Acta Univ. Sapientiae Informatica **14,** 2 (2022) 199–207

DOI: 10.2478/ausi-2022-0012

Computational complexity of network vulnerability analysis

Murat Erşen BERBERLER
Faculty of Science,
Department of Computer Science,
Dokuz Eylül University,
35160, Izmir, TÜRKİYE
orcid=0000-0002-9227-2040
email: murat.berberler@deu.edu.tr

Abstract. Residual closeness is recently proposed as a vulnerability measure to characterize the stability of complex networks. Residual closeness is essential in the analysis of complex networks, but costly to compute. Currently, the fastest known algorithms run in polynomial time. Motivated by the fast-growing need to compute vulnerability measures on complex networks, new algorithms for computing node and edge residual closeness are introduced in this paper. Those proposed algorithms reduce the running times to $\Theta\left(n^3\right)$ and $\Theta\left(n^4\right)$ on unweighted networks, respectively, where n is the number of nodes.

1 Introduction

Networks with complex topology describe a wide range of systems in nature and society. The research for complex networks is a significant area of multidisciplinary studies including applied and theoretical sciences such as informatics, computer science, physics, biology, mathematics, chemistry, social sciences [18, 21]. An interconnection network is composed of nodes and edges between

Computing Classification System 1998: G.2.2, I.2.8
Mathematics Subject Classification 2010: 05C40, 68M10, 68R10
Key words and phrases: Vulnerability, complex networks, design of algorithms, graph algorithms

those nodes. The nodes are the elementary components of the system and the edges connect pair of nodes that mutually interact exchanging information. The robustness of an interconnection network is of prime significance for network designers. Stability measures are considerably intriguing in the research of network vulnerability. The robustness of a network is the measurement of the global strength of its underlying graph. The vulnerability of complex networks can be either node or edge vulnerability meaning that how different classes of network topologies are affected by the removal of a finite number of nodes and/or edges. There exist diversity of vulnerability parameters proposed to measure the stability of networks. The earliest measure of vulnerability of a network is connectivity and it is a considerable measure [16]. It gives the minimum cost to disrupt the network. However it does not consider what remains after disintegration and this is the disadvantage of connectivity. Other improved measures were introduced and studied to overcome this disadvantage, such as integrity [5], toughness [9, 28], tenacity [4, 17], scattering number [15], rupture degree [29, 30]. On the contrary to connectivity, improved parameters not only consider the cost to damage a network but also how badly the network is damaged. These measures are efficient if the underlying graph has become disconnected or trivial after network failure. The problem of measuring these stability parameters of graphs is NP-complete in general from an algorithmic point of view [7, 8, 10, 11, 14, 19, 22]. Therefore these parameters are not efficient enough in the study of complex networks. The notion of residual closeness is introduced recently as a novel graph vulnerability measure by Dangalchev [6]. Residual closeness is a graph-based approach for network vulnerability analysis. This parameter measures the vulnerability of networks more susceptible than some other parameters in being. The main object of residual closeness is to measure the vulnerability even when the disruption of nodes/edges do not disconnect the graph. The necessity and advantages of residual closeness for measuring the vulnerability of graphs are detailed by Dangalchev. Instances reveal that the residual closeness can state the robustness of graphs better than or independently from the other measures existing in literature. Residual closeness can be seen the most proper measure for modeling the stability of network topologies in case of potential node or edge disruption [6]. The closeness and residual closeness of a graph has been studied by several authors, including [1, 2, 3, 12, 13, 24, 25, 26, 27, 31, 32, 33].

In this paper, the graphs are simple, finite and undirected without loops and multiple edges. For a graph $G = (V, E)$ with n nodes and m edges, $|V| = n$ and $|E| = m$, respectively. The length of a shortest path between two nodes i

and j in G is the *distance* $d_G(i,j)$. If the nodes i and j are not connected, then $d_G(i,j) = \infty$, and for $i = j$, $d_G(i,j) = 0$.

The paper proceeds as follow. The theoretical background on the residual closeness is presented in Section 2. Section 3 describes the new developed algorithms for computing the node and edge residual closeness that require less computational effort than the existing algorithms in literature. Section 4 concludes the paper.

2 Residual closeness

The *closeness* of a graph is defined as $C = \sum_{i} C(i)$, where $C(i)$ is the closeness of a vertex i and $C(i) = \sum_{j \neq i} 2^{-d(i,j)}$ [6].

Let $d_k(i,j)$ be the distance between vertices i and j in the graph, received from the original graph where all edges incident to vertex k are deleted. Then the closeness after removing vertex k is defined as $C_k = \sum_{i} \sum_{j \neq i} 2^{-d_k(i,j)}$. The *vertex residual closeness* (VRC) of the graph is defined as $R = \min_{k} \{C_k\}$ [6].

Let $d_{(k,p)}(i,j)$ be the distance between vertices i and j in the graph, received from the original graph where only the edge (k,p) is deleted. Then, the closeness after removing edge (k,p) is defined as $C_{(k,p)} = \sum_{i} \sum_{j \neq i} 2^{-d_{(k,p)}(i,j)}$. The *edge (link) residual closeness* (LRC) of the graph is defined as $R = \min_{(k,p)} \left\{ C_{(k,p)} \right\}$ [6].

3 Design of the algorithms

The *all-pair-shortest-paths* problem (APSP) is one of the most well-studied problems in algorithm design. The problem is to compute the shortest-path distance between every pair of nodes, that is finding a path between two nodes of a graph such that the sum of the weights of its connecting edges is minimized, together with a representation of these shortest paths. The Floyd-Warshall algorithm [20] is one of the algorithms most used for determining the least cost path between every pair of nodes in an edge weighted directed graph. For the APSP problem, this algorithm has a worst-case runtime of $\Theta\left(n^3\right)$ for graphs with n nodes. For unweighted undirected graphs, a traditional method to solve the APSP problem of a graph G is to run *breadth-first-search* (*BFS*), once from every node of G which takes $O(nm)$ time.

For computing the node and edge vulnerability in networks via residual closeness, algorithms with polynomial time complexities $\Theta\left(n^4\right)$ and $\Theta\left(n^5\right)$, respectively, have been proposed previously in [33] and [31]. These algorithms were designed by the use of Floyd-Warshall algorithm. Although these algorithms are efficient, they are still time-consuming when applied on large-scale complex networks.

In this section, a polynomial time algorithm is proposed in order to calculate the node and edge residual closeness for any simple finite undirected unweighted connected graph without loops and multiple edges by using the below exploration algorithm *BFS* [23]. The above function *BFS* returns the distances from a source node v to all other nodes in graph G. The adjacency matrix A is used to store the neighbors of each vertex. The running time of function $\mathsf{BFS}(G, v)$ is $O(|V| + |E|)$ and the function $\mathsf{BFS}(G, v)$ runs for all $v \in V(G)$.

```
function BFS(G,v);
Input: Graph G = (V,E), start node v
Output: for all nodes u reachable from v, dist[u] is set to the distance from v to u

for all u ∈ V do
|    dist[u] ← ∞;
end
dist[v] ← 0;
Q ← [v];
while Q is not empty do
|    u ← eject(Q);
|    for all edges (u,w) ∈ E do
|    |    if dist[v] = ∞ then
|    |    |    INJECT(Q,v);
|    |    |    dist[w] ← dist[u] + 1;
|    |    end
|    end
end
```

Algorithm. Node Residual Closeness

Input: Graph $G=(V,E)$ and the adjacency matrix A

Output: Node residual closeness

```
for all w ∈ V do
    for all (w, j) ∈ E do
        A[w, j] = 0;
        A[j, w] = 0;
    end
    for all v ∈ V do
        BFS(G, v);
        for all u ∈ V do
            D[v, u] = dist[u];
        end
    end
    temp = 0;
    for all v ∈ V and u ∈ V do
        if D[v, u] ≠ 0 and D[v, u] ≠ ∞ then
            temp = temp + 1/2^D[v,u];
        endif
    end
    if temp < nrc then
        nrc = temp;
        nrc_vertex = w;
    endif
    for all (w, j) ∈ E do
        A[w, j] = 1;
        A[j, w] = 1;
    end
end
```

Node residual closeness calculation algorithm runs in polynomial time. The inequality of $|E| \leqslant |V|\,(|V|-1)/2$ yields the complexity of $|V|^2$, by the outermost *for* loop, the total time complexity will be $\Theta\left(|V|^3\right)$.

Algorithm. Link Residual Closeness

Input: Graph $G=(V,E)$ and the adjacency matrix A

Output: Link residual closeness

```
for all (i, j) ∈ E do
    A[i, j] = 0;
    A[j, i] = 0;
    for all v ∈ V do
        BFS(G, v);
        for all u ∈ V do
            D[v, u] ← dist[u];
        end
    end
    temp = 0;
    for all v ∈ V and u ∈ V do
        if D[v, u] ≠ 0 and D[v, u] ≠ ∞ then
            temp = temp + 1/2^D[v,u];
        endif
    end
    if temp < lrc then
        lrc = temp;
        lrc_link = (i, j);
    endif
    A[i, j] = 1;
    A[j, i] = 1;
end
```

Edge (link) residual closeness calculation algorithm runs in polynomial time. The inequality of $|E| \leqslant |V|(|V|-1)/2$ yields the complexity of $|V|^2$, by the outermost *for* loop, the total time complexity will be $\Theta\left(|V|^4\right)$.

4 Conclusion

The nodes or the edges which are responsible for fast communication flow can be identified within a network by the use of residual closeness. The nodes and the edges giving the residual closeness of a network are fast in distributing information through the network. We can conclude that vulnerability of complex networks can be more efficiently and sensitively measured by the use of the new proposed algorithms. For unweighted graphs, the shortest paths from one node to all other nodes are computed by the use of *BFS* in the shortest possible time, that is, in complexity $O(|V| + |E|)$. In that case, if *BFS* is run for each node in the graph, that is, n times, then the shortest paths from all nodes to all other nodes are calculated in a much shorter time than all existing algorithms in the literature. Thereafter, at each step, one node/edge is removed from the graph one by one independent from each other and the closeness of the remaining subgraph is computed by the calculation of all pairs of shortest paths. The smallest closeness among all the closeness values of the nodes/edges is determined as the residual closeness of the related graph.

References

[1] A. Aytaç, Z. N. Berberler, Residual closeness for helm and sunflower graphs, *TWMS J. App. Eng. Math.* **7**, 2 (2017) 209–220. ⇒200

[2] A. Aytaç, Z. N. Berberler, Robustness of regular caterpillars, *Int. J. Found. Comp. Sci.* **28**, 7 (2017) 835–841. ⇒200

[3] A. Aytaç, Z. N. Odabaþ Berberler, Network robustness and residual closeness, *RAIRO Oper. Res.* **52**, 3 (2018) 839–847. ⇒200

[4] B. L. Piazza, F. S. Robertst, S. K. Stueckle, Edge-tenacious networks, *Networks* **25** (1995) 7–17. ⇒200

[5] C. A. Barefoot, R. Entringer, H. Swart, Vulnerability in graphs—a comparative survey, *J. Combin. Math. Combin. Comput.* **1** (1987) 13–22. ⇒200

[6] C. Dangalchev, Residual Closeness in Networks, *Physica A* **365** (2006) 556–564. ⇒200, 201

[7] D. Bauer, S. L. Hakimi, E. Schmeichel, Recognizing tough graphs is NP-hard, *Discrete Applied Mathematics* **28** (1990) 191-195. ⇒200

[8] D. E. Mann, *The tenacity of trees*, Ph.D. Thesis, Northeastern University (1993). ⇒200

[9] D. Gusfield, Connectivity and edge disjoint spanning trees, *Inform. Process. Lett.* **16** (1983) 87-89. ⇒200

[10] D. Kratsch, T. Kloks, H. Müller, Measuring the vulnerability for classes of intersection graphs, *Discrete Applied Mathematics* **77** (1997) 259–270. ⇒200

[11] D. Moazzami, An Algorithm for the Computation of Edge Integrity, *International Journal of Contemporary Mathematical Sciences* **6**, 11 (2011) 507-516. ⇒ 200

[12] E. Yiğit, Z. N. Berberler, A Note on the Link Residual Closeness of Graphs Under Join Operation, *International Journal of Foundations of Computer Science* **30**, 3 (2019) 417-424. ⇒ 200

[13] E. Yiğit, Z. N. Berberler, Link failure in wheel type networks, *International Journal of Modern Physics C* **30**, 9 (2019) 1950072. ⇒ 200

[14] F. Li, Q. Ye, X. Li, Tenacity and Rupture Degree of Permutation Graphs of Complete Bipartite Graphs, *Bulletin of the Malaysian Mathematical Sciences Society* **34**, 3 (2011) 423-434. ⇒ 200

[15] H. A. Jung, On a class of posets and the corresponding comparability graphs, *Journal of Combinatorial Theory, Series B* **24**, 2 (1978) 125–133. ⇒ 200

[16] H. Frank, I. T. Frisch, Analysis and design of survivable networks, *IEEE Transactions on Communications Technology COM-18* **567** (1970). ⇒ 200

[17] M. Cozzens, D. Moazzami, S. Stueckle, The tenacity of a graph, *Seventh International Conference on the Theory and Applications of Graphs*, Wiley, New York, 1995, pp. 1111–1122. ⇒ 200

[18] M. E. J. Newman, The structure and function of complex networks, *SIAM Review* **45**, 2 (2003) 167-256. ⇒ 199

[19] M. R. Fellows, S. Stueckle, The immersion order, forbidden subgraphs and the complexity of network integrity, *J. Combin. Math. Combin. Comput.* **6** (1989) 23-32. ⇒ 200

[20] R. W. Floyd, Algorithm 97: Shortest path, *Communications of the ACM* **5**, 6 (1962) 345. ⇒ 201

[21] S. Boccaletti, V. Latora, Y. Moreno, M. Chavez, D. U. Hwanga, Complex networks: structure and dynamics, *Physics Reports* **424** (2006) 175-308. ⇒ 199

[22] S. Zhang, X. Li, X. Han, Computing the scattering number of graphs, *International Journal of Computer Mathematics* **79**, 2 (2002) 179-187. ⇒ 200

[23] T. H. Cormen, C. Stein, R. L. Rivest, C. E. Leiserson, *Introduction to Algorithms* (2nd Edition), McGraw-Hill Higher Education, 2001. ⇒ 202

[24] T. Turacı, Computing the closeness centralities of a type of the probabilistic neural networks, *Romanian Journal of Mathematics and Computer Science* **11**, 2 (2021) 1-12. ⇒ 200

[25] T. Turacı, M. Ökten, Vulnerability of Mycielski Graphs via Residual Closeness, *Ars Combinatoria* **118** (2015) 419-427. ⇒ 200

[26] T. Turacı, V. Aytaç, Residual Closeness of Splitting Networks, *Ars Combinatoria* **130** (2017) 17-27. ⇒ 200

[27] V. Aytaç, T. Turacı, Closeness Centrality in Some Splitting Networks, *Computer Science Journal of Moldova* **26**, 3 78 (2018) 251-269. ⇒ 200

[28] V. Chvàtal, Tough graphs and Hamiltonian circuits, *Discrete Math.* **5** (1973) 215–228. ⇒ 200

[29] Y. K. Li, *On reliability parameters of graphs*, MA Thesis, Northwestern Polytechnical University, 2003. ⇒ 200

[30] Y. K. Li, S. G. Zhang, X. L. Li, Rupture degree of graphs, *Int. J. Comput. Math.* **82**, 7 (2005) 793-803. ⇒200

[31] Z. N. Berberler, E. Yiğit, Computing the link vulnerability of networks in polynomial time, *I. International University Industry Cooperation, R&D and Innovation Congress,* 2017. ⇒200, 202

[32] Z. N. Berberler, E. Yiğit, Link vulnerability in networks, *International Journal of Foundations of Computer Science* **29**, 3 (2018) 447-456. ⇒200

[33] Z. N. Berberler, *Residual closeness in communication networks*, Ph.D. Thesis, Ege University, Turkey, 2014. ⇒200, 202

Received: October 5, 2022 • *Revised: November 13, 2022*