Identifying Pair of Parallel Edges in Directed Graphs when treated as Undirected Graphs

Anshul Anand
Indian Institute of Information Technology , Allahabad
Email: icm2014501@iiita.ac.in

Akanshu Gupta
Indian Institute of Information Technology , Allahabad
Email: iwm2014501@iiita.ac.in

Amit Khanna
Indian Institute of Information Technology , Allahabad
Email: iim2014004@iiita.ac.in

***Abstract*—Consider a graph G with a given Adjacency Matrix A , in this work we propose an algorithm to find all pairs of parallel edges in a given directed graph when treated as undirected graph. Further we evaluate the proposed algorithm's time complexity against existing/naive approaches to the problem of finding all pairs of parallel edges in a given directed graph when treated as undirected graph.**

I. Introduction

An interconnection of points is known as Graphs , or in a more formal parlance , A graph $\boldsymbol{G}$ is a set of $\boldsymbol{E}$ and $\boldsymbol{V}$, where $\boldsymbol{E}$ denotes the set of edges and $\boldsymbol{V}$ denotes the set of Vertices . Here a vertice $v \in \boldsymbol{V}$ represents a point or node and an edge $e \in \boldsymbol{E}$ is a connection between two points vertices. If an edge e_{ij} connects the vertices v_i and v_j then v_i and v_j are known as **endpoints** of that edge.

If all connections are unidirectional then the graph is known as **directed graph** or digraph. In case of all edges being bi-directional we call the graph undirected graph. A vertex u is **adjacent** to vertex v if they are joined by an edge.
Two edges connecting the same pair of points (and pointing in the same direction if the graph is directed) are called **parallel** or **multiple/multi edges.**

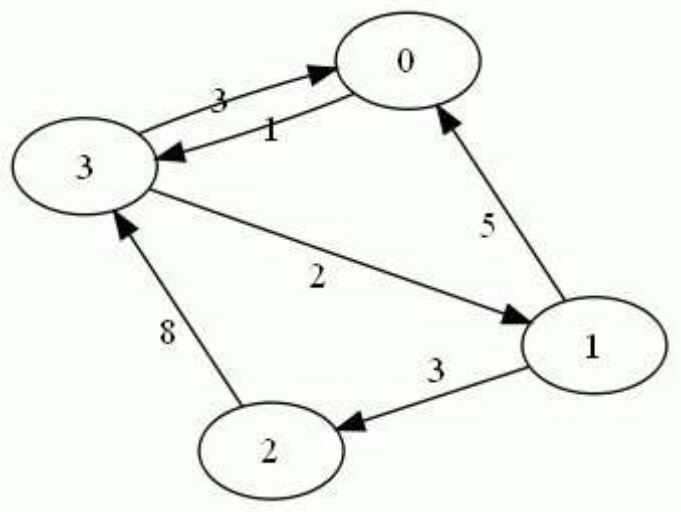

Fig. 1. An example diagraph showing parallel edges

In the given digraph the vertices 1 & 3 have two parallel edges between them .

One of the most popular form of representing a graph is called the Adjacency Matrix . An adjacency matrix representation for a simple graph or digraph G = (V,E) is a |V|× |V|matrix A, where A[i,j] = 1 if there is an edge from vertex i to vertex j; A[i,j] = 0 otherwise.

In our work we will be finding all the parallel edges in the graph provided the adjacency matrix , we will be considering the direction to be immaterial for accounting parallel and consider common end points to be sufficient for edges to be parallel .

II. Motivation

We have to traverse the graph atleast once to find the parallel edges , while doing traversal of an adjacency matrix for two nodes to check for common end points via a naive loop based approach we end up repeating a particular edge twice or checking the case for a single node when both end points are the same . Instead we can skip checking for these points (i.e self loops or for the same end points two times) and save the pair only once. Hence this leads to a decrease in time complexity , reduction of which is our primary motivation

III. Algorithm Proposed

A. Algorithm 1 - Naive Approach

Algorithm 1 Naive Approach

$i \leftarrow 0, j \leftarrow 0, S \leftarrow \phi$**for** i from 1 to n: **do**
for j from 1 to n: **do**
  if i == j **then**
    continue
  end if
  if $A[i][j] >= 1 \& A[j][i] >= 1$ **then**
    $S \leftarrow ((i, j), (j, i))$
  end if
end for
end for

B. Algorithm 2 - Proposed Algorithm

Algorithm 2 Proposed Algorithm

$i \leftarrow 0, j \leftarrow 0, S \leftarrow \phi$**for** i from 1 to n-1: **do**
  for j from i+1 to n: **do**
    if $A[i][j]$ >= 1&$A[j][i]$ >= 1 **then**
      $S \leftarrow ((i, j), (j, i))$
    end if
  end for
end for

IV. Results

The following are the plots comparing Algorithm 1 (blue) and Algorithm 2 (red) for Number of nodes against time taken by algorithm. Both algorithms were run on randomly generate graphs with size ranging from 1 to 300 nodes.

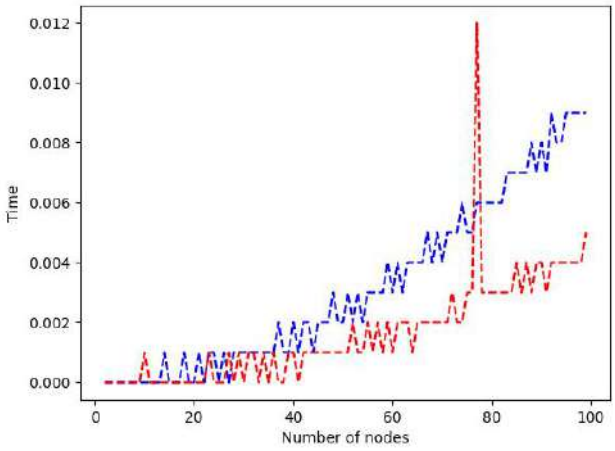

Fig. 2. Plot comparing Algorithm 1 and Algorithm 2

V. Conclusion

For a graph with N number of nodes, time complexity to find parallel edges is O(N*(N-1)/2).

References

[1] Kopka Helmut, W. Daly Patrick, *Guide to LaTeX*, 4^{th} Edition. Available at `http://www.amazon.com`.

[2] Jonathan L. Gross, Jay Yellen, *Handbook of Graph Theory*.

[3] StackOverflow `https://stackoverflow.com/`.