Topological Sorting

Manju Muralidharan Priya
CS 560
Computer Science Department
Illinois Institute of Technology
Chicago, Illinois 60616
pmanjumu@hawk.iit.edu

Abstract - **A topological sort is used to arrange the vertices of a directed acyclic graph in a linear order. In this paper we introduce topological sorting and discuss algorithms for the same, along with its properties and applications. This paper discusses directed acyclic graphs with interdependent vertices. Partial ordering and total ordering relation to topological sorting has also been discussed in this paper. This paper serves as an introductory document for the topic of topological sorting.**

Keywords - Topological sort, Directed acyclic graph, ordering, sorting algorithms.

INTRODUCTION

I. *Problem definition*

In graph theory, a topological sort or topological ordering of a directed acyclic graph (DAG) is a linear ordering of its nodes in which each node comes before all nodes to which it has outbound edge.
i.e., if a DAG has vertices *u* and *v* and edge from *u* to *v*, in the sorted order u must appear before *v*.
Some notations and definitions used in this paper are defined below.

Definition 1.1 A Directed acyclic graph is a directed graph where no path starts and ends on the same vertex. In a DAG with nodes *x* and *y* with edge *x->y* there is no *y->x*. A DAG is depicted as D= (V,E), where V is a vertex and E is an edge between two vertices.

Definition 1.2 Topological sort A topological ordering *ordD*, of a directed acyclic graph D = (V, E) maps each vertex to a priority value such that *ordD(x) < ordD(y)* holds for all edges.

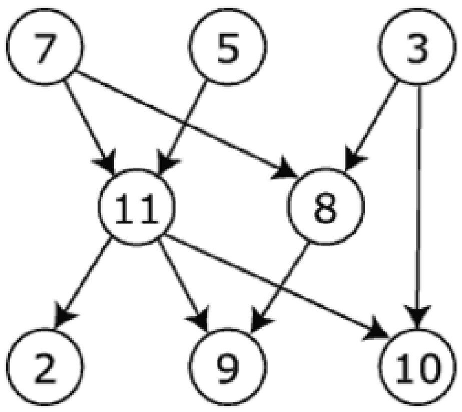

FIGURE 1
SAMPLE DIRECTED ACYCLIC GRAPH.

One of the many topological orders of the FIGURE 1 is

7 -> 5-> 3-> 11-> 8-> 9->2-> 10

ALGORITHMS FOR TOPOLOGICAL SORTING

Many algorithms for topological sorting have been published. Some of the important algorithms are discussed.

I. *Breadth First Search – Kahn (1969)*

The very first algorithm was written by Kahn (1969) [7] in this algorithm (lift from Wikipedia). In this algorithm, the sorted order is started by accessing the nodes in the graph which do not have any incoming edges. As topological sort can be only done on acyclic graphs, this set must contain at least one element that does not have any incoming edges. Graph traversal is started with the vertices in the above set. The vertices on the outgoing edge are considered and that edge is removed. That node is checked again for anymore incoming edges, if there are none it is inserted into the set of sorted elements.

The algorithm is as follows

```
L ← Empty list that will contain the sorted elements
S ← Set of all nodes with no incoming edges
while S is non-empty do
    remove a node n from S
    insert n into L
    for each node m with an edge e from n to m do
        remove edge e from the graph
        if m has no other incoming edges then
            insert m into S
if graph has edges then
    return error (graph has at least one cycle)
else
    return L (a topologically sorted order)
```

The solution for this breadth first search solution for FIGURE 1 will be

7-> 5-> 3-> 11-> 8-> 2-> 9-> 10

This algorithm uses a breadth first search. This technique finds the shortest path solution from start to end.

II. *Depth first Algorithm*

Depth First algorithm traverses the graph from the start node and progresses downwards. When a node with more incoming nodes is detected, back tracking is done and a previous level node is traversed.
The algorithm is as follows [3]

```
DFS(Node start) {
        initialize stack s to hold start
        mark start as visited
        while(s is not empty) {
        next = s.pop()
        for each node u adjacent to next
        if(u is not marked)
        mark u and push onto s
        }
}
```

The depth first solution of FIGURE 1 will be

7-> 5-> 11-> 2-> 3-> 8-> 9-> 10

III. *Generating All solutions*

An algorithm to generate all possible solutions for a topological sort was published by Donald Knuth (1964)[5]. This algorithm is an extension of the previous algorithm written by Knuth for interaction between linked and sequential forms of data representation. This algorithm uses a deque D which acts as a counter array for the sort. Backtracking methods are used for multiple traversals of the DAG. Since the need for recursion arises in the algorithm, it is made iterative by the use of a stack for storing certain local variables.

IV. *Dynamic Algorithm For Topological sort*

A new algorithm is introduced by Pearce D and Kelly P [1] ,where a new dynamic algorithm for topological sorting is introduced and compared against other algorithms for improved time complexity.

This algorithm uses both forward depth first search and a backward depth first search from two edges and sorts the elements accordingly into two lists. The two lists are sorted separately in the topological order and merged together. The algorithm checks for cycles in the graph only during the forward depth first search. The stated time complexity for the given algorithm is

$$\boldsymbol{\Theta((\delta xy \log \delta xy) + \ll \delta xy \gg)}$$

PROPERTIES

Topological sort have certain properties that they possess. These properties can be constrains on the input of the topological sort or the properties of the output. Properties of a topological sort are discussed in this section.

I. Uniqueness property

The topological sort's output is not a unique one. Every topological sort can have multiple solutions depending on the type of algorithm used for sorting.
The sort solution also depends on the way the algorithm peruses through the graph, breadth first or depth first sort.
Certain DAGs have exclusively one solution, if they are a list.

II. Relation with partial orders

Topological sorting algorithms are also used in mathematics to linearly order a partially ordered list.
A partially ordered set/list has elements which are related to each other with an inequality relation. This set/list is used as an input for the topological sort algorithm and the out put produced is a total ordered list which is ordered in either an ascending or descending order of the elements.

Definition 3.1 An inequality is represented with $\geq$ symbol.

Definition 3.2 A Partial order contains elements of the form, $x \geq y$, $z \geq x$. A total order is of the form z, x, y which linearly orders the partial order.

III. Other properties

DAGs must have at least one vertex what does not have any incoming edge. Topological sorts can be only done on graphs that are acyclic. Edges on a cyclic graph cannot be removed for linear ordering because of cyclic edges between two or more vertices.

In FIGURE 2, we notice that there is a cycle where node 7 and node 11 are inter-dependent, and hence an order cannot be determined. Hence an acyclic graph is mandatory.

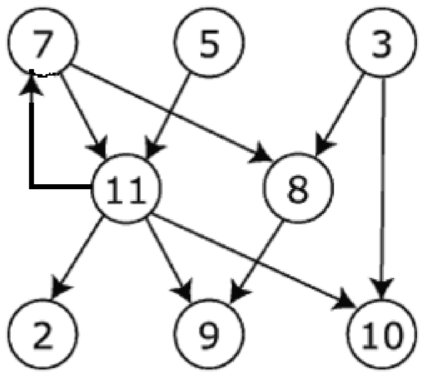

FIGURE 2
SAMPLE CYCLIC GRAPH FOR FIGURE 1

APPLICATIONS

Topological sorting can be used in a lot of different scenarios that involve scheduling a number of tasks which have some inter dependencies.

- While creating complex database tables, some tables have interdependencies . Topological sort can determine the order in which we create them.
- Determining what order to take courses and their pre-requisites in to complete a degree.
- Formulating a lesson plan for a course
- Tsort algorithm in UNIX rearranges source files to define all the methods before they are used in the file.
- It is used to detect cycles in a graph.

UNIX AND TOPOLOGICAL SORTING

UNIX uses topological sort as an inbuilt keyword. This keyword allows the object files of the source to be ordered sequentially so the linker can process them in the same order.

One application of tsort in UNIX is to arrange and declare all the functions in a program before they are used.

Tsort uses a command line input to read the file name and the option for what has to be done with the file[6]

```
tsort [option] [file]
```

CONCLUSION

Topological sort has been introduced in this paper. The properties for the input of the topological sort, i.e. a directed acyclic graph, are discussed. The problem for topological sorting has been defined along with the notations used in the paper. Different algorithms have been explained using a sample directed acyclic graph and the solutions have been found.

REFERENCES

[1] Pearce D, Kelly P , A Dynamic Topological Sort Algorithm for Directed Acyclic Graphs *http://www.doc.ic.ac.uk/~phjk/Publications/DynamicTopoSortAlg-JEA-07.pdf*

[2] Jianjun Z, Müller M, Depth-First Discovery Algorithm for incremental topological sorting of directed acyclic graphs,, *Information Processing Letters 88 (2003) 195–200*

[3] Washington University CSE332 Course slides, *http://www.cs.washington.edu/education/courses/cse332/10sp/lectures/lecture16.pdf*

[4] University of Iowa, Data Structures course slides, *http://homepage.cs.uiowa.edu/~hzhang/c21/notes/18TopoSort.pdf*

[5] A structures Program to generate All topological Sorting Arrangements (1974), Donald Knuth, Jayme L. Szwarcfiter, *Literate Programming Donald Knuth (1992).*

[6] UNIXHelp for Users, *http://unixhelp.ed.ac.uk/*

[7] Wikipedia, *http://en.wikipedia.org/wiki/Topological_sorting*