Approximating the position of a hidden agent in a graph

Hannah Guggiari[*], Alexander Roberts[*], Alex Scott[*†]

May 13, 2018

Abstract

A cat and mouse play a pursuit and evasion game on a connected graph G with n vertices. The mouse moves to vertices $m_1, m_2, \ldots$ of G where m_i is in the closed neighbourhood of m_{i-1} for $i \geq 2$. The cat tests vertices $c_1, c_2, \ldots$ of G without restriction and is told whether the distance between c_i and m_i is at most the distance between c_{i-1} and m_{i-1}. The mouse knows the cat's strategy, but the cat does not know the mouse's strategy. We will show that the cat can determine the position of the mouse up to distance $O(\sqrt{n})$ within finite time and that this bound is tight up to a constant factor. This disproves a conjecture of Dayanikli and Rautenbach.

1 Introduction

There are a wide variety of different pursuit and evasion games – games where one player tries to localise a second player moving along the edges of a graph by using information about the current position of the second player.

One of the most widely studied games is Cops and Robbers, which was introduced by Nowakowski and Winkler [6] and independently by Quilliot [7]. One player controls a set of cops and the other player is the robber. At the start of the game, the cops are placed on vertices of G. The robber then chooses a vertex. The players take alternate turns with the cops playing first.

[*]Mathematical Institute, University of Oxford, Oxford, OX2 6GG, United Kingdom. Email:{guggiari,robertsa,scott}@maths.ox.ac.uk

[†]Supported by a Leverhulme Trust Research Fellowship.

With thanks to the Bellairs Research Institute where some of this work was done.

On the cops' turn, each cop moves to a vertex within the closed neighbourhood of their current position and multiple cops may occupy the same vertex. The robber plays in a similar fashion, moving to a vertex within its current closed neighbourhood. The cops win if one of them is on the same vertex as the robber after a finite period of time; otherwise the robber wins. In this game, the players have perfect information and so they know the location of both the cops and robber at all times. One of the main open questions is Meyniel's Conjecture which states that, on any n-vertex graph, $O(\sqrt{n})$ cops can catch the robber. See [1] for a general account and [3, 5, 8] for the best current bounds.

Haslegrave [4] proposed a different game, Cat and Mouse, in which the cat wins by finding the mouse in finite time. As in Cops and Robbers, the players take alternate turns. However, in this game, the cat has no information about where the mouse is and, on its turn, it is not constrained by the edges of G and may test any vertex to see if the mouse is there. The mouse knows the cat's strategy in advance. The mouse is also required to move to a neighbouring vertex on each turn. These differences mean that the strategies employed by the players in Cat and Mouse differ greatly from those used in Cops and Robbers.

In this paper, we will study a variation of the Cat and Mouse pursuit and evasion game introduced by Dayanikli and Rautenbach [2]. The rules are as described above, except the mouse is not required to move on its turn. A time step consists of one turn for each player with the mouse playing first. After the mouse has moved, the cat chooses any vertex and is told whether the distance between itself and the mouse has increased compared with the previous time step. The cat wins if it is able to localise the mouse up to a given distance d within finite time (i.e. it can determine a vertex v such that the mouse is at distance at most d from v). Otherwise, the mouse wins. A more formal description of the game, including a precise definition of what it means for the cat to localise the mouse, is given in Section 2.

Dayanikli and Rautenbach [2] proved that, if G is a tree with maximum degree Δ and radius h, then the cat can localise the mouse up to distance $4\Delta - 6$ within time $O(h\Delta)$. They also showed that the cat can localise the mouse up to distance 8 within time $O(\log n)$ on the $n \times n$ grid. These results led them to make the following conjecture.

Conjecture 1.1. *The cat can localise the mouse up to distance $O(\log n)$ on any connected graph of order n.*

In Section 3, we will show that Conjecture 1.1 is false. We will prove the following theorem.

Theorem 1.2. *For all n sufficiently large, there exists a tree T on n vertices such that, regardless of the strategy employed by the cat, the mouse is able to avoid being localised up to distance $\sqrt{n}/12$ on T for an infinite period of time.*

In Section 4, we will prove that the cat can localise the mouse up to distance $O(\sqrt{n})$ on any simple, undirected and connected graph of order n within time $O(\sqrt{n})$. In particular, we will prove the following theorem.

Theorem 1.3. *The cat can localise the mouse up to distance $\sqrt{32n}$ by time $\sqrt{2n}$.*

Together, these two results show that the cat can localise the mouse up to distance $O(\sqrt{n})$ on any simple, undirected and connected graph of order n, and that this bound is tight up to a constant factor.

2 Rules of the Game

There are two players, the *cat* and the *mouse*, and the game is played on a simple, undirected, connected graph G of order n. It proceeds in discrete time steps, which are labelled by the positive integers.

The mouse is only able to move along the edges of G. Let m_i be the position of the mouse at time i. For $i \geq 2$, m_i belongs to the closed neighbourhood of m_{i-1}, that is m_i is either m_{i-1} or a neighbour of m_{i-1}.

The cat may test any vertex of G without any restrictions. At time i, the cat chooses a vertex c_i. At the end of time step $i \geq 2$, the cat is told the value of b_i where

$$b_i = \begin{cases} 1 & \text{if } d(c_i, m_i) \leq d(c_{i-1}, m_{i-1}) \\ 0 & \text{if } d(c_i, m_i) > d(c_{i-1}, m_{i-1}). \end{cases}$$

where $d(u, v)$ denotes the length of a shortest path connecting vertices u and v. The value of b_i tells the cat whether it is further away from the mouse at step i than it was at step $i-1$. The cat has no information about the location of the mouse when it chooses c_1 and c_2. However, for $i \geq 3$, the cat can use $b_1, \ldots, b_{i-1}$ to help it decide which vertex to choose for c_i. For each i, the cat can calculate M_i, the set of possible positions for m_i. A vertex v belongs to M_i if and only if there are vertices $\widetilde{m}_1, \ldots, \widetilde{m}_i$ satisfying:

- $v = \widetilde{m}_i$
- $\widetilde{m}_j$ is in the closed neighbourhood of $\widetilde{m}_{j-1}$ for every $2 \leq j \leq i$

- for every $2 \leq j \leq i$, $d(c_j, \widetilde{m}_j) \leq d(c_{j-1}, \widetilde{m}_{j-1})$ if and only if $b_j = 1$.

Let $f : \bigcup_{i\in\mathbb{N}}\{0,1\}^i \to V(G)$. We say that the cat *follows strategy* (c_1, c_2, f) if c_1 and c_2 are the first two vertices it tests and $c_i = f(b_2, \ldots, b_{i-1})$ for $i \geq 3$. We say that *the cat can localise the mouse up to distance d within time t on G* if there is some strategy f such that, whenever the cat follows strategy f, there exists $i \leq t$ with $\mathrm{rad}_G(M_i) \leq d$, where the *radius* of a set W of vertices is defined as

$$\mathrm{rad}_G(W) = \min\big\{\max\{d(v,w) : w \in W\} : v \in V(G)\big\}.$$

The cat only knows G and the values b_i (and hence can calculate the sets M_i). The mouse has more information. As well as G, the mouse also knows about the strategy followed by the cat. Therefore, the mouse knows how the cat will choose its vertices and can adapt its own strategy accordingly.

Note that the cat must follow a strategy $f : \bigcup_{i\in\mathbb{N}}\{0,1\}^i \to V(G)$ which has been decided before the game begins. We do not allow the cat to follow a random strategy (as we require the cat to be certain of localising the mouse within given time and distance).

3 Lower Bound

Before we prove Theorem 1.2 for every $n \in \mathbb{N}$, we will first show that it holds for some special cases. We will prove the following proposition.

Proposition 3.1. *Let t be a multiple of 12 and take $n = t^2 + 1$. There exists a tree T on n vertices such that, regardless of the strategy employed by the cat, the mouse is able to avoid being localised up to distance $\frac{t}{12}$ on T for an infinite period of time.*

Throughout the rest of this section, fix $n = t^2 + 1$ where t is divisible by 12. We define T to be the tree on n vertices formed from the star $K_{1,t}$ by subdividing each edge exactly $t - 1$ times. We will call the centre vertex u and refer to the subdivided edges as branches. We do not regard u to be on any branch. The structure of T means (informally) that, at each turn, the cat is only able to check whether the mouse lies on a single branch of the graph.

In the proof of Theorem 3.1, we will provide a strategy for the mouse where m_i is always on a different branch to c_{i-1} and c_i and hence the cat can never determine from its tests which branch the mouse is on. We will also consider a second set of possible positions (w_i) for the mouse such that the cat cannot tell the difference between a mouse at m_i and a mouse at w_i. For

every i, we will ensure that $d(m_i, w_i) > \frac{t}{6}$ and hence $\mathrm{rad}_G(M_i) > \frac{t}{12}$. This is sufficient to ensure that cat cannot localise the mouse up to distance $\frac{t}{12}$ in finite time.

Proof of Theorem 3.1. Suppose that the cat is trying to localise the mouse to up distance $\frac{t}{12}$ on T. We will provide a strategy for the mouse such that $\mathrm{rad}_G(M_i) > \frac{t}{12}$ for every $i \in \mathbb{N}$. Hence, the cat is never able to localise the mouse on T.

To simplify the numbers, the first turn is at time $t = 0$. At this time, the cat only knows the graph T. The mouse knows what T looks like as well as the strategy employed by the cat. The mouse will sometimes give the cat additional information about its position or movements. We will see later that this does not impact the result.

The mouse employs the following strategy, making the moves given by (m_i). In what follows, we will also consider a second set of possible moves (w_i) for the mouse. At each turn, we will ensure that both $m_i, w_i \in M_i$ and $d(m_i, w_i) > \frac{t}{6}$. Hence $\mathrm{rad}_G(M_i) > \frac{t}{12}$ as required.

At several stages in the mouse's strategy, it needs to choose a branch which the cat will not check for a given period of time τ where $\tau < t$. As T has t branches and the cat can only check one per turn, such a branch certainly exists. In order to find such a branch, it is necessary to look ahead at the cat's strategy. Note that, as long as the cat does not play on the branch within the given time period, it does not matter which branch the mouse chooses because its movements, and consequently the cat's movements, will be the same.

Here is the strategy which the mouse will follow:

1. The mouse chooses a branch which the cat will not check for $\frac{2t}{3}$ turns and chooses m_0 to be the vertex at distance $\frac{t}{4}$ from u on this branch. Similarly, let w_0 be a vertex at distance $\frac{t}{4}$ from u on another branch that the cat will not check for $\frac{2t}{3}$ turns. The mouse tells the cat that it is at distance $\frac{t}{4}$ from u so $M_0 = \{v \in V(T) : d(v, u) = \frac{t}{4}\}$. The mouse also tells the cat $\frac{t}{4}$ branches which it is not located on (none of which include m_0 or w_0).

2. Over the next $\frac{t}{6}$ turns, the cat will gradually lose information about the position of the mouse. For $1 \leq i \leq \frac{t}{6}$, define m_i and w_i as follows:

   - If $d(c_i, u) \leq d(c_{i-1}, u)$, then m_i is one vertex closer to the u than m_{i-1} and $w_i = w_{i-1}$.
   - If $d(c_i, u) > d(c_{i-1}, u)$, then $m_i = m_{i-1}$ and w_i is one vertex further away from u than w_{i-1}.

Note that, in the first case, we have $b_i = 1$ and, in the second case, we have $b_i = 0$. In both cases, the cat receives the same value of b_i for a mouse at m_{i-1} and then m_i and a mouse at w_{i-1} followed by w_i and so cannot tell the difference between these positions. At time $\frac{t}{6}$, we have $\frac{t}{3} \leq d(m_{t/6}, w_{t/6}) \leq \frac{2t}{3}$.

3. Let $d = d(m_{t/6}, u)$ so $\frac{t}{12} \leq d \leq \frac{t}{4}$. The mouse tells the cat that it will now run towards u for d time steps and proceeds to do so. For $\frac{t}{6} < i \leq \frac{t}{6} + d$, we have m_i and w_i are both one vertex closer to u than m_{i-1} and w_{i-1} respectively.

4. At time $\frac{t}{6} + d$, we have $m_{t/6+d} = u$ and $d(w_{t/6+d}, u) = \frac{t}{6}$. The mouse now chooses a branch on which the cat will not play in the next $\frac{11t}{12}$ turns (and different from the branch containing w_i). This is the branch on which the (m_i) will now lie. The (w_i) will remain on their original branch.

5. The mouse now tells the cat it is running away from the centre for $\frac{t}{4}$ time steps and does so. For $\frac{t}{6} + d < i \leq \frac{5t}{12} + d$, both m_i and w_i are one vertex further away from u than m_{i-1} and w_i respectively.

6. Let $j = \frac{5t}{12} + d$. At time j, we have $d(m_j, u) = \frac{t}{4}$. Let B be a branch which the cat has not checked in the previous $\frac{t}{4}$ turns and will not check for the next $\frac{2t}{3}$ turns, and which is different to the branch the mouse is currently on (i.e. $m_j \notin B$). Redefine w_j to be the vertex at distance $\frac{t}{4}$ from u on B. The mouse then tells the cat it is at distance $\frac{t}{4}$ from u. The situation is now exactly the same as in step 1.

Stage 6 is identical to stage 1, both in the positions of the mouse and the information known by the cat. Thus, the mouse may repeat stages 2 through 6 to avoid being localised by the cat indefinitely.

Let us now justify the mouse's choice of branch in Step 1. Provided that the mouse chooses a branch that the cat will not play on in the next $\frac{2t}{3}$ rounds, the sequence (b_i) and the branches the cat plays on do not depend on which branch the mouse is on. Therefore, as the mouse knows the cat's strategy, the mouse can look into the future, simulate how the cat will play and hence find a suitable branch.

At stage 4, we have $m_{t/6+d} = u$. The cat has no idea which branch the mouse will run down and so all of the branches it previously eliminated now need checking again. However, the cat cannot tell the difference between a mouse at $m_{t/6+d}$ and one at $w_{t/6+d}$ and so it does not know that the mouse is at u. As $d(w_{t/6+d}, u) = \frac{t}{6}$ but the cat does not know which branch $w_{t/6+d}$ is on, we still have $\mathrm{rad}_G(M_{t/6+d}) > \frac{t}{12}$.

Between times $\frac{t}{6}+d$ and j, the cat has eliminated at most $\frac{t}{4}$ branches for a mouse that reached the centre at time $\frac{t}{6}+d$. At time j, the cat (depending on its play for the previous $\frac{t}{4}$ turns) may realise that the mouse was indeed at the centre at time $\frac{t}{4}+d$, but this is no longer helpful as the mouse is now at distance $\frac{t}{4}$ from the centre along an unknown branch.

Now consider the situation where the cat is not given any information except for T and the values of the b_i for $i \geq 2$. The mouse can employ exactly the same strategy as it used above but without telling the cat anything. Let M_i' be the possible positions of the mouse at time i and M_i be the sets given by the above argument where the cat has extra information. This additional information enables the cat to narrow down the mouse's possible position more accurately and so, for every i, we find that $M_i \subseteq M_i'$

Therefore, as the cat cannot localise the mouse up to distance $\frac{t}{4}$ when it is told some information about its position and movements, it definitely cannot localise the mouse when it does not have access to this extra knowledge. □

Theorem 1.2 follows as a corollary of Proposition 3.1.

Proof of Theorem 1.2. Suppose $n = t^2 + r$ where t is a multiple of 12 and $r \in [24t + 144]$. Let T be the tree described in Proposition 3.1 - the tree obtained by subdividing each edge of the star $K_{1,t}$ exactly $t-1$ times. Create the tree T' on n vertices by adding a branch B of $r-1$ vertices to T. The cat and mouse play on T'. The mouse only plays on the subgraph T and is never located on B. It uses the same strategy as in Proposition 3.1 to avoid being localised up to distance $\frac{t}{12}$ on T'. □

4 Upper Bound

In this section, we will prove Theorem 1.3. In fact we prove a slightly more precise result.

Theorem 4.1. *(i) Let $c > 0$. Then the cat can localise the mouse up to distance $\left(\frac{8}{c} + c\right)\sqrt{n}$ by time $\frac{4}{c}\sqrt{n}$.*

(ii) The cat can localise the mouse up to distance $\frac{9}{2}\sqrt{n}$ by time n.

Theorem 1.3 follows from (i) by setting $c = \sqrt{8}$.

The theorem will follow from two lemmas giving upper bounds on how well a cat can localise a mouse in a graph.

The first lemma depends on how easily we can cover the vertices of the graph G with balls. Thus we get stronger bounds when G is a "fat" graph with lots of clustering.

Lemma 4.2. *Let L, k be natural numbers. Let $G = (V, E)$ be a graph whose vertices may be covered by L balls of radius k. Then the cat can localise the mouse up to distance $4L + k$ in G by time $2L$.*

Assuming this lemma, we can prove Theorem 4.1(i).

Proof of Theorem 4.1 (i). Let $c > 0$ and let $G = (V, E)$ be a connected graph. Let $\mathcal{R}$ be a maximal subset of V with pairwise distance at least $c\sqrt{n}$. Since the $\frac{c\sqrt{n}-1}{2}$-balls around vertices in $\mathcal{R}$ are disjoint and G is connected, $|\mathcal{R}| \leq \frac{2}{c}\sqrt{n}$. Thus the vertices of G are covered by $\frac{2}{c}\sqrt{n}$ balls of radius $c\sqrt{n}$. So by Lemma 4.2 the cat can localise the mouse up to distance $\left(\frac{8}{c} + c\right)\sqrt{n}$ by time $\frac{4}{c}\sqrt{n}$. □

The second lemma depends on a different property of the graph G. The lemma gives stronger bounds when G is a "thin" graph. We define the *diameter* of any connected graph G to be $\mathrm{diam}(G) = \max\{d(u, v) : u, v \in V(G)\}$.

Lemma 4.3. *Let K be a natural number. Let $G = (V, E)$ be a graph such that for all $v \in V$, there exists an $\ell = \ell(v) < K$ such that $|\{w \in V : d(v, w) = \ell\}| < \frac{\ell}{4}$. Let D be the diameter of G. Then the cat can localise the mouse up to distance $\frac{3K}{2}$ in G by time $D/2$.*

Assuming this lemma, we can prove Theorem 4.1 (ii).

Proof of Theorem 4.1 (ii). Let $G = (V, E)$ be a graph with diameter D. For any $v \in V$, by the pigeonhole principle, there exists an $\ell < 3\sqrt{n}$ such that $|\{w \in V : d(v, w) = \ell\}| < \frac{\ell}{4}$. We may then appeal to Lemma 4.3 with $K = 3\sqrt{n}$ and $D \leq n$. □

Let L, k be natural numbers. Let $G = (V, E)$ be a graph and suppose $\{u_1, \ldots, u_L\} \subseteq V$ is such that $\bigcup_{i \in [L]} B_k(u_i) = V$. Consider the strategy for the cat given by Algorithm 1.

Initialization *Set $i = 1$, $w_i = 1$;*
while $i < L$ **do**
  Set $c_{2i-1} = u_{w_i}$, $c_{2i} = u_{i+1}$;
  if $d(c_{2i}, m_{2i}) \leq d(c_{2i-1}, m_{2i-1})$ **then**
    increase i and set $w_i = i$;
  else
    increase i and set $w_i = w_{i-1}$;
  end if
end while

Algorithm 1: Locating the mouse in a "fat" graph.

Note that if the mouse doesn't move, then this algorithm determines which of the u_i is closest to the mouse. It takes them two at a time and chooses the closer one. When we run the same algorithm with a moving mouse, then the mouse may add noise, but only at a bounded rate. Lemma 4.2 follows immediately from the following claim.

Claim 4.4. *Independent of $(m_i)_{i\in\mathbb{N}}$, we have $d(m_{2L-1}, u_{w_L}) \leq 4L + k$.*

Proof. Let $i \in [L-1]$. Then w_{i+1} is either w_i or $i+1$. In the former case, since the mouse moves along an edge once per time step,

$$\begin{aligned} d(u_{w_{i+1}}, m_{2i+1}) &= d(u_{w_i}, m_{2i+1}) \\ &\leq d(u_{w_i}, m_{2i-1}) + 2. \end{aligned}$$

Otherwise $d(u_i, m_{2i}) < d(c_{u_i}, m_{2i-1})$ and $w_i = i$ in which case

$$\begin{aligned} d(u_{w_{i+1}}, m_{2i+1}) &= d(u_i, m_{2i+1}) \\ &\leq d(u_i, m_{2i}) + 1 \\ &\leq d(u_{w_i}, m_{2i-1}) + 1. \end{aligned}$$

In either case, we have $d(u_{w_{i+1}}, m_{2i+1}) \leq d(u_{w_i}, m_{2i-1}) + 2$. By induction, we then have for all $i \leq L$

$$d(u_{w_k}, m_{2L-1}) \leq d(u_{w_i}, m_{2i-1}) + 2(L-i).$$

Now for each $i = 2, \ldots, L$, we have $d(u_{w_i}, m_{2i-1}) \leq d(u_i, m_{2i-2}) + 1$. Therefore for all $i = 2, \ldots, L$

$$\begin{aligned} d(u_i, m_{2L-1}) &\geq d(u_i, m_{2i-2}) - 2(L-i) - 1 \\ &\geq d(u_{w_i}, m_{2i-1}) - 2(L-i) - 2 \\ &\geq d(u_{w_L}, m_{2i-1}) - 4(L-i) - 2 \\ &\geq d(u_{w_L}, m_{2i-1}) - 4L. \end{aligned}$$

Since $w_1 = 1$, the above bound also holds for $i = 1$. Rearranging this, we get that $d(u_{w_L}, m_{2i-1}) \leq 4L + d(u_i, m_{2L-1})$ for all $i \leq L$. But $(B_k(u_i))_{i\in[L]}$ forms a cover of V and so $d(u_i, m_{2L-1})$ must be at most k for one of the i and so $d(u_{w_k}, m_{2i-1}) \leq 4L + k$. □

Let K be a natural number. Let $G = (V, E)$ be a graph such that, for all $v \in V$, there exists an $\ell = \ell(v) < K$ such that $|\{w \in V : d(v, w) = \ell\}| < \frac{\ell}{4}$. Let D be the diameter of G and for each vertex v fix $\ell(v)$ such that $|\{w \in V : d(v, w) = \ell\}| < \frac{\ell}{4}$. Consider Algorithm 2 which dictates a strategy for the

cat given the movements of the mouse $(m_i)_{i\in\mathbb{N}}$. Note that T_i is not necessary for the strategy but we include it to simplify the analysis.

Initialization *Set* $i = 0$, $j = 1$, $T_1 = 0$ *and pick* $v_1 \in V$ *arbitrarily*;
while $i < D$ **do**
  Set $U = \{w \in V : d(v_j, w) = \ell(v_j)\}$;
  Set $u_j = v_j$;
  while $U \neq \emptyset$ **do**
    Increase i ;
    Pick $w \in U$;
    Set $c_{2i-1} = u_j$, $c_{2i} = w$;
    if $d(c_{2i}, m_{2i}) \leq d(c_{2i-1}, m_{2i-1})$ **then**
      Set $u_j = w$;
    end if
    Set $U = U - w$;
  end while
  Set $T_{j+1} = i$, $v_{j+1} = u_j$;
  Increase j
end while

Algorithm 2: Locating the mouse in a "thin" graph.

Lemma 4.3 follows immediately from the following claim.

Claim 4.5. *$d(v_n, m_{2T_n-1}) \leq \frac{3}{2}K$ for all $T_n \geq \frac{D}{2} - \frac{3}{4}K$.*

Proof. Suppose we run Algorithm 2. We claim that, for all $n \geq 1$, $T_{n+1} - T_n \leq \frac{\ell(v)}{4}$ and

$$d(v_{n+1}, m_{2T_{n+1}-1}) \leq \max\left\{d(v_n, m_{2T_n-1}) - \frac{\ell(v_n)}{2}, \frac{3}{2}K\right\}. \tag{1}$$

Assuming these inequalities, we see that $T_n \leq \frac{1}{4}\sum_{i=1}^{n-1} \ell(v_i)$ and

$$d(v_n, m_{2T_n-1}) \leq \max\left\{d(v_1, m_1) - \frac{1}{2}\sum_{i=1}^{n-1} \ell(v_i), \frac{3}{2}K\right\}.$$

Since we have $d(v_1, m_1) \leq D$, it follows that, if $\frac{1}{2}\sum_{i=1}^{n-1} \ell(v_i) \geq D - \frac{3}{2}K$, then $d(v_n, m_{2T_n-1}) \leq \frac{3}{2}K$. Therefore, if $T_n \geq \frac{1}{2}D - \frac{3}{4}K$, then $d(v_n, m_{2T_n-1}) \leq \frac{3}{2}K$.

It remains to show that for all $n \geq 1$, $T_{n+1} - T_n \leq \frac{\ell(v)}{4}$ and (1) holds. First suppose that $d(v_n, m_{2T_n-1}) \geq K$. Let $\ell < K$ be such that $|\{w \in V : d(v_n, w) = \ell\}| < \frac{\ell}{4}$. Note that there must be some w such that

$d(v_n, w) = \ell$ and w is in a shortest $v_n - m_{2T_n-1}$ path. So then $d(w, m_{2T_n-1}) = d(v_j, m_{2T_n-1}) - \ell$. Let $i' \geq i$ be such that $c_{2i} = w$. Then

$$\begin{aligned} d(c_{2i}, m_{2i}) &\leq d(w, m_{2T_n-1}) + 2i - (2T_n - 1) \\ &= d(v_n, m_{2T_n-1}) - \ell + 2i - (2T_n - 1). \end{aligned}$$

On the other hand, $d(v_{n+1}, m_{2T_{n+1}-1}) \leq d(c_{2i}, m_{2i}) + 2T_{n+1} - 1 - 2i$, and so

$$\begin{aligned} d(v_{n+1}, m_{2T_{n+1}-1}) &\leq d(v_n, m_{2T_n-1}) - \ell + 2i - (2T_n - 1) + 2T_{n+1} - 1 - 2i \\ &= d(v_n, m_{2T_n-1}) + 2(T_{n+1} - T_n) - \ell. \end{aligned}$$

$T_{n+1} - T_n = |\{w \in V : d(v_n, w) = \ell\}|$ and so we have that

$$d(v_{n+1}, m_{2T_{n+1}-1}) \leq d(v_n, m_{2T_n-1}) - \frac{\ell}{2}.$$

Now suppose that $d(v_n, m_{2T_n-1}) < K$. Since $T_{n+1} - T_n \leq \frac{K}{4}$, the mouse moves at most $\frac{K}{2}$ steps. Therefore

$$d(v_{n+1}, m_{2T_{n+1}-1}) \leq d(v_n, m_{2T_n-1}) + \frac{K}{2} \leq \frac{3K}{2}.$$

In either case, a simple induction on $n \geq 1$, we have $T_{n+1} - T_n \leq \frac{1}{4}\ell(v_{T_n})$ and that either $d(v_{T_n}, m_{2T_n-1}) \leq \frac{3K}{2}$ or

$$d(v_{n+1}, m_{2T_{n+1}-1}) \leq d(v_n, m_{2T_n-1}) - \frac{\ell}{2}.$$

□

5 Conclusion

While we have established $\Theta(\sqrt{n})$ as a general upper bound for a cat localising a mouse on a graph, it would be nice to get better bounds depending on the structure of G. Trivially we have $\mathrm{diam}(G)$ as an upper bound. In addition to this, when the diameter of G is $n - R$, we may appeal to Lemma 4.3 with $K = O(R)$ to get that the cat can localise the mouse up to distance $O(R)$. Unfortunately, nothing more can be said when the diameter is between these two extreme cases. Indeed, consider lengthening two of the branches of the tree given in Section 3. Lemmas 4.2 and 4.3 are perhaps steps in the right direction here. Lemma 4.2 works well when there is lots of clustering whilst Lemma 4.3 works well when there is very little clustering.

Another interesting direction to consider is the case where multiple cats are trying to localise the mouse exactly. In particular, suppose that there is a collection[1] of k cats $c^1, \ldots, c^k$. At turn i, each cat c^j chooses a vertex c_i^j and is told the value of b_i^j. There is a simple argument which shows that 7 cats is sufficient to localise the mouse exactly on any tree in finite time. We believe that this is not optimal and suggest the following questions.

Question 5.1. *What is the minimum number of cats that are needed to localise the mouse exactly on any tree?*

Question 5.2. *What happens on a general graph G? How accurately can k cats localise a mouse?*

References

[1] W. Baird and A. Bonato, Meyniel's conjecture on the cop number: a survey, *Journal of Combinatorics* **3** (2012), 225–238.

[2] D. Dayanikli and D. Rautenbach, Approximately locating an invisible agent in a graph with relative distance queries, `https://arxiv.org/abs/1801.02370` (2018).

[3] A. Frieze, M. Krivelevich and P. Loh, Variations on Cops and Robbers, *Journal of Graph Theory* **69** (2012), 383–402.

[4] J. Haslegrave, An evasion game on a graph, *Discrete Mathematics* **314** (2014), 1–5.

[5] L. Lu and X. Peng, On Meyniel's conjecture of the cop number, *Journal of Graph Theory* **71** (2012), 192–205.

[6] R. Nowakowski and P. Winkler, Vertex-to-vertex pursuit in a graph, *Discrete Mathematics* **43** (1983), 235–239.

[7] A. Quilliot, Jeux et pointes fixes sur les graphes, Ph.D. Dissertation, Université de Paris VI, (1978).

[8] A. Scott and B. Sudakov, A bound for the cops and robbers problem, *SIAM Journal on Discrete Mathematics* **25** (2011), 1438–1442.

[1] It seems there is a large choice of collective nouns for a collection of cats, including clowder, clutter, destruction (wild cats only), dout, glare, glorying, nuisance, and pounce [9].

[9] D. Tersigni, Animal collective nouns, http://www.thealmightyguru.com/Pointless/AnimalGroups.html, (2003–2017).