Journal of Algorithms and Computation

journal homepage: http://jac.ut.ac.ir

A Survey on Complexity of Integrity Parameter

Mahmood Shabankhah, *[1]

[1]University of Tehran, College of Engineering, Department of Engineering Science.

ABSTRACT

Many graph theoretical parameters have been used to describe the vulnerability of communication networks, including toughness, binding number, rate of disruption, neighbor-connectivity, integrity, mean integrity, edge-connectivity vector, l-connectivity and tenacity. In this paper we discuss Integrity and its properties in vulnerability calculation. The integrity of a graph G, $I(G)$, is defined to be $min(\mid S \mid +m(G-S))$ where $S \subset V(G)$ and $m(G-S)$ is the maximum order of the components of $G-S$. Similarly the edge-integrity of G is $I'(G) := min(\mid S \mid +m(G-S))$ where now $S \subseteq E(G)$. Here and through the remaining sections, by an I-set (with respect to some prescribed graph G) we will mean a set $S \subset V(G)$ for which $I(G) = \mid S \mid +m(G-S)$. We define an I'-set similarly.

In this paper we show a lower bound on the edge-integrity of graphs and present an algorithm for its computation.

ARTICLE INFO

Article history:
Received 10, May 2015
Received in revised form 11, January 2016
Accepted 2, March 2016
Available online 12, April 2016

Keyword: Integrity parameter, toughness, neighbor-connectivity, mean integrity, edge-connectivity vector, l-connectivity and tenacity

AMS subject Classification: Primary 05C78, 05C70

*Email:*shabankhah@ut.ac.ir*;

1 Introduction

The concept of integrity of a graph G was introduced in [3] as a useful measure of the "vulnerability" of G. If we think of the graph as modeling a network, the vulnerability measures the resistance of the network to disruption of operation after the failure of certain stations. In [3] Barefoot, Entringer and Swart compared integrity, connectivity, binding number and toughness for several classes of graphs. Their results suggested that integrity is well suited to measuring vulnerability in that it is best able to distinguish between graphs that intuitively should have different measures of vulnerability. In [9] we also compared these measures with a new invariant for graphs. We also in [10] showed some interesting and new results.
We use the terminology of Bondy and Murty [4] for the most part; any exceptions will be noted. In particular, we require our graphs to be simple, i.e., no loops or multiple edges are permitted. Here $\lfloor x \rfloor$ and $\lceil x \rceil$ are the usual greatest integer and least integer functions, respectively.

Computational complexity of integrity
Clark, Entringer, and Fellows [5] showed that the vertex integrity problem is NP-complete but that for each fixed value of k it is decidable in time $O(n^2)$ whether an arbitrary graph G of order n satisfies $I(G) \leq k$. This last result is obtained by a simple application of the powerful results of Robertson and Seymour on graph minors.
All graphs considered in [5] are simple, without loops or multiple edges. A simple graph H is a *minor* of the simple graph G if H can be obtained from G by a sequence of operations of the following two kinds:
(a) replace a graph by a subgraph of itself or
(b) contract an edge.
If the minor H of G satisfies $H \ncong G$ then H is a *proper minor* of G. Clark, Entringer and Swart used the following result of Robertson and Seymour [14].
Theorem 1. (Robertson and Seymour). Let F be a minor-closed class of graphs such that some planar graph is not in F. Then there is an algorithm to determine membership of $G = (V, E)$ in F with running time $O(\mid V \mid^2)$.
Consider the following decision problem.
VERTEX INTEGRITY
Input: A graph $G = (V, E)$ and an integer k.
Question: Is $I(G) \leq k$?
Theorem 2. VERTEX INTEGRITY is NP-complete, even for input restricted to planar graphs.
Proof. The problem is clearly in NP since it is easy to verify, given $S \subseteq V$, that

$| S | + m(G - S) \leq k$. □

To show that VERTEX INTEGRITY is NP-hard we reduce from the following decision problem which is known to be NP-complete [7].

VERTEX COVER

Input: A planar graph $G = (V, E)$ and an integer k.

Question: Is there a set $V' \subseteq V$ with $| V' | \leq k$ such that every edge of G is incident with some vertex of V'?

Let $| V | = n$. We can assume $k \leq n - 3$. (Indeed, by the four-color theorem, G has an independent set of at least $\frac{n}{4}$ vertices and so a vertex cover with at most $\frac{3n}{4}$ vertices.)

Given G and k, we describe how to compute in polynomial time a graph G' and integer k' such that $I(G') \leq k'$ if and only if G has a vertex cover of cardinality at most k.

Let W denote the wheel with $2n - 2$ spokes. The order of W is then $2n - 1$. The graph G' is obtained from G adding to each vertex v of G an edge to all the rim vertices of a different copy of W. Let $k' = 2n + k$. The graph G' is planar, since G is planar.

Assume G has a vertex cover $S \subseteq V$ with $| S | \leq k$. Since every edge of G is incident with some vertex of S, $G - S$ has no edges and $m(G' - S) \leq 2n$. This implies $I(G') \leq k + 2n = k'$.

Conversely, suppose $I(G') \leq k' = k + 2n$. Then, for some set $S' \subseteq V'$, $| S' | + m(G' - S') \leq k + 2n$. In particular, $| S' | \leq k + 2n < 3n$,so some component C of $G' - S'$ has order at least $2n - 2$. This, in turn, implies that $| S' | \leq k + 2 < n$ and so, in fact, some component C of $G' - S'$ has order at least $2n$, which implies $| S' | \leq k$.

If any component C of $G' - S'$ contains two or more vertices of G (considered as a subgraph of G') then C has order at least $4n - k > 2n + k$, a contradiction. We may conclude that $G' - S'$ contains no edge of G. But this implies that $G - (S' \cap V)$ has no edges, i.e. $S' \cap V$ is a vertex cover of G of no more than k vertices. □

In contrast to this result, the next theorem shows that it is easy to decide whether an arbitrary graph of order n has integrity at most k, for each fixed value of k.

Clark, Entringer and Fellows first showed that the class of graphs $G_k = \{G \mid I(G) \leq k\}$ is closed under the minor ordering. We remark that this is not true for edge-integrity.

Theorem 3. For every positive integer k, it is decidable in time $O(n^2)$ whether an arbitrary graph G of order n satisfies $I(G) \leq k$.

The principal subject of [6] is the computational complexity of vertex and edge integrity, two "trade-off" metrics of network vulnerability. It has previously been shown that for each positive integer k, the family of graphs $G_k = \{G \mid I(G) \leq k\}$ is a lower ideal in the partial ordering of graphs by minors [5]. This has the consequence that for each fixed value of k, membership in G_k can be decided in time $O(n^2)$ for a graph of order n, by the deep results of Robertson and Seymour [11,12,13,14]. From a practical perspective, the small degree polynomial-time bound is somewhat deceptive. The algorithms are only

proven to exist (the proof is nonconstructive) and even if known, they would involve astronomical constants. It is an important open probem for many of the applications of the Robertson-Seymour theorems to determine whether alternative and "reasonable" small-degree polynomial-time algorithms can be found [8]. In [6] Fellows and Stueckle showed that this is indeed the case for both vertex and edge integrity. It is shown that for each positive integer k, the family of finite graphs G'_k of graphs G with $I'(G) \leq k$ is a lower ideal in the partial ordering of graphs by immersions. Some information on the obstruction sets for these families is obtained. It is shown that for every fixed positive integer k it is decidable in time $O(n)$ for an arbitrary graph G of order n whether $I(G) \leq k$, and also whether $I'(G) \leq k$. For variable k, the problem of determining whether $I'(G)$ is at most k is shown to be NP-complete, complementing a similar previous result concerning $I(G)$. The picture of the computational complexity of the vulnerability metrics of vertex and edge integrity appears now to be quite complete. For fixed k, the vertex integrity class is closed in the minor ordering, and the edge integrity class is closed in the immersion ordering. Thus encouraged to look for small degree polynomial-time algorithms, we have seen that constructive algorithms with time bound $O(n)$ having reasonable hidden constants exponential in k are to be had, while for variable k both problems are NP- complete.

Some bounds for the Edge - Integrity of Trees

In this section we show a lower bound on the edge-integrity of graphs and present an algorithm for its computation.

In [2] Bagga, Beineke, Pippert, Sedlmeyer and Lipman derived a new lower bound on the edge-integrity of graphs, but most of their results concern trees. After showing several bounds on the edge-integrity of trees they presented an algorithm for its computation, along with a proof of the correctness of the algorithm.

The following result in [2] provides a new lower bound for $I'(G)$:

Theorem 4. If G is a graph with edge connectivity λ, then $I'(G) \geq min\{\lceil\sqrt{2n\lambda}\rceil, n\}$.

Let T be a tree, and let z denote the largest number of end-vertices adjacent to a vertex in T. As expected, the results in this section are improvements on the bounds on the edge-integrity of arbitrary graphs.

Theorem 5. If T is a tree with $\Delta(T) \geq \frac{n}{2}$, then $I'(T) = \Delta(T) + 1$.

Proof. Let v be a vertex of degree $\Delta(T) = d$ in T. Assume that $T - v$ consists of r trivial components and s nontrivial components. Let S consist of the s edges joining v to the nontrivial components, and assume that the largest of these components has order n_1. Thus $n \geq n_1 + 2(s-1) + r + 1$. Since $d = r + s$ and $d \geq \frac{n}{2}$, it follows that $n_1 \leq r + 1$. Therefore, the largest component of $T - S$ has order $r+1$. Hence $I'(T) \leq s+r+1 = d+1$. By Theorem 3, $I'(T) = d + 1$. □

The next two theorems in [2] provide information when the maximum degree is less than

$\frac{n}{2}$.

Theorem 6. If T is a tree with $\Delta(T) < \frac{n}{2}$, then $I'(T) \leq \lfloor \frac{n+3}{2} \rfloor$.

Theorem 7. Let $n \geq 5$ and let d satisfy $\frac{n}{4} \leq d < \frac{n}{2}$. Then there exists a tree T having order n, maximum degree $\Delta(T) = d$, and edge-integrity $I'(T) = \lfloor \frac{n+3}{2} \rfloor$.

An immediate consequence of the definition of edge-integrity is that for any vertex v, $I'(T) \leq m(T - v) + d_G(v)$. Under certain circumstances we get a similar lower bound which is sometimes an improvement on Theorem 3, $I'(G) \geq \Delta(G) + 1$.

Theorem 8. Let v be a vertex of a tree T with $d_G(v) = d$. If the smallest component of $T - v$ has order $t \leq d + 1$, then $I'(T) \geq t + d$.

Proof. Let S be any set of edges of T, and let $s =\mid S \mid$ and $m = m(T-S)$. By considering cases, we will show that $s + m \geq t + d$.

First assume $s < d$. Then in $T - S$ at least $d - s$ of the components of $T - v$ are adjacent to v and have no edge removed from them. Hence $m \geq 1 + (d - s)t \geq t + d - s$, so that the desired inequality holds in this case.

Now assume $s \geq d$. Suppose that the conclusion does not hold, so that $s + m < t + d$ and hence $t > m$. Let h denote the number of edges at v which are in s. Then the component of $T - S$ which contains v has order at least $d - h + 1$. Therefore, $d - h + 1 \leq m < t + d - s$, so that $s < d + h$ since $t - 1 \leq d$ by hypothesis. Consequently, some component of $T - v$ contains no edge of s. Such a component is contained in a component of $T - S$, so $m \geq t$. This contradicts our supposition and completes the proof. □

It seems reasonable that further restrictions on the maximum degree should yield improved bounds on the edge-integrity of trees. The next theorem gives an example for $\Delta(T) = 3$.

Theorem 9. If $\Delta(T) = 3$, then $I'(T) \leq \frac{15\lceil\sqrt{2n}\rceil}{8}$.

Clearly, Theorem 5 gives the precise value of $I'(T)$. The upper bound in Theorem 6 is exact for paths with $n \leq 8$, and there are trees with $\Delta = 3$ which achieve the upper bound for each value of $n \leq 10$. However, since the bound of Theorem 6 is linear and that of Theorem 9 is $O(n^{\frac{1}{2}})$, Theorem 9 gives better bounds for large values of n (approximately the same near $n = 20$ and strictly better for $n \geq 28$). With fine-tuning, the bound of Theorem 9 can probably be improved, and it would be interesting to know whether the least upper bound is still $O(n^{\frac{1}{2}})$. Similar results can be obtained for other values of Δ, but the amount of effort required to increase with the magnitude of Δ.

An algorithm for the computation of $I'(T)$

It will be convenient to have the following additional piece of terminology: The pruned tree of T, denoted prT, is the subtree of T that remains when all end-vertices of T are deleted. The algorithm presented here employs several facts about the values of $\mid S \mid$ and

m which satisfy $\mid S \mid + m(T - S) = I'(T)$:
1. If S is a minimal set achieving I', then S does not contain any end-edges, so only edges from prT are candidates for membership in S.
2. Let z denote the largest number of end-vertices adjacent to a vertex in T. From the above, for any minimal S achieving $I'(T)$, $m \geq 1 + z$.
3. Each upper bound on I' yields upper and lower bounds for $s = \mid S \mid$ and $m = m(T - S)$ such that $s + m = I'(T)$, as follows. Suppose $B \geq I'(T)$. Deleting s edges from T leaves exactly $s + 1$ components. Since each of them has at most m vertices, $(s + 1)m \geq n$. Since $m + s \leq B$, $m + \frac{n}{m} - 1 \leq B$. Then m is constrained to lie between the roots of the quadratic equation $m^2 - (B + 1)m + n = 0$. For convenience, denote the larger root by upper(B).

The algorithm successively computes a minimal set S for each candidate for $m = m(T-S)$, and returns the minimum among all the values $m+ \mid S \mid$. Given an upper bound M for $m(T - S)$, we select S by pruning T from the end-vertices inwards, cutting only when we must. Notice that this procedure produces $I'(T)$ and a specific S which achieves $I'(T)$, and also tells how expensive it is to achieve each M inside a specified range. The algorithm as we present it is in "implementation-free" form.

The Algorithm:
Step 1: {check for application of Theorem 6.} Compute $\Delta(T)$. If $\Delta(T) \geq \frac{n}{2}$, return $I'(T) = 1 + \Delta(T)$ and STOP. Otherwise , set ESTIMATE to $\lceil \frac{n}{2} \rceil + 1$.
Step 2: { compute the initial upper bound for $m(T - S)$.} Compute the largest number of end-vertices of a vertex z. Let $M = z + 1$.
Step 3: {Assign initial weights.} Assign weight 1 to each end-vertex. For every vertex v in prT, if v is adjacent to r end-vertices and has degree $r + 1$, assign weight $r + 1$ to v. Some vertices will have no weight. Every initial weight is less than or equal to M.
Step 4: {Construct S.} Let $S = \phi$. Until every vertex in prT has been assigned a weight $w \leq M$, do the following:

Step A: Select an unweighted vertex v in prT such that all of its neighbors except possibly one are weighted. Let $w = \{w_1, \cdots, w_r\}$ be the weights of the neighbors of v, listed in non-increasing order. Let $F = \{e_j\}$ be the set of edges incident to v, listed in the same order. Assign to v weight $w(x) = w_1 + \cdots + w_r + 1$.

Step B: {Update S.} If $w(v) \geq M$, select the smallest i so that $w = w_i + \cdots + w_r + 1 \leq M$. For each $j < i$ add edge e_j to S. Assign to v the new weight $w(v) = w$.
Step 5: {Check whether finished.} The construction of S determines $m(T - S)$. Set ESTIMATE equal to the minimum of ESTIMATE and $\mid S \mid + m(T - S)$. Compute upper (ESTIMATE), the maximum possible m for ESTIMATE. If $M < upper(ESTIMATE)$, set M to $M + 1$, reset the weighting of T to the initial weighting, and go back to Step 4.

Otherwise return $I'(T) = ESTIMATE$ and STOP.

Theorem 10. The algorithm is of complexity $O(n^3)$.

Proof. We use as a measure of the complexity the number of vertices that must be examined to compute $I'(T)$. Each vertex is examined once to establish n, Δ, z, and the initial weighting. Then, for each value of M, each vertex in prT is weighted and whenever it is necessary to add an edge to S, the neighbors of a vertex are searched to find the maximum weighting.

For a given value of M we consider the two parts of the algorithm separately. Each vertex in prT must be examined, which requires some number of steps less than n. The number of searches is at most $upper(ESTIMATE) - M \leq \frac{n+3}{2} - M$, and the number of vertices in each search is at most $\Delta < \frac{n}{2}$. Therefore, for each value of M the number of vertices examined is fewer than $\frac{(n+\frac{n+3}{2}-M)n}{2} < \frac{n^2}{2}$.

Since M ranges from $z+1$ to at most $\frac{n+1}{2}$, the total complexity of the algorithm is at most

$$\sum_{M=2}^{\frac{n+1}{2}} \frac{n^2}{2} \leq \frac{n^3}{4} \qquad \square$$

It is natural to ask for the order of an optimal algorithm. In particular, since many parameters can be determined for trees in linear time, it is possible that a linear algorithm exists.

This algorithm implemented in LISP. A tree is stored as a list of lists, one list for each vertex in prT. The list for v looks like (v e(v) $y_1 \cdots y_k$), where $e(v)$ is the number of end-vertices adjacent to v and the rest of the entries in the list are the neighbors of v in prT.

Step 4 is performed by means of a queue, vertices being put on the queue when they are ready to be weighted, that is, when all but at most one of their neighbors has a weight.

The algorithm as described makes one pass through Step 4 for each value of M. In general several values of M must be checked. Further, every pass through Step 4 produces a potentially better upper bound for $I'(T)$, which in turn reduces the maximum potential value for M that needs to be checked. Therefore, a compromise must be made between choosing an M which might prove to be larger than needed, and choosing $M's$ too close together and therefore generating the same set S more than once. We can modify Step 5 in our implementation by selecting values for M according to the following heuristic:

1. Start with $M = z + 3$.
2. If m is achieved and S for $m-1$ has not been computed, let $M = m - 1$.
3. Otherwise let $M = \frac{upper(ESTIMATE)+m+1}{2}$.

This is still a crude heuristic. Better starting values for M are certainly possible. It might be more clever to select something other than the middle of the unexpected range to jump to. Of course none of these choices affects the worst case complexity of the algorithm.

Acknowledgement
This work supported by University of Tehran. My special thanks go to University of Tehran, College of Engineering and Department of Engineering Science for providing all the necessary facilities available to me for successfully conducting this research.

References

[1] K.S. Bagga, L.W. Beineke, M.J. Lipman, R.E. Pippert, On the edge-integrity of graphs, Congressus Numerantium 60 (1987), 141 - 144.

[2] K.S. Bagga, L.W. Beineke, M.J. Lipman, R.E. Pippert and Sedlmeyer, Some Bounds and an Algorithm for the Edge-Integrity of Trees, Ars Combin. 35 (1993), A, 225-238.

[3] C. A. Barefoot, Roger Entringer and Henda Swart, Vulnerability in Graphs - a comparative survey, J. Combin. Math. Combin. Comput. 1 (1987), 13-22.

[4] J. A. Bondy and U.S.R. Murty, Graph Theory with Applications, American Elsevier, New York, 1976.

[5] L.C. Clark, R.C. Entringer, and M. R. Fellows, Computational Complexity of Integrity, J. Combin. Math. Combin. Comput. 2 (1987), 179-191.

[6] M.R. Fellows and S. Stueckle, The Immersion Order, Forbidden subgraphs and the complexity of Integrity, J. Combin. Math. Combin. Comput. 6 (1989), 23-32.

[7] M.R. Garey and D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W.H. Freeman and Co., San Francisco, 1979.

[8] D.S. Johnson, The Many Faces of Polynomial Time, in The NP-CompleteColumn: An Ongoing Guide, J. Algorithms 8 (1987), 285-303.

[9] D. Moazzami, Vulnerability in Graphs - a Comparative Survey, J. Combin. Math. Combin. Comput. 30 (1999), 23-31.

[10] D. Moazzami, Stability Measure of a Graph - a Survey, Utilitas Mathematica, 57 (2000), 171-191.

[11] N. Robertson and P. Seymour, Disjoint paths - a Survey, SIAM. Alg. Disc. Math. 6 (1985), 300-305.

[12] N. Robertson and P. Seymour, Graph Minors - a Survey, in Surveys in Combinatorics (I. Anderson, ed.), Cambridge Univ. Press (1985), 153-171.

[13] N. Robertson and P. Seymour, Graph Minors IV. Tree-Width and Well-Quasi- Ordering, The disjoint paths problem, September 1986 (manuscript).

[14] N. Robertson and P. Seymour, Graph Minors XIII, The disjoint paths problem, September 1986 (manuscript).