SE284: Introduction to Graph Algorithms

Katerina Taškova

Email: katerina.taskova@auckland.ac.nz
Room: 303S.483

Outline

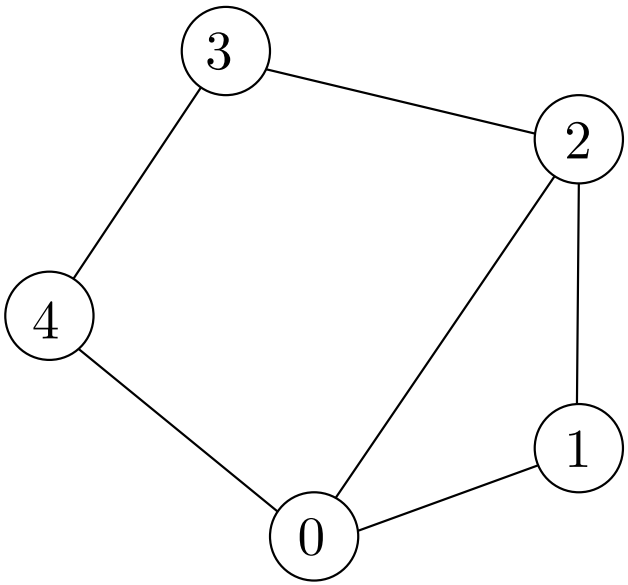
Girth, connectivity, and components

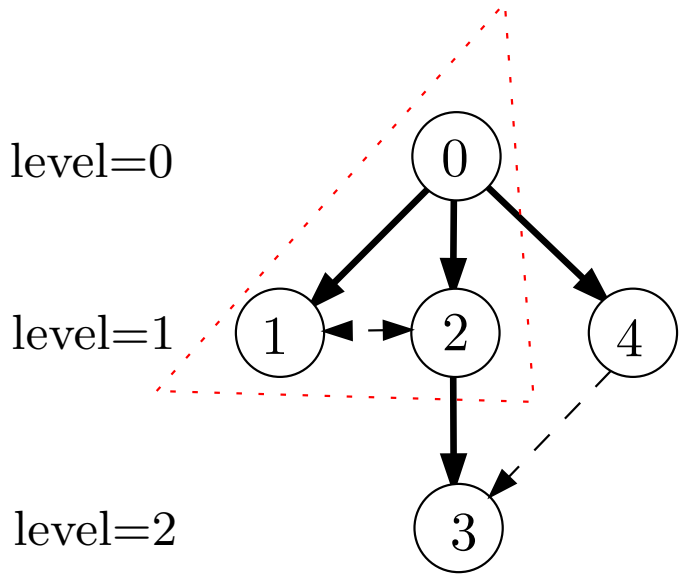
Lecture Notes 27, Textbook 5.7-8

Acknowledgment for slide content: Michael Dinneen, Simone Linz

Using BFS to find cycles in graphs

Cycles can also be easily detected in a graph using BFS. Finding a cycle of minimum length in a graph is not difficult using BFS (better than DFS).





BFS : cross edges (undirected graphs)

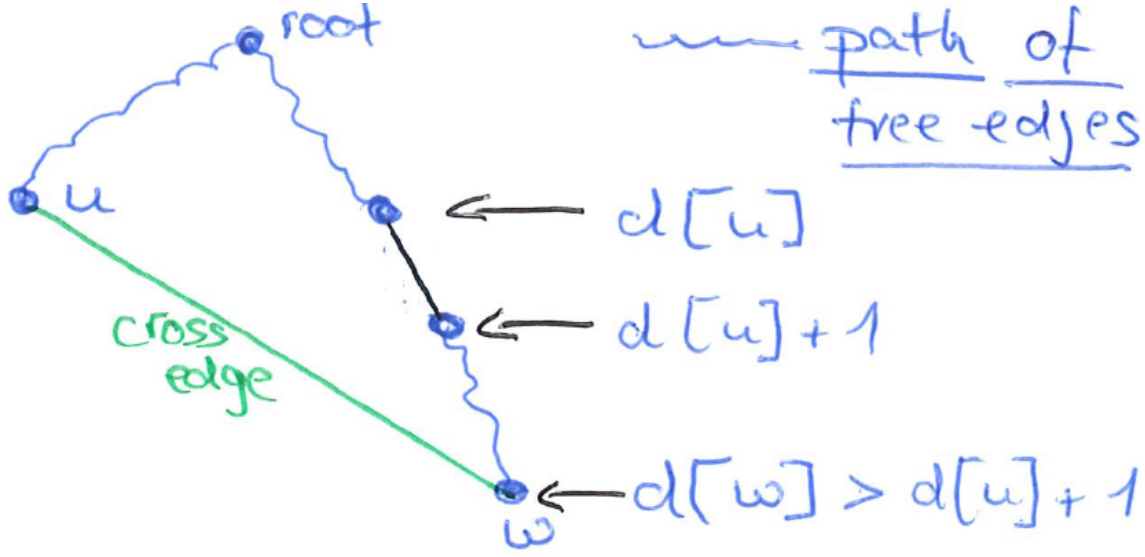
Let $\{u, w\}$ be a cross edge. Then

either $d[u] = d[w]$

or $|d[u] - d[w]| = 1$

Sketch of proof

Suppose $|d[u] - d[w]| > 1$



$\Rightarrow d[w] >$ distance from root to w

(shortest path from root to w contains the cross edge)

$\lightning$ contradiction

Girth of a graph (digraph) – reminder

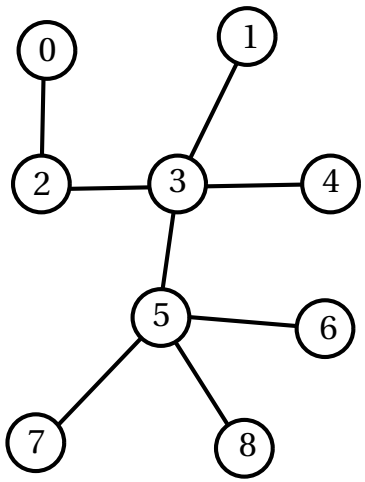
Definition

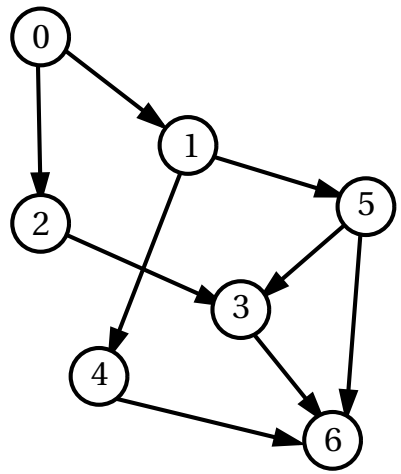
For a graph (with a cycle), the length of the shortest cycle is called the girth of the graph. If the graph has no cycles then the girth is undefined but may be viewed as $+\infty$.

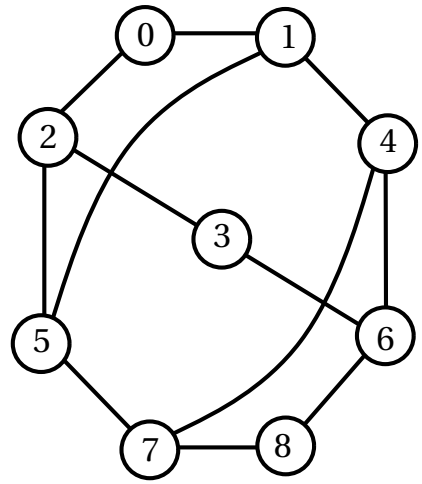
Fact

For a digraph we use the term girth for its underlying graph and the (maybe non-standard) term directed girth for the length of the smallest directed cycle.

Girth of a graph (digraph) – example







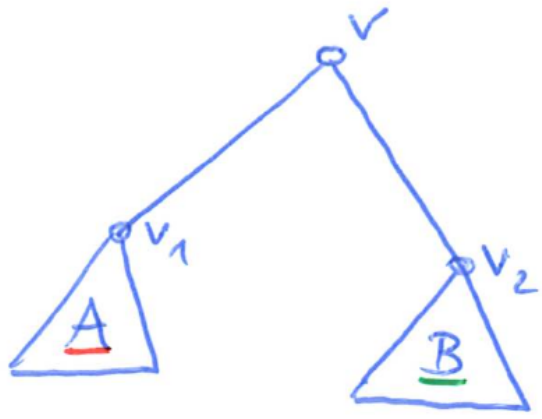
The first graph has no cycle, the DAG in the middle has girth 3, and the graph on the right has girth 4.

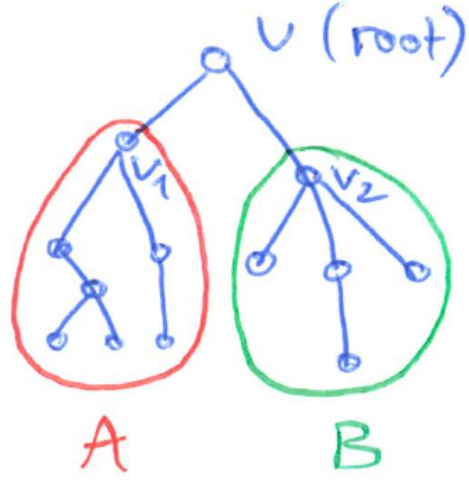
Computing girth of a graph

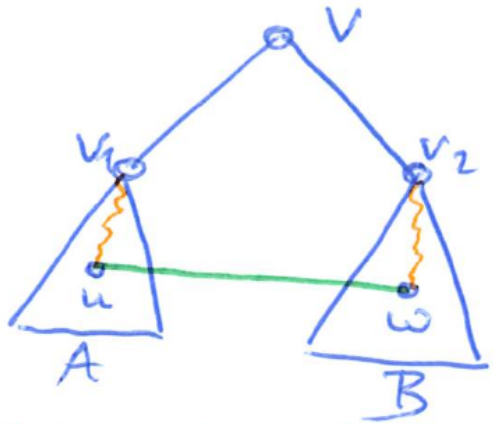
- If vertex v is on at least one cycle then BFS starting at v will find it.
- On detection of a cross edge between descendants u and w, determine whether u and w are in different subtrees below v (the root of the tree).
  - If yes, then a cycle of length $d[u] + d[w] + 1$ is found.
  - If no, then a cycle of shorter length is found (but avoids v).

Computing girth of a graph

- If vertex v is on at least one cycle then BFS starting at v will find it.
- On detection of a cross edge between descendants u and w, determine whether u and w are in different subtrees below v (the root of the tree).
  - If yes, then a cycle of length $d[u] + d[w] + 1$ is found.
  - If no, then a cycle of shorter length is found (but avoids v).


v
v1
v2
A
B

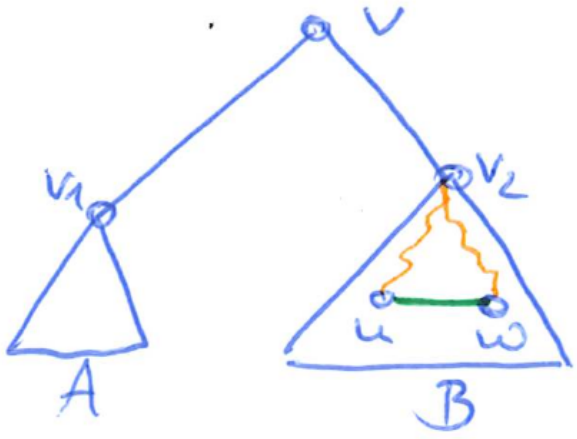

v (root)
v1
v2
A
B



cross edge $\{u, w\}$

distance from root v

$d[u] + d[w] + 1$
(length of cycle containing v)


v
v1
v2
u
w
A
B

Computing girth of a graph

- If vertex v is on at least one cycle then BFS starting at v will find it.
- On detection of a cross edge between descendants u and w, determine whether u and w are in different subtrees below v (the root of the tree).
  - If yes, then a cycle of length $d[u] + d[w] + 1$ is found.
  - If no, then a cycle of shorter length is found (but avoids v).
- If $d[u] = d[w]$ then odd length, where v common ancestor.
- Otherwise, WLOG, $d[u] + 1 = d[w]$ and even length.

cross edge $\{u, w\}$

$d[u] = d[w]$

cycle length : $2d[u] + 1$

(cross edge)

$\Rightarrow$ odd length

$d[u] + 1 = d[w]$

cycle length : $d[u] + d[w] + 1$

$= d[u] + d[u] + 1 + 1$

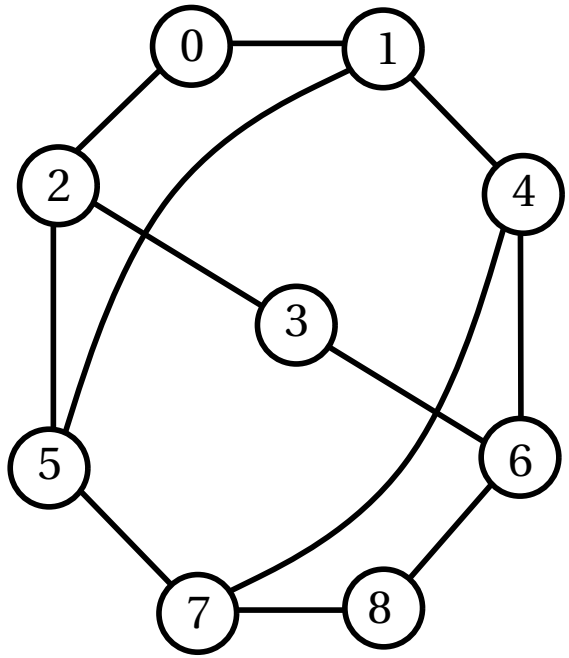
$= 2d[u] + 2$

$\Rightarrow$ even length

Computing girth of a graph

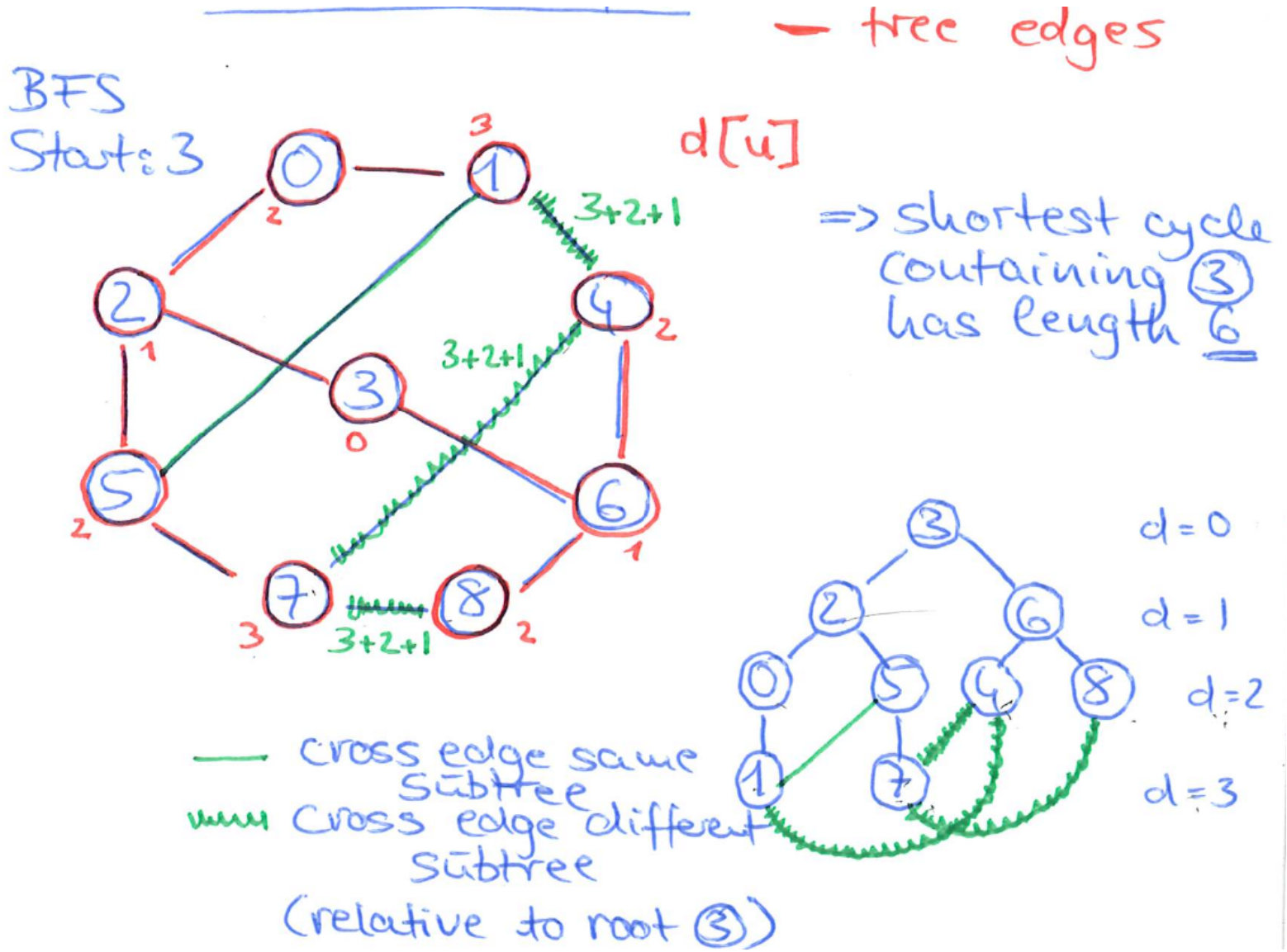
- If vertex v is on at least one cycle then BFS starting at v will find it.
- On detection of a cross edge between descendants u and w, determine whether u and w are in different subtrees below v (the root of the tree).
  - If yes, then a cycle of length $d[u] + d[w] + 1$ is found.
  - If no, then a cycle of shorter length is found (but avoids v).
- If $d[u] = d[w]$ then odd length, where v common ancestor.
- Otherwise, WLOG, $d[u] + 1 = d[w]$ and even length.
- **To compute girth, we run the above procedure once for each $v \in V(G)$ and take minimum.**

Computing girth of a graph – an example

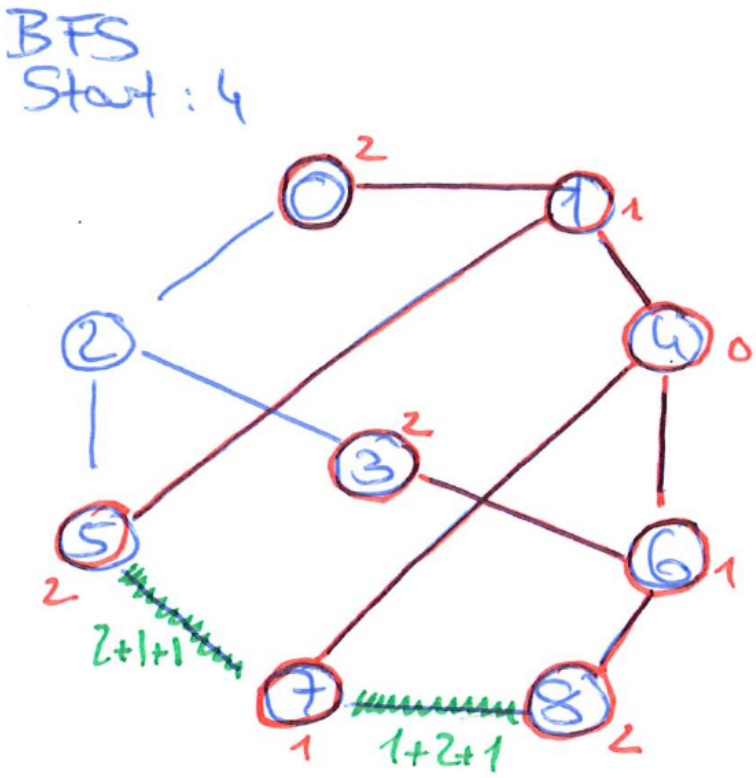


Run BFS for 3, shortest cycle has length 6.
Run BFS for any other vertex, shortest cycle has length 4.
Girth of the graph is 4.

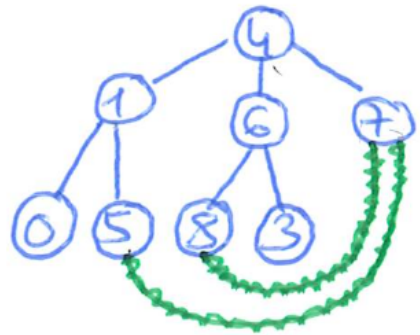
Computing girth of a graph – an example



Computing girth of a graph – an example



=> shortest cycle containing (4) has length 4

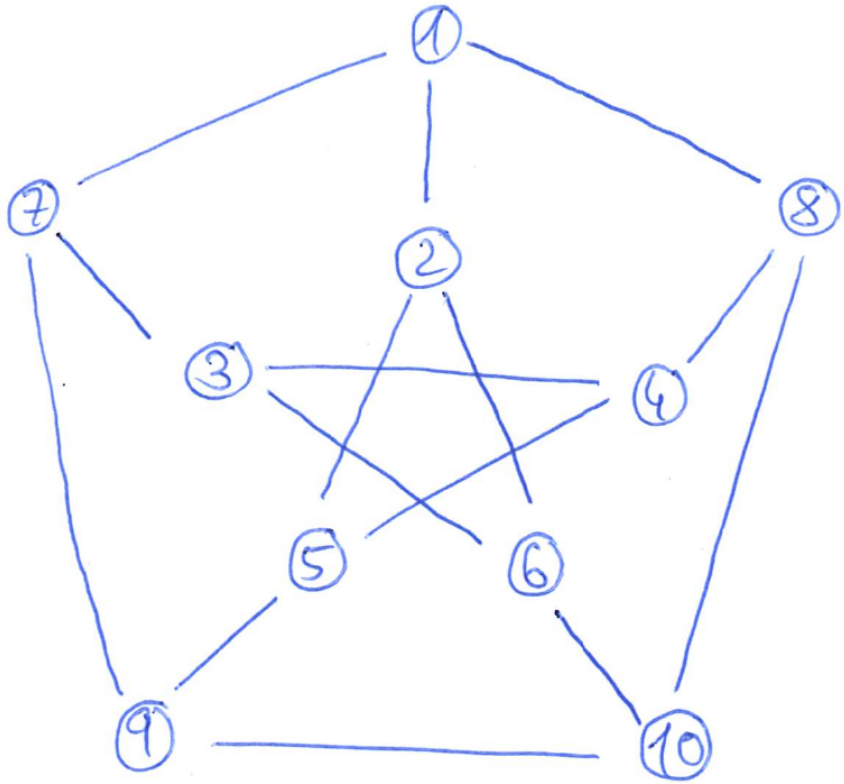


d = 0

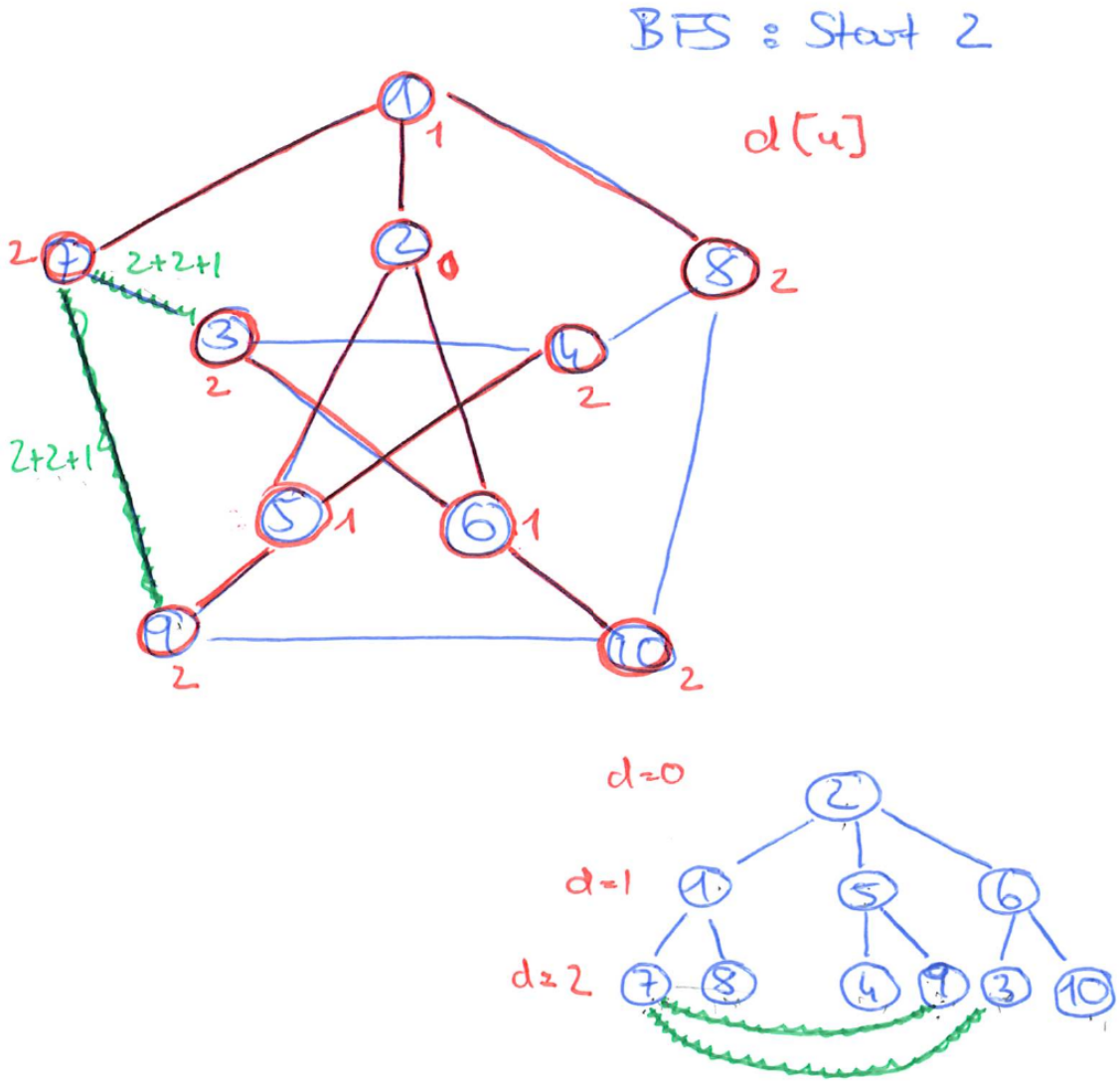
d = 1

d = 2

What is the girth of Petersen graph?

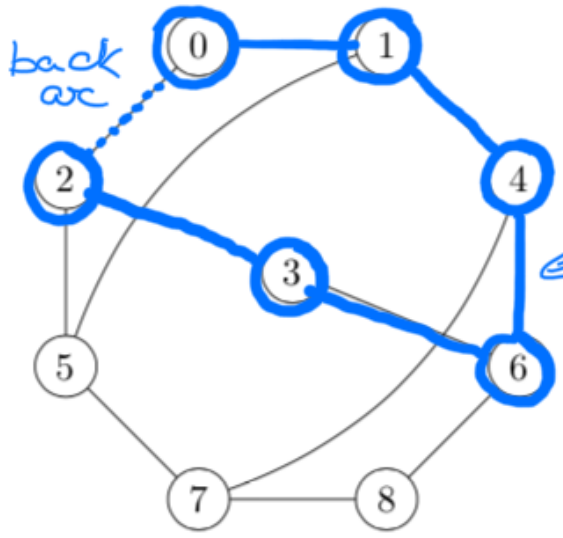


What is the girth of Petersen graph?



The girth is 5 = 2 x 2 + 1 (cross-edge)

Example 26.2. An easy-to-implement DFS idea may not work properly. Consider the DFS tree originating from vertex 0 of the graph below. Which is the smallest cycle it finds with the back edges? Which smaller cycle does it miss?



cycle of length 6, but there is a shorter cycle that contains vertex 0.

26.1 Finding the girth of a graph with BFS

How to compute the girth of a graph? Here is an algorithm for finding the length of a shortest cycle containing a given vertex v in a graph G.

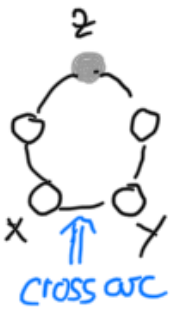
- Perform `BFSvisit` starting at v.
- If we meet a grey neighbour (that is, we are exploring edge $\{x, y\}$ from x and we find that y is already grey), continue only to the end of the current level and then stop.
- For each edge $\{x, y\}$ as above on this level, if v is the lowest common ancestor of x and y in the BFS tree, then there is a cycle containing x, y, v of length $l = d(x) + d(y) + 1$.
- Report the minimum value of l obtained along the current level.

Example 26.1. Prove above algorithm is correct by showing that:

1. meeting a grey neighbour means that you have indeed found a cycle;
2. any cycle that exists will be found in such a way; and
3. the cycle found in this way will be the shortest going through v.

1. There is a cycle that contains x, y, and z, where z is the lowest ancestor of x and y in the search tree. The length of this cycle is $d[x] + d[y] + 1 - 2\,d[z]$.

2. Let C be any cycle. Let z be the fist vertex of C visited by BFS. Let x, y be vertices of C that have the greatest distance from z such that $d[x] \leq d[y]$. Then z is the lowest ancestor of x and y and there is an edge $\{x, y\}$ which is a cross arc in the resulting search forest.



Graph connectivity

Definition

A graph G is connected if for each pair of vertices $u, v \in V(G)$, there is a path between them.

Definition

A graph G is disconnected if it is not connected and the maximal induced connected subgraphs are called the components of G.

Theorem

Let G be a graph and suppose that DFS or BFS is run on G. Then the connected components of G are precisely the subgraphs spanned by the trees in the search forest.

Graph connectivity

Definition

A graph G is connected if for each pair of vertices $u, v \in V(G)$, there is a path between them.

Definition

A graph G is disconnected if it is not connected and the maximal induced connected subgraphs are called the components of G.

Theorem

Let G be a graph and suppose that DFS or BFS is run on G. Then the connected components of G are precisely the subgraphs spanned by the trees in the search forest.

Graph connectivity

Definition

A graph G is connected if for each pair of vertices $u, v \in V(G)$, there is a path between them.

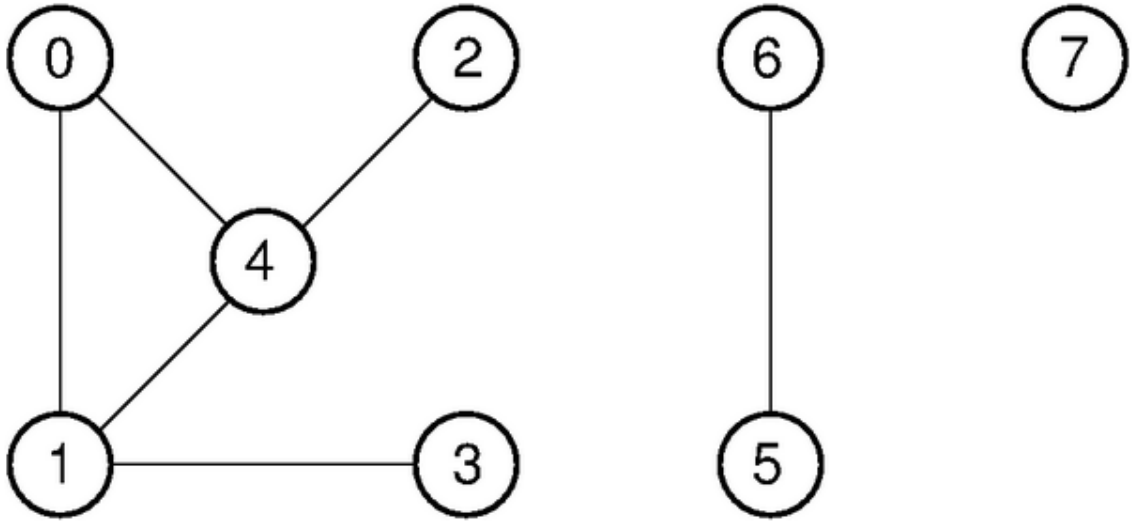
Definition

A graph G is disconnected if it is not connected and the maximal induced connected subgraphs are called the components of G.

Theorem

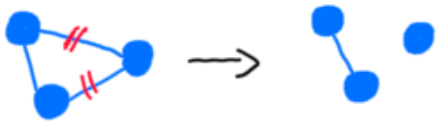
Let G be a graph and suppose that DFS or BFS is run on G. Then the connected components of G are precisely the subgraphs spanned by the trees in the search forest.

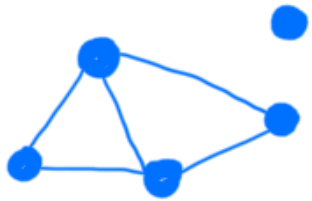
Connected components



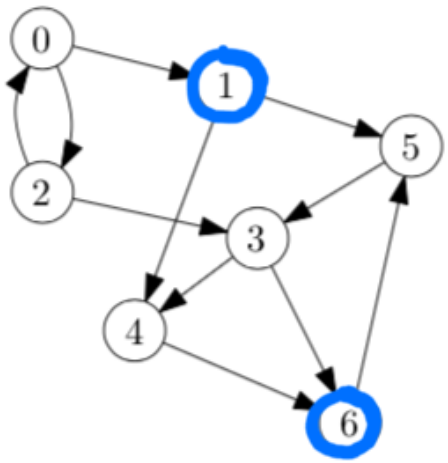
Examples

Example 26.6. The graph obtained by deleting two edges from a triangle has 2 connected components. Draw a graph with 5 vertices, 5 edges and 2 connected components.





Example 26.8. This digraph appears to be "all in one piece" as its underlying graph is connected. But can you get from any node to any other node following the direction of the arcs?

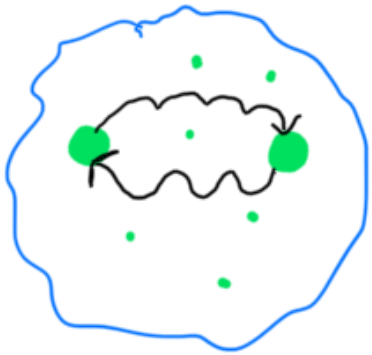


There is no directed path from 6 to 1.

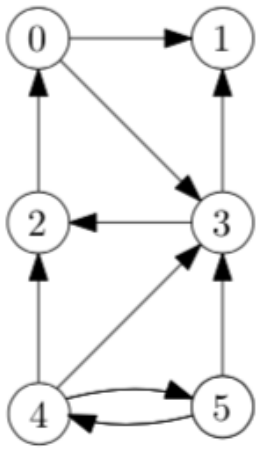
Not strongly connected digraph

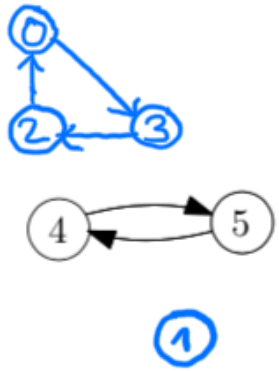
Example 26.10. Show that a strongly connected digraph of order at least two contains at least one cycle.

≥2 verti.



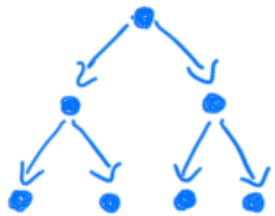
Example 26.12. This digraph has three strongly connected components. Find and draw the two missing ones.





Note that there are arcs of the digraph not included in the strongly connected components.

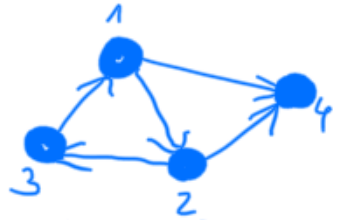
Example 26.13. Find a counter-example to show that the method for finding connected components of graphs in Theorem 26.7 fails at finding strongly connected components of digraphs.



not strongly connected, yet BFS would return a single tree in the search forest

Example 26.14. Using `BFSvisit` or `DFSvisit`, devise an algorithm that will determine whether or not G is strongly connected (i.e., has a single strongly connected component). What is the running time of the algorithm?

Run DFS/BFS for each vertex and check whether or not each resulting search forest contains a single tree only that spans the entire graph



DFS starting at 1, o.k.

DFS starting at 4, not o.k !!

Thank you!