Isolation and Vulnerability Index : Computational approach for uncertainty quantification in disaster risk scenarios. Remote Foreign Training 2020

Harshal Dupare

Indian Institute of Techonolgy Kharagpur (IIT Kharagpur), India
National Institute for Space Research (INPE), Sao Jose dos Campos/SP, Brazil

Abstract

This Report includes the description of the Isolation Index(Jeferson Feitosa) and a New Algorithm(Harshal Dupare) with time complexity $O(V+E)$ per node which is improvement on the Previous Algorithm(Jeferson Feitosa) with time complexity $O(V \cdot E \cdot logE + V^2)$ per node and the results comparing the Index with Vulnerability Index. For the implementation of graphs we have used python-igraph library, the complexity of the previous algorithm depends on the library used, but in the best case it is $O(V^2 + V \cdot E \cdot logV)$. The improvement in the new algorithm amounts for change in the runtime of algorithm from about 1-3min to about 1-2sec for a graph with size $V = 100$ and $E = 800$ which is equivalent to 100 times improvement in the runtime.

Under Guidance of: ***Prof. Leonardo Santos****, National Center for Monitoring and Early Warning of Natural Disasters (Cemaden), Sao Jose dos Campos/SP, Brazil, santoslbl@gmail.com*

Major Collaborators: ***Jeferson Feitosa****, Sao Paulo State University (UNESP), Sao Jose Dos Campos/SP, Brazil, jeferson.feitosa8@gmail.com*

GitHub Link

1 Introduction

As the effect of global change due extreme weather events are expected to increase in frequency and intensity and cause more social and economic impacts in several sectors,such as Critical Infrastructures, like Transportation systems.

Hence it becomes extremely important to develop proper methods to assess the possible impact of disaster even before they happen namely the method for for the Disaster Risk Reduction (DRR). In DRR Community "Vulnerability" is a key concept (Wisner et al.,1994). The measurement and mapping of vulnerability constitute a subject of global interest.

The Complex Networks approach may offer a valuable perspective considering one type of vulnerability specially related to DRR on critical infrastructures which can be modeled as Complex Networks: the topological vulnerability (Santos et al., 2019).

2 Isolation Index

Given a graph $G = (N, L)$, let L_k be the set of edges connected to a node k where $k \in N$. The isolation index of the node k is given by Q_k, which is the number of "infinite length paths" between any pair of nodes in the graph $H = (N, (L - L_k))$. This is equivalent to the number of disconnected pairs of nodes (i.e. there does not exist a path joining one node to another). The intuition behind the Isolation index is the following: it quantifies the inaccessibility in the network when a specific element (node or edges) is disconnected, like in a flooding or construction work.

3 Algorithm

3.1 Definitions

Connected Component : A connected component C of an undirected graph G is a maximal set of nodes such that each pair of nodes $u, v \in C$ are connected by a path. Connected components form a partition of the set of graph vertices, meaning that connected components are non-empty, they are pairwise disjoints, and the union of connected components forms the set of all vertices.

If we revisit the definition of Isolation Index we can observe that after we disconnect a node form the graph we might get some set of connected components say $C_1, C_2, C_3 ...., C_m$ and for any node pair $u, v \in G$ if v, u belong to same connected component then they are connected else they are disconnected i.e. if $v \in C_i$ and $u \in C_j$ if $i = j$ then v, u are connected else v, u are disconnected. Hence if we know all the connected components of $H = (N, (L - L_k))$ then we can calculate the Isolation index of k.

We will use this fact in calculating the Isolation index with the following Algorithm.

3.2 Algorithm pseudocode

Isolation Index without normalization

Isolation_Index(G,v):

$H \leftarrow G.disconnect\_node(v)$
$component\_size \leftarrow Connected\_Component(H).sizes$
$total \leftarrow G.node\_count$
$isolation\_index \leftarrow 0$
for $component\_size \in component\_sizes$:
$total \leftarrow total - component\_size$
$isolation\_index \ += 2 \cdot total \cdot component\_size$
end for
return $isolation\_index$

Normalized Isolation Index

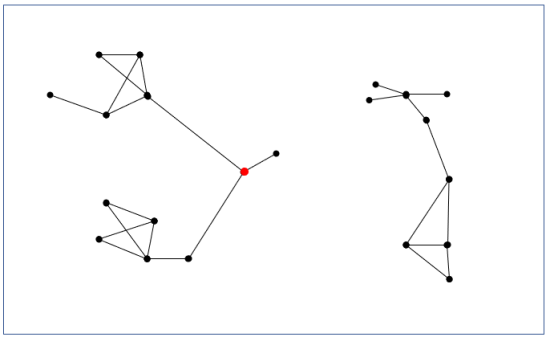
$base\_index \leftarrow Isolation\_Index(G, None)$
$V \leftarrow G.node\_count$
$max\_index \leftarrow V \cdot (V - 1) - base\_index$

Normalize_Index(Isolation_Index_List):

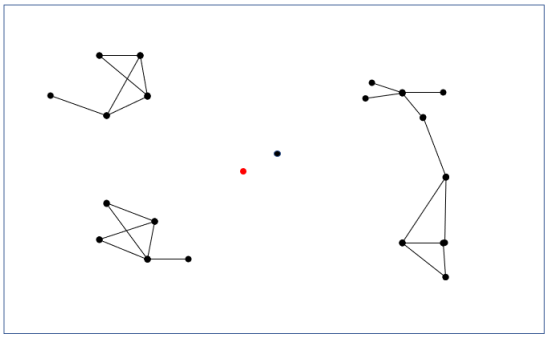
$Normalized\_List = \Phi$
for $isolation\_index \in Isolation\_Index\_List$:
$Normalized\_List.add(\frac{isolation\_index - base\_index}{max\_index})$
end for
return Normalized_List

3.3 Algorithm process Diagram for a small example

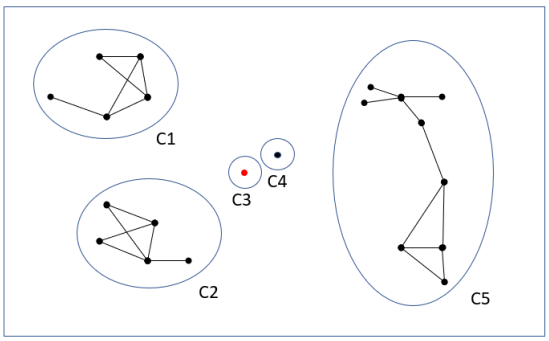
1. Node of interest is the red color node we will show the computation of isolation index for it, same process follows for all other nodes.



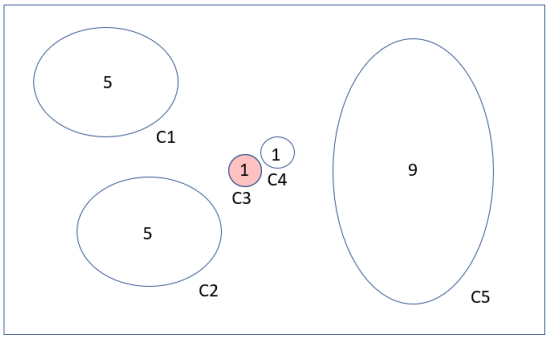
2. Disconnect the red node gives the resultant graph H.



3. Identifying the connected components

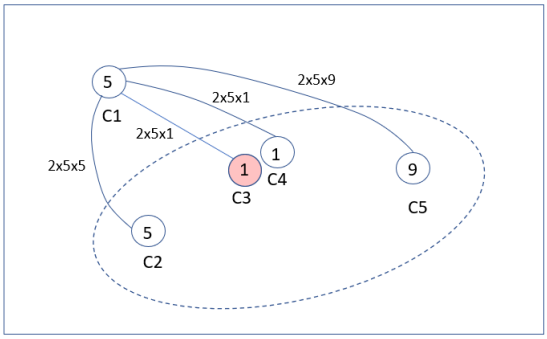


4. Getting the sizes of the connected components

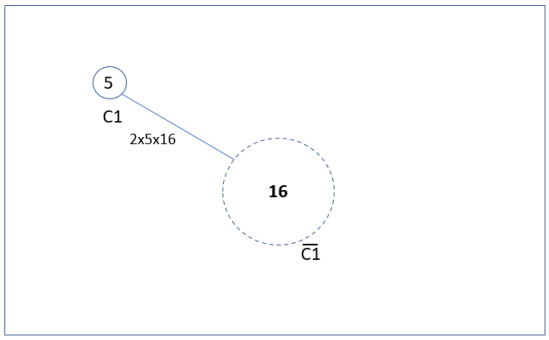


5. Initially set Isolation index of node to 0. The lines joining 2 components represent the number of disconnected ordered pairs for nodes between them. The diagram shows calculation for the 1^{st} component C_1 w.r.t all other components (This

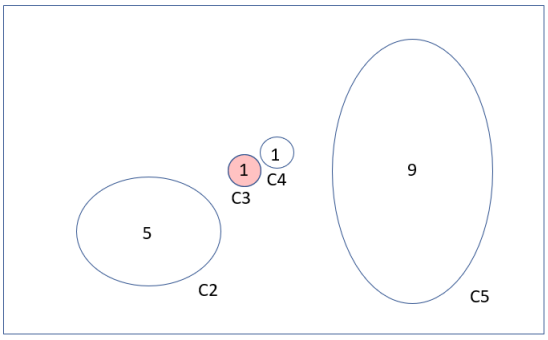
step is shown just for explanation sake in algorithm we directly calculate by the next method)



6. Simplified calculation for 1^{st} component using the compliment of the component as $\bar{C_1}$ add the total to the isolation index count. This is where the variable $Total$ is used for representing the number of nodes in the compliment component for each component as we progress to calculate the isolation index



7. Moving on to the contributions from the next component, hence removing the 1^{st} component as it's contribution is already taken into account. Repeat this process (from 5/6) for all other components and adding the contribution from each component gives the Isolation index.



The above diagrams represent the algorithm for Isolation Index without normalization, to account for the normalization we simply calculate the base-Isolation-Index and maximum isolation index of graph and use it in the last stage to normalize the Isolation index.

$$Normalized\_Index = \frac{Isolation\_Index - Base\_Index}{Max\_Index}$$

3.4 Algorithms Complexity

Let E is number of edges in G and V number of nodes in G, let time-complexity of $Isolation\_Index(G, v)$ be $II(V, E)$, time-complexity of $G.disconnect\_node(v)$ be $DC(V, E, v)$, time-complexity of $Connected\_Component(H).sizes$ be $CC(V, E)$ and number of Connected Components be C

Hence,

$$II(V, E) = DC(V, E, v) + CC(V, E) + C$$

As number of connected components is at max V

$$C = O(V)$$

Since connected components can be found by BFS or DFS and sizes can be returned in C time we have

$$CC(V, E) = O(V + E)$$

And as node can be disconnected and new graph can be returned in $O(V + E)$ time.

$$DC(V, E, v) = O(V + E)$$

we have

$$II(V, E) = O(V + E)$$

4 Results

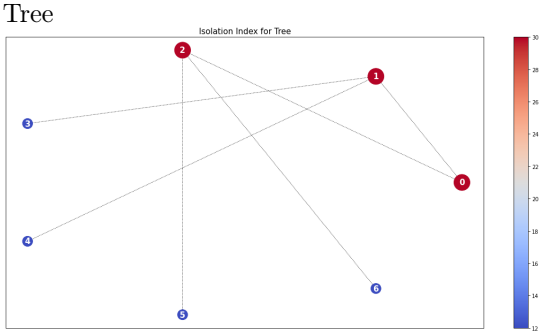
4.1 Isolation Index

The table below represents the values of Isolation Index for the types of graphs on 7 vertices

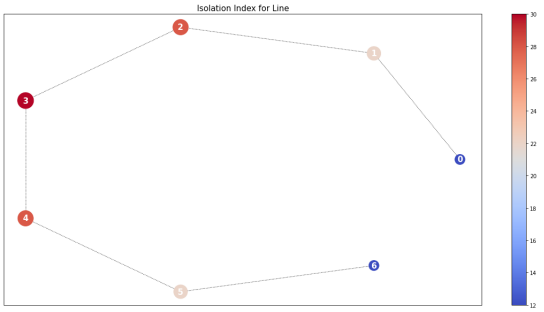
Index	Isolation						
node	0	1	2	3	4	5	6
Graphs							
tree	30.0	30.0	30.0	12.0	12.0	12.0	12.0
star	42.0	12.0	12.0	12.0	12.0	12.0	12.0
complete	12.0	12.0	12.0	12.0	12.0	12.0	12.0
line	12.0	22.0	28.0	30.0	28.0	22.0	12.0
ring	12.0	12.0	12.0	12.0	12.0	12.0	12.0

Graphs for visualizing of the Isolation index

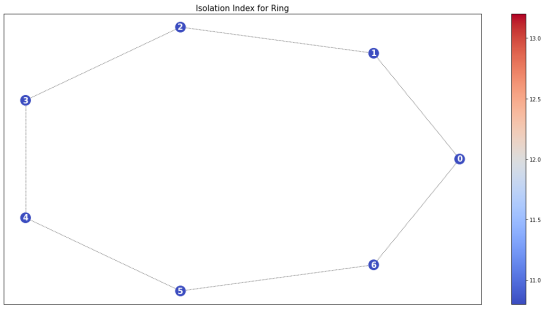
1. Tree



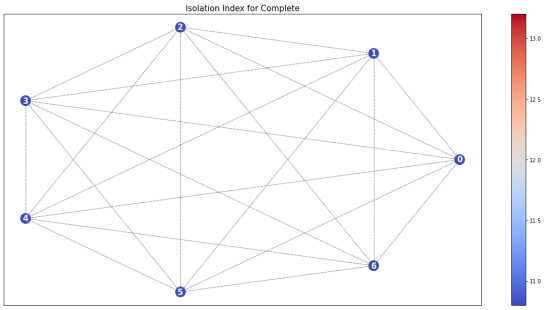
2. Line



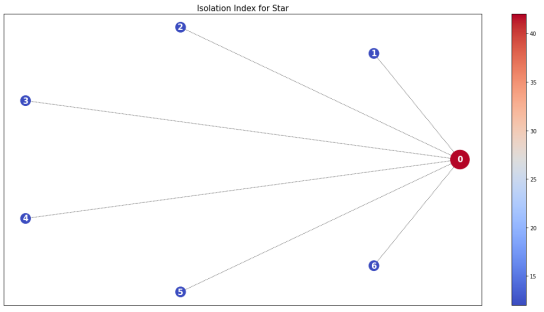
3. Ring



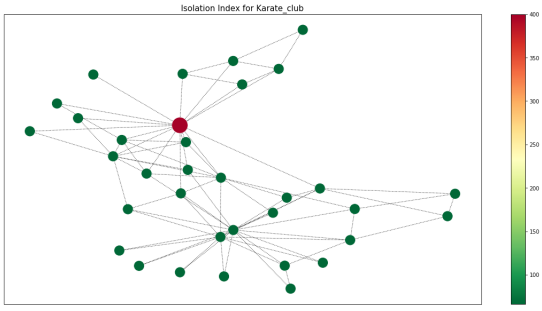
4. Complete



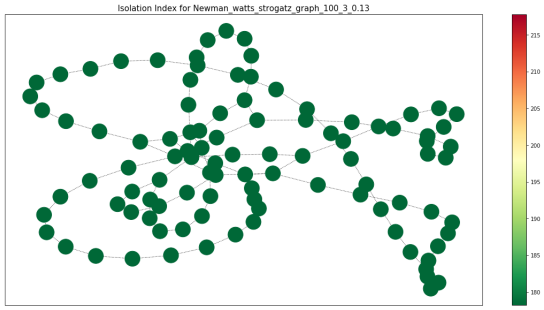
5. Star



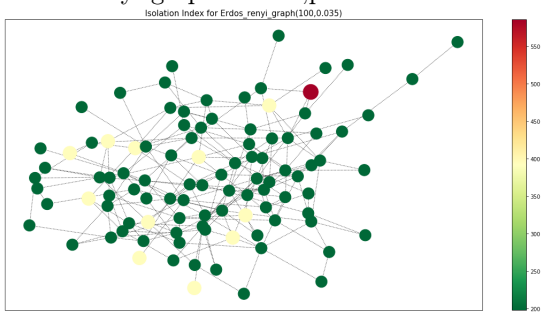
6. Karate Club graph



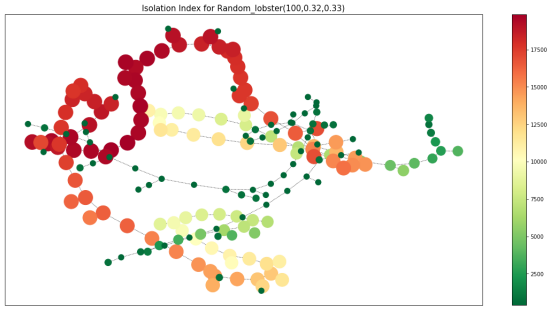
7. Newman Watts Strogatz graph n=100,k=3,p=0.13



8. Erdos Renyi graph n=100,p=0.035



9. Random Lobster graph n=100,p1=0.32,p2=0.33



4.2 Vulnerability Index

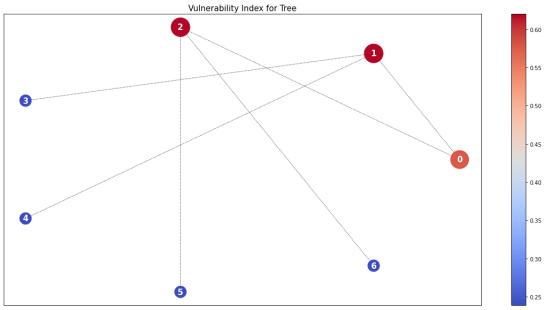
The table below represents the values of Vulnerability Index for the types of graphs on 7 vertices

Index	Vulnerability						
node	0	1	2	3	4	5	6
Graphs							
tree	0.577	0.620	0.620	0.239	0.239	0.239	0.239
star	1.000	0.259	0.259	0.259	0.259	0.259	0.259
complete	0.286	0.286	0.286	0.286	0.286	0.286	0.286
line	0.220	0.425	0.522	0.552	0.522	0.425	0.220
ring	0.322	0.322	0.322	0.322	0.322	0.322	0.322

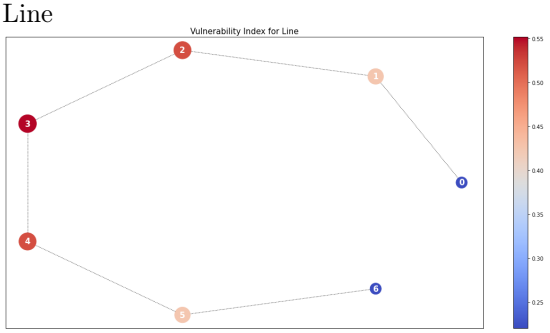
Graphs for visualizing of the Vulnerability index

1. Tree

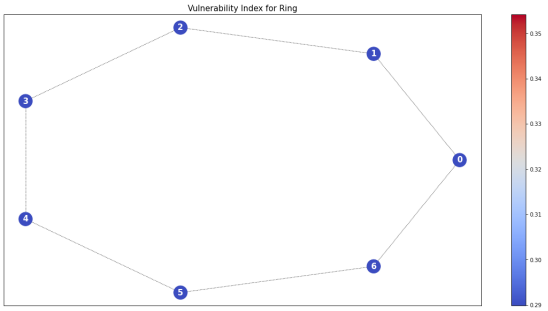
1. Tree



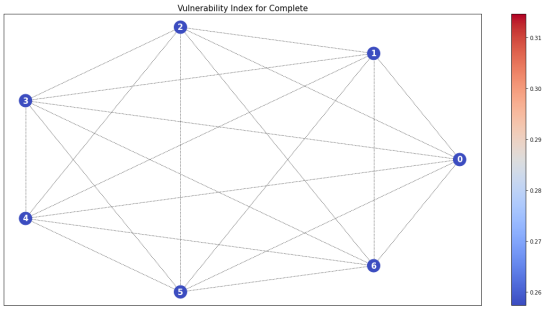
2. Line



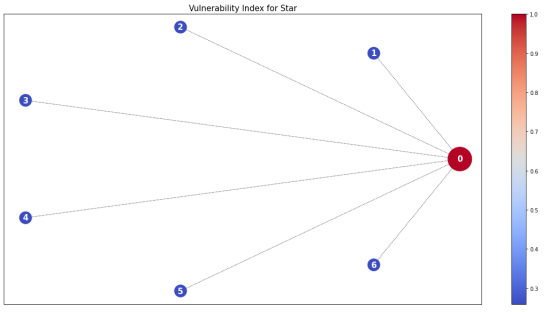
3. Ring



4. Complete



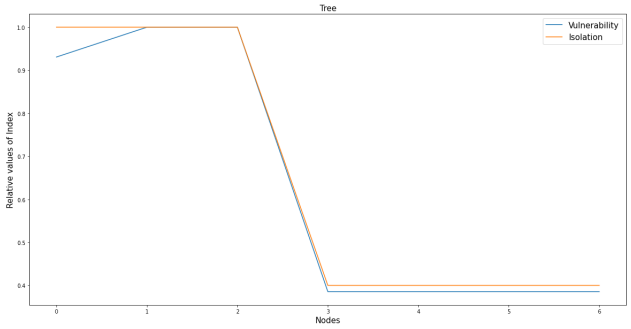
5. Star



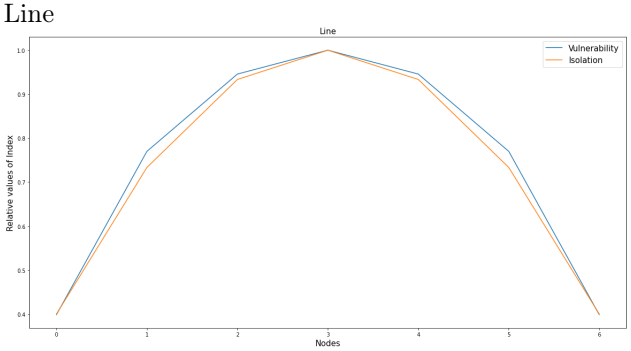
4.3 Comparison among both index

Below are the graphs displaying relative values of index for all 5 graphs.

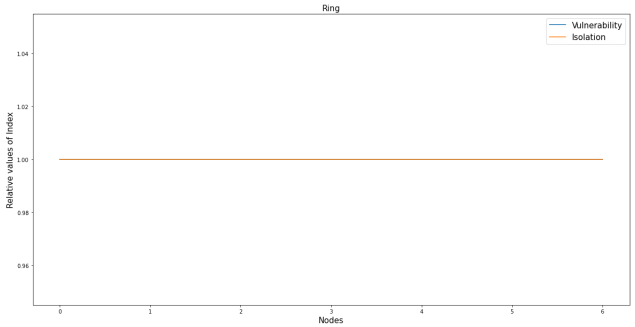
1. Tree



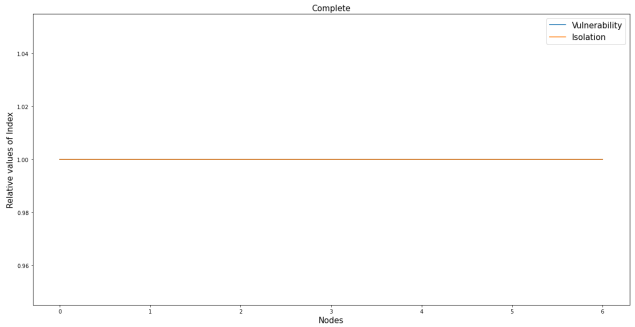
2. Line



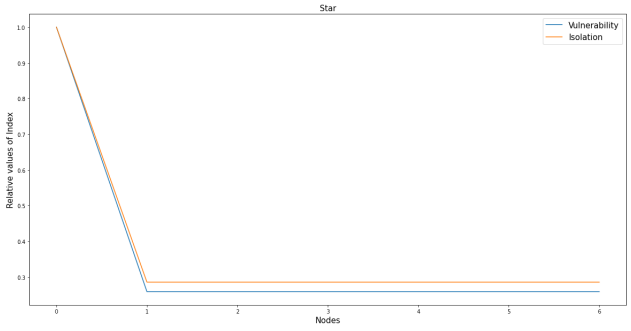
3. Ring



4. Complete



5. Star



5 Contributions

- Highly efficient algorithm to calculate the isolation index
- Method to Normalize the index, accounting for the differences in graph sizes and structure.
- The proposed algorithm also reveals the insights and relationship between the connected components and isolation index which leads to the faster calculation of the index.
- With the help of different results we observe the diversity and power of isolation index to identify the critical points of the network.
- We also study the relationship between the Isolation index and Vulnerability Index and find that they seem to be correlated.