

# Applications of Depth First Search

Depth-first search (DFS) is an algorithm (or technique) for traversing a graph.

Following are the problems that use DFS as a building block.

**1) For an unweighted graph, DFS traversal of the graph produces the minimum spanning tree and all pair shortest path tree.**

## **2) Detecting cycle in a graph**

A graph has cycle if and only if we see a back edge during DFS. So we can run DFS for the graph and check for back edges. (See [this](#) for details)

## **3) Path Finding**

We can specialize the DFS algorithm to find a path between two given vertices u and z.

- i) Call DFS(G, u) with u as the start vertex.
- ii) Use a stack S to keep track of the path between the start vertex and the current vertex.
- iii) As soon as destination vertex z is encountered, return the path as the contents of the stack

See [this](#) for details.

## **4) Topological Sorting**

Topological Sorting is mainly used for scheduling jobs from the given dependencies among jobs. In computer science, applications of this type arise in instruction scheduling, ordering of formula cell evaluation when recomputing formula values in spreadsheets, logic synthesis, determining the order of compilation tasks to perform in makefiles, data serialization, and resolving symbol dependencies in linkers [2].

## **5) To test if a graph is bipartite**

We can augment either BFS or DFS when we first discover a new vertex, color it opposite to its parents, and for each other edge, check it doesn't link two vertices of the same color. The first vertex in any connected component can be red or black! See [this](#) for details.

**6) Finding Strongly Connected Components of a graph** A directed graph is called strongly connected if there is a path from each vertex in the graph to every other vertex. (See [this](#) for DFS based algo for finding Strongly Connected Components)

**7) Solving puzzles with only one solution**, such as mazes. (DFS can be adapted to find all solutions to a maze by only including nodes on the current path in the visited set.)

## **Sources:**

<http://www8.cs.umu.se/kurser/TDBAfl/VT06/algorithms/LEC/LECTUR16/NODE16.HTM>

[http://en.wikipedia.org/wiki/Depth-first\\_search](http://en.wikipedia.org/wiki/Depth-first_search)

<http://www.personal.kent.edu/~rmuhamma/Algorithms/MyAlgorithms/GraphAlgor/depthSearch.htm>

<http://www3.algorithmdesign.net/handouts/DFS.pdf>

