

Breadth First Search

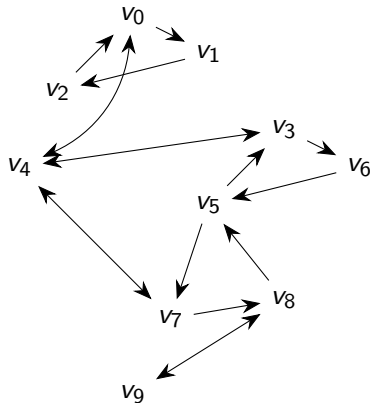
Madhavan Mukund

<https://www.cmi.ac.in/~madhavan>

Mathematics for Data Science 1
Week 10

Reachability in a graph

- Mark source vertex as reachable
- Systematically mark neighbours of marked vertices
- Stop when target becomes marked



Reachability in a graph

- Mark source vertex as reachable
- Systematically mark neighbours of marked vertices
- Stop when target becomes marked
- Choose an appropriate representation
 - Adjacency matrix
 - Adjacency list

	0	1	2	3	4	5	6	7	8	9
0	0	1	0	0	1	0	0	0	0	0
1	0	0	1	0	0	0	0	0	0	0
2	1	0	0	0	0	0	0	0	0	0
3	0	0	0	0	1	0	1	0	0	0
4	1	0	0	1	0	0	0	1	0	0
5	0	0	0	1	0	0	0	1	0	0
6	0	0	0	0	0	1	0	0	0	0
7	0	0	0	0	1	0	0	0	1	0
8	0	0	0	0	0	1	0	0	0	1
9	0	0	0	0	0	0	0	0	1	0

0	{1,4}
1	{2}
2	{0}
3	{4,6}
4	{0,3,7}

5	{3,7}
6	{5}
7	{4,8}
8	{5,9}
9	{8}

Reachability in a graph

- Mark source vertex as reachable
- Systematically mark neighbours of marked vertices
- Stop when target becomes marked
- Choose an appropriate representation
 - Adjacency matrix
 - Adjacency list
- Strategies for systematic exploration
 - Breadth first — propagate marks in “layers”
 - Depth first — explore a path till it dies out, then backtrack

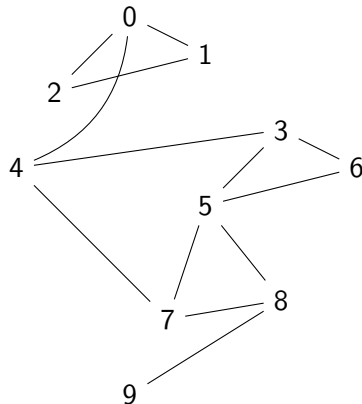
	0	1	2	3	4	5	6	7	8	9
0	0	1	0	0	1	0	0	0	0	0
1	0	0	1	0	0	0	0	0	0	0
2	1	0	0	0	0	0	0	0	0	0
3	0	0	0	0	1	0	1	0	0	0
4	1	0	0	1	0	0	0	1	0	0
5	0	0	0	1	0	0	0	1	0	0
6	0	0	0	0	0	1	0	0	0	0
7	0	0	0	0	1	0	0	0	1	0
8	0	0	0	0	0	1	0	0	0	1
9	0	0	0	0	0	0	0	0	1	0

0	{1,4}
1	{2}
2	{0}
3	{4,6}
4	{0,3,7}

5	{3,7}
6	{5}
7	{4,8}
8	{5,9}
9	{8}

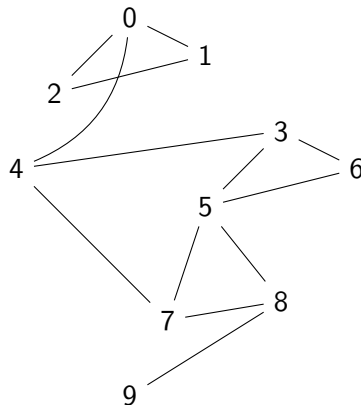
Breadth first search (BFS)

- Explore the graph level by level
 - First visit vertices one step away
 - Then two steps away
 - ...



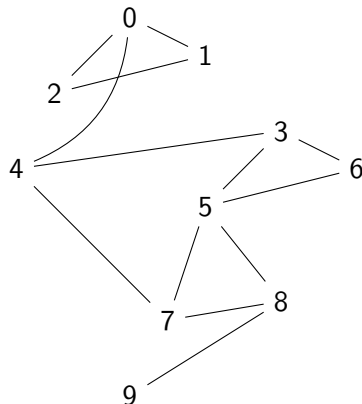
Breadth first search (BFS)

- Explore the graph level by level
 - First visit vertices one step away
 - Then two steps away
 - ...
- Each **visited** vertex has to be **explored**
 - Extend the search to its neighbours
 - Do this only once for each vertex!



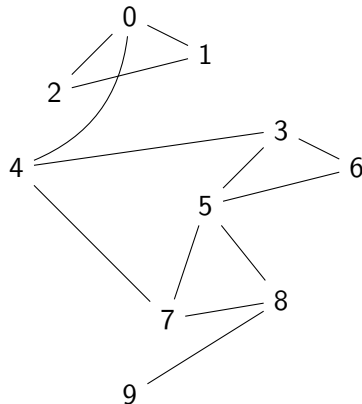
Breadth first search (BFS)

- Explore the graph level by level
 - First visit vertices one step away
 - Then two steps away
 - ...
- Each **visited** vertex has to be **explored**
 - Extend the search to its neighbours
 - Do this only once for each vertex!
- Maintain information about vertices
 - Which vertices have been visited already
 - Among these, which are yet to be explored



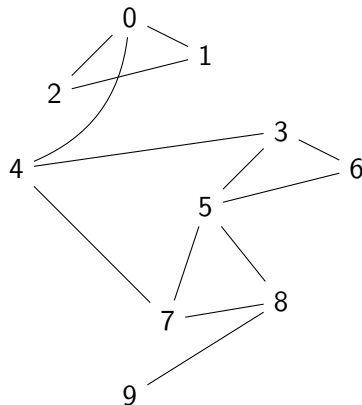
Breadth first search (BFS) ...

- Assume $V = \{0, 1, \dots, n-1\}$



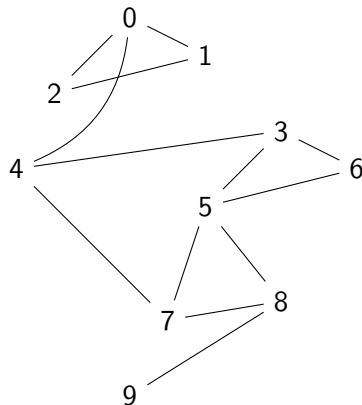
Breadth first search (BFS) ...

- Assume $V = \{0, 1, \dots, n-1\}$
- $\text{visited} : V \rightarrow \{\text{True}, \text{False}\}$ tells us whether $v \in V$ has been visited
 - Initially, $\text{visited}(v) = \text{False}$ for all $v \in V$



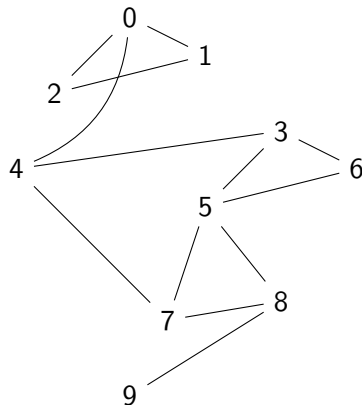
Breadth first search (BFS) ...

- Assume $V = \{0, 1, \dots, n-1\}$
- $\text{visited} : V \rightarrow \{\text{True}, \text{False}\}$ tells us whether $v \in V$ has been visited
 - Initially, $\text{visited}(v) = \text{False}$ for all $v \in V$
- Maintain a sequence of visited vertices yet to be explored
 - A **queue** — first in, first out
 - Initially empty



Breadth first search (BFS) ...

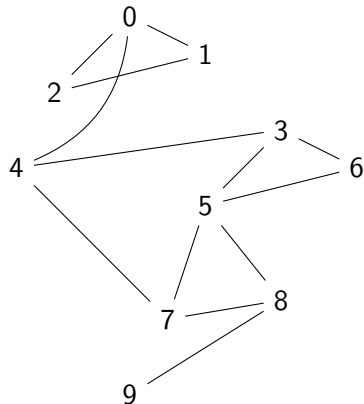
- Assume $V = \{0, 1, \dots, n-1\}$
- $\text{visited} : V \rightarrow \{\text{True}, \text{False}\}$ tells us whether $v \in V$ has been visited
 - Initially, $\text{visited}(v) = \text{False}$ for all $v \in V$
- Maintain a sequence of visited vertices yet to be explored
 - A **queue** — first in, first out
 - Initially empty
- Exploring a vertex i
 - For each edge (i, j) , if $\text{visited}(j)$ is **False**,
 - Set $\text{visited}(j)$ to **True**
 - Append j to the queue



Breadth first search (BFS) ...

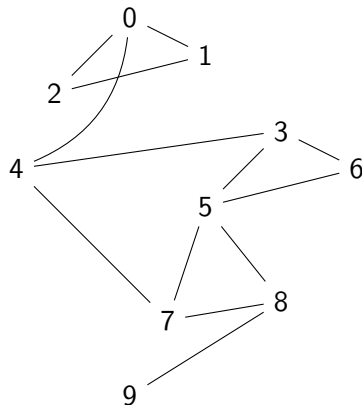
- Initially

- $\text{visited}(v) = \text{False}$ for all $v \in V$
- Queue of vertices to be explored is empty



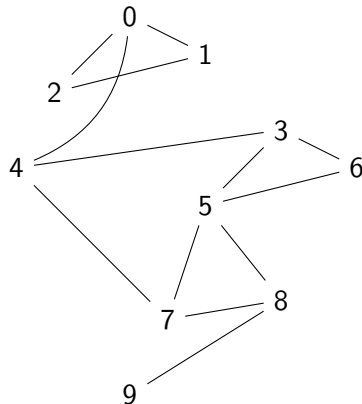
Breadth first search (BFS) ...

- Initially
 - $\text{visited}(v) = \text{False}$ for all $v \in V$
 - Queue of vertices to be explored is empty
- Start BFS from vertex j
 - Set $\text{visited}(j) = \text{True}$
 - Add j to the queue



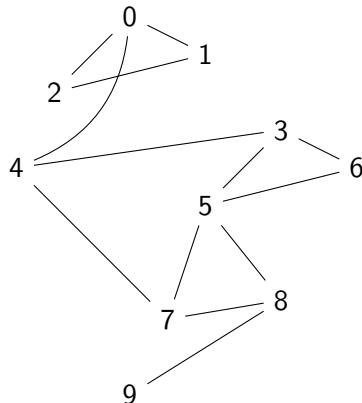
Breadth first search (BFS) ...

- Initially
 - $\text{visited}(v) = \text{False}$ for all $v \in V$
 - Queue of vertices to be explored is empty
- Start BFS from vertex j
 - Set $\text{visited}(j) = \text{True}$
 - Add j to the queue
- Remove and explore vertex i at head of queue
 - For each edge (i, j) , if $\text{visited}(j)$ is **False**,
 - Set $\text{visited}(j)$ to **True**
 - Append j to the queue



Breadth first search (BFS) ...

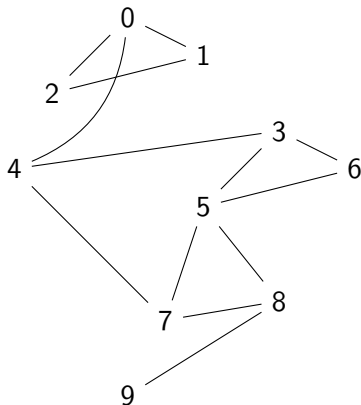
- Initially
 - $\text{visited}(v) = \text{False}$ for all $v \in V$
 - Queue of vertices to be explored is empty
- Start BFS from vertex j
 - Set $\text{visited}(j) = \text{True}$
 - Add j to the queue
- Remove and explore vertex i at head of queue
 - For each edge (i, j) , if $\text{visited}(j)$ is **False**,
 - Set $\text{visited}(j)$ to **True**
 - Append j to the queue
- Stop when queue is empty



BFS from vertex 7

Visited	
0	False
1	False
2	False
3	False
4	False
5	False
6	False
7	False
8	False
9	False

To explore queue									

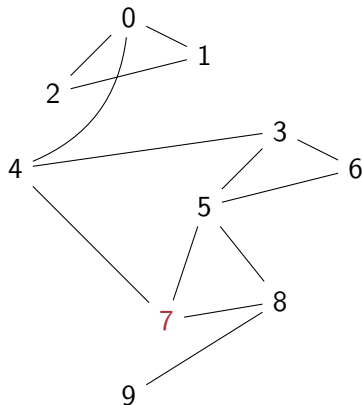


BFS from vertex 7

Visited	
0	False
1	False
2	False
3	False
4	False
5	False
6	False
7	True
8	False
9	False

To explore queue									
7									

- Mark 7 and add to queue

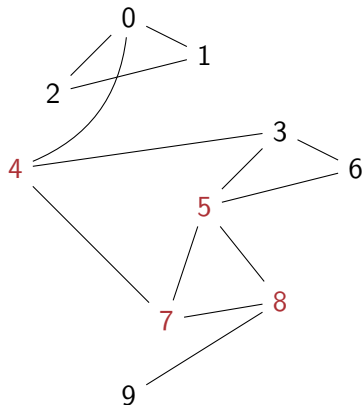


BFS from vertex 7

Visited	
0	False
1	False
2	False
3	False
4	True
5	True
6	False
7	True
8	True
9	False

To explore queue								
4	5	8						

- Mark 7 and add to queue
- Explore 7, visit {4,5,8}

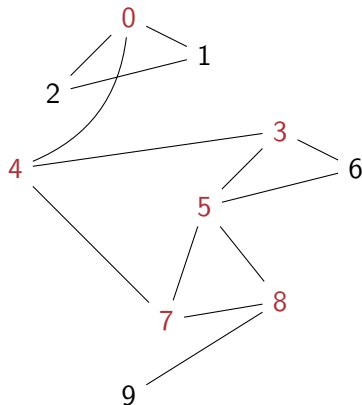


BFS from vertex 7

Visited	
0	True
1	False
2	False
3	True
4	True
5	True
6	False
7	True
8	True
9	False

To explore queue								
5	8	0	3					

- Mark 7 and add to queue
- Explore 7, visit {4,5,8}
- Explore 4, visit {0,3}

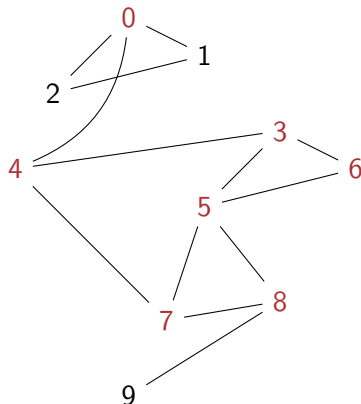


BFS from vertex 7

Visited	
0	True
1	False
2	False
3	True
4	True
5	True
6	True
7	True
8	True
9	False

To explore queue								
8	0	3	6					

- Mark 7 and add to queue
- Explore 7, visit {4,5,8}
- Explore 4, visit {0,3}
- Explore 5, visit {6}

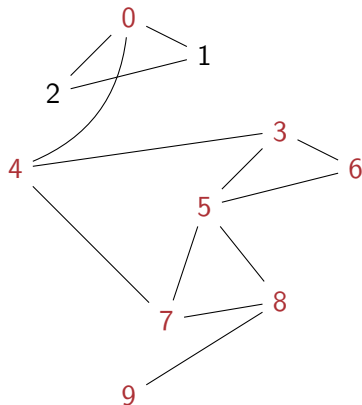


BFS from vertex 7

Visited	
0	True
1	False
2	False
3	True
4	True
5	True
6	True
7	True
8	True
9	True

To explore queue								
0	3	6	9					

- Mark 7 and add to queue
- Explore 7, visit {4,5,8}
- Explore 4, visit {0,3}
- Explore 5, visit {6}
- Explore 8, visit {9}

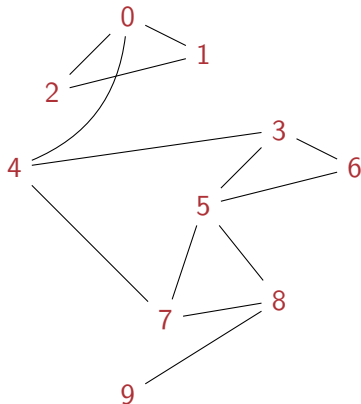


BFS from vertex 7

Visited	
0	True
1	True
2	True
3	True
4	True
5	True
6	True
7	True
8	True
9	True

To explore queue									
3	6	9	1	2					

- Mark 7 and add to queue
- Explore 7, visit {4,5,8}
- Explore 4, visit {0,3}
- Explore 5, visit {6}
- Explore 8, visit {9}
- Explore 0, visit {1,2}

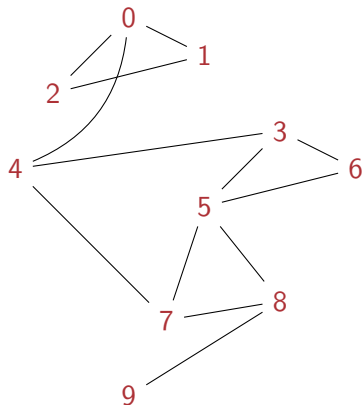


BFS from vertex 7

Visited	
0	True
1	True
2	True
3	True
4	True
5	True
6	True
7	True
8	True
9	True

To explore queue								
6	9	1	2					

- Mark 7 and add to queue
- Explore 7, visit {4,5,8}
- Explore 4, visit {0,3}
- Explore 5, visit {6}
- Explore 8, visit {9}
- Explore 0, visit {1,2}
- Explore 3

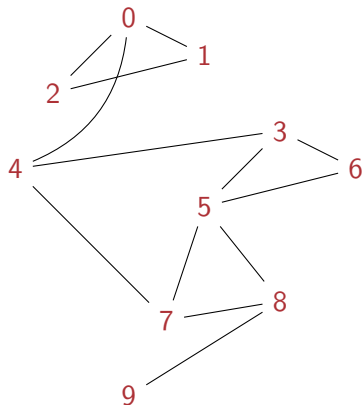


BFS from vertex 7

Visited	
0	True
1	True
2	True
3	True
4	True
5	True
6	True
7	True
8	True
9	True

To explore queue								
9	1	2						

- Mark 7 and add to queue
- Explore 7, visit {4,5,8}
- Explore 4, visit {0,3}
- Explore 5, visit {6}
- Explore 8, visit {9}
- Explore 0, visit {1,2}
- Explore 3
- Explore 6

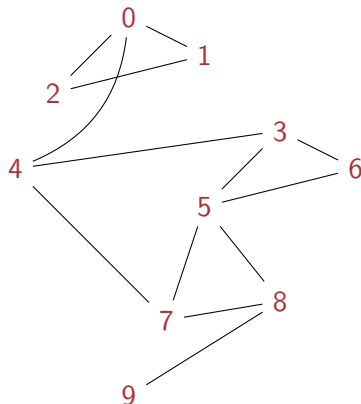


BFS from vertex 7

Visited	
0	True
1	True
2	True
3	True
4	True
5	True
6	True
7	True
8	True
9	True

To explore queue								
1	2							

- Mark 7 and add to queue
- Explore 7, visit {4,5,8}
- Explore 4, visit {0,3}
- Explore 5, visit {6}
- Explore 8, visit {9}
- Explore 0, visit {1,2}
- Explore 3
- Explore 6
- Explore 9

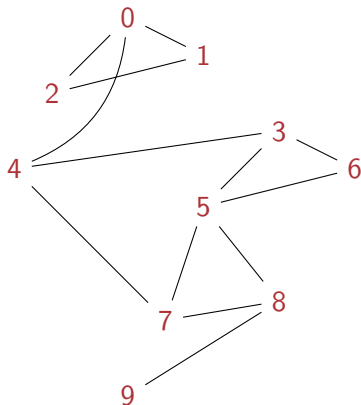


BFS from vertex 7

Visited	
0	True
1	True
2	True
3	True
4	True
5	True
6	True
7	True
8	True
9	True

To explore queue									
2									

- Mark 7 and add to queue
- Explore 7, visit {4,5,8}
- Explore 4, visit {0,3}
- Explore 5, visit {6}
- Explore 8, visit {9}
- Explore 0, visit {1,2}
- Explore 3
- Explore 6
- Explore 9
- Explore 1

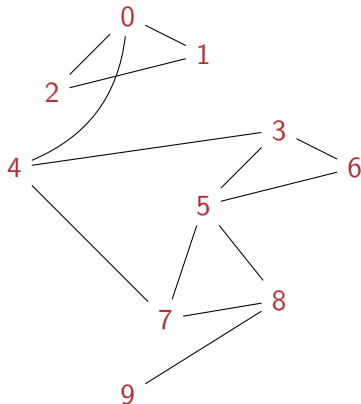


BFS from vertex 7

Visited	
0	True
1	True
2	True
3	True
4	True
5	True
6	True
7	True
8	True
9	True

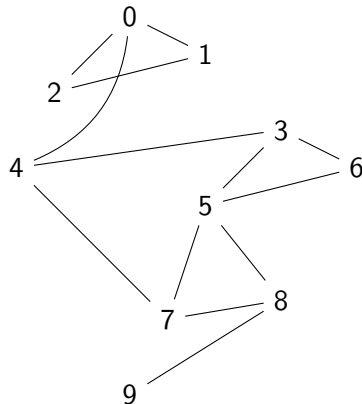
To explore queue								

- Mark 7 and add to queue
- Explore 7, visit {4,5,8}
- Explore 4, visit {0,3}
- Explore 5, visit {6}
- Explore 8, visit {9}
- Explore 0, visit {1,2}
- Explore 3
- Explore 6
- Explore 9
- Explore 1
- Explore 2



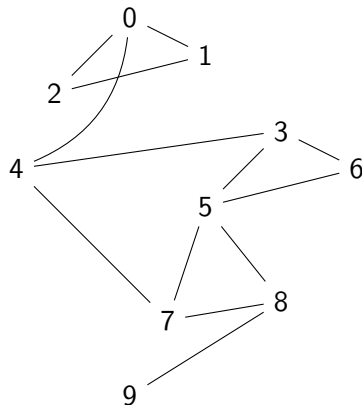
Enhancing BFS to record paths

- If BFS from i sets $\text{visited}(j) = \text{True}$, we know that j is reachable from i



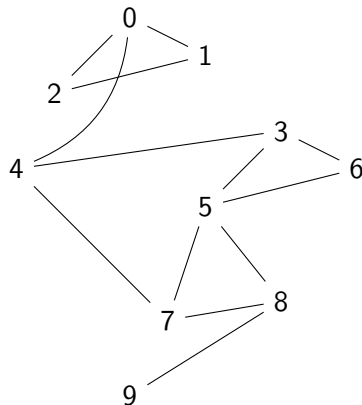
Enhancing BFS to record paths

- If BFS from i sets $\text{visited}(j) = \text{True}$, we know that j is reachable from i
- How do we recover a path from i to j ?



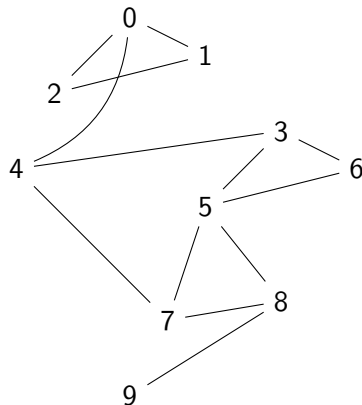
Enhancing BFS to record paths

- If BFS from i sets $\text{visited}(j) = \text{True}$, we know that j is reachable from i
- How do we recover a path from i to j ?
- $\text{visited}(j)$ was set to True when exploring some vertex k



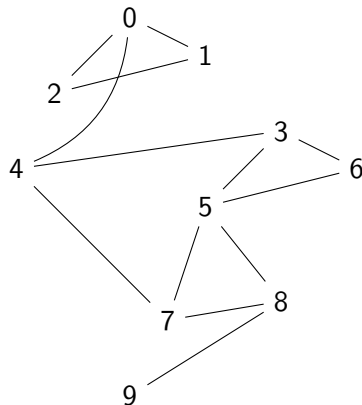
Enhancing BFS to record paths

- If BFS from i sets $\text{visited}(j) = \text{True}$, we know that j is reachable from i
- How do we recover a path from i to j ?
- $\text{visited}(j)$ was set to True when exploring some vertex k
- Record $\text{parent}(j) = k$



Enhancing BFS to record paths

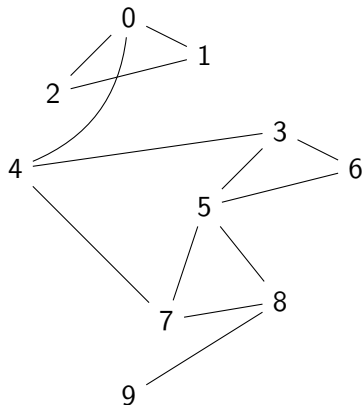
- If BFS from i sets $\text{visited}(j) = \text{True}$, we know that j is reachable from i
- How do we recover a path from i to j ?
- $\text{visited}(j)$ was set to True when exploring some vertex k
- Record $\text{parent}(j) = k$
- From j , follow parent links to trace back a path to i



BFS from vertex 7 with parent information

	Visited	Parent
0	False	
1	False	
2	False	
3	False	
4	False	
5	False	
6	False	
7	False	
8	False	
9	False	

To explore queue									

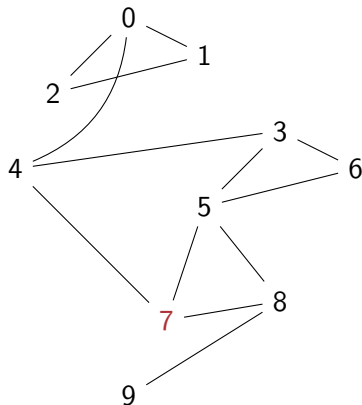


BFS from vertex 7 with parent information

	Visited	Parent
0	False	
1	False	
2	False	
3	False	
4	False	
5	False	
6	False	
7	True	
8	False	
9	False	

To explore queue									
7									

- Mark 7, add to queue

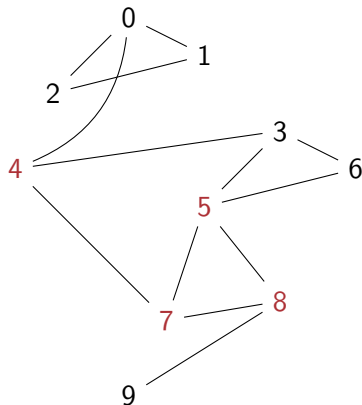


BFS from vertex 7 with parent information

	Visited	Parent
0	False	
1	False	
2	False	
3	False	
4	True	7
5	True	7
6	False	
7	True	
8	True	7
9	False	

To explore queue								
4	5	8						

- Mark 7, add to queue
- Explore 7, visit {4,5,8}

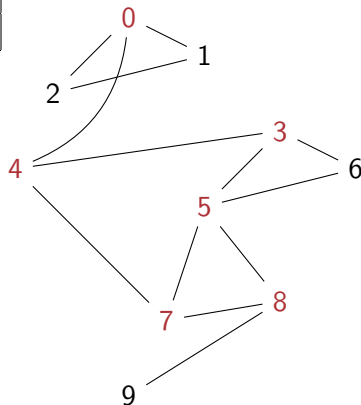


BFS from vertex 7 with parent information

	Visited	Parent
0	True	4
1	False	
2	False	
3	True	4
4	True	7
5	True	7
6	False	
7	True	
8	True	7
9	False	

To explore queue									
5	8	0	3						

- Mark 7, add to queue
- Explore 7, visit {4,5,8}
- Explore 4, visit {0,3}

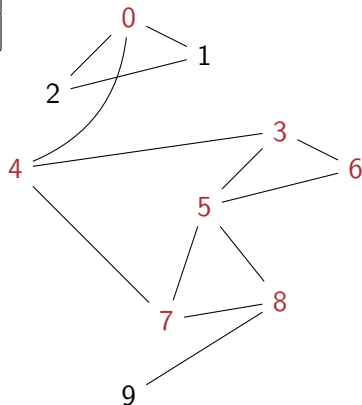


BFS from vertex 7 with parent information

	Visited	Parent
0	True	4
1	False	
2	False	
3	True	4
4	True	7
5	True	7
6	True	5
7	True	
8	True	7
9	False	

To explore queue									
8	0	3	6						

- Mark 7, add to queue
- Explore 7, visit {4,5,8}
- Explore 4, visit {0,3}
- Explore 5, visit {6}

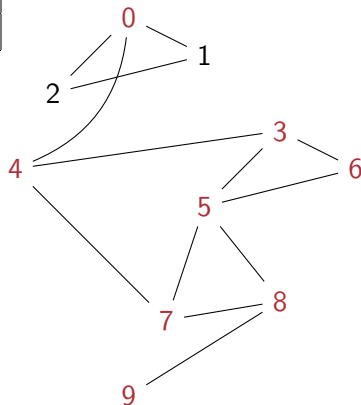


BFS from vertex 7 with parent information

	Visited	Parent
0	True	4
1	False	
2	False	
3	True	4
4	True	7
5	True	7
6	True	5
7	True	
8	True	7
9	True	8

To explore queue									
0	3	6	9						

- Mark 7, add to queue
- Explore 7, visit {4,5,8}
- Explore 4, visit {0,3}
- Explore 5, visit {6}
- Explore 8, visit {9}

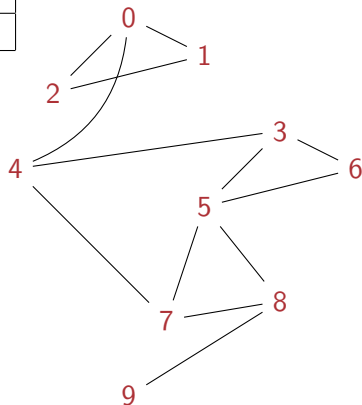


BFS from vertex 7 with parent information

	Visited	Parent
0	True	4
1	True	0
2	True	0
3	True	4
4	True	7
5	True	7
6	True	5
7	True	
8	True	7
9	True	8

To explore queue									
3	6	9	1	2					

- Mark 7, add to queue
- Explore 7, visit {4,5,8}
- Explore 4, visit {0,3}
- Explore 5, visit {6}
- Explore 8, visit {9}
- Explore 0, visit {1,2}

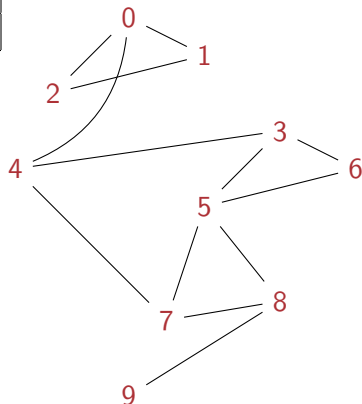


BFS from vertex 7 with parent information

	Visited	Parent
0	True	4
1	True	0
2	True	0
3	True	4
4	True	7
5	True	7
6	True	5
7	True	
8	True	7
9	True	8

To explore queue									
6	9	1	2						

- Mark 7, add to queue
- Explore 7, visit {4,5,8}
- Explore 4, visit {0,3}
- Explore 5, visit {6}
- Explore 8, visit {9}
- Explore 0, visit {1,2}
- Explore 3

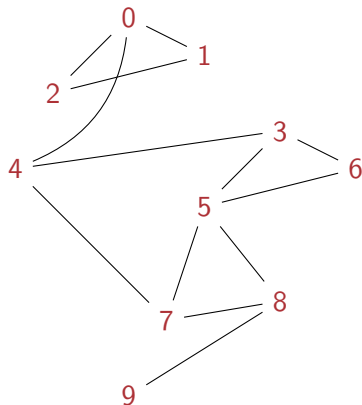


BFS from vertex 7 with parent information

	Visited	Parent
0	True	4
1	True	0
2	True	0
3	True	4
4	True	7
5	True	7
6	True	5
7	True	
8	True	7
9	True	8

To explore queue								
9	1	2						

- Mark 7, add to queue
- Explore 7, visit {4,5,8}
- Explore 4, visit {0,3}
- Explore 5, visit {6}
- Explore 8, visit {9}
- Explore 0, visit {1,2}
- Explore 3
- Explore 6

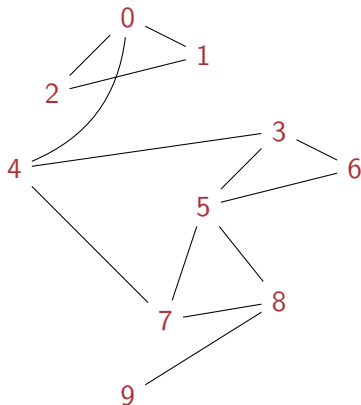


BFS from vertex 7 with parent information

	Visited	Parent
0	True	4
1	True	0
2	True	0
3	True	4
4	True	7
5	True	7
6	True	5
7	True	
8	True	7
9	True	8

To explore queue								
1	2							

- Mark 7, add to queue
- Explore 7, visit {4,5,8}
- Explore 4, visit {0,3}
- Explore 5, visit {6}
- Explore 8, visit {9}
- Explore 0, visit {1,2}
- Explore 3
- Explore 6
- Explore 9

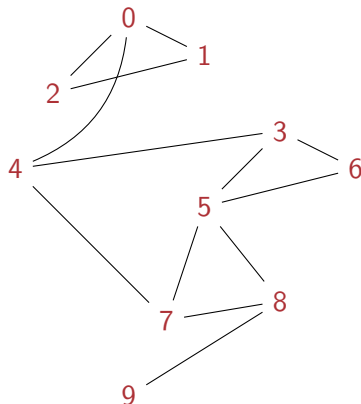


BFS from vertex 7 with parent information

	Visited	Parent
0	True	4
1	True	0
2	True	0
3	True	4
4	True	7
5	True	7
6	True	5
7	True	
8	True	7
9	True	8

To explore queue									
2									

- Mark 7, add to queue
- Explore 7, visit {4,5,8}
- Explore 4, visit {0,3}
- Explore 5, visit {6}
- Explore 8, visit {9}
- Explore 0, visit {1,2}
- Explore 3
- Explore 6
- Explore 9
- Explore 1

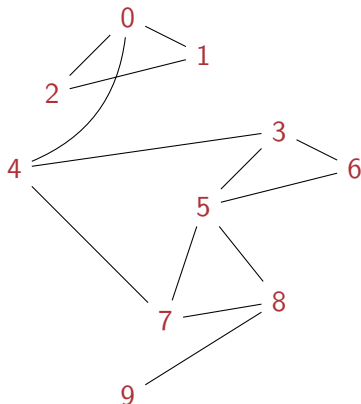


BFS from vertex 7 with parent information

	Visited	Parent
0	True	4
1	True	0
2	True	0
3	True	4
4	True	7
5	True	7
6	True	5
7	True	
8	True	7
9	True	8

To explore queue									

- Mark 7, add to queue
- Explore 7, visit {4,5,8}
- Explore 4, visit {0,3}
- Explore 5, visit {6}
- Explore 8, visit {9}
- Explore 0, visit {1,2}
- Explore 3
- Explore 6
- Explore 9
- Explore 1
- Explore 2



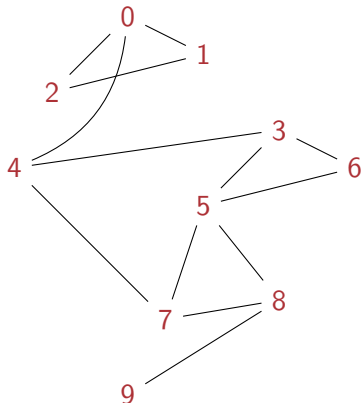
BFS from vertex 7 with parent information

	Visited	Parent
0	True	4
1	True	0
2	True	0
3	True	4
4	True	7
5	True	7
6	True	5
7	True	
8	True	7
9	True	8

Path from 7 to 6 is
7-5-6

To explore queue								

- Mark 7, add to queue
- Explore 7, visit {4,5,8}
- Explore 4, visit {0,3}
- Explore 5, visit {6}
- Explore 8, visit {9}
- Explore 0, visit {1,2}
- Explore 3
- Explore 6
- Explore 9
- Explore 1
- Explore 2



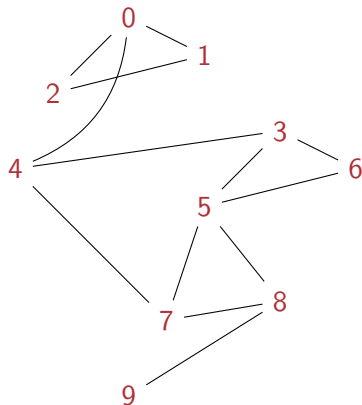
BFS from vertex 7 with parent information

	Visited	Parent
0	True	4
1	True	0
2	True	0
3	True	4
4	True	7
5	True	7
6	True	5
7	True	
8	True	7
9	True	8

Path from 7 to 2 is
7-4-0-2

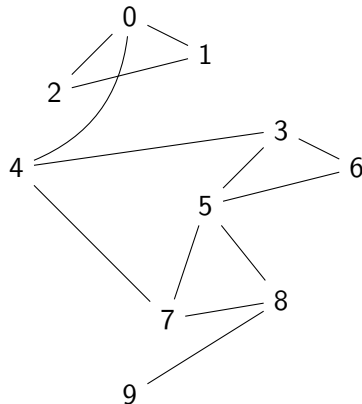
To explore queue									

- Mark 7, add to queue
- Explore 7, visit {4,5,8}
- Explore 4, visit {0,3}
- Explore 5, visit {6}
- Explore 8, visit {9}
- Explore 0, visit {1,2}
- Explore 3
- Explore 6
- Explore 9
- Explore 1
- Explore 2



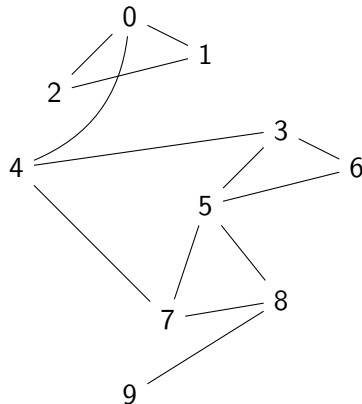
Enhancing BFS to record distance

- BFS explores neighbours level by level



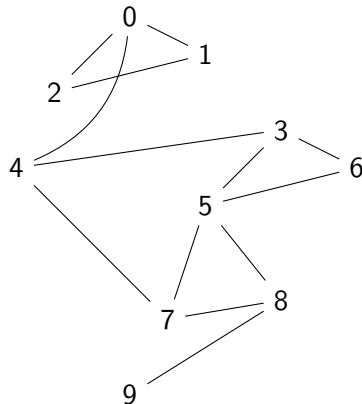
Enhancing BFS to record distance

- BFS explores neighbours level by level
- By recording the level at which a vertex is visited, we get its distance from the source vertex



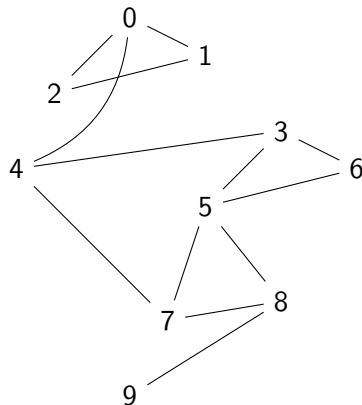
Enhancing BFS to record distance

- BFS explores neighbours level by level
- By recording the level at which a vertex is visited, we get its distance from the source vertex
- Instead of `visited(j)`, maintain `level(j)`



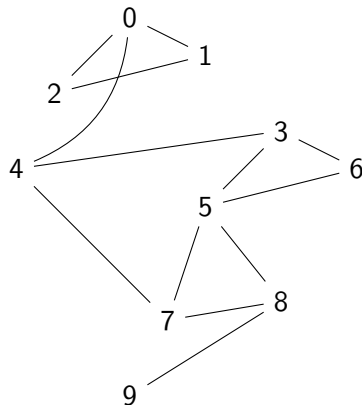
Enhancing BFS to record distance

- BFS explores neighbours level by level
- By recording the level at which a vertex is visited, we get its distance from the source vertex
- Instead of `visited(j)`, maintain `level(j)`
- Initialize `level(j) = -1` for all j



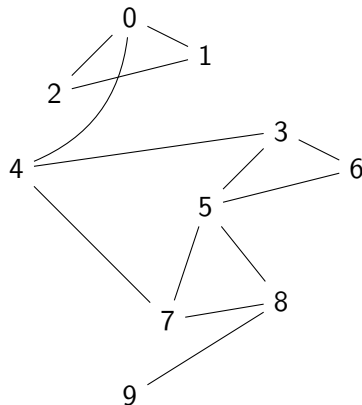
Enhancing BFS to record distance

- BFS explores neighbours level by level
- By recording the level at which a vertex is visited, we get its distance from the source vertex
- Instead of `visited(j)`, maintain `level(j)`
- Initialize `level(j) = -1` for all j
- Set `level(i) = 0` for source vertex



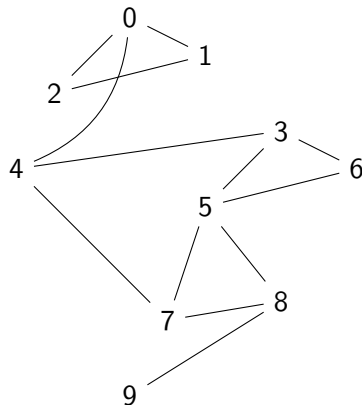
Enhancing BFS to record distance

- BFS explores neighbours level by level
- By recording the level at which a vertex is visited, we get its distance from the source vertex
- Instead of `visited(j)`, maintain `level(j)`
- Initialize `level(j) = -1` for all j
- Set `level(i) = 0` for source vertex
- If we visit j from k , set `level(j)` to `level(k) + 1`



Enhancing BFS to record distance

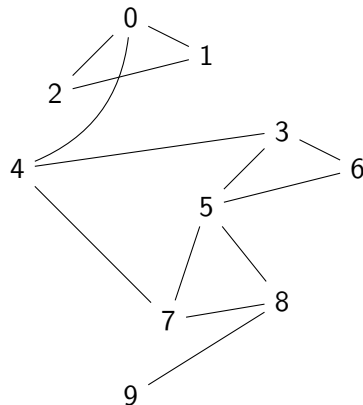
- BFS explores neighbours level by level
- By recording the level at which a vertex is visited, we get its distance from the source vertex
- Instead of `visited(j)`, maintain `level(j)`
- Initialize `level(j) = -1` for all j
- Set `level(i) = 0` for source vertex
- If we visit j from k , set `level(j)` to `level(k) + 1`
- `level(j)` is the length of the shortest path from the source vertex, in number of edges



BFS from vertex 7 with parent and distance information

	Level	Parent
0	-1	
1	-1	
2	-1	
3	-1	
4	-1	
5	-1	
6	-1	
7	-1	
8	-1	
9	-1	

To explore queue									

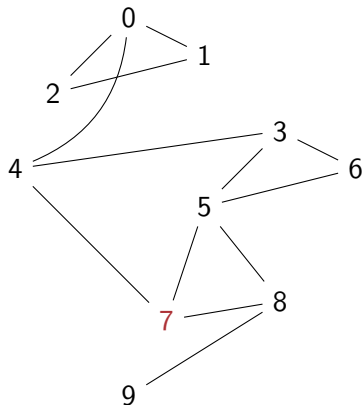


BFS from vertex 7 with parent and distance information

	Level	Parent
0	-1	
1	-1	
2	-1	
3	-1	
4	-1	
5	-1	
6	-1	
7	0	
8	-1	
9	-1	

To explore queue									
7									

- Mark 7, add to queue

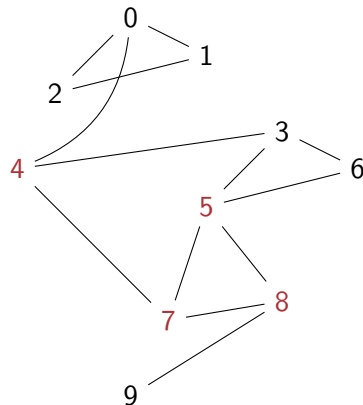


BFS from vertex 7 with parent and distance information

	Level	Parent
0	-1	
1	-1	
2	-1	
3	-1	
4	1	7
5	1	7
6	-1	
7	0	
8	1	7
9	-1	

To explore queue								
4	5	8						

- Mark 7, add to queue
- Explore 7, visit {4,5,8}

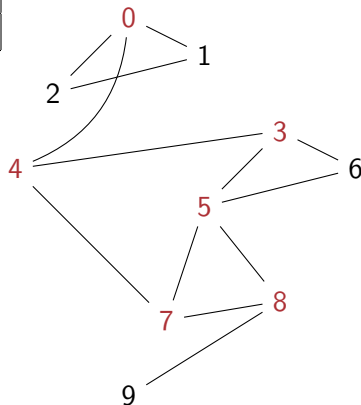


BFS from vertex 7 with parent and distance information

	Level	Parent
0	2	4
1	-1	
2	-1	
3	2	4
4	1	7
5	1	7
6	-1	
7	0	
8	1	7
9	-1	

To explore queue									
5	8	0	3						

- Mark 7, add to queue
- Explore 7, visit {4,5,8}
- Explore 4, visit {0,3}

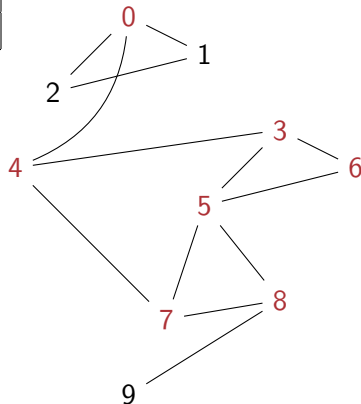


BFS from vertex 7 with parent and distance information

	Level	Parent
0	2	4
1	-1	
2	-1	
3	2	4
4	1	7
5	1	7
6	2	5
7	0	
8	1	7
9	-1	

To explore queue									
8	0	3	6						

- Mark 7, add to queue
- Explore 7, visit {4,5,8}
- Explore 4, visit {0,3}
- Explore 5, visit {6}

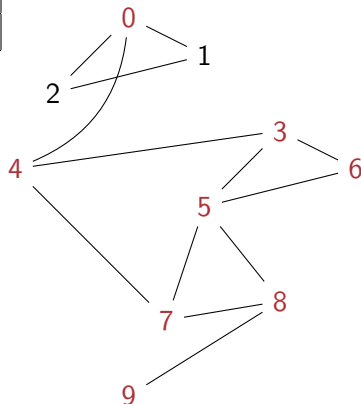


BFS from vertex 7 with parent and distance information

	Level	Parent
0	2	4
1	-1	
2	-1	
3	2	4
4	1	7
5	1	7
6	2	5
7	0	
8	1	7
9	2	8

To explore queue									
0	3	6	9						

- Mark 7, add to queue
- Explore 7, visit {4,5,8}
- Explore 4, visit {0,3}
- Explore 5, visit {6}
- Explore 8, visit {9}

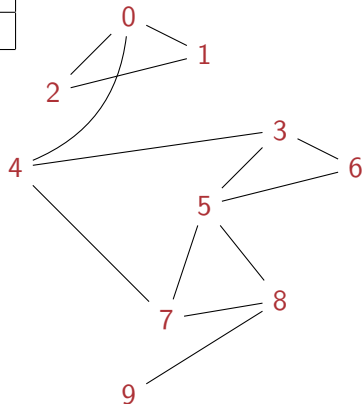


BFS from vertex 7 with parent and distance information

	Level	Parent
0	2	4
1	3	0
2	3	0
3	2	4
4	1	7
5	1	7
6	2	5
7	0	
8	1	7
9	2	8

To explore queue									
3	6	9	1	2					

- Mark 7, add to queue
- Explore 7, visit {4,5,8}
- Explore 4, visit {0,3}
- Explore 5, visit {6}
- Explore 8, visit {9}
- Explore 0, visit {1,2}

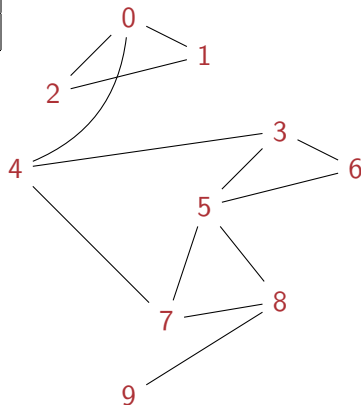


BFS from vertex 7 with parent and distance information

	Level	Parent
0	2	4
1	3	0
2	3	0
3	2	4
4	1	7
5	1	7
6	2	5
7	0	
8	1	7
9	2	8

To explore queue									
6	9	1	2						

- Mark 7, add to queue
- Explore 7, visit {4,5,8}
- Explore 4, visit {0,3}
- Explore 5, visit {6}
- Explore 8, visit {9}
- Explore 0, visit {1,2}
- Explore 3

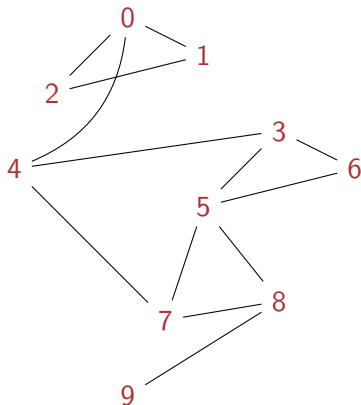


BFS from vertex 7 with parent and distance information

	Level	Parent
0	2	4
1	3	0
2	3	0
3	2	4
4	1	7
5	1	7
6	2	5
7	0	
8	1	7
9	2	8

To explore queue								
9	1	2						

- Mark 7, add to queue
- Explore 7, visit {4,5,8}
- Explore 4, visit {0,3}
- Explore 5, visit {6}
- Explore 8, visit {9}
- Explore 0, visit {1,2}
- Explore 3
- Explore 6

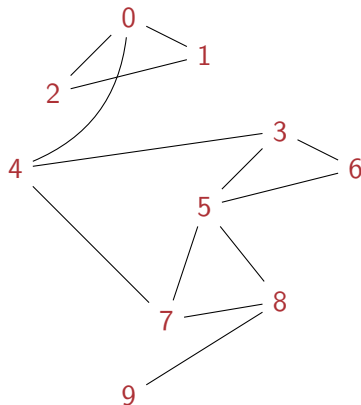


BFS from vertex 7 with parent and distance information

	Level	Parent
0	2	4
1	3	0
2	3	0
3	2	4
4	1	7
5	1	7
6	2	5
7	0	
8	1	7
9	2	8

To explore queue								
1	2							

- Mark 7, add to queue
- Explore 7, visit {4,5,8}
- Explore 4, visit {0,3}
- Explore 5, visit {6}
- Explore 8, visit {9}
- Explore 0, visit {1,2}
- Explore 3
- Explore 6
- Explore 9

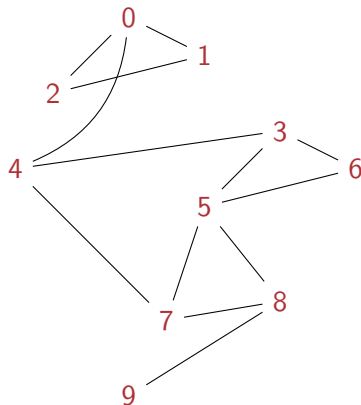


BFS from vertex 7 with parent and distance information

	Level	Parent
0	2	4
1	3	0
2	3	0
3	2	4
4	1	7
5	1	7
6	2	5
7	0	
8	1	7
9	2	8

To explore queue									
2									

- Mark 7, add to queue
- Explore 7, visit {4,5,8}
- Explore 4, visit {0,3}
- Explore 5, visit {6}
- Explore 8, visit {9}
- Explore 0, visit {1,2}
- Explore 3
- Explore 6
- Explore 9
- Explore 1

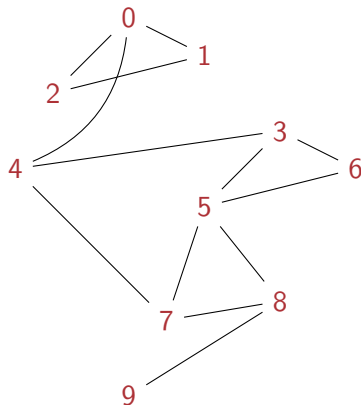


BFS from vertex 7 with parent and distance information

	Level	Parent
0	2	4
1	3	0
2	3	0
3	2	4
4	1	7
5	1	7
6	2	5
7	0	
8	1	7
9	2	8

To explore queue									

- Mark 7, add to queue
- Explore 7, visit {4,5,8}
- Explore 4, visit {0,3}
- Explore 5, visit {6}
- Explore 8, visit {9}
- Explore 0, visit {1,2}
- Explore 3
- Explore 6
- Explore 9
- Explore 1
- Explore 2



Summary

- Breadth first search is a systematic strategy to explore a graph, level by level

Summary

- Breadth first search is a systematic strategy to explore a graph, level by level
- Record which vertices have been visited

Summary

- Breadth first search is a systematic strategy to explore a graph, level by level
- Record which vertices have been visited
- Maintain visited but unexplored vertices in a queue

Summary

- Breadth first search is a systematic strategy to explore a graph, level by level
- Record which vertices have been visited
- Maintain visited but unexplored vertices in a queue
- Add parent information to recover the path to each reachable vertex

Summary

- Breadth first search is a systematic strategy to explore a graph, level by level
- Record which vertices have been visited
- Maintain visited but unexplored vertices in a queue
- Add parent information to recover the path to each reachable vertex
- Maintain level information to record length of the shortest path, in terms of number of edges

Summary

- Breadth first search is a systematic strategy to explore a graph, level by level
- Record which vertices have been visited
- Maintain visited but unexplored vertices in a queue
- Add parent information to recover the path to each reachable vertex
- Maintain level information to record length of the shortest path, in terms of number of edges
- In general, edges are labelled with a **cost** (distance, time, ticket price, ...)

Summary

- Breadth first search is a systematic strategy to explore a graph, level by level
- Record which vertices have been visited
- Maintain visited but unexplored vertices in a queue
- Add parent information to recover the path to each reachable vertex
- Maintain level information to record length of the shortest path, in terms of number of edges
- In general, edges are labelled with a **cost** (distance, time, ticket price, ...)
- Will look at **weighted graphs**, where shortest paths are in terms of cost, not number of edges