# A Compendious Review on the Research Paper – Circuit Detection in Web and Social Network Graphs

Sreehari R, Rohan R Pillai, Indulekha T.S

## I. INTRODUCTION

Every single day, tremendously large amount of data is generated, and the significant role that the web, social networks, and networks play in enabling this is something that needs no introduction. The rapidly growing abundance of information in social and web networks directly affects the efficiency of information searching algorithms, and algorithms that require variables that can be interpreted with ease when the networks are represented as graphs. Depending on the requirement, and scenario, graph-based representations can be used very flexibly, with the nodes and edges representing different features and relationships at different points in time for the same data. These graphs tend to be highly, and densely connected and are generally depicted as directed graphs for purpose of analysis. Circuits in a web graph can be viewed to be critical problem that reduces the effectiveness of the relationship determination, community detection, and information retrieval. Identifying circuits in a web graph can be used for better and efficient page ranking, web crawling, network analysis and other applications. The sheer vastness of these web graphs and social network graphs can however make it a very expensive operation to detect the significant circuits.

## II. RELATED STATE OF THE ART WORKS & METHODOLOGY

There exist multiple backtracking-based circuit searching algorithms which detect elementary cycles in a graph. However, the primary limitation observed in all such algorithms, is that a node or an edge may be frequently revisited during the searching process.

SZWARCFITER & LAUER proposed an algorithm that detects circuits once they're generated anywhere in the current path under examination, and observes a time complexity of $O(n / e(c + 1))$, for a digraph with $n$ vertices, $e$ edges and $c$ elementary cycles.
LOIZOU & THANISCH improved the abovementioned algorithm by pre-processing the digraph, which reduced the complexity of cycle enumeration process, and then using backtracking and DFS. The pre-processing done detects strongly connected blocks, after which the digraph can be broken down into smaller digraphs. DFS is done to check whether any alternate extension to the path would be viable at that time. In the sparse digraph, the worst-case time complexity observed is exponential.
JOHNSON proposed an algorithm with a time complexity $O((n + e)(c + 1))$ and a space complexity $O(n + e)$, for a graph with $n$ vertices, $e$ edges and $c$ elementary circuits, in which each edge is considered at most twice within the same circuit. For avoiding duplicate circuits, a node $v$ is marked whenever it becomes a part of an elementary path originating from node $s$, and remains unchanged until every path from $v$ to $s$ intersects the marked path at a vertex other than $s$. Also, only nodes that are trivial vertexes in at least one elementary circuit qualify to be considered as a root node for forming the elementary paths.
TIERNAN proposed a method that examines each cycle only once during the entire search.
However, the drawback with this approach is that all minor, irrelevant and insignificant circuits are taken into consideration often.

## III. PROPOSED METHODOLOGY

Communities in a graph represent densely connected nodes, which represents a strong relation between respective nodes, and conversely can be used for identifying the most relevant nodes from the graph. While finding communities directly in a graph by employing known methods can prove to be a complex task, detecting graph properties similar to communities (but exhibit a relation with communities) such as bi-connected components and strongly connected components is a lot easier. Further, since all strongly connected components have circuits in them, they can be detected using the former. The algorithms proposed in this paper require the understanding of three concepts centrally.

1. Strongly Connected Components

A directed graph *G* is said to be strongly connected if there exists a directed path from every vertex $v_i$ to every other vertex $v_j$ . A maximal sub graph of *G* which is strongly connected is called a strongly connected component (*SCC*) of *G*. Notably, *Kosaraju's Algorithm* detects the strongly connected components in a graph in linear time – *O(n + e)*.

2. Bi-connected Components

A bi-connected component (*BCC*) of *G* is a sub graph of *G* with no articulation point. A DFS based linear time algorithm with complexity *O(n + e)* can be used to detect all the bi-connected components.

3. Link Analysis using HITS Algorithm

Link Analysis is a technique used for evaluating the relation between nodes of a graph.
The HITS (*Hyperlink-Induced Topic Search*) algorithm, which is employed for link analysis in the algorithms proposed, assigns two scores to the web graph – *hub score* and *authority score*, which are calculated as follows, respectively.

$$hub(v) = \sum_{i=1}^{n} auth(i) \qquad auth(v) = \sum_{i=1}^{n} hub(i)$$

A node is assigned a higher authority score if it is being linked with nodes recognized as hub information and vice versa.

It is noted that although traditionally social networks use digraphs, in some cases, undirected graphs would prove to be more effective in representing relationships. Analysis reflecting these cases was also recorded and made available.

In the initial step for both algorithms proposed, using HITS, the authority scores are determined for all nodes. Both algorithms also require a definition of a threshold value, $\delta$, which can be set as per requirement.
The first algorithm proposed eliminates retrieval of insignificant circuits by improving upon the conventional method for cycle detection using SCCs, by considering only nodes with an authority score > $\delta$ in the DFS step during detection of the SCCs.
In the second algorithm proposed, a triad of nodes $< v_1, v_2, v_3 >$ is identified such that $a_{score}(v_3) - a_{score}(v_2) \leq \delta$ and $a_{score}(v_2) - a_{score}(v_1) \leq \delta$. Next, all possible circuits that can be created by connecting the triads found are explored. If $v_3$ of a triad $T_1$ is $v_1$ of a triad $T_2$, then the two triads are considered to be connected. This process is repeated until *k*-circuits are identified.

## IV. RESULTS

The algorithms were developed by studying the relations between SCCs, circuits and communities.
First, the results of their study and analysis of the relationship between community structure, circuits, BCCs, and SCCs considering different graph samples is presented, and a tabulated record illustrating the same is made available. It is then pointed out that not all communities are made up of SCCs. Sample graphs have been used to illustrate that communities made with SCCs have circuits present, and consist of highly coherent nodes, which signifies the relevance of the respective circuits.
Although the proposed algorithms are claimed to have been tested on various random graphs with different initial hub vectors and $\delta$s, no additional data concerning the same have been provided.

## V. CONCLUSION

Identifying the significant circuits in web and/or social network graphs can prove to be vital for searching, and effectively using the data that can be interpreted from the network. The primary issue with existing circuit detection algorithms for web and social network graphs has been comprehensively pontificated to be the extortionate overhead that is incurred due to redundant circuit traversals, and consideration of irrelevant, insignificant circuits, which prove to have little to no influence on the network itself.
Thus, this paper proposes and analyses two methods, which perform link analysis first, and then proceed to identify the *k*-most relevant and significant circuits in a respective graph, that qualify the required conditions, which are numerically represented by the thresholds defined.