Measuring and Improving the Core Resilience of Networks

Ricky Laishram[1] Ahmet Erdem Sarıyüce[2] Tina Eliassi-Rad[3]
Ali Pinar[4] Sucheta Soundarajan[1]

[1]Syracuse University, Syracuse, NY, USA

[2]University at Buffalo, Buffalo, NY, USA

[3]Northeastern University, Boston, MA, USA

[4]Sandia National Laboratories, Livermore, CA, USA

The Web Conference 2018

Introduction

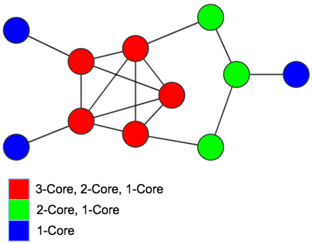
Centrality measures are important for studying the properties of networks. k-Core is one such centrality measure.

k-Core: Maximal subgraph in which every node has at least k neighbors.

Nodes in higher k-cores are considered to be more central to the network.

For example:

- In the autonomous systems (AS) network, the nodes in the higher k-cores are the more important ones.
- We might want to allocate more resources (security, capacity etc.) for these nodes.



Motivating Applications

- Anomaly detection: CORE-A (Shin et al. 2016) is a method of detecting anomalous nodes in a network G based on the nodes core number and degree.
- Community detection: By finding the central region of the network, k-core can be used to accelerate community detection (Peng et al. 2014).

Core Resilience

- In many network, edges (or nodes) might get deleted.
- In such cases, it is important to understand how resilient the k-core structure is.

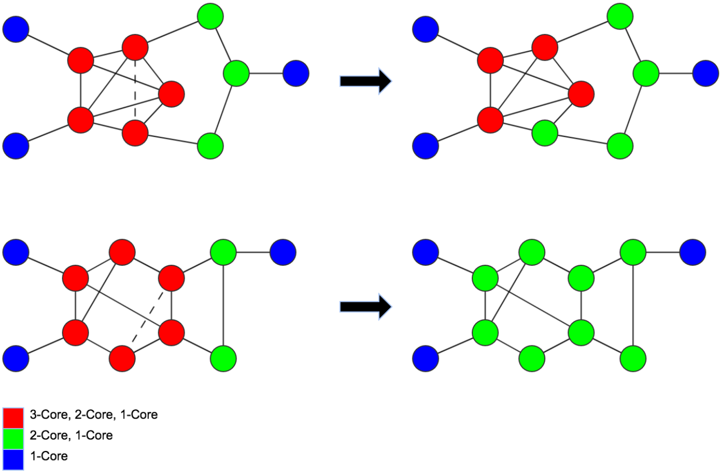
For example, in the AS network, communication links might get dropped or some routers might go offline.

Problems

1. How resilient is the k-core structure of a network to missing edges (or nodes)?
2. How can we characterize the *Core Resilience* with node-level properties?
3. How can we improve the *Core Resilience* of a given network?

1. Core Resilience of Networks
2. Characterizing Core Resilience
3. Improving Core Resilience

Core Resilience of Networks



The Core Resilience tells us how likely is the *k*-core structure to change if there are missing information (edges or nodes).

Computing Core Resilience (The Naive Method)

How do we measure Core Resilience?

1. Randomly remove p edges (or nodes) from G, to get G'.
2. Perform k-core decomposition on G and G'.
3. Order the nodes in G, G' by decreasing (or increasing) core number. Call this L_1, L_2 respectively.
4. Compute rank correlation between L_1, L_2.
5. Repeat multiple times, and for different values of p.

Slow to compute!!

Motivating Application: Anomaly Detection

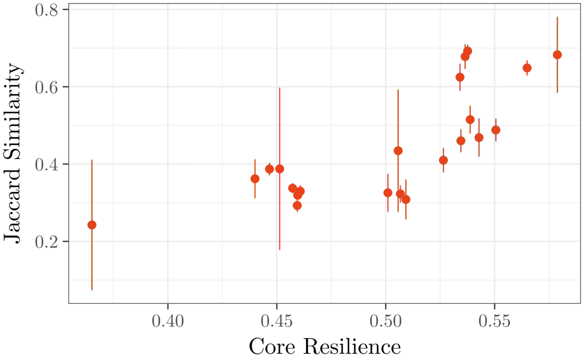
Anomalies are identified as nodes whose ranking when ordered by degree and core number are very different. (Shin et al. 2016).

Hypothesis: If G has high core resilience, the anomalies found in G' will be more similar to those found in G.

Experimental Setup:

- Find anomalies in network G. Call it A.
- Generate a random edge sample, G' of network G.
- Find anomalies in G'. Call it A'.
- Compute similarity between A, A'. (Jaccard Similarity).

Motivating Application: Anomaly Detection



Anomalies found on the subgraph agree more with those in the original network, if the original network have high core resilience.

1. Core Resilience of Networks
2. Characterizing Core Resilience
3. Improving Core Resilience

Characterizing Core Resilience

Computing the core resilience is time consuming!

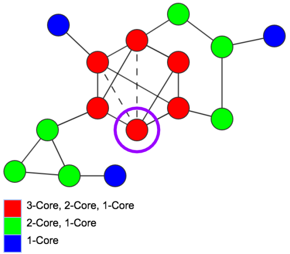
Can we characterize the core resilience with other measures that are easier (and faster) to compute?

We propose three measures:

- Core Strength [node-level property]
- Core Influence [node-level property]
- Core Influence-Strength [network-level property]

Core Strength [Node Level Property]

Core Strength: The minimum number of a node's neighbors that needs to be disconnected for its core number to drop.



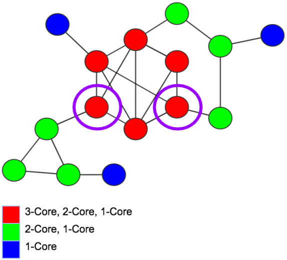
- A node with a high core strength has many extra connections.
- So, it is less likely to drop its core number if its connections are deleted.

Can be computed in $O(|E|)$.

Core Influence [Node Level Property]

Core Influence: A measure of the extent to which the node affects the core numbers of neighbor nodes.

- A node with high core influence is critical to the k-core structure.
- A lot of other nodes are affected if its core number drops.



Can be computed in $O(|E|)$.

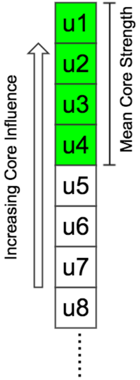
Core Influence-Strength [Network Level Property]

Core Influence-Strength is a proxy for Core Resilience.

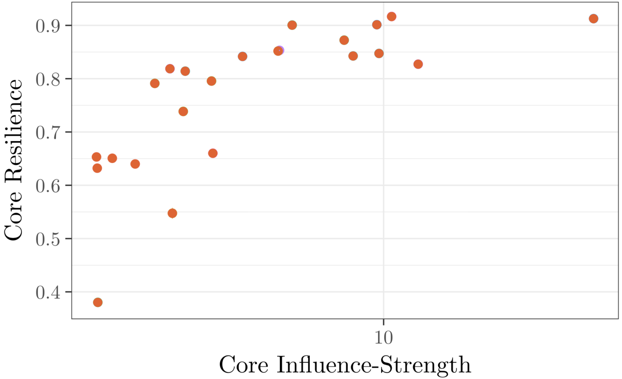
Core Influence-Strength: Mean Core Strength of top n nodes as ranked by Core Influence.

- High CIS → High Core Resilience
- Low CIS → Low Core Resilience

Can be computed in $O(|V| \log |V| + |E|)$.



Core Influence-Strength is correlated to Core Resilience



Networks in which the Core Influence-Strength is high have high Core Resilience.

1. Core Resilience of Networks
2. Characterizing Core Resilience
3. Improving Core Resilience

Motivating Example

- Consider the AS network, G.
- The nodes that are in high k-core would have been allocated more security and capacity resources.
- If the Core Resilience is low, the nodes in high k-core will change frequently.
- So, we will need to reallocate the security and capacity resources frequently.

So, it is important that we improve the core resilience in such networks.

Problem

Given an undirected, unweighted network G and an edge budget b, which b edges should we add to G so that the core resilience of the modified network G' is as high as possible and core numbers are not changed?

Based on our observations on Core Strength and Core Influence, we propose an algorithm called Maximize Resilience of k-Core (`MRKC`).

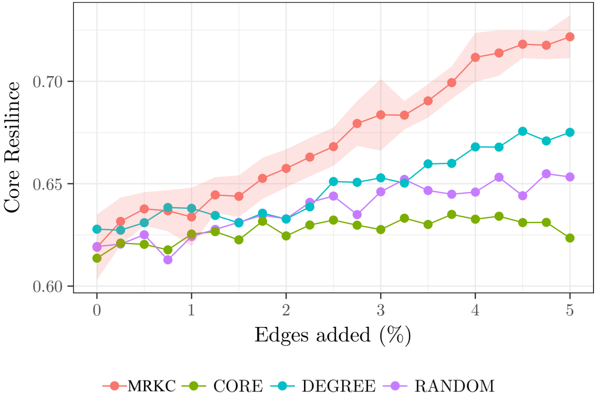
MRKC

- E': Set of edges that can be added without changing core number.
- If we add an edge (u, v) to G, the node with higher core number will not be affected.
- So, for all $(u, v) \in E'$, `MRKC` assigns priority as,

$$\rho(u, v) = \begin{cases} \frac{CI(u,G)}{CS(u,G)}, & \text{if } K(u, G) < K(v, G) \\ \frac{CI(v,G)}{CS(v,G)}, & \text{if } K(u, G) > K(v, G) \\ \frac{CI(u,G)}{CS(u,G)} + \frac{CI(v,G)}{CS(v,G)}, & \text{if } K(u, G) = K(v, G) \end{cases}$$

- At each step, `MRKC` selects the edge with the highest priority and adds it to the network until it reaches the budget.

Experimental Results



MRKC achieves a significant improvement (16%) in the Core Resilience by adding 5% additional edges.

Dataset: Autonomous Systems

Conclusion

We propose Core Resilience of a network.
Important in applications such as anomaly detection, community detection etc.

- Core Strength [node-level property]
- Core Influence [node-level property]
- Core Influence-Strength [network-level property]

Core Influence-Strength can tell us if a network has high (or low) Core Resilience, and it is fast to compute.

Maximize Resilience of *k*-core (`MRKC`): Method to improve core resilience.
Improves the core resilience of a network by 11.1% on average.
The best baseline method gives an improvement of only only 2%.

Thank You

Questions?

rlaishra@syr.edu