LOCATING NUMBERS AND SETS OF DISCONNECTED GRAPHS

JONATHON P. SPAW

Abstract. In graph theoretic contexts, a locating code for a vertex is an ordered k-vector containing distances to a specified subset of vertices in the graph. If locating codes are unique for distinct vertices in the graph, then this subset is considered to be a locating set. Moreover, the minimum number of vertices required to create a locating set in a particular graph its locating number. We consider, for a finite graph $G(V, E)$, minimum locating sets L and its locating number $loc(G)$. Results are presented establishing criteria for L to be a locating set when $G(V, E)$ is disconnected. We also give results pertaining to the locating number of a disconnected graph. In addition, we provide an algorithm for computing the locating number and minimum locating sets for any connected or disconnected graph as implemented in the mathematical platform Sage. Furthermore, we provide generalized characterization results for various locating numbers.

1. Introduction

When locating numbers and locating sets were first developed, it was with the intention of using them to examine algebraic properties through graphical representations of commutative rings. In particular, Pirzada et. al. studied zero-divisor graphs (which were first introduced and developed in [4]) through the lens of locating sets and locating numbers. All of the results presented in [1] and [2] depended on the assumption that the graphs in question were connected (because zero-divisor graphs are always connected). We explore new results with the assumption that the graphs being explored are disconnected. We hope this will enable the study of graphical representations of commutative rings that are not necessarily connected, such as irreducible divisor graphs [7].

Locating sets and locating numbers were first defined and explored in [1] by Pirzada et. al. in 2013. The concept was then further studied two years later by the same authors in [2] with emphasis placed on its application to algebra, particularly in commutative ring theory. The foundations of locating numbers and locating sets have been well developed, particularly with regards to connected graphs that have locating numbers 1, 2, or $n-1$. Pirzada et. al. have explored locating sets in specific graph structures including paths, cycles, complete graphs, an bipartite graphs. Not only are there additional graph theoretic properties to be studied, but there are also many algebraic structures that can be examined in the context of locating sets and numbers.

Since the exploration of locating sets and locating numbers is computationally demanding, we develop tools to assist in generating and computing the necessary graphs and properties. Using Sage we can generate all of the graphs that we wish to examine as well as implement the necessary graph theoretic algorithms for computing locating numbers and locating sets.

Section 2 presents the foundational definitions for all of the objects and attributes being examined in this paper. These include rudimentary graph-theoretic definitions as well as a discussion of locating numbers and locating sets. In section 3, we examine conditions for locating sets and minimum locating sets of disconnected graphs and a method for computing their locating numbers. Section 4 contains an extension of past results from [1] pertaining to the characterization of graphs with particular locating numbers. Finally, section 5 includes an algorithm that computes the locating number of any graph.

2. Definitions and Past Results

We begin by defining the notion of a graph, which will be used throughout this work.

Definition 2.1. A *graph* is an ordered pair $G = (V, E)$, where V is the set of vertices and E is the set of edges between vertices.

Given a graph G, the set of vertices and the set of edges of G are denoted as $V(G)$ and $E(G)$, respectively. We will always consider G to be a finite, undirected, simple graph. The *order* of a graph G is the number elements in $V(G)$. We define a *path* of length n between two vertices $v_1, v_n \in V(G)$ to be an ordered sequence of distinct vertices and edges $\{v_1, e_1, v_2, e_2, \ldots, v_n\}$ where $e_i \in E(G)$ is incident on v_i and v_{i+1}. Additionally, we define a cycle on n vertices in $V(G)$ to be an ordered sequence of distinct vertices and distinct edges $\{v_1, e_1, v_2, e_2, \ldots, v_n, e_n, v_1\}$. The *distance* between two vertices $v_1, v_2 \in V(G)$, denoted as $d(v_1, v_2)$, is the length of the shortest length path between v_1 and v_2. If no such path exists between v_1 and v_2, then $d(v_1, v_2) = \infty$. Moreover, $d(v, v) = 0$. All graphs considered will be finite, undirected, simple graphs.

Definition 2.2. A graph G is *connected* if there exists a path between any two distinct vertices in $V(G)$. Otherwise, G is *disconnected*.

Definition 2.3. A *connected component* A, or simply *component*, of a graph G is a subgraph in which any two vertices in A are connected by a path and are not connected to any vertices that are not in the component.

As we examine the locating sets and locating numbers of certain graphs, there will be specific graph structures that will be mentioned. Each of these graph categories have been examined in the context of locating numbers and locating sets in [1]. We will define these graph structures rigorously, as they will be referenced in many results.

Definition 2.4. A vertex $v \in V(G)$ is said to be an *isolated vertex* if there are no edges incident on v.

Definition 2.5. The *complete graph* on n vertices K_n is a graph on n vertices in which each pair of vertices is connected by an edge.

Definition 2.6. The *null graph* on n vertices is a graph on n vertices in which all vertices are isolated.

Definition 2.7. The *path* on n vertices P_n is a graph on n vertices consisting of a single path.

Definition 2.8. The *cycle* on n vertices C_n is a graph on n vertices containing a single cycle through all vertices.

The notion of a locating set was first developed in [1]. Locating sets for a graph are dependent on locating codes for individual vertices, which we define formally below. The definition provided is altered slightly from the one given by Pirzada et. al. in [1] in order to allow for disconnected graphs and graph components consisting of only a single vertex.

Definition 2.9. Let G be an n vertex graph where $n \geq 1$. For an ordered set $W = \{w_1, w_2, \dots w_k\}$ of vertices of G and a vertex v of G, the locating code of v with respect to W is the k-vector

$$c_W(v) = (d(v, w_1), d(v, w_2), \cdots, d(v, w_k)).$$

We consider W to be a *locating set* if each of the locating codes with respect to W is unique. Moreover, if a locating set contains the minimum number of vertices, we call it a *minimum locating set*. We define the *locating number* of a graph G, denoted $loc(G)$, to be the number of vertices in a minimum locating set for G. Note that if $L \subseteq W$ and L is a locating set, then W is also a locating set.

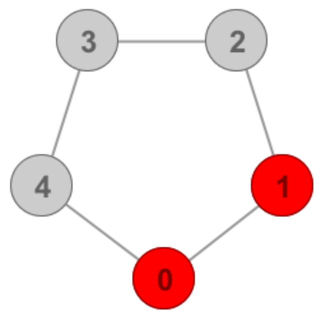

FIGURE 1. A cycle C_5 on 5 vertices with $loc(C_5) = 2$. Vertices in a minimum locating set are shaded.

Example 2.1. Consider the cycle graph C_5 on 5 vertices and the set $W = \{1, 5\}$. Computing the locating codes for each of the vertices in C_5 with respect to W yields

$$c_W(1) = (0, 1), \qquad c_W(2) = (2, 2), \qquad c_W(3) = (1, 2), \qquad c_W(4) = (2, 1), \qquad c_W(5) = (1, 0)$$

We see that each of the locating codes are distinct. Moreover, if we were to construct W to have fewer than 2 vertices, we would not have distinct locating codes. Thus, W is a locating set for C_5 and $loc(C_5) = 2$.

In [1], Pirzada et. al. presented a characterization result that included the locating numbers for paths, cycles, and complete graphs. These lemmas will serve as the building blocks for further generalized characterization results later on. We write $G \cong G'$ if G is isomorphic to G'.

Lemma 2.1. [1, Lemma 2.1] *A connected graph G of order n has locating number 1 if and only if $G \cong P_n$ where P_n is a path on n vertices.*

Lemma 2.2. [1, Lemma 2.2] *A connected graph G of order $n \geq 1$ has locating number $n - 1$ if and only if $G \cong K_n$.*

Lemma 2.3. [1, Lemma 2.3] *For $n \geq 3$, the locating number of a cycle is 2.*

3. Locating Numbers of Disconnected Graphs

As previously discussed, locating sets and locating numbers were initially developed and studied in the context of examining graphical representations of algebraic objects. In particular, Pirzada et. al. used these graph theoretic properties to study zero-divisor graphs for commutative rings. These graphs, which were first developed by I. Beck in [4], are always connected as shown by Anderson et. al. in [3]. Thus, the early development of locating sets was influenced by the connectedness for zero-divisor graphs.

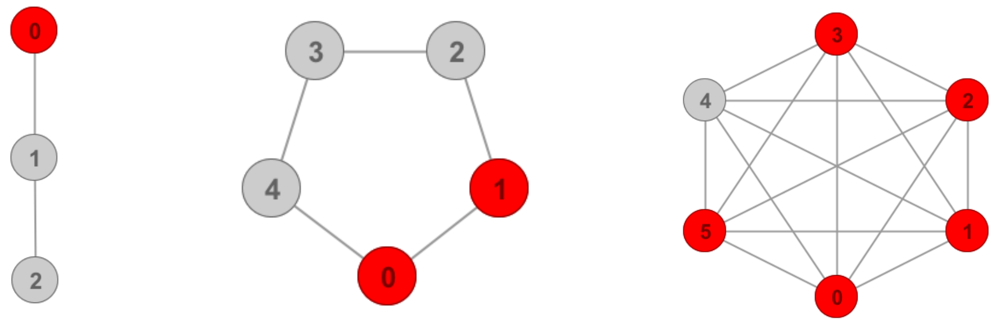

FIGURE 2. A path, cycle graph, and complete graph and their corresponding locating numbers. Shaded vertices denote a minimum locating set. Notice that $loc(P_3) = 1$, $loc(C_5) = 2$, and $loc(K_6) = 5$.

A logical next step would be to use locating sets to examine other graphical representations of commutative rings, such as irreducible divisor graphs, which are covered extensively in [7] and [6]. Doing so requires that we are able to consider locating sets for disconnected graphs. The following results establish attributes for such locating sets and will play an important role in examining locating numbers for disconnected graphs.

Theorem 3.1. *Let G be a disconnected graph that consists of k connected components A_i for $1 \le i \le k$ where $|A_i| \ge 2$. Then W is a locating set for G if and only if for each A_i there exists a $W_i \subseteq W$ such that W_i is a locating set for A_i.*

Proof. ($\Rightarrow$) Let W be a locating set G. We will show $W \cap V(A_i) \subseteq W$ is a locating set for A_i. For $a, b \in V(A_i)$ with $a \neq b$, we have $c_W(a) \neq c_W(b)$ because W is a locating set for G. Thus, $c_{W\cap V(A_i)}(a) \neq c_{W\cap V(A_i)}(b)$, which gives that $W \cap V(A_i)$ is a locating set for A_i.

($\Leftarrow$) We will show that $W = \bigcup_{i=1}^{k}(W \cap A_i)$ is a locating set for G. Let $a \in V(A_i), b \in V(A_j)$ and suppose $i = j$. Since W contains a locating set for W_i and any superset of a locating set is a locating set, it follows that $W \cap V(A_i)$ is a locating set for A_i. Thus, $c_{W\cap V(A_i)}(a) \neq c_{W\cap A_i}(b)$. Since $W \cap V(A_i) \subseteq W$, we have $c_W(a) \neq c_W(b)$. Suppose $i \neq j$. Then a and b belong to different components, so $d(a, c) = n$ implies $d(b, c) = \infty$. Without loss of generality, let $c \in W \cap V(A_i)$. This gives $d(a, c) = n$ for some $n \in \mathbb{N}$ so $d(b, c) = \infty$. Hence, $c_W(a) \neq c_W(b)$ and W is a locating set for G. $\square$

Example 3.1. Consider the graph $G = P_3 \cup \{3, 4\}$ in Figure 3.1 and let $W = \{0\}$. Notice that W is a not a locating set for G because $c_W(3) = c_W(5) = (\infty)$. Although this seemingly contradicts Theorem 3.1, we see that G does not consist entirely of components A_i such that $|A_i| \ge 2$.

Example 3.2. Consider $G = P_3 \cup P_3$ in Figure 3.2 and let $W = \{0, 3\}$. It can be easily verified that W is a locating set for G. Moreover, $\{0\}$ and $\{3\}$ are locating sets for their respective components.

Studying the locating sets of disconnected graphs naturally leads to questions about their locating numbers. The relationship between locating sets of disconnected graphs and those of its components hints at a connection between their locating numbers as well. Conveniently, the locating number of a disconnected graph can be determined from the locating numbers for each component.

Lemma 3.1. *Let G be a disconnected graph that consists of k components $A_1, A_2, \dots, A_k$ such that $loc(A_i) = a_i$ and $|A_i| \ge 2$. Then $loc(G) = \sum_{i=1}^{k} a_i$.*

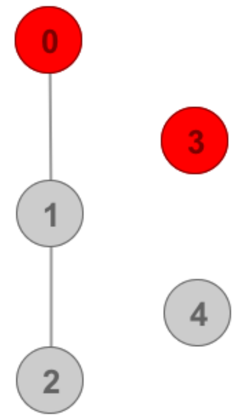

FIGURE 3. The set $W = \{1\}$ is not a locating set for $G = P_3 \cup \{3, 4\}$.

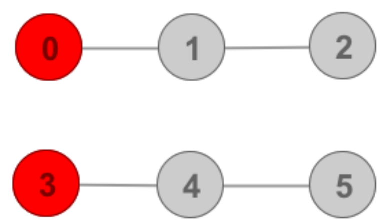

FIGURE 4. The set $W = \{0, 3\}$ is a locating set for $G = P_3 \cup P_3$.

Proof. Assume W_i is a minimum locating set for a component A_i. Let $W = \bigcup_{i=1}^{k} W_i$ be the union of the minimum locating sets for each component of G. To show that W is a locating set, we must show that all distinct vertices in G have distinct locating codes with respect to W.

Let $a, b \in V(G)$ such that $a \neq b$. If $a, b \in V(A_i)$ for $1 \leq i \leq k$, then their locating codes are distinct with respect to W_i since W_i is a locating set for A_i. This gives that their locating codes are distinct with respect to W.

Now consider $a \in V(A_i), b \in V(A_j)$ for $i \neq j$ and let $c \in W_i \subseteq V(A_i)$. Then $d(a, c)$ is finite and $d(b, c)$ is infinite. Thus, $d(a, c) \neq d(b, c)$ so $c_{W_i}(a) \neq c_{W_i}(b)$ which gives $c_W(a) \neq c_W(b)$. Hence, W is a locating set and $loc(G) \leq \sum_{i=1}^{k} a_i$.

Since $\bigcap_{i=1}^{k} V(A_i) = \emptyset$, by Theorem 3.1 we have that $loc(G) \geq \sum_{i=1}^{k} a_i$. Therefore, we have the desired equality $loc(G) = \sum_{i=1}^{k} a_i$. Since all of the components are mutually disjoint, $|W| = \sum_{i=1}^{k} a_i$. Hence, W is a minimum locating set of G.

□

It is important to note that the previous lemma assumes that none of the components are isolated vertices. The following result considers graphs that are comprised of only isolated vertices.

Lemma 3.2. *Let G be a graph consisting of n isolated vertices. Then $loc(G) = n - 1$.*

Proof. Let $W = \{v_1, v_2, \ldots, v_{n-1}\}$ where $v_i \in V(G)$. We will show that W is a locating set and that removing any vertices from W will result in W no longer being a locating set.

Consider $v_j \in W$ where $1 \leq j \leq n - 1$. The corresponding locating code $c_W(v_j)$ will have 0 in the j^{th} component because $d(v_j, v_j) = 0$. Similarly, the j^{th} component will not be 0 for all other locating codes with respect to W. Hence, $c_W(v_j)$ is distinct. Now consider $v_n \in V(G)$ where $v_n \neq v_i$ for $1 \leq i \leq n-1$. The corresponding locating code $c_W(v_n)$ has ∞ in every component. Since this is not the case for any other vertices in $V(G)$, we have that $c_W(v_n)$ is distinct. Therefore, all locating codes with respect to W are unique so W is a locating set and we have that $loc(G) \leq n-1$ Since G has at least two isolated vertices, in any locating set W all but at most one of the isolated vertices must be contained in W. To show this, suppose at least two isolated vertices $v_i, v_j \in V(G)$ are not contained in a possible locating set W'. Then $c_{W'}(v_i) = c_{W'}(v_n) = (\infty, \ldots, \infty)$, so W' cannot be a locating set. This gives $loc(G) > n-2$. Hence, $loc(G) = n-1$ and W is a minimum locating set. □

We may also encounter graphs that include both isolated vertices as well as components with two or more vertices. The following theorem provides the most general case for locating numbers of disconnected graphs.

Theorem 3.2. *Let G be a disconnected graph that such that $G = (\bigcup_{i=1}^{k} A_i) \cup \{v_1, v_2, \ldots, v_n\}$ for $n \geq 1$, where $|A_i| \geq 2$, $loc(A_i) = a_i$, and each v_i is an isolated vertex. Then $loc(G) = \sum_{i=1}^{k} a_i + (n-1)$.*

Proof. We will first prove the case when $n = 1$ and continue by induction.

Consider $G = (\bigcup_{i=1}^{k} A_i) \cup \{v_1\}$. We have $loc(G - \{v_1\}) = \sum_{i=1}^{k} a_i$ by Lemma 3.1. Let W be a minimum locating set for $G - \{v_1\}$. Then $c_W(v_1) = (\infty, \ldots, \infty)$ because v_1 is an isolated vertex. Moreover, since v_1 is the only isolated vertex in $V(G)$, we have that its locating code is distinct. Thus, W is a minimum locating set for G and $loc(G) = \sum_{i=1}^{k} a_i = \sum_{i=1}^{k} a_i + (1-1)$.

Now assume that $loc(G_m) = \sum_{i=1}^{k} a_i + (m-1)$ for every natural number up to m where $G_m = (\bigcup_{i=1}^{k} A_i) \cup \{v_1, v_2, \ldots, v_m\}$. We will now consider $n = m+1$. Let W_m be a minimum locating set for G_m and consider both $W_{m+1} = W_m \cup \{v_{m+1}\}$ and $G_{m+1} = G_m \cup \{v_{m+1}\}$. By construction, W_{m+1} is a locating set for G_m. We also have that $c_{W_{m+1}}(v_{m+1})$ is unique because $v_{m+1} \in W_{m+1}$. Thus, W_{m+1} is a locating set for G_{m+1}.

Now we must show that W_{m+1} is minimum. Since there are two isolated vertices from G_{m+1} that are not in $W_{m+1} - \{v_j\}$ for $1 \leq j \leq m+1$, we have that $W_{m+1} - \{v_j\}$ is not a locating set for G_{m+1}. Furthermore, $W_{m+1} - \{x\}$ is not a locating set by Theorem 3.1 where $x \in V(A_i)$ for some $1 \leq i \leq k$. Following the same reasoning as in Lemma 3.2, whenever a graph has two or more isolated vertices, it must be the case that any locating set all but at most one of them. This gives that W_{m+1} is a minimum locating set. Hence, $loc(G_{m+1}) = |W_{m+1}| = |W_m| + 1 = loc(G_m) + 1 = \sum_{i=1}^{k} a_i + (m-1) + 1 = \sum_{i=1}^{k} a_i + ((m+1)-1)$.

□

Example 3.3. Consider the graph $G = P_3 \cup P_3 \cup \{6, 7, 8\}$, which is shown below in Figure 3.3. By Theorem 3.2, we have $loc(G) = loc(P_3) + loc(P_3) + (3-1) = 1 + 1 + 2 = 4$.

4. Characterizations for Locating Numbers

As previously discussed, Pirzada et. al. provided characterization results for certain locating numbers numbers in connected graphs [1]. Having the framework to study disconnected graphs in the context of locating sets, we are now able to provide more generalized results. We proceed with these characterizations by splitting up cases depending on the number of connected components and isolated vertices.

Theorem 4.1. *Let G be a graph on n vertices. Then $loc(G) = n-1$ if and only if $G \cong K_n$ or G is the null graph on n vertices.*

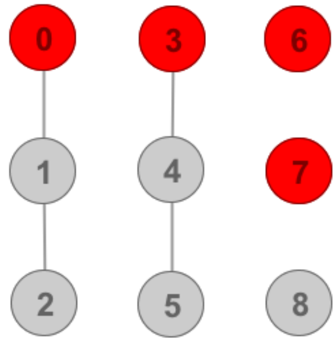

FIGURE 5. The graph $G = P_3 \cup P_3 \cup \{6, 7, 8\}$ has locating number 4.

Proof. ($\Leftarrow$) If $G \cong K_n$, then $loc(G) = n - 1$ by [1, Lemma 2.2]. If G consists of n isolated vertices, then $loc(G) = n - 1$ by Lemma 3.2.

($\Rightarrow$) Suppose $loc(G) = n - 1$.
Case 1: If G is connected, then $G \cong K_n$ by Lemma 2.2.

Case 2: Suppose A_i is the only connected component with $|A_i| \geq 2$. Let $|A_i| = x$. Then there are $n - x$ isolated vertices in G. By Theorem 3.2, we have $loc(G) = loc(A_i) + (n - x - 1)$. Since $loc(A_i) < |A_i|$, we have $loc(G) < n - 1$. Thus, there do not exist any non-trivial components in G. Therefore, G consists entirely of isolated vertices.

Case 3: Suppose G is disconnected and there exist at least two disjoint components $A_i, A_j \subseteq G$. From [1], we have $loc(A_i) \leq |A_i| - 1$ and $loc(A_j) \leq |A_j| - 1$. Thus, by Lemma 3.1 we have that $loc(G) \leq n - 2 < n - 1$. Hence, there cannot be two or more disjoint components in G. $\square$

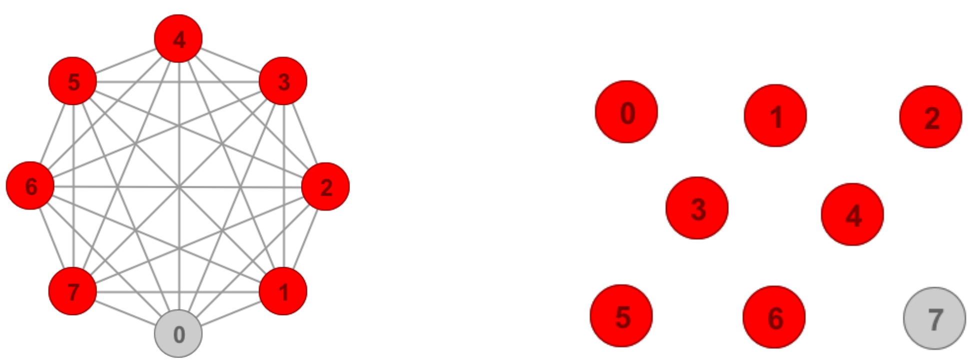

FIGURE 6. Graphs on n vertices with locating number $n - 1$.

Theorem 4.2. *Let G be a graph. Then $loc(G) = 1$ if and only if one of the following is true.*

i. $G \cong P_n$
ii. $G \cong P_n \cup \{v_1\}$
iii. $G \cong \{v_1, v_2\}$

Proof. ($\Leftarrow$) If $G \cong P_n$, then $loc(G) = 1$ by Lemma 2.1. If $G \cong P_n \cup \{v_1\}$ or $G \cong \{v_1, v_2\}$, then $loc(G) = 1$ by Theorem 3.2.

($\Rightarrow$) Suppose $loc(G) = 1$. Similar to the previous result, we will consider several cases.

Case 1: Suppose G is connected. Then $G \cong P_n$ by Lemma 2.1.

Case 2: Suppose G is disconnected and there exists exactly one component $A_1 \subseteq G$ with $|A_1| \geq 2$. Then G must contain m isolated vertices for some $m \in \mathbb{N}$. By Theorem 3.2, we have $loc(G) = loc(A_1) + (m - 1) = 1$. By assumption, we have that $m \geq 1$. Since $loc(A_1) \geq 1$, $loc(G) = 1$ implies that $m = 1$. Now $loc(G) = loc(A_1) = 1$. Thus, by Lemma 2.1, we have $A_1 \cong P_n$. Hence, $G \cong P_n \cup \{v_1\}$.

Case 3: Suppose G is disconnected and consists of n isolated vertices. Then by Theorem 4.1 we have $loc(G) = n - 1 = 1$, which gives $n = 2$. Thus, $G \cong \{v_1, v_2\}$.

Case 4: Suppose G is disconnected and there exist $k \geq 2$ connected components $A_1, \ldots, A_k \subseteq G$ with $|A_i| \geq 2$ for all $i \in \{1, \ldots, k\}$. Then $loc(G) = \sum_{i=1}^{k} loc(A_i) + (m - 1)$ by Theorem 3.2. Since each A_i is a connected component, we have $loc(A_i) \geq 1$ for all $1 \leq i \leq k$. This gives $\sum_{i=1}^{k} loc(A_i) \geq k \geq 2$. Now $1 = \sum_{i=1}^{k} loc(A_i) + (m - 1) \geq \sum_{i=1}^{k} loc(A_i) \geq 2$, which is a contradiction. Therefore, G cannot have $k \geq 2$ connected components having two or more vertices. $\square$

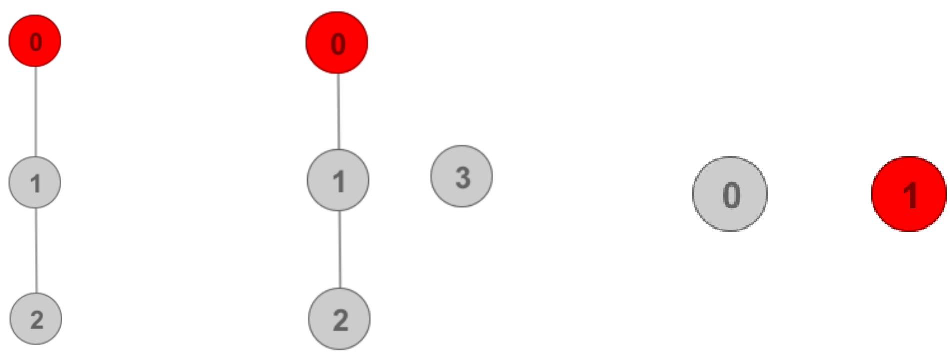

FIGURE 7. Graphs with locating number 1.

Theorem 4.3. *Let G be a graph. Then $loc(G) = 2$ if and only if one of the following is true:*

i. $G \cong C_n$
ii. $G \cong C_n \cup \{v_1\}$
iii. $G \cong P_n \cup P_\ell$
iv. $G \cong P_n \cup P_\ell \cup \{v_1\}$
v. $G \cong P_n \cup \{v_1, v_2\}$
vi. $G \cong \{v_1, v_2, v_3\}$

Proof. ($\Leftarrow$) It is easily verified that all of the listed graphs have locating number 2.

($\Rightarrow$) Suppose $loc(G) = 2$. We consider several cases.

Case 1: If G is connected, then $G \cong C_n$ by Lemma 2.3.

Case 2: Suppose G is disconnected with exactly one connected component $A_1 \subseteq G$ with $|A_i| \geq 2$. Then G must contain $m \geq 1$ isolated vertices for some $m \in \mathbb{N}$ since G is disconnected. Suppose $m = 1$. This gives $2 = loc(G) = loc(A_1) + (1-1) = loc(A_1)$ by Theorem 3.2. Thus, $A_1 \cong C_n$ by Lemma 2.3 so $G \cong C_n \cup \{v_1\}$. Suppose $m = 2$. Then $loc(G) = loc(A_1) + (2-1) = loc(A_1) + 1$, so $loc(A_1) = 1$ and now $A_1 \cong P_n$ by Lemma 2.1. Hence, $G \cong P_n \cup \{v_1, v_2\}$. Suppose $m \geq 3$. Then $m - 1 \geq 2$ and since A_1 is a connected component with $|A_1| \geq 2$, we have $loc(A_1) \geq 1$. Now $loc(G) = loc(A_1) + (m-1) \geq 2 + 1$, but $loc(G) = 2$ by assumption. We have reached a contradiction, so we cannot have $m \geq 3$.

Case 3: Suppose G is disconnected with with two connected components $A_1, A_2 \subseteq G$ with $|A_1|, |A_2| \geq 2$. Suppose $m = 0$. Then $loc(G) = loc(A_1) + loc(A_2) = 2 = 1 + 1$. Since $loc(A_1) \geq 1$ and $loc(A_2) \geq 1$, we must have $A_1 \cong P_n$ and $A_2 \cong P_\ell$, so $G \cong P_n \cup P_\ell$. Suppose $m = 1$. Then $loc(G) = loc(A_1) + loc(A_2) + (1-1) = loc(A_1) + loc(A_2) = 2$. Again, this gives $A_1 \cong P_n$ and $A_2 \cong P_\ell$, so $G \cong P_n \cup P_\ell \cup \{v_1\}$. Suppose $m \geq 2$. This gives $m - 1 \geq 1$. We also have $loc(A_1) \geq 1$ and $loc(A_2) \geq 1$. Hence, $loc(G) = loc(A_1) + loc(A_2) + (m-1) = 2 \geq 3$, which is a contradiction.

Case 4: Suppose G is disconnected and contains $k \geq 3$ connected components A_i. We have $loc(A_i) \geq 1$ for all $1 \leq i \leq k$, so $\sum_{i=1}^{k} loc(A_i) \geq k \geq 3$. This gives $loc(G) = 2 \geq k \geq 3$, which is a contradiction since $k \geq 3$.

Case 5: Suppose G consists only of isolated vertices. Then by Lemma 3.2, we have $G \cong \{v_1, v_2, v_3\}$. $\square$

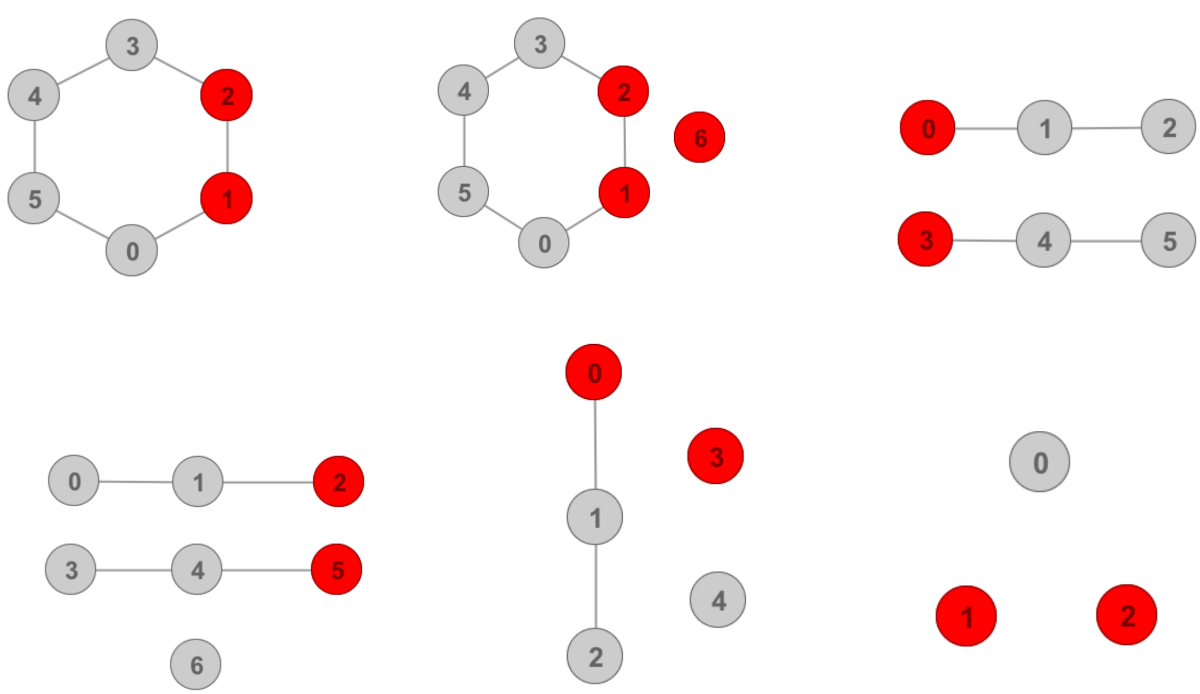

FIGURE 8. Graphs with locating number 2.

5. Computing Locating Numbers and Minimum Locating Sets using Sage

Locating numbers become increasingly difficult to compute by hand as the complexity of a graph's structure increases. Fortunately, the use of computational tools can substantially accelerate the process of computing locating numbers and allows the exploration of complex graphs. The following is an algorithm for computing the locating number of any graph and was implemented in the open source software Sage. This algorithm generates each combination of n vertices from a graph G beginning with $n = 1$ and increasing n incrementally. Next, we compute the locating codes with respect to each combination and test whether or not each locating code is unique. Once a locating set is found, the program returns the locating number of the graph. Interestingly, the algorithm is independent of whether or not the graph is connected/disconnected or even undirected/directed. The pseudocode is given below for computing the locating number for a graph G. Additionally, the Sage implementation of this algorithm is provided as an appendix.

```
For i in [1, |V(G)|]:
  Generate combinations of i vertices
  For each combination:
    Compute locating codes for all v in V(G)
    If locating codes are distinct:
      loc(G) = i
      Return loc(G)
      Break
```

In generating combinations of i vertices, beginning with $i = 1$ and increasing i by 1 in each iteration, the algorithm will terminate if a locating set is found. Since $loc(G) \leq n - 1$ for all graphs on n vertices, the algorithm will yield an output by the $n - 1$st iteration. Moreover, this algorithm will return the locating number regardless of the connectedness of the graph, including cases with isolated vertices.

6. Acknowledgements

This research was conducted at Millikin University under supervision of Dr. Joe Stickles. I would like to thank the Millikin University Undergraduate Research Fellows Program for providing funding during the 2014-2015 year. In addition, I thank the Millikin University Department of Mathematics for guidance and support throughout the research process.

References

[1] S. Pirzada, R. Raja, S. Redmond, Locating Sets and Numbers of Graphs Associated to Commutative Rings, *Journal of Algebra and Its Applications* **13** (2013), 1-17.

[2] R. Raja, S. Pirzada, S. Redmond, On Locating Numbers and Codes of Zero Divisor Graphs Associated with Commutative Rings, *Journal of Algebra and Its Applications, Vol 15, No. 1 (2016).*

[3] D.F. Anderson, P. Livingstron, The Zero-Divisor Graph of a Commutative Ring, *J. Algebra* **217** (1999), 434-447.

[4] I. Beck, Colorings of commutative rings, *J. Algebra* **116** (1988), 208-226

[5] M. Axtell, J. Stickles, Irreducible Divisor Graphs in Commutative Rings with Zero Divisors, *Communications in Algebra* **36** (2008), 1883-1893.

[6] M. Axtell, N. Baeth, J. Stickles, Graphical Representations of Factorizations in Commutative Rings, *Rocky Mountain Journal of Mathematics, Vol 43, Issue 1 (2013)*

[7] J. Coykendall, J. Maney, Irreducible Divisor Graphs, *Comm. Alg.* **35** (2007), 885-895.

```
def find_locating_number(G):
  A = G.vertices()
  for i in range(1,len(A)+1):
    combs = Combinations(A,i)
    locating_number=0
    locating_sets = []
    for k in combs:
      loc_set = list(k)
      codes=[]
      for vertex in G:
        code = []
        for h in loc_set:
          code.append(G.distance(vertex,h))
        codes.append(code)
        if len(set(codes)) == len(codes):
          locating_sets.append(k)
    if len(locating_sets) != 0:
      min_len = len(A)
      for l in locating_sets:
        if len(l) < min_len:
          min_len = len(l)
      print('The locating number is '+str(min_len))
    if len(locating_sets) != 0:
      break
```