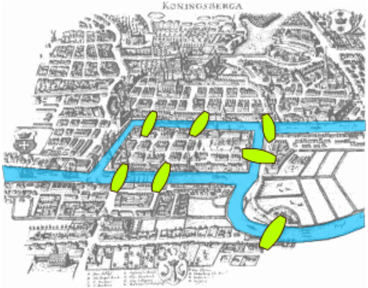
Motivation

Königsberg Seven Bridges Problem

In Königsberg, there were two islands connected to each other and the mainland by seven bridges, as shown in the figure below.



Question:
Is it possible to take a walk and cross over each bridge exactly once?

Euler showed that it is not possible, but he proved it?

(image source:
`https://simple.wikipedia.org/wiki/Seven_Bridges_of_K%C3%B6nigsberg`)

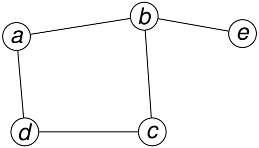
CPT108 Data Structures and Algorithms

Lecture 22

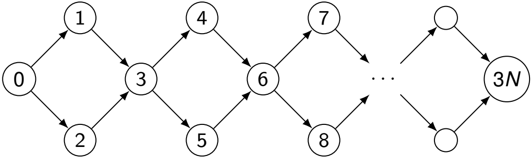
Graph Traversal: Breadth First Search

Traversing a Graph

- Many algorithms on graphs depend on traversing all or some nodes
  - i.e., given a graph representation and a vertex *s* in the graph, find *ALL* paths from *s* to other vertices
- Can we use recursion when traversing a graph?
  - Possible, but with caution due to *cycle*



- Even in acyclic graphs, can get combinatorial explosions:



Treat “0” as the root and do recursive traversal down the two edges out of each node: $O(2^N)$ operations!

- So, typically try to visit each node constant number of times (e.g., once)

Traversing a Graph (cont.)

- Two common graph traversal algorithms
  - Depth first search (DFS)
  - Breadth first search (BFS)

Breadth First Search (BFS)

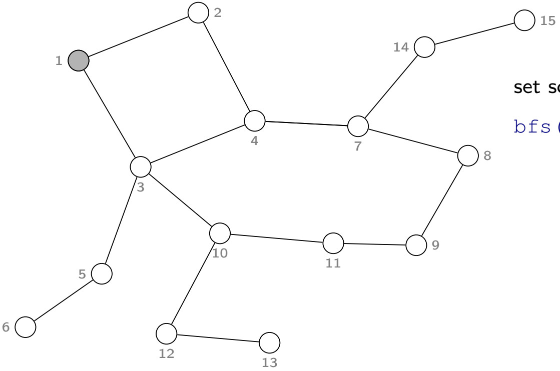
Applications

- Find shortest path between nodes in *unweighted graph*
- Cycle detection in *undirected graph*
- GPS navigation for neighboring locations
- Find person in social networks
- Devices connected to a particular network
- Crawlers in Search Engines

Breadth First Search (BFS)

Main concept

- Visit other nodes at increasing distances from a source node s
  - What do we mean by "distances"?
    - the number of edges on a path from s



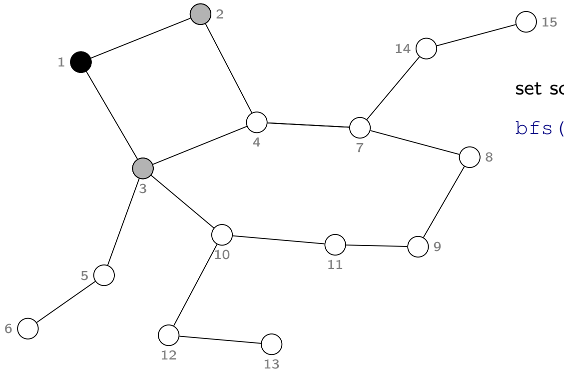
set source $s = \{1\}$

`bfs(s)` $=\{1\}$

Breadth First Search (BFS)

Main concept

- Visit other nodes at increasing distances from a source node s
  - What do we mean by “distances”?
    - the number of edges on a path from s



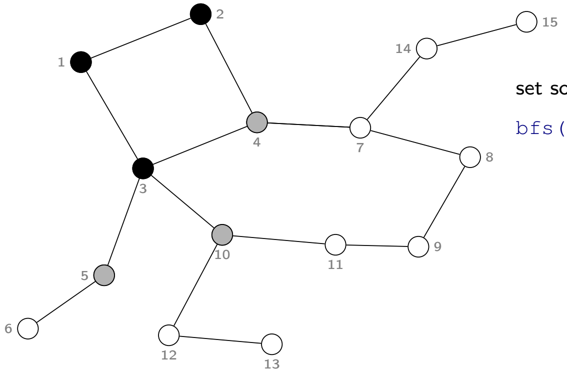
set source $s = \{1\}$

`bfs(s)` $= \{\ 1,$
$2, 3\ //d = 1$
$\}$

Breadth First Search (BFS)

Main concept

- Visit other nodes at increasing distances from a source node s
  - What do we mean by “distances”?
    - the number of edges on a path from s



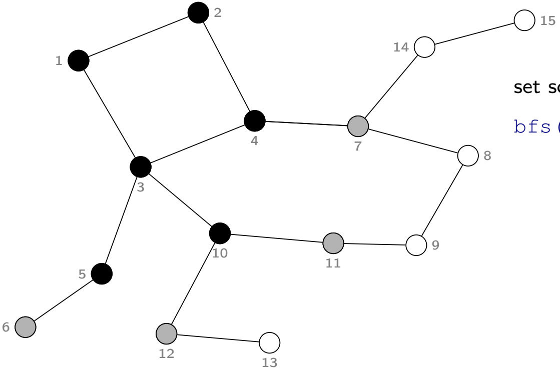
set source $s = \{1\}$

`bfs(s)` $= \{\ 1,$
$2, 3,\quad //d = 1$
$4, 5, 10\ //d = 2$
$\}$

Breadth First Search (BFS)

Main concept

- Visit other nodes at increasing distances from a source node s
  - What do we mean by "distances"?
    - the number of edges on a path from s



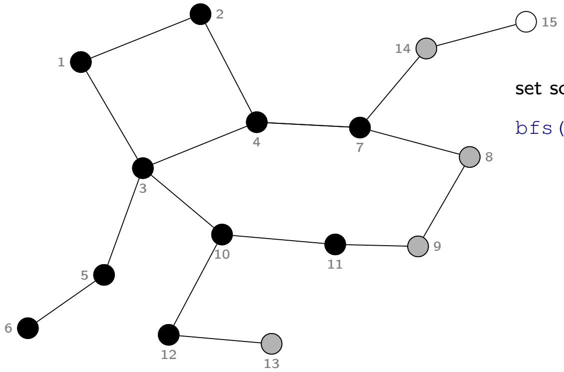
set source $s = \{1\}$

$$
\texttt{bfs(s)} = \{\ 1,\\
2, 3, \quad //d = 1\\
4, 5, 10, \quad //d = 2\\
6, 7, 11, 12 \quad //d = 3\\
\}
$$

Breadth First Search (BFS)

Main concept

- Visit other nodes at increasing distances from a source node s
  - What do we mean by "distances"?
    - the number of edges on a path from s



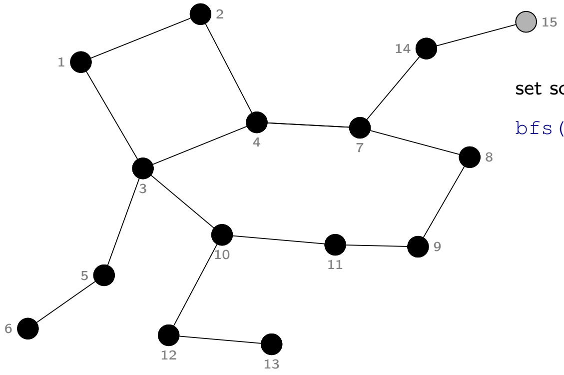
set source $s = \{1\}$

$$\texttt{bfs(s)} = \{ \begin{array}{ll} 1, & \\ 2, 3, & //d = 1 \\ 4, 5, 10, & //d = 2 \\ 6, 7, 11, 12, & //d = 3 \\ 8, 9, 13, 14 & //d = 4 \\ \} & \end{array}$$

Breadth First Search (BFS)

Main concept

- Visit other nodes at increasing distances from a source node s
  - What do we mean by "distances"?
    - the number of edges on a path from s



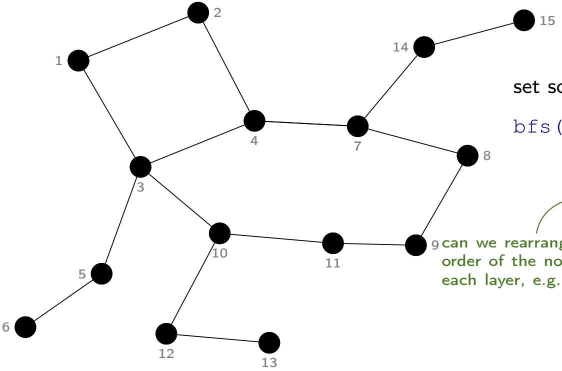
set source $s = \{1\}$

`bfs(s)` $= \{$ 1,
2, 3, $//d = 1$
4, 5, 10, $//d = 2$
6, 7, 11, 12, $//d = 3$
8, 9, 13, 14, $//d = 4$
15 $//d = 5$
$\}$

Breadth First Search (BFS)

Main concept

- Visit other nodes at increasing distances from a source node s
  - What do we mean by “distances”?
    - the number of edges on a path from s



set source $s = \{1\}$

`bfs(s)` $= \{$ 1,
2, 3, $//d = 1$
4, 5, 10, $//d = 2$
6, 7, 11, 12, $//d = 3$
8, 9, 13, 14, $//d = 4$
15 $//d = 5$
$\}$

can we rearrange the order of the nodes at each layer, e.g., here?

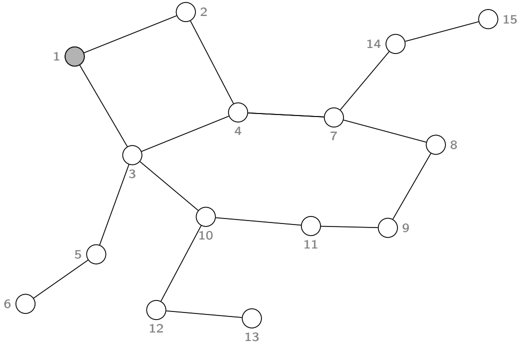
$\Rightarrow$ BFS completed!

Breadth First Search (BFS)

Main concept

- Visit other nodes at increasing distances from a source node s
  - What do we mean by "distances"?
    - the number of edges on a path from s

set source $s = \{1\}$

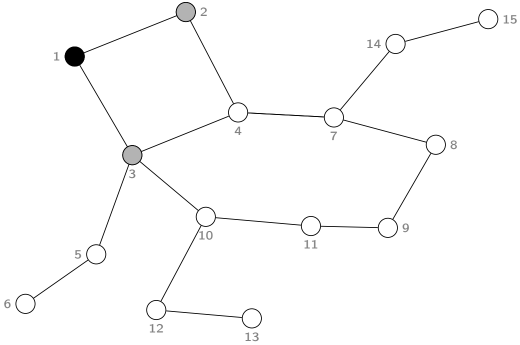


Breadth First Search (BFS)

Main concept

- Visit other nodes at increasing distances from a source node s
  - What do we mean by "distances"?
    - the number of edges on a path from s

set source $s = \{1\}$

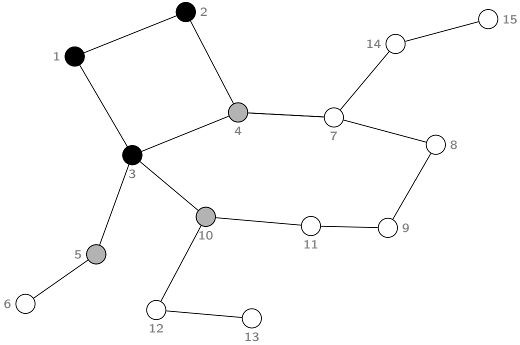


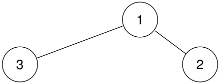
Breadth First Search (BFS)

Main concept

- Visit other nodes at increasing distances from a source node s
  - What do we mean by “distances”?
    - the number of edges on a path from s

set source $s = \{1\}$



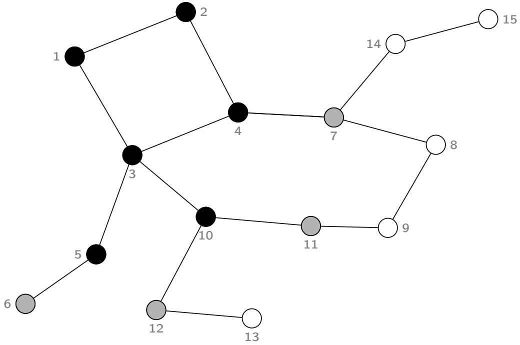


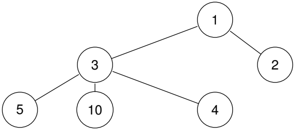
Breadth First Search (BFS)

Main concept

- Visit other nodes at increasing distances from a source node s
  - What do we mean by "distances"?
    - the number of edges on a path from s

set source $s = \{1\}$



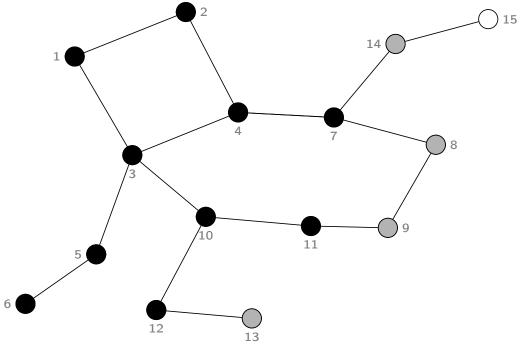


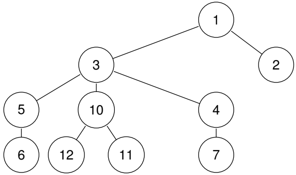
Breadth First Search (BFS)

Main concept

- Visit other nodes at increasing distances from a source node s
  - What do we mean by “distances”?
    - the number of edges on a path from s

set source $s = \{1\}$



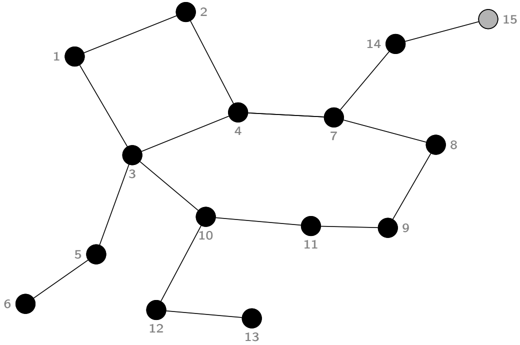


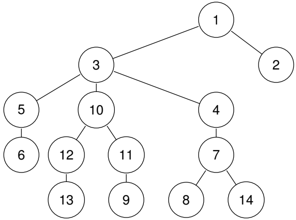
Breadth First Search (BFS)

Main concept

- Visit other nodes at increasing distances from a source node *s*
    - What do we mean by “distances”?
        - the number of edges on a path from *s*

set source $s = \{1\}$

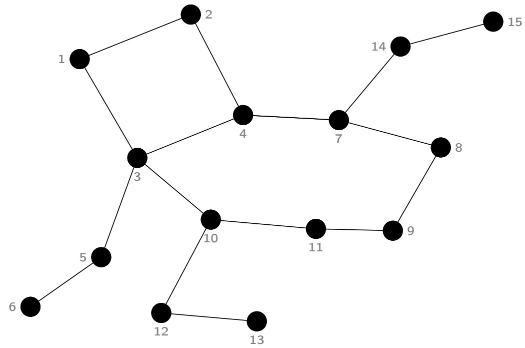




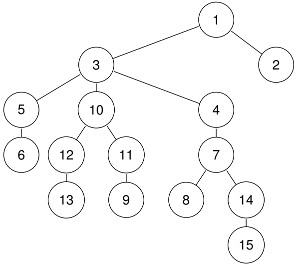
Breadth First Search (BFS)

Main concept

- Visit other nodes at increasing distances from a source node s
  - What do we mean by "distances"?
    - the number of edges on a path from s



set source $s = \{1\}$



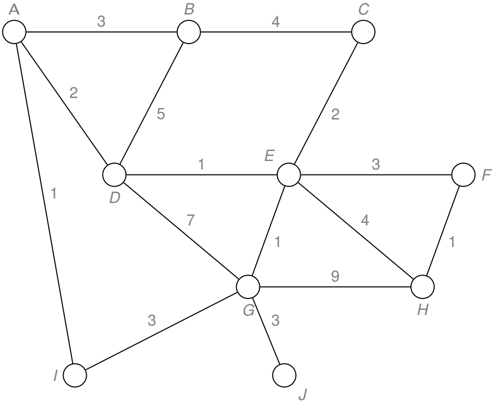
BFS tree / *BFS forest*

- What would a "Level" order traversal tell you?

Breadth First Search (BFS) (cont.)

Exercise

Report on the order of the vertices encountered on a BFS starting from vertex A. Break all ties by picking the vertices in alphabetic order (i.e., A before Z).



Breadth First Search (BFS) (cont.)

Pseudocode

1. **Initialization**: Enqueue the starting node into a queue and mark it as visited
2. **Exploration**: While the queue is not empty
    - Dequeue a node from the queue and visit it (e.g., print its value)
    - For each unvisited neighbor of the dequeued node:
        - Enqueue the neighbor into the queue
        - Mark the neighbor as visited
3. **Termination**: Repeat Step 2 until the queue is empty

Recap: Adjacency list vs Adjacency matrix

Given a graph $G = (V, E)$:
n = number of vertex, and
m = number of edges

- **Adjacency list**
  - More compact than adjacency matrix if graph has few edges
  - Requires a scan of adjacency list to check if an edge exists
  - Requires a scan to obtain all edges!
- **Adjacency matrix**
  - Always require n^2 space
    - This can waste a lot of space if the number of edges are sparse
  - Find if an edge exist if $O(1)$
  - Obtain all edges in $O(n^2)$

Breadth First Search (BFS) (cont.)

Time complexity

Using adjacency list

(n = number of vertex
and m = number of edges)

Recall: $\Sigma_{v \in V} \texttt{deg}(v) = 2m$

Total running time of the while loop

$= \Sigma_{v \in V} (\texttt{deg}(v) + 1)$

$= \Sigma_{v \in V} \texttt{deg}(v) + \Sigma_{v \in V} 1$

$= O(2m + n)$

$= O(m + n)$ (or $O(|E| + |V|)$)

```
BFS(G, s)
for each vertex u ∈ G.V
    flag[u] = false
Q = empty queue
flag[s] = true
ENQUEUE(Q, s)
while Q is not empty
    v = DEQUEUE(Q)
    for each w adjacent to v
        if flag[w] = false
            flag[w] = true
            ENQUEUE(Q, w)
```

Each vertex will enter Q at most once

Each iteration takes time proportional to $\texttt{deg}(v) + 1$ (the number 1 is to account for the case where $\texttt{deg}(v) = 0$ — the work required is 1, not (0).

Breadth First Search (BFS) (cont.)

Time complexity

Using adjacency matrix

(n = number of vertex
and m = number of edges)

BFS(G, s)

```
for each vertex u ∈ G.V
    flag[u] = false
Q =empty queue
flag[s] = true
ENQUEUE(Q, s)
while Q is not empty
    v = DEQUEUE(Q)
    for each w adjacent to v
        if flag[w] = false
            flag[w] = true
            ENQUEUE(Q, w)
```

Each vertex will enter Q at most once

Finding the adjacent vertices of v requires checking all elements in the row, which takes $O(n)$.

Total running time of the while loop

$= \Sigma_{v \in V}(n)$

$= O(n \times n)$

$= O(n^2)$ (or $O(|V|^2)$)

which is independent to the number of edges m

Reading

- Chapter 20, Cormen (2022)

References I

Geeksforgeeks.org (2024). *Breadth First Search or BFS for a Graph*. Online: https://www.geeksforgeeks.org/breadth-first-search-or-bfs-for-a-graph. [last accessed: 20 Mar 2024].

Programiz (2024). *Breadth first search*. Online: https://www.programiz.com/dsa/graph-bfs. [last accessed: 20 Mar 2024].