Abstract

We consider time constraints for four models of searching graphs for intruders. One model is the standard cops and robber vertex-searching model with complete visibility. The second model differs from the preceding one only in that none of the searchers can see the intruder. The third model is a vertex-searching model in which searchers and an intruder move simultaneously and none of the searchers can see the intruder. The fourth model is simultaneous edge searching with an arbitrarily fast intruder.

1. Introduction

There has been considerable research over the last thirty years dealing with clearing a graph of an intruder. The search models fall into two broad classes: those for which an intruder may be located at vertices or along edges, and those for which an intruder may be located only at vertices. We use the general term edge searching for the former and vertex searching for the latter. We frequently use just the term searching when it is clear which version of searching is being discussed. See the two surveys [1,4] for a discussion of many of the models employed for searching. We shall say a few words about particular models as we come to them.

For graph theory terminology, we refer the reader to [7]. One exception is that we use valency of a vertex rather than degree. We use val_χ(u) to denote the valency of the vertex u in a graph X. When the graph X is clear from context, we use val(u). Another exception is that we use reflexive multigraph if both loops and multiple edges are allowed. Continuing in this vein, multigraphs allow multiple edges, but not loops, and graphs have neither loops nor multiple edges.

Essentially all of the research on searching revolves around the notion of the minimum number of searchers required to capture an intruder no matter how the intruder moves subject to constraints built into the model. This is a natural concept to consider and was mentioned in the first paper on the subject [2] and then extensively first developed in [10]. However, there are contexts, such as using mobile agents to search computer networks, for which
the cost of searchers is negligible. In such a situation it is feasible to attempt to capture an intruder quickly by flooding the graph with more searchers than the minimum required for eventual capture of any intruder. This was studied in [3,8] and is the subject of this paper. Viewing this as time constraints, we mention that Fomin and Golovach [5] have considered searching with cost constraints.

2. Time constrained cops and robbers

The vertex-searching model that has been studied the longest is based on a complete information model. In this model, the set of searchers locate themselves on vertices followed by an intruder choosing a vertex. Everybody knows everybody else’s location. The searchers and intruder then alternate moves with some subset of the searchers moving first. On their respective turns, searchers and the intruder either may stay put on their current vertex or move to an adjacent vertex. Capture takes place when a searcher and the intruder occupy the same vertex at the same time. We refer to this model as the BCR search model (basic cops and robber model). The other extreme situation in which the searchers do not know an intruder’s location, or whether there is an intruder in the graph, is considered in Section 3.

If we are going to introduce extra searchers into the BCR search model, the best we can hope for is that the searchers can capture the intruder on their very first move, that is, they can capture any intruder on the first tick of the clock. We call this one-tick searching.

Definition 2.1. If $X$ is a reflexive multigraph, then the minimum number of searchers required to capture any intruder in $X$, under the BCR search model, in one tick of the clock is called the one-tick cop number of $X$. It is denoted by $1-cn(X)$.

We use $\gamma(X)$ to denote the domination number of a reflexive multigraph $X$. The next result is easily seen to hold.

Proposition 2.2. If $X$ is a reflexive multigraph, then $1-cn(X) = \gamma(X)$.

Neither loops nor multiple edges play any role in either domination or BCR-searching. Thus, one loses nothing by restricting one’s attention to the family of graphs. We are stating these results for the family of reflexive multigraphs for clarity, but in our exposition we often mention only graphs.

Corollary 2.3. Given a reflexive multigraph $X$ and a positive integer $k$, the problem of determining whether $1-cn(X) \leq k$ is NP-complete.

Proof. The result follows immediately from Proposition 2.2 and the fact that determining whether $\gamma(X) \leq k$ is NP-complete (see [6]).

The above results establish the equivalence between the determination of the one-tick search number of a reflexive multigraph $X$ and the determination of the domination number of $X$. We now consider what happens as we allow repeated searcher moves.

Definition 2.4. Let $X$ be a reflexive multigraph. The minimum number of searchers required to capture any intruder in $X$, under the BCR search model, in $k$ ticks of the clock is called the $k$-tick search number of $X$. It is denoted by $k-cn(X)$.

The preceding definition is built around the notion of the time at which capture takes place, so be careful not to confuse this with the number of searcher moves. The $\alpha$th searcher move happens at time $2\alpha - 1$ and corresponds to $(2\alpha - 1)$-tick searching.

Definition 2.5. Given a vertex $u$ in a reflexive multigraph $X$, the closed $r$-neighborhood centered at $u$, denoted $N_r[u]$, is the set of vertices in $X$ whose distances from $u$ are less than or equal to $r$.

Definition 2.6. A subset $U$ of vertices of $X$ is called a distance $r$-dominating set of $X$ provided that the collection \{ $N_r[u] : u \in U$ \} covers the vertex set of $X$. The distance $r$-domination number of $X$, denoted $\gamma_r(X)$, is the smallest cardinality of a distance $r$-dominating set of $X$.

Definition 2.7. Given a reflexive multigraph $X$, the graph underlying $X$ is obtained by removing all loops from $X$ and replacing any edge of multiplicity two or more with a single edge.
We need the preceding concept because $X$ itself may have loops or multiple edges which are short cycles, whereas, the graph underlying $X$ may have no short cycles, that is, large girth. The next theorem indicates that the relationship between $(2r - 1)\text{-cn}(X)$ and $\gamma_r(X)$ is more complex than the relationship between domination and one-tick searching.

**Theorem 2.8.** If $X$ is a reflexive multigraph, then

$$\gamma_r(X) \leq (2r - 1)\text{-cn}(X).$$

Moreover, if $r = 1$ or $r \geq 2$ and the graph underlying $X$ has girth at least $4r - 1$, then

$$\gamma_r(X) = (2r - 1)\text{-cn}(X).$$

**Proof.** We know the result is true when $r = 1$ so we may assume that $r > 1$. Let $U$ be the set of vertices initially occupied by searchers in an optimal $(2r - 1)$-tick search ($r$ searcher moves) for $X$. If any vertex $v$ of $X$ has distance $r + 1$ or more from $U$, then an intruder could locate himself on $v$, not move, and not be captured during the first $r$ searcher moves. This contradicts the fact we are assuming that we have a $(2r - 1)$-tick search for $X$. Thus, $U$ is an $r$-dominating set for $X$ and the general inequality $\gamma_r(X) \leq (2r - 1)\text{-cn}(X)$ holds.

Letting $Y$ denote the graph underlying $X$, we need to show that equality holds when $Y$ has girth at least $4r - 1$. Choose an $r$-dominating set $U$ for $Y$ – and thus also for $X$ – of minimum cardinality. Place a single searcher on each vertex of $U$. If the intruder locates himself on a vertex of $U$ or on a vertex that is distance one from $U$, then he is captured immediately or on the first tick of the clock, respectively. Hence, we may assume the intruder locates himself on a vertex $v$ that has distance at least two from $U$.

Consider the closed $(2r - 1)$-neighborhood $N_{2r-1}[v]$ centered at $v$. Let $T_1$ denote the subgraph of $Y$ induced by $Y$ on $N_{2r-1}[v]$. The subgraph $T_1$ is a tree because the girth of $Y$ is at least $4r - 1$. Similarly, the subgraph induced by $Y$ on $N_{r-1}[v]$ is a tree $T_2$ and, more importantly, $T_2$ is a subtree of $T_1$.

The intruder’s first $r - 1$ moves describes a walk $W$ of length $\ell \leq r - 1$ because he may choose to stay at a vertex on any given move. The intruder starts at $v$ so that $W$ is contained in $T_2$.

Once the intruder has chosen $v$ as his initial location, the following strategy is employed. Any searcher not initially located at a vertex of $T_1$ simply does not move throughout the search. On the other hand, let $\kappa$ be a searcher whose initial location is on a vertex $u$ of $T_1$. There is a unique shortest path from $u$ to $v$ in $T_1$ because it is a tree. The searcher $\kappa$ moves along this path toward $v$. The intruder now has three options for his first move. He may move toward $\kappa$ along the shortest path joining $u$ to $v$, he may stay where he is, or he may move from $v$ into a branch of $T_1$ not containing $u$. In all three cases, after two ticks of the clock the shortest path from $\kappa$ to the intruder lies in $T_1$ and the length of the path has not increased, indeed, it may have decreased.

In general, after the intruder has moved, he is located at some vertex of $T_2$. If $\kappa$ is in a different branch than the intruder, then the shortest path between them passes through $v$ and $\kappa$ moves along this path towards $v$. If $\kappa$ is in the same branch as the intruder and is closer to $v$ than the intruder, then $\kappa$ cannot have started in this branch because at some point they would have passed each other in opposite directions on the shortest path joining them, that is, capture would have taken place earlier. This implies that $\kappa$ entered the branch from $v$ in pursuit of the intruder. In this case, $\kappa$ moves towards the intruder along the shortest path joining them. If $\kappa$ is in the same branch as the intruder but is further from $v$ than the intruder, then $\kappa$ moves towards the intruder along the shortest path joining them.

From the preceding discussion we see that any searcher who starts at a vertex of $T_1$ never leaves $T_1$ and has a unique move each time it is the searchers’ turn to move. We now show that for any possible walk $W$ traced out by the intruder on his first $r - 1$ moves, there is a searcher poised to capture him on the next move if capture has not taken place earlier.

Let $W$ be an arbitrary walk that the intruder may follow for $r - 1$ intruder moves. Let $w$ be the terminal vertex of $W$. As mentioned earlier, $w \in N_{r-1}[v]$. The distance $\delta$ from $w$ to $U$ is at most $r$ because $U$ is an $r$-dominating set. Let $u$ be an arbitrary vertex of $U$ at distance $\delta$ from $w$ (there may be more than one such vertex). Note that $u \in N_{2r-1}[v]$ so that there is a searcher $\kappa$ whose initial location is vertex $u$ and who will be following the strategy described above. Let $E(W)$ denote the edges that appear in $W$. Clearly, $E(W)$ is a subtree of $T_2$ and $|E(W)| \leq r - 1$ because edges may be traversed more than once in a walk.

Let $P$ denote the path in $T_1$ from $u$ to $w$. As we traverse $P$ starting at $u$, let $w'$ denote the first vertex of $E(W)$ on $P$. Let $P[u, w']$ denote the subpath of $P$ with end vertices $u$ and $w'$. Since $|E(W)| \leq r - 1$, the shortest path from $u$ to any vertex of $E(W)$ is the juxtaposition of $P[u, w']$ with the path from $w'$ to the vertex in $E(W)$. Hence, no matter
where the intruder is located in $E(W)$, the searcher $\kappa$ moves along $P$ towards $E(W)$ until reaching $w'$. One extreme possibility is that $w' = w$ and $P$ has length $r$. In this case, after $r - 1$ moves for all participants, the searcher is located on the vertex of $P$ adjacent to $w$ and the intruder is located at $w$ unless he was captured earlier by another searcher. In the former case, $\kappa$ captures the intruder on the next move.

Thus, we may assume that searcher $\kappa$ reaches $w'$ before his $r$th move. What $\kappa$ does from this point on depends on where the intruder is located in $E(W)$ when $\kappa$ moves onto vertex $w'$. If $w' = w$ and the intruder also is on $w$, then capture has taken place. If $w' = w$ and the intruder is not on $w$, then capture eventually takes place because the walk $W$ leads the intruder to vertex $w$ and $\kappa$ is moving towards the intruder in $E(W)$. Assume that $w' \neq w$. Consider the branches of $E(W)$ at $w'$. If the intruder is in a different branch than $w$, then capture will take place because the walk $W$ terminates at $w$ and $\kappa$ always is moving towards the intruder. If the intruder is in the same branch as $w$, then $\kappa$ will capture the intruder at some point because the length of the path from $u$ to $w$ is at most $r$ and $\kappa$ keeps moving towards the intruder. ■

We now want to show that the girth condition in Theorem 2.8 is the best possible. Consider a cycle of length $4r - 2$ and a path of length $2r$ amalgamated at a vertex of the cycle and an end vertex of the path (see Fig. 1 for $r = 2$). The unique $r$-dominating set of this graph has cardinality 2 and consists of the central vertex of the path and the vertex of the cycle diametrically opposed to the vertex of valency 3. If two searchers are going to capture any intruder in $r$ searcher moves, they must locate themselves at the two vertices of the $r$-dominating set. The intruder now locates himself on the vertex of valency 3. On the first tick, the searcher on the path moves towards the intruder. The searcher on the cycle has a problem. If $\kappa$ remains on his current vertex, then the intruder moves along the cycle and now remains on this vertex for good. Neither searcher can reach him in the next $r - 1$ moves. If the searcher on the cycle moves to either adjacent vertex, then the intruder moves on the cycle so that he again is opposite the searcher on the cycle. $\kappa$ remains on this vertex and neither searcher can reach him in the next $r - 1$ moves. Hence, we need at least three searchers to capture any intruder in $r$ moves for this graph and it is easy to see that three searchers suffice.

The tradeoff between number of searchers and number of searcher moves suggests several potential research directions. One such direction is to consider the convergence problem as we increase the length of the search. For a one-tick search the number of searchers required is $1 - cn(X)$, whereas, for a search with no time constraints we require $cn(X)$ searchers. For example, when $X$ is the path $P_n$ of length $n - 1$, we have $1-cn(P_n) = \lceil n/3 \rceil$ and $cn(P_n) = 1$. There is a strategy for a single searcher in $P_n$ to capture any intruder in $\lceil n/2 \rceil$ or fewer moves. So as we increase the number of searcher moves from 1 to $\lceil n/2 \rceil$, the number of searchers required goes from $1-cn(P_n)$ to $cn(P_n)$. What can we say about the convergence for arbitrary graphs or particular families of graphs?

Define $s_k$ to be the product of $k$ and $(2k - 1)-cn(X)$ for $k = 1, 2, \ldots$ until we reach the first $k$ for which $(2k - 1)-cn(X) = cn(X)$. What can we say about the sequence $s_1, s_2, s_3, \ldots$?

3. Other search models

The BCR search model assumes that the participants have complete information about the locations of the other participants. We describe this by saying that the participants have complete visibility. It is not hard to imagine situations for which the searchers have no information about the location of an intruder or intruders. Indeed, the searchers may not even know whether there is an intruder in the graph. If the only change we make to BCR searching is that searchers cannot see the location of an intruder, we call this zero visibility pursuit-evasion searching and use the acronym ZCR for this search model. Since the searchers must follow a search strategy that enables them to be certain that they capture any number of intruders in the graph, we lose nothing by assuming that intruders do know the locations of the searchers.
Definition 3.1. If $X$ is a reflexive multigraph, then the minimum number of searchers required to capture any intruder in $X$, under the ZCR search model, is called the zero visibility cop number of $X$ and is denoted $\text{zcn}(X)$. The minimum number of searchers required for capturing any intruder, under the ZCR search model, in one tick of the clock is called the one-tick zero visibility search number of $X$. We use $1\text{-zsn}(X)$ to denote this number.

There has been very little work on zero visibility searching, but recent M.Sc. theses by Tang [11] and Jeliázkova [9] deal with it and other restrictions on visibility. We now look at one-tick zero visibility searching.

It is not hard to see that there is a big difference between complete visibility and zero visibility. The $m$-star $K_{1,m}$ satisfies $1\text{-cn}(K_{1,m}) = 1$ according to Theorem 2.2 because the central vertex of the star is a dominating set. We claim that $1\text{-zcn}(K_{1,m}) = m$. By placing $m$ searchers on the central vertex of the star and then having one searcher move from the center to each of the pendant vertices at the first tick, any intruder in the graph will be captured. This shows that $1\text{-zsn}(K_{1,m}) \leq m$.

In order to capture an intruder who may be located at a pendant vertex, either a searcher must be initially located there or must move from the central vertex to the given pendant vertex on the first tick. Thus, we need at least one searcher for each pendant vertex. This implies that $1\text{-zcn}(K_{1,m}) \geq m$ from which the claim follows.

We remind the reader that an edge cover of a reflexive multigraph $X$ is a set $E_1$ of edges of $X$ so that every vertex of $X$ is an end of at least one edge in $E_1$. A minimum edge cover of $X$ is an edge cover with the fewest number of edges over all edge covers of $X$. Let $\beta'(X)$ denote the number of edges in a minimum edge cover of $X$.

Theorem 3.2. If $X$ is a reflexive multigraph with no isolated vertices, then $1\text{-zsn}(X) = \beta'(X)$.

Proof. Let $X$ be a reflexive multigraph. Consider a minimum edge cover $E$ of $X$ that has the fewest number of loops over all minimum edge covers. We claim that the only loops in $E$ come from the components of $X$ that consist of single vertices incident with some number of loops. If $u$ is a vertex of $X$ with an edge $e$ joining $u$ to another vertex $v$ of $X$ and $E$ has a loop covering $u$, we could remove the loop and replace it with the edge $e$. The new set of edges is another minimum edge cover, but has fewer loops than $E$. This proves the claim about loops belonging to $E$.

The subgraph $X'$ of $X$ induced by the non-loop edges of $E$ is a forest. This follows because if there is a cycle in $X'$, then any edge of the cycle may be removed to produce an edge cover with fewer edges than $E$. In addition, if any of the trees forming the components of $X'$ have a path of length longer than 2, then we may remove an internal edge of such a path and again obtain an edge cover with fewer edges.

Thus, we know that the edges of $E$ induce a subgraph of $X$ whose components are stars and loops. Above we saw that the one-tick zero visibility cop number of a star is the size (number of edges) of the star. Since it also takes one searcher for each vertex of a loop, we see that $|E|$ searchers suffice for all of $X$. Thus, $1\text{-zsn}(X) \leq \beta'(X)$.

In order to establish the reverse inequality, assume we have $1\text{-zcn}(X)$ searchers carrying out a one-tick zero visibility search of $X$. We define a set $E$ of edges in the following way. If a given searcher $\kappa$ is initially located at vertex $u_\kappa$ and moves to an adjacent vertex $v_\kappa$ at the first tick, then place the edge $u_\kappa v_\kappa$ in $E$. If a searcher $\kappa$ is initially located at a vertex $w_\kappa$ and does not move at the first tick, then all vertices of $X$ have a searcher on them at either the initial placement or at the first tick because we have a one-tick zero visibility search. Since $w_\kappa$ is not isolated, arbitrarily choose an edge or loop incident with $w_\kappa$ to place in $E$. We then see that $E$ is an edge cover of $X$. Hence, $\beta'(X) \leq 1\text{-zsn}(X)$, thereby, completing the proof. 

We introduce one last searching model for which also we consider one-tick searching. This model arises from the applications for computer networks and similar situations covered by the model below. The alternate moves that are used in BCR and ZCR search seem to have their origins in viewing this kind of searching as a combinatorial game, where players take alternate turns. However, in an actual setting where vertex searching is appropriate, there is not much reason to expect an intruder to “wait his turn”. The relevant question then becomes how fast one lets intruders move. We are going to consider a “slow” intruder. In this model, at each tick of the clock, both intruders and searchers may move to adjacent vertices. We are maintaining zero visibility so that the searchers do not know the location of any intruders. Finally, one last important difference is that searchers neither capture nor detect an intruder if they happen to use the same edge at a given tick of the clock. This means that a searcher could go from $u$ to $v$ and an intruder go from $v$ to $u$ at the same tick of the clock without capture taking place or the searcher even knowing an intruder has “passed by”. We call this model SCR searching.
Suppose that if \( X \) is a connected reflexive multigraph of order \( n \), then

\begin{equation}
(2)
\end{equation}

\begin{equation}
(3)
\end{equation}

A one-tick search that captures any intruder and uses 1-scn (zero visibility simultaneous search number) of \( X \) is initially located at \( v \) where they start. The only unoccupied vertex after this move is the vertex \( u \) and \( X \) that are not joined by an edge. Initially locate one searcher at every vertex of \( X \) consisting of three vertex-disjoint copies \( X_1, X_2, X_3 \) of the complete graph \( K_3 \) such that every vertex of \( X_1 \) is joined with every vertex of \( X_2 \), every vertex of \( X_2 \) is joined with every vertex of \( X_3 \), and there are no edges between vertices of \( X_1 \) and \( X_3 \). If we initially locate one searcher on every vertex in \( V(X_1) \cup V(X_2) \), and then have the searchers from \( V(X_i) \) move on some perfect matching between \( X_i \) and \( X_{i+1} \) to the vertices in \( V(X_{i+1}) \), \( i = 1, 2 \), on the first tick of the clock, it is easy to see that any intruders will be captured. The next theorem shows that we cannot do any better than \( 2t \) searchers. Hence, this example also shows the lower bound in the next theorem is sharp.

**Theorem 3.4.** If \( X \) is a reflexive multigraph of order \( n \), then 1-scn\((X) \geq 2n/3\).

**Proof.** A one-tick search that captures any intruder and uses 1-scn\((X) \) searchers is called optimal. Choose an optimal search and partition the vertices of \( V(X) \) as follows. Let \( U \) be the vertices not initially containing a searcher, let \( B \) be the vertices initially containing a searcher that have at least one neighbor in \( U \), and let \( A \) be the vertices initially occupied by searchers having no neighbor in \( U \). We have

\[ |A| + |B| + |U| = n \]

(1)

because we have partitioned the vertices, and

\[ 1-scn(X) \geq |A| + |B| \]

(2)

because each vertex of \( A \cup B \) initially is occupied by at least one searcher. After the first tick of the clock, each vertex of \( B \cup U \) must contain a searcher and this implies that

\[ 1-scn(X) \geq |B| + |U| \]

(3)

If \( |U| \leq n/3 \), (1) and (2) imply 1-scn\((X) \geq |B| + |A| \geq 2n/3 \) and we are done. If \( |U| > n/3 \) and \( |B| \geq |U| \), then (3) implies that 1-scn\((X) > 2n/3 \) and we are done. Thus, we may assume that \( |U| > n/3 \) and \( |B| < |U| \). This means there is a vertex \( v \in B \) with two searchers \( \kappa_1 \) and \( \kappa_2 \) that move from \( v \) to distinct vertices \( u_1 \in U \) and \( u_2 \in U \), respectively. Consider the following search strategy instead. All searchers do as before except \( \kappa_1 \) is initially located at \( u_1 \) and remains there on the first tick. This is an optimal strategy with a strictly smaller set \( U \). This optimal strategy shrinks the cardinality of \( U \) by one and increases the cardinality of \( B \) by one. By repeating this, either we reach an optimal strategy with \( |U| \leq n/3 \) or \( |U| + |B| > 2n/3 \) completing the proof. 

Complete graphs force the maximum number of searchers as the next result shows.

**Theorem 3.5.** If \( X \) is a connected reflexive multigraph of order \( n \), then

\[ 1-scn(X) = n \]

if and only if \( X \) contains a complete subgraph of order \( n \).

**Proof.** Suppose that \( X \) is a reflexive multigraph that contains a complete subgraph of order \( n \). If \( n - 1 \) searchers locate themselves on the vertices of \( X \), then at least one vertex \( u \) is unoccupied by searchers. An intruder then locates himself on \( u \). At the first tick of the clock, the \( n - 1 \) searchers move and some vertex \( v \) must be unoccupied after they complete their moves. If the intruder moves to \( v \), then he is not captured. Hence, 1-scn\((X) > n - 1 \). On the other hand, \( n \) searchers capture any intruder by initially locating one searcher on each vertex of \( X \).

Now suppose that \( X \) does not contain a complete subgraph of order \( n \). In particular, let \( u, v \) be distinct vertices of \( X \) that are not joined by an edge. Initially locate one searcher at every vertex of \( X \) other than \( v \). Since \( X \) is connected and \( u \) and \( v \) are not adjacent, there is a shortest path \( u u_2 \ldots u_k v \) of length \( k \geq 2 \) joining \( u \) and \( v \) in \( X \). The searchers located on the vertices \( u, u_2, \ldots, u_k \) move along the path to the next vertex towards \( v \). All other searchers remain where they start. The only unoccupied vertex after this move is the vertex \( u \). Since \( u \) is not adjacent to \( v \), the intruder
cannot reach this vertex on one tick of the clock. Thus, the intruder is captured. This implies that $1\text{-ssn}(X) \leq n - 1$.

4. Edge searching

We now move to edge searching and shall consider just one edge searching model. In this model, searchers initially locate themselves on vertices of a reflexive multigraph. An intruder then chooses a location. At each tick of the clock (starting at 0), any subset of searchers move, where each individual searcher who is moving traverses an edge $uv$ from its current location $u$ to $v$. The traversal is viewed as taking place at a uniform rate over the appropriate interval $[i, i + 1]$. The intruder, on the other hand, moves through the multigraph according to a continuous function $I$ mapping $[0, \infty)$ to the points of the graph. Thus, the intruder can move arbitrarily fast. (This is what is meant by a “fast” intruder.) Capture takes place if the intruder and a searcher are at the same point of the graph at the same time. One can view the multigraph as embedded in 3-space in order to have the definitions make sense, but it turns out that the algorithms designed to clear multigraphs operate independently of any embedding so that one may think of the multigraphs abstractly without creating problems.

Definition 4.1. The preceding model of edge searching reflexive multigraphs is called simultaneous edge searching. The minimum number of searchers required to clear a reflexive multigraph $X$ under this model is denoted $ssn(X)$. We are interested in the minimum number of searchers required to clear $X$ in one tick of the clock and call this the one-tick simultaneous sweep number of $X$. We denote this by $1\text{-ssn}(X)$.

Determining $1\text{-ssn}(X)$ for an arbitrary reflexive multigraph $X$ was first considered by Chang in [3] and the connection with the maximum order induced bipartite subgraph problem was first studied in [8]. The following three results, Lemma 4.2, Theorem 4.3, and Corollary 4.5, are essentially contained in [8], but the first two are not explicitly stated while the third is. In addition, proofs of the first two are contained in the exposition once a reader has these two results in mind. Of course, the corollary is proved.

The main reason for the differences in the exposition of [8] and this paper is that Hsiao, Tang, and Chang emphasize the theoretical relationship between solutions for the two problems, while we emphasize developing an expression for $1\text{-ssn}(X)$. We also include proofs for the next two results because our proofs are based on a nice application of orientations of odd length cycles. The proofs are crisp and easily understood.

Lemma 4.2. Let $X$ be a multigraph of order $n$. If $Y$ is an induced bipartite subgraph of $X$ with parts $A$ and $B$ having cardinalities $a$ and $b$, respectively, then

$$1\text{-ssn}(X) \leq |E(X)| + n - a - b.$$  

Proof. Let $n' = |V(X)| - a - b$. (If $X$ itself is bipartite, then $n' = 0$.) Arbitrarily label the vertices not in $A \cup B$ as $u_1, u_2, \ldots, u_{n'}$. Obtain an orientation $\overrightarrow{X}$ of $X$ by orienting all edges with one end in $A$ away from $A$, all edges with one end in $B$ towards $B$, and any edge joining $u_i$ and $u_j$ from $u_i$ to $u_j$ if and only if $i < j$. Now station a searcher at each of the $n' = n - a - b$ vertices $u_1, u_2, \ldots, u_{n'}$. These searchers will remain in place. Then for every arc of $\overrightarrow{X}$ station one searcher at the tail of the arc. This searcher will traverse the corresponding edge from tail to head of the arc. This gives us $|E(X)|$ searchers in motion on the interval $[0, 1]$. It is easy to see that this strategy captures any intruder in $X$. $\blacksquare$

Theorem 4.3. If $X$ is a multigraph of order $n$, then

$$1\text{-ssn}(X) = |E(X)| + m,$$

where $n - m$ is the largest order induced bipartite submultigraph of $X$.

Proof. Suppose that $1\text{-ssn}(X) = |E(X)| + m$. We need to show that $n - m$ is the largest order of an induced bipartite subgraph of $X$. We know that all edges of $X$ are cleared so that we obtain an orientation $\overrightarrow{X}$ of $X$ as follows. If an edge is traversed in only one direction by searchers, we orient the edge in that direction. If an edge is traversed in both directions by searchers, we arbitrarily orient the edge in one of the two possible directions. No two searchers traverse the same edge in the same direction or else we could clear $X$ with fewer searchers. Hence, there are $m$ searchers who
are not used to give the orientation to the edges of \( X \). Each such searcher either remains stationary at a vertex or traverses an edge \( uv \) in the direction opposite its orientation in \( X \).

Call a vertex \( v \in V(X) \) contrary if either a searcher remains stationary at \( v \) during the interval \([0, 1]\), or a searcher traverses an edge \( uv \) incident with \( v \) from \( v \) to \( u \), and the edge is oriented from \( u \) to \( v \) in \( X \). From Lemma 4.2, we know that \( m \leq n - 2 \) because every multigraph contains an induced bipartite subgraph with at least 2 vertices. Since there are precisely \( m \) searchers not being used to give the orientation to the edges of \( X \), there are at most \( m \) contrary vertices in \( X \). Hence, there are at least \( n - m \) non-contrary vertices in \( X \). We know that \( n - m \geq 2 \). Let \( X' \) be the subgraph induced by \( X \) on the non-contrary vertices.

If \( X' \) has order 2, then it is bipartite because multigraphs do not have loops. If \( X' \) has order more than 2 and has any odd length cycles, then no matter how the edges of \( X' \) are oriented, there is a directed subpath of length 2 on the cycle. Let \( u_1u_2u_3 \) be such a directed subpath of length 2. Since all three vertices are non-contrary, one searcher traverses the edge \( u_1u_2 \) from \( u_1 \) to \( u_2 \), another searcher traverses the edge \( u_2u_3 \) from \( u_2 \) to \( u_3 \), and no searcher remains on \( u_2 \). Thus, an intruder could start at the midpoint of the edge \( u_1u_2 \) and travel through the vertex \( u_2 \) stopping at the midpoint of \( u_2u_3 \). In this way an intruder can avoid capture because no searcher has remained stationary on \( u_2 \). Therefore, \( X' \) has no odd length cycles which implies it is bipartite. There can be no induced bipartite submultigraph of larger order by Lemma 4.2.

**Corollary 4.4.** For the complete graph \( K_n, n \geq 2 \), we have

\[
1\text{-ssn}(K_n) = \frac{n^2 + n - 4}{2}.
\]

**Proof.** The proof follows immediately from Theorem 4.3 because the largest induced bipartite subgraph of a complete graph is \( K_2 \).

**Corollary 4.5.** Given a reflexive multigraph \( X \) and positive integer \( k \), the problem of determining whether \( 1\text{-ssn}(X) \leq |E(X)| + k \) is NP-complete.

**Proof.** This follows immediately from Theorem 4.3 and the fact that determining the largest order of an induced bipartite subgraph of a multigraph is known to be NP-complete.

5. Two-tick simultaneous edge searching

We have seen in the preceding section that one-tick simultaneous edge searching leads quickly to difficult problems. This suggests that allowing more ticks may lead to even more difficult problems. The picture is not clear, but there are some interesting results for two-tick simultaneous edge searching upon restricting considerations to certain graphs. The next proposition follows immediately because any given searcher can clear at most two edges in two ticks.

**Proposition 5.1.** If \( X \) is a multigraph with \( |E(X)| \) edges, then

\[
2\text{-ssn}(X) \geq \left\lceil \frac{|E(X)|}{2} \right\rceil.
\]

What can we say about multigraphs for which equality holds in Proposition 5.1? Recall that the size of \( X \) is \( |E(X)| \). Suppose that \( |E(X)| \) is even. It is easy to see that all searchers must clear one edge at each tick of the clock, and each edge of \( X \) is traversed by exactly one searcher. Thus, \( X \) could not have any loops because the only way a loop may be cleared is for one searcher to remain on the vertex with which the loop is incident and another searcher to traverse the loop. This is why the proposition is restricted to multigraphs.

If \( u \) is a vertex from which a searcher \( \kappa \) departs at the first tick of the clock, then the edge traversed by \( \kappa \) is not cleared unless searchers traverse all other edges incident with \( u \), starting at \( u \), at the first tick of the clock.

Now consider a vertex \( v \) containing a searcher \( \kappa \) after the first move. If there are \( r \) searchers located at \( v \) at this point, then \( r \) edges incident with \( v \) have been cleared. At the second tick, each of these searchers must clear a distinct edge incident with \( v \) and there can be no uncleared edges incident with \( v \) after they leave \( v \) or else the edges do not remain cleared. Hence, the vertex \( v \) has valency \( 2r \).
Let $U$ be the vertices initially containing searchers and let $V$ be the vertices containing searchers after the first move. It is not hard to see from the above discussion that $U \cap V = \emptyset$, and both $U$ and $V$ are independent sets of vertices.

Let $W$ be the vertices containing searchers after the second move. It is easy to see that $W \cap (U \cup V) = \emptyset$ which then implies that $W$ also is an independent set of vertices. Therefore, $X$ is bipartite with parts $V$ and $U \cup W$ such that every vertex in $V$ has even valency, and each vertex of $V$ has half of its neighbors in $U$ and half of its neighbors in $W$.

On the other hand, let $X$ be a bipartite multigraph of even size with parts $A$ and $V$ such that $A$ can be partitioned into two non-empty sets $U$ and $W$ so that every vertex in $V$ has half its neighbors in $U$ and half its neighbors in $W$. Then place $\text{val}(u)$ searchers on vertex $u$ for each $u \in U$. On the first tick, let the searchers traverse the edges from $U$ to $V$. The number of searchers on each vertex of $v \in V$ equals the number of uncleaned edges from $v$ to $W$. On the second tick, let the searchers traverse the edges from $V$ to $W$. It is clear that $X$ is cleared after the second move.

The preceding discussion tells us exactly which graphs of even size satisfy the equality in Proposition 5.1. Let us now move to the case that $X$ has odd size. In this case we know that we have exactly one searcher who may "waste" a move, that is, a searcher who may remain at a vertex for one tick, or a searcher who traverses an edge without clearing it, or two searchers who together clear just a single edge. This now allows $X$ to have a loop.

Suppose $X$ has a loop incident with vertex $u$ and that this loop is cleared at the first tick. This means a searcher $k_1$ remains on $u$ while another searcher $k_2$ traverses the loop. On the second tick both $k_1$ and $k_2$ must clear distinct edges $uw_1$ and $uw_2$ incident with $u$. Let $X'$ be the multigraph obtained from $X$ by removing the loop and two edges $uw_1, uw_2$. Then $X'$ has even size and has a vertex partition into $U, V, W$ determined by the remaining searchers according to the earlier discussion.

If $u$ is an isolated vertex in $X'$, then it has even valency in $X$ and both $w_1$ and $w_2$ must belong to $W$ or else recontamination takes place in $X$ on one of the two moves. Thus, we may arbitrarily say $u \in U$.

If $u$ is not isolated in $X'$, it may belong to either $U$ or $V$. If $u \in U$, then all edges incident with $u$, other than the loop and $uw_1, uw_2$, join $u$ to vertices in $V$. Both $w_1$ and $w_2$ must be in $W$. If $u \in V$, then $r$ edges incident with $u$ are cleared on the first move and there are $r + 2$ edges from $u$ to $W$ to clear on the second move.

If the loop is cleared on the second tick, then all searchers traverse and clear edges on the first move. This means that $u \in V$. It forces $r \geq 2$ searchers to be located at $u$ with only $r - 2$ uncleaned edges from $u$ to $W$.

The preceding discussion completely describes the structure of $X$ when there is a single loop. We do not continue with details for the case that $X$ has no loops, but mention that the analysis is essentially the same. The conclusions for this case as well as for the above discussion are summarized in the next theorem. In order to simplify the statement of the theorem, we give a definition.

**Definition 5.2.** A bipartite multigraph $X$, with parts $U'$ and $V$, is *neighborly balanced* if $U'$ can be partitioned into two non-empty subsets $U$ and $W$ so that every vertex of $V$ has half its neighbors in $U$ and the other half in $W$. In particular, every vertex of $V$ must have even valency.

**Theorem 5.3.** If $X$ is a reflexive multigraph, then

$$2-\text{ssn}(X) = \left\lceil \frac{|E(X)|}{2} \right\rceil$$

if and only if one of the following holds:

(i) $X$ has even size and is a neighborly balanced bipartite multigraph;
(ii) $X$ has odd size and is obtained from a neighborly balanced bipartite multigraph $X'$, with parts $V$ and $U \cup W$, by adding a loop at a vertex of $u \in (U \cup V)$ and two new edges joining $u$ to any vertices in $W$, or adjoining a new vertex $u$ with a loop at $u$ and two edges from $u$ to any vertices in $W$;
(iii) $X$ has odd size and is obtained from a neighborly balanced bipartite multigraph $X'$, with parts $V$ and $U \cup W$, by removing an edge incident with a vertex of $V$, or by adding an edge joining two vertices $v_1, v_2 \in V$ and deleting one edge from each of $v_1$ and $v_2$ to one of $U$ or $W$.

We are going to use Theorem 5.3 to examine two-tick simultaneous searching of trees. Recall that the *level* of a vertex in a rooted tree is its distance from the root. In particular, the root itself is the unique vertex of level 0.
Corollary 5.4. Let $T$ be a tree of order $n$, and let $A$ and $B$ be the parts in the unique bipartition of $T$. Let $a$ and $b$ be the number of vertices of odd valency in $A$ and $B$, respectively. If $c = \min\{a, b\}$, then

$$2\text{-ssn}(T) \leq \left\lceil \frac{n - 1 + c}{2} \right\rceil.$$ 

Proof. Note that the result holds trivially for the tree of order 2 so that we assume $T$ has order at least 3. Without loss of generality we may assume that $c = a$. Choose any vertex $u \in B$ to be the root of $T$. We now use the nomenclature of Theorem 5.3. Let $V$ consist of all vertices on odd levels in the tree rooted at $u$. Arbitrarily put $u$ in $U$. If a vertex $v$ at level 1 has even valency $d$, then arbitrarily put $d/2$ of its descendants in $W$ and the remaining $(d - 2)/2$ descendants in $U$. Not every vertex at level 1 can be a leaf because of the choice of the root coming from $B$. Thus, if there are no vertices of even valency on level 1, there is at least one vertex of odd valency bigger than 1. If vertex $v$ on level 1 has odd valency $d > 1$, then arbitrarily put $(d + 1)/2$ of its descendants in $W$, the other $(d - 3)/2$ of its descendants in $U$, and insert a new edge joining the vertex to $u$.

Since not all of the vertices on level 1 can be leaves, there is at least one vertex in $W$ on level 2. If vertex $v$ on level 1 is a leaf, insert an edge joining $v$ to any vertex in $W$ on level 2.

We repeat a similar procedure for the vertices on level 3 except that the roles of $U$ and $W$ may be reversed because the unique ancestor of a vertex on level 3 may be in $U$ or $W$. We then continue for all the vertices on odd levels. When we have finished, exactly one edge is added to $T$ for each vertex of odd valency on odd levels, that is, we have added $c$ edges to $T$ to form a neighborly balanced bipartite multigraph $T'$ with $(n - 1 + c)$ edges. We have $2\text{-ssn}(T') = (n - 1 + c)/2$ by Theorem 5.3. It is easy to see how to modify the two-tick edge search of $T'$ to give a two-tick edge search of $T$. This completes the proof.

There are trees for which the two-tick simultaneous search number is smaller than the bound given in Corollary 5.4. The tree shown in Fig. 2 has three vertices of odd valency in each part so that the corollary gives an upper bound of 5. However, it is not difficult to verify that it requires four searchers to carry out a two-tick edge search.

Corollary 5.5. If $T$ is a tree of order $n$ for which the distances between all pairs of vertices of odd valency have the same parity, then

$$2\text{-ssn}(T) = \left\lceil \frac{n - 1}{2} \right\rceil.$$ 

Proof. The only trees with two vertices of odd valency are paths and the result clearly holds for paths. Let $T$ be any tree with three or more vertices of odd valency. Suppose two of them are at odd distance from one another and $P = u_1u_2\cdots u_r$ is the path joining them. Let $u_t$ denote another vertex of odd valency in $T$. If the path $Q$ from $u_1$ to $u_t$ also has odd length, then the path from $u_r$ has even length. Thus, if the distances between all pairs of vertices of odd valency have the same parity, then these distances all are even. Therefore, all vertices of odd valency are in the same part of the unique bipartition of $T$ and the result follows from Corollary 5.4.

We are going to let $T(k; m)$ denote the complete $k$-ary tree with depth $m$. This tree is rooted with the root occurring on level 0. The root has $k$ neighbors on level 1. Every vertex on level $i$, $1 \leq i \leq m - 1$, has one neighbor on level $i - 1$ and $k$ neighbors on level $i + 1$. The vertices on level $m$ are leaves with one neighbor on level $m - 1$.

Corollary 5.6. If $k$ is a positive odd integer, then the complete $k$-ary tree $T(k; m)$ satisfies

$$2\text{-ssn}(T(k; m)) = \left\lceil \frac{k(k^m - 1)}{2(k - 1)} \right\rceil.$$ 


Proof. The complete $k$-ary tree has $\lceil k(m - 1)/(k - 1) \rceil$ edges. When $m$ is even, all vertices of odd valency are at even distance from each other and the result follows from Corollary 5.5. When $m$ is odd, the number of edges is odd and one of the parts in the unique bipartition of the tree has one vertex of odd valency. The result follows from Theorem 5.3.

The next result is a generalization of Corollary 5.6. It follows from Corollary 5.4 because one part in the unique bipartition of the tree has at most one vertex of odd valency.

Corollary 5.7. Let $T$ be a tree with at most one non-leaf vertex of odd valency. If $u$ is the non-leaf vertex of odd valency and all distances from $u$ to leaves have the same parity, or if the only vertices of $T$ with odd valency are leaves and there exists a vertex $u$ in $T$ all of whose distances to leaves have the same parity, then

$$2\text{-ssn}(T) = \left\lceil \frac{|E(T)|}{2} \right\rceil.$$ 

Acknowledgement

This research was supported by NSERC under Grant A-4792.

References