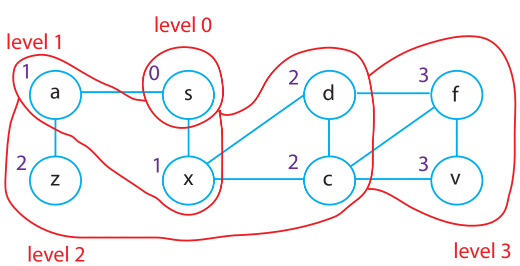
Figure : Illustrating Breadth-First Search

- build level $i > 0$ from level $i - 1$ by trying all outgoing edges, but ignoring vertices from previous levels

Breadth-First-Search Algorithm

```
BFS (V,Adj,s):                              See CLRS for queue-based implementation
    level = { s: 0 }
    parent = {s : None }
    i = 1
    frontier = [s]                          # previous level, i − 1
    while frontier:
        next = [ ]                          # next level, i
        for u in frontier:
            for v in Adj [u]:
                if v not in level:          # not yet seen
                    level[v] = i            ♯ = level[u] + 1
                    parent[v] = u
                    next.append(v)
        frontier = next
        i + =1
```

Example



$\text{frontier}_0 = \{s\}$
$\text{frontier}_1 = \{a, x\}$
$\text{frontier}_2 = \{z, d, c\}$
$\text{frontier}_3 = \{f, v\}$
(not x, c, d)

Figure : Breadth-First Search Frontier

Analysis:

- vertex V enters next (& then frontier) only once (because level$[v]$ then set)

  base case: $v = s$

- $\implies$ Adj[v] looped through only once

$$\text{time} \ = \sum_{v \in V} |Adj[V]| = \begin{cases} |E| \text{ for directed graphs} \\ 2|E| \text{ for undirected graphs} \end{cases}$$

- $\implies$ $O(E)$ time

- $O(V+E)$ ("LINEAR TIME") to also list vertices unreachable from v (those still not assigned level)

Shortest Paths:

- for every vertex v, fewest edges to get from s to v is

$$\begin{cases} \text{level}[v] \text{ if } v \text{ assigned level} \\ \infty \qquad \text{else (no path)} \end{cases}$$

- parent pointers form shortest-path tree = union of such a shortest path for each v
$\implies$ to find shortest path, take v, parent[v], parent[parent[v]], etc., until s (or None)