


# Cat Herding Game Played on Infinite Trees

Rylo Ashmore 

Memorial University of Newfoundland, St. John's, Canada

Sophie Pinchinat 

IRISA/University of Rennes, France

---

## Abstract

The game of Cat Herding is played on a graph between two players, the cat and the herder. The game setup consists of the cat choosing a starting vertex for their cat token. Then, both players alternate turns, beginning with the herder: they delete (any) one edge, called a cut, and the cat moves along a path to a new vertex. While this game has been studied on finite graph arenas regarding how optimally herder wins, we shift our attention to an infinite version of the game where the cat may now survive indefinitely. We show that cat winning positions in an infinite tree can be characterized by a second-order monadic statement, also amounting to having a complete infinite binary tree minor, or having uncountably many distinct rays. We take advantage of the logical characterization of cat winning positions to generalize a measure known as the cat number, to ordinals.

**2012 ACM Subject Classification** Theory of computation → Higher order logic; Theory of computation → Verification by model checking; Theory of computation → Automata over infinite objects; Theory of computation → Tree languages

**Keywords and phrases** Pursuit-evasion games, Cat Herding, Cat number, Infinite trees, Monadic Second Order Logic, Ordinals

**Digital Object Identifier** 10.4230/LIPIcs.FSTTCS.2025.10

**Funding** *Rylo Ashmore*: Rennes Métropole, Incoming International Mobility Grant for doctoral researchers.

## 1 Introduction

Pursuit-evasion games is a family of games played on graphs that allow for modeling problems arising from piratical applications, such as Robotics, Network Security, and Traffic Management to mention a few. Many results on those games exist, that mostly focus on algorithms and computational complexity of the problem. For a few examples, the quintessential pursuit-evasion game of cops and robbers is known to be EXP-complete [11], the game of firefighting is NP-hard [1], and a broader survey of complexity of pursuit-evasion games can be found [6]. Some of these games also extend naturally to infinite versions, either directly or with slight rules adjustments [15, 12, 14, 4], which are typically analyzed through a lens of structural classifications, rather than algorithmic complexity. In the case of firefighting, this gives open problems, such as whether one firefighter suffices to contain a fire on a hexagonal infinite grid [9]. Such infinite games yield natural difficulties with finding algorithmic results (what is the size of the input?). This paper introduces a novel approach to tackling these difficulties, by taking a logical approach from theoretical computer science to yield deeper insight to a particular pursuit-evasion game: the *Cat Herding* game.

The game of Cat Herding, introduced in [2], is played on a graph between two players, the *cat* and the *herder*. The game setup consists of the cat choosing a starting vertex for their cat token, which we also refer to as the cat. After this, both players alternate turns, beginning with the herder: they delete (any) one edge, called a *cut*, and the cat moves along a path to a new vertex. Note that in this context, a cut refers to a single edge cut, which may or may not disconnect the graph. Note also that the cat may not stay still. Once the



© Rylo Ashmore and Sophie Pinchinat;

licensed under Creative Commons License CC-BY 4.0

45th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2025).

Editors: C. Aiswarya, Ruta Mehta, and Subhajit Roy; Article No. 10; pp. 10:1–10:15



Leibniz International Proceedings in Informatics

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

cat is on an isolated vertex and cannot move (which must happen within  $|E|$  cuts), the game terminates. The objective of the herder is to minimize the number of cuts required to isolate the cat, while the objective of the cat is to maximize the number of cuts required to isolate them. We also refer to the cat having no moves available as being *captured*.

While the introductory paper [2] introduces the game and analyzes it primarily on complete graphs, in this paper we shift our attention to an infinite version of the game. Here, we are concerned with if the cat can survive indefinitely. We will later find that this is not equivalent to the cat surviving arbitrarily long, and we will also provide some analysis of that. We will restrict our attention to trees due to our techniques being essentially with tree automata. For a more complete picture of general infinite Cat Herding, we direct the reader's attention to [3]. This paper's main objective is to show that a certain monadic second order logic statement is satisfied on a given tree  $T$  if, and only if, the cat can win on that tree  $T$ . We note that [5] analyzes infinite (locally finite) graphs and their localization numbers, and their results are similarly related to ends and rays. We also show that cat-win trees and trees that satisfy our Monadic Second Order Logic (MSO) statement are also equivalent to having a complete infinite binary tree minor, or having uncountably many distinct rays. Our argument allows us to also extract an explicit herder strategy on graphs that are herder-win, and this strategy involves a notion of transfinite cat numbers. We show that there are graphs with unbounded transfinite cat number (up to the countable ordinal). Our argument also allows us to extract a decision procedure for deciding if regular trees (those with a structure representable by a regular language) are cat-win or not.

The paper is organized as follows. In Section 2 we introduce the context of our study and recall some known results on the Cat Herding game on finite arenas. Then Section 3 contains our first contribution where we make use of MSO for solving the Cat Herding game on infinite tree arenas. Our second contribution on transfinite cat numbers is developed in Section 4. We end this paper with perspectives on studying the Cat Herding on other kinds of infinite arenas than trees.

## 2 The Cat Herding game

### 2.1 Basics on graphs and trees

For standard graph definitions of degree, paths, cycles, and complete graphs, we defer to [20], as well as for other more technical notions that we briefly recall here. Given a finite (undirected) graph  $G = (V(G), E(G))$ , where  $V(G)$  is the set of vertices and  $E(G) \subseteq V(G) \times V(G)$  is the set of edges, a subgraph  $H$  is a *minor* of  $G$  when it is obtainable through a sequence of vertex and edge deletions, along with edge contractions (where the edge  $uv$  is deleted, vertices  $u, v$  are identified and their neighbourhoods are merged). Minors can also be defined for infinite graphs by relaxing the former definition: we allow any set of edges and vertices to be deleted, along with any connected set of vertices to be contracted to a point analogously, etc. We also recall that any (finite or infinite) connected graph which does not contain cycles is a *tree graph*. Formally, we define a *tree* as a structure  $T = (N, r, succ)$ , that arises from some tree graph  $G = (N, E)$ , where  $r$  is a particular vertex called the *root* and  $succ \subseteq N \times N$  is given by:  $(u, v) \in succ$  if  $(u, v) \in E$  and  $u$  is closer to  $r$  than vertex  $v$ . For finite trees, the *height* is the length of a longest branch.

When working on trees, we may use the word “node” instead of “vertex”. In the following, we will write  $succ(u, v)$  instead of  $(u, v) \in succ$ , and we will write  $\leq$  for the reflexive and transitive closure of the binary relation  $succ$ . Namely,  $u \leq v$  means either  $u = v$ , or  $u$  and  $v$  are on a same path from the root  $r$  in the underlying tree graph and node  $u$  is closer to  $r$

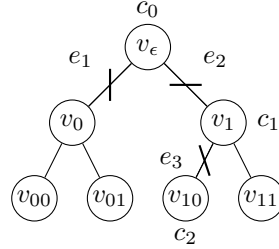
than node  $v$ ; we say that  $u$  is an *ancestor* of  $v$  (and  $v$  is a *descendant* of  $u$ ). In an infinite tree  $T = (N, r, \text{succ})$ , a *ray* (infinite branch) is an infinite sequence of vertices  $(v_i)_{i \in \mathbb{N}}$  such that  $\text{succ}(v_i, v_{i+1})$ , for all  $i$ .

## 2.2 Cat Herding game on finite graphs

The game of Cat Herding is played on a graph, where the cat begins by choosing a vertex to start on. Then the players take turns, with the herder permanently deleting any edge from the graph, and the cat moving along any nontrivial path to a new vertex. In a finite graph, the game ends when the cat cannot move, and the score is the number of edges deleted over the course of the game. The cat is trying to maximize this number, while the herder is trying to minimize it. We will explore the infinite case on trees, but first we recall the main results on the Cat Herding game when played on finite graphs as studied in [2].

► **Definition 1.** Let  $G = (V(G), E(G))$  be a graph. For each  $v \in V(G)$ , we define  $\text{cat}(G, v)$  to be the minimum number of cuts to terminate the game (e.g., isolate the cat) played on arena  $G$  when the cat starts at  $v$ , and we let  $\text{cat}(G) = \max_v \text{cat}(G, v)$ .

► **Example 2.** Actually, it can be shown by induction on the height of binary trees<sup>1</sup> that the cat number of any vertex is equal to its height in vertices, counting itself. For example, consider playing on the graph  $B_3$  (the complete binary tree of height of 3 vertices). In Figure 1, diagrammed herder cuts of edges correspond to the following play where the initial position of the cat is  $v_\epsilon$ . The herder makes any move, say deleting  $(v_\epsilon, v_0)$ . The cat responds by moving to the root of the subtree that was not cut, say  $v_1$ . The herder may cut  $(v_\epsilon, v_1)$ , forcing the cat down a level again, say to  $v_{10}$ . Finally the herder cuts  $(v_1, v_{10})$ , isolating the cat in 3 cuts.



■ **Figure 1** An optimal game on  $B_3$ . Vertex  $c_i$  denotes the cat's position on turn  $i$ , and  $e_i$  denotes the  $i$ th edge that the herder cut.

Regarding known results on the cat number, we know that it is monotonic with respect to subgraphs.

► **Theorem 3** ([2]). If  $H$  is a subgraph of  $G$ , then  $\text{cat}(H) \leq \text{cat}(G)$ . Further, if  $v \in V(H)$ , then  $\text{cat}(H, v) \leq \text{cat}(G, v)$ .

Next, we recall some preliminary values for the cat number of some known classes of graphs. As we will be interested in trees, we start with paths and stars. Let us write  $P_n$  for a path with  $n$  vertices.

<sup>1</sup> a tree is binary if it is non-empty and its internal nodes have at most two children.

## 10:4 Cat Herding Game Played on Infinite Trees

► **Theorem 4** ([2]). *For all  $n \geq 2$ ,  $\text{cat}(P_n) = \lceil \log_2 n \rceil$*

Paths roughly correspond to binary search, providing some intuition for the  $\log_2 n$  term. After one cut, cycles are paths. Let us write  $C_n$  the cycle graph with  $n$  vertices and  $W_n$  the wheel graph with a cycle of  $n - 1$  vertices and an extra vertex “in the middle” connected to all of them.

► **Theorem 5** ([2]). *We have  $\text{cat}(C_1) = 0$  and for  $k > 1$ ,  $\text{cat}(C_{2k+1}) = \text{cat}(C_{2k}) = \lceil \log_2 2k \rceil + 1$ .*

*For  $n \geq 3$ , we have  $\text{cat}(W_n) = n + 1$ .*

Since the cat must move after a herder’s cut, we have the following result on stars. Let us write  $S_n$  for the star graph with  $n$  vertices.

► **Theorem 6** ([2]). *For all  $n \leq 2$ ,  $\text{cat}(S_n) = 2$ .*

The reader interested in more details may refer to [2].

### 3 Cat Herding played on Infinite Trees

In this section, we classify the cat-win infinite trees. Without loss of generality, for any non-empty tree, fixing a node as the root does not change the game of Cat Herding (since the cat can travel both ways along edges of the graph).

Since our analysis will rely on monadic second-order logic, we preliminarily recall which logic we mean to use.

#### 3.1 Monadic Second Order Logic on Trees

The Monadic Second-Order Logic (MSO) formulas we consider are meant to be interpreted on trees of arbitrary degree. Our presentation is inspired from [19].

The syntax of our monadic second-order logic (MSO) is based on a set  $Var$  composed of first-order variables  $x, y, z, x', y' \dots$  and of second-order variables  $X, Y, Z, \dots$  which are meant to represent elements and sets, respectively. The full syntax of the logic MSO is given by :

$$\varphi, \psi := \text{Succ}(x, y) \mid X(x) \mid \varphi \wedge \psi \mid \neg \varphi \mid \exists X : \varphi \mid \exists x : \varphi$$

We also allow for the usual syntactic sugar for  $\varphi \vee \psi$ ,  $\varphi \Rightarrow \psi$ ,  $\varphi \Leftrightarrow \psi$ ,  $\forall x : \varphi$ , and  $\forall X : \varphi$ . Also  $x \leq y$  is a short-cut for  $x = y \vee x < y$ , and  $X \subseteq Y$  is a short cut for  $\forall x : X(x) \Rightarrow Y(x)$ .

Further, we interpret MSO-formulas on an arbitrary tree  $T = (N, r, \text{succ})$ . We will use symbols  $u, v, \dots$  for elements ranging over  $N$  and symbols  $V, V', \dots$  for subsets of  $N$ .

Given an MSO-formula  $\varphi$  and a valuation of the first-order and second-order variables  $\nu : Var \rightarrow 2^N$  we write  $T, \nu \models \varphi$  to express that formula  $\varphi$  holds in tree  $T$  for valuation  $\nu$ , and define it by induction over  $\varphi$  as follows – we take the convention to write  $\nu[x \mapsto v]$  for the valuation that is as  $\nu$  but that maps first-order variable  $x$  onto vertex  $v$ , and similarly for  $\nu[X \mapsto V]$ .

- $T, \nu \models \text{Succ}(x, y)$  whenever  $\text{succ}(\nu(x), \nu(y))$  in  $T$
- $T, \nu \models X(x)$  whenever  $\nu(x) \in \nu(X)$
- $T, \nu \models \neg \varphi$  whenever  $T, \nu \not\models \varphi$
- $T, \nu \models \varphi \wedge \psi$  whenever  $T, \nu \models \varphi$  and  $T, \nu \models \psi$
- $T, \nu \models \exists x : \varphi$  whenever there exists  $v \in N$  such that  $T, \nu[x \mapsto v] \models \varphi$
- $T, \nu \models \exists X : \varphi$  whenever there exists  $V \subseteq N$  such that  $T, \nu[X \mapsto V] \models \varphi$

With a very standard construction, and thanks to second-order quantification, one can define a formula written  $x \leq y$ , whose semantics is the reflexive and transitive closure of the binary relation *succ* (see Section 3.3).

In the following, and for succinctness of MSO-statements, we use formulas of the form  $x \leq y$  with the following semantics.

- $T, \nu \models x \leq y$  whenever  $\nu(x) \leq \nu(y)$  in  $T$

Finally, we let the formula  $x < y := x \leq y \wedge \neg(y \leq x)$ .

As for a closed<sup>2</sup> MSO-formula  $\varphi$ , the valuation plays no role, we simply write  $T \models \varphi$ . Also we loosely write  $T, [x \mapsto v] \models \varphi$  to set the value of free occurrences of variable  $x$  to  $v$  in  $\varphi$ , and similarly for second-order variables.

We end this section by recalling the major result on MSO over infinite trees, namely Rabin's Theorem, that holds for the class of infinite complete trees with a countable degree<sup>3</sup>. This theorem will be useful to establish that solving the Cat Herding game over an infinite countable-degree regular tree arena is decidable, as studied in the next section.

► **Theorem 7** ([17, 10]). *For every  $d \in \mathbb{N}$ , one can decide if an input MSO-formula holds in  $T_d$ , the complete tree of degree  $d$ .*

We now focus on playing the Cat Herding on infinite graphs that have a tree structure.

### 3.2 MSO for Cat Herding game on infinite tree arenas

We present the core MSO-formulas over trees we will need to analyze the Cat Herding game. In our explanation for each of them, the term “path” relates to a path in the tree graph underlying the tree under concern.

$\text{HasDiPath}(X, x, y) := x \leq y \wedge \forall z: (z \leq y \wedge x \leq z \Rightarrow X(z))$   
 //  $y$  is a descendant of  $x$  and  $X$  contains the unique path  
 from  $x$  to  $y$

$\text{DiPath}(X, x, y) := \text{HasDiPath}(X, x, y) \wedge \forall Y: (\text{HasDiPath}(Y, x, y) \Rightarrow X \subseteq Y)$   
 // since  $X$  is the minimal set achieving **HasDiPath**,  $X$  is  
 // exactly composed of the vertices between (and including).  
 //  $x$  and  $y$ .

$\text{Path}(Z, x, y) := \exists X \exists Y: (\forall z: Z(z) \Leftrightarrow (X(z) \vee Y(z))) \wedge$   
 $(\exists z: \text{DiPath}(X, z, x) \wedge \text{DiPath}(Y, z, y))$   
 //  $Z$  is a set of vertices that includes a path from  $x$  to  $y$ .

$\text{DB}(x, y, z) := \exists X \exists Y: \text{Path}(X, x, y) \wedge \text{Path}(Y, x, z) \wedge$   
 $(\forall x': X(x') \wedge Y(x') \Rightarrow x' = x)$   
 // there are edge-disjoint  $x, y$ - and  $x, z$ -paths  
 //(DB means “disjoint branches”).

<sup>2</sup> that is all variables are under the scope of a quantifier.

<sup>3</sup> the degree of a tree is the maximum number of children of the nodes in the tree.

## 10:6 Cat Herding Game Played on Infinite Trees

We finally introduce the main formula  $\text{DBChildin}(X, x)$  whose meaning will be further investigated and that plays a central role in characterizing cat winning positions in an infinite tree.

$$\text{DBChildin}(X, x) := \exists y \exists z: x < y \wedge x < z \wedge X(y) \wedge X(z) \wedge \text{DB}(x, y, z) \quad (1)$$

We now establish a few lemmas for the proof of our main result (Theorem 15) of a logical characterization of all trees where the cat has a winning strategy (that is, the cat can survive indefinitely). From now on and up to Lemma 14 included, we fix a tree  $T = (N, r, \text{succ})$ . For each natural number  $i$ , we define  $V_i \subseteq N$  as follows.

- $V_0 := N$ , and
- for each  $i > 0$ ,  $V_i := \{v \in N \mid T, [X \mapsto V_{i-1}, x \mapsto v] \models \text{DBChildin}(X, x)\}$ .

► **Lemma 8.** *For every  $i \geq 0$  and  $v \in N$ , we have  $v \in V_i$  if, and only if,  $v$  is the root of a binary tree minor of  $T$  of height  $i$ .*

**Proof.** We proceed by induction on  $i$ . The case  $i = 0$  is trivial, as a vertex itself is a binary tree minor of height 0. Let  $i > 0$ .

( $\Leftarrow$ ) Suppose  $v \in N$  is the root of a complete binary tree minor of height  $i$ . We must show  $v \in V_i$ . We proceed by observing the left and right children from  $v$  in this minor, say  $u_0, u_1$ . These nodes are both a root of complete binary tree minor of height  $i - 1$ . Nodes  $u_0$  and  $u_1$  are the good candidates to set the two existential quantified variables  $y$  and  $z$  in formula  $\text{DBChildin}(X, x)$ : indeed, they are both descendants of  $v$  with edge-disjoint paths (by construction), and by induction, we have  $u_0, u_1 \in V_{i-1}$ , giving us that  $v \in V_i$ .

( $\Rightarrow$ ) Suppose that  $v \in V_i$ . There must be  $u_0, u_1$  such that  $v < u_0, v < u_1$ ,  $u_0, u_1 \in V_{i-1}$ , and there are edge-disjoint  $v, u_0$  and  $v, u_1$ -paths. By induction, since  $u_0, u_1 \in V_{i-1}$ , both  $u_0, u_1$  must be roots of complete binary tree minors of height  $i - 1$ . Combined with edge-disjoint  $v, u_0$  and  $v, u_1$ -paths,  $v$  must be the root of a binary tree minor of height  $i$ . ◀

We now extend our semantics to capture what we loosely call here “transfinite iterations” of  $\text{DBChildin}$ , and that corresponds to the fix-point of a well-identified function in Lemma 12.

► **Definition 9.** *As earlier, let  $V_0 := N$ . Then, for every successor ordinal  $\alpha + 1$ , let  $V_{\alpha+1} := \{v \in N \mid T, [X \mapsto V_\alpha, x \mapsto v] \models \text{DBChildin}(X, x)\}$ , and for every limit ordinal  $\alpha = \{\beta \mid \beta < \alpha\}$ , let  $V_\alpha := \bigcap_{\beta \in \alpha} V_\beta$ .*

► **Lemma 10.** *For every ordinal  $\alpha$ , we have  $v \in V_\alpha$  if, and only if, for all  $\beta < \alpha$ ,  $v$  has edge-disjoint descendant paths to a pair of vertices in  $V_\beta$ .*

**Proof.** We proceed by transfinite induction on  $\alpha$ .

This is vacuously true for  $\alpha = 0$ . If  $\alpha + 1$  is a successor ordinal, then  $v \in V_{\alpha+1}$  implies there are edge-disjoint descendant paths to vertices in  $V_\alpha$ . But vertices in  $V_\alpha$  must have descendant paths to vertices in  $V_\beta$  for every  $\beta < \alpha$ . Composing these paths, we obtain edge-disjoint paths to every  $\beta < \alpha$ , as well as edge-disjoint paths to a pair of vertices in  $V_\alpha$ , thus giving every ordinal  $\beta < \alpha + 1$ . Conversely, if  $v$  has edge-disjoint paths to pairs of vertices in  $V_\beta$  for all  $\beta < \alpha + 1$ , then in particular  $v$  has edge-disjoint paths to a pair of vertices in  $V_\alpha$  since  $\alpha < \alpha + 1$ , and thus  $v \in V_{\alpha+1}$ .

If  $\alpha$  is a limit ordinal, then  $v \in V_{\beta+1}$  gives the paths to vertices in  $V_\beta$ , for any  $\beta < \alpha$ . Conversely, if  $v$  has a pair of edge-disjoint paths to all  $\beta < \alpha$ , then it is in all  $V_\beta$  by induction, and thus is in the intersection, so is in  $V_\alpha$ . ◀

With the sets  $V_\alpha$  and the preliminary properties expressed, we can consider the following function  $f$  over the complete lattice  $(2^N, \subseteq)$ .

$$\begin{aligned} f : 2^N &\rightarrow 2^N \\ V &\mapsto \{v \in N \mid T, [X \mapsto V, x \mapsto v] \models \text{DBChildin}(X, x)\} \end{aligned}$$

One can verify that the function  $f$  is  $\subseteq$ -monotonic. By the Knaster-Tarski Theorem [18], the function  $f$  has a greatest fix-point, that we write  $\nu V.f(V)$ <sup>4</sup>.

► **Lemma 11.** *If  $\beta \leq \alpha$ , then  $V_\alpha \subseteq V_\beta$ .*

**Proof.** We show that  $V_\alpha$  is contained within all  $V_\beta$  for  $\beta \leq \alpha$  by transfinite induction on  $\alpha$ .

If  $\alpha$  is a limit ordinal or 0, this is trivial. Otherwise, we only need to show that  $V_{\alpha+1} \subseteq V_\alpha$ , as this implies  $V_{\alpha+1} \subseteq V_\alpha \subseteq V_\beta$  for all  $\beta \leq \alpha$ , the same as all  $\beta < \alpha + 1$ . As such, we must show that  $f(V_\alpha) \subseteq V_\alpha$ . For  $w \in f(V_\alpha)$ , we know that there are two edge-disjoint descendant paths to  $u, v \in V_\alpha$ . Each of these must have edge-disjoint paths to vertices in  $V_\beta$  for all  $\beta < \alpha$  by Lemma 10. By combining these paths, we obtain that  $w \in V_\alpha$  as was to be desired. ◀

With no surprise, we obtain the following.

► **Lemma 12.** *For every  $\alpha \leq |N|$ ,  $\nu V.f(V) \subseteq V_\alpha$ . Further,  $\bigcap_{\alpha \leq |N|} V_\alpha = \nu V.f(V)$ .*

**Proof.** This is a fairly standard proof, one may skip reading.

Since there is no set of all ordinals, we must bound the ordinals. However, each non-stabilizing iteration of  $f$  removes at least one vertex, thus  $V_\alpha$  need only be computed up and including to the order of  $N$ , making this a well-defined set.

We show  $\nu V.f(V) \subseteq V_\alpha$  for all ordinals  $\alpha$  by transfinite induction. Observe  $\alpha = 0$  is trivial. If  $\alpha + 1$  is a successor ordinal, then we observe that  $w \in \nu V.f(V)$  implies  $w$  has edge-disjoint paths to  $u, v$  also in the fix-point. But then  $u, v \in V_\alpha$  by induction. Thus  $w \in V_{\alpha+1}$ . Finally if  $\alpha = \{\beta \mid \beta < \alpha\}$  is a limit ordinal, then  $v \in \nu V.f(V)$  implies  $v \in V_\beta$  for all  $\beta < \alpha$  by induction. Thus  $v$  in the fix-point implies  $v$  in the limit ordinal. Now,  $\bigcap_{\alpha \leq |N|} V_\alpha \supseteq \nu V.f(V)$  follows immediately from the first argument.

We now show  $\bigcap_{\alpha \leq |N|} V_\alpha \subseteq \nu V.f(V)$ . We show  $f(\bigcap_{\alpha \leq |N|} V_\alpha) = \bigcap_{\alpha \leq |N|} V_\alpha$ , so that  $\bigcap_{\alpha \leq |N|} V_\alpha$  is a fix-point, and by  $\nu X$  being a greatest fix-point, we obtain the subset relation. If  $v \in \bigcap_{\alpha \leq |N|} V_\alpha$ , then  $v$  has edge-disjoint descendant path pairs to vertices in  $V_\alpha$  for all  $\alpha$  (using  $v \in V_{\alpha+1}$ ). Thus  $v \in f(\bigcap_{\alpha \leq |N|} V_\alpha)$ . Conversely, if  $v \in f(\bigcap_{\alpha \leq |N|} V_\alpha)$ , then  $v$  has edge-disjoint descendant paths to  $x, y \in \bigcap_{\alpha \leq |N|} V_\alpha$ . Trivially  $v \in V_0$ . Next,  $v \in V_\alpha$  implies  $v \in V_{\alpha+1}$  since  $v$  has paths to  $x, y \in V_\alpha$ . Finally, for limit ordinal  $\alpha$ , if  $v \in V_\beta$  for all  $\beta < \alpha$ , then by definition  $v \in V_\alpha$ . Thus  $v \in \bigcap_{\alpha \leq |N|} V_\alpha$ , and  $\bigcap_{\alpha \leq |N|} V_\alpha$  is a fix-point of  $f$ . ◀

We will show that property  $\nu V.f(V) = \emptyset$  characterizes the herder-win trees, namely trees where herder has a strategy to capture the cat (whichever node the cat starts at). In service of this, we will provide herder strategies when a given rooted subtree has bounded ordinal  $V_\alpha$  vertices. To do so, we will make frequent use of a herder strategy to force the cat off of a ray.

► **Lemma 13.** *If  $R$  is a ray in tree  $T$ , then within finitely many moves, the herder may capture the cat, or force the cat off of  $R$ .*

<sup>4</sup> a classical notation, as for mu-calculus [13].



**Proof.** If the cat is not initially on  $R$ , then the herder disconnects  $R$  from the cat. On a cat move to  $x$  along this ray, the herder deletes the next vertex along the ray, isolating the cat to finitely many vertices of  $R$ . After finitely many moves, the herder may separate all of these vertices, and then make a passing move if necessary to force the cat off of the last one the cat has access to, and finally make a cut separating the cat from that vertex. Observe that as only the cat's initial move may choose how many vertices of  $R$  remain accessible after the herder's cut, removing the cat's access to this ray must be attainable within  $\omega$  cuts – recall that  $\omega$  is the least infinite ordinal greater than all the finite ordinals. ◀

We further explain how herder captures the cat on subtrees of bounded  $V_\alpha$ .

► **Lemma 14.** *Let  $\alpha$  be an ordinal, and let  $v_0 \in N$ . If for all  $v \geq v_0$ ,  $v \notin V_\alpha$ , then the herder can capture the cat starting at  $v_0$ .*

**Proof.** Regardless of  $\alpha$ 's type, the herder begins by cutting  $v_0$  from its parent if any, otherwise herder cuts an edge outgoing  $v_0$ . If the game does not terminate here, on the cat move to  $v > v_0$ , we split the argument on the type of ordinal  $\alpha$ .

If  $\alpha = 0$ , the property trivially holds as it cannot be that  $v \notin V_0 (= V)$ .

If  $\alpha$  is a successor ordinal, then consider where descendants in  $V_{\alpha-1}$  lie in the subtree below  $v_0$ . If ever two incomparable  $u, u' \in V_{\alpha-1}$ , then their deepest common ancestor necessarily below or equal to  $v_0$  would be in  $V_\alpha$ , a contradiction. Thus all elements of  $V_{\alpha-1}$  are comparable, and thus lie on a single ray (or path). Apply Lemma 13 to force the cat off of all elements of  $V_{\alpha-1}$ , and apply the (transfinite) induction hypothesis on the resulting tree the cat is on, since all its vertices are outside  $V_{\alpha-1}$ .

If  $\alpha = \{\beta \mid \beta < \alpha\}$  is a limit ordinal, then consider the set of  $v_0$ -rooted rays  $\mathcal{R} = \{R \mid R \text{ is a ray, } V_\beta \cap R \neq \emptyset \text{ for all } \beta < \alpha\}$ . If  $\mathcal{R}$  contains at least two distinct rays, then there must be a first vertex at which they deviate, say  $r$ . By Lemma 10,  $r$  must be in  $V_\alpha$ , a contradiction. Thus at most one ray may contain vertices in  $\bigcap_{\beta < \alpha} V_\beta$ . We follow Lemma 13 to remove the cat's access to this ray, and let  $x \geq v_0$ , be the new cat position. Now, every descendant vertex  $w \geq x$  is outside some  $V_{\beta_w}$ , with  $\beta_w < \alpha$ . We show  $\sup_{w \geq x} \{\beta_w\} < \alpha$ .

Suppose for contradiction  $\sup_{w \geq x} \{\beta_w\} = \alpha$ . By Lemma 11, there must be infinitely many  $\beta < \alpha$  such that  $V_\beta \neq \emptyset$ . Choose a representative  $w \geq x$  with  $w_\beta \in V_\beta$ , so that  $w_\beta$  is as close to the root as possible. Suppose  $\beta < \beta'$  have incomparable representatives  $w_\beta, w_{\beta'}$ . Then their least common ancestor should have been the candidate for  $\beta$ . Thus every representative is comparable, and they must lie on the same ray. But then we have another cat-accessible ray  $r$  with  $R \cap V_\beta \neq \emptyset$  for all  $\beta < \alpha$ , a contradiction with the earlier herder-play. Thus  $\{\beta_w \mid w \geq v\}$  is bounded, say by  $\beta < \alpha$ . We now use the transfinite induction where vertex  $x \notin V_\beta$  can be used as the new root  $v_0$  of Lemma 14.

Thus the ordinal has been reduced for the entire subtree and the cat's location may again be interpreted as a root. ◀

Note that these ray-themed arguments are reminiscent of the argument that the cat wins implies the existence of an infinite binary tree minor, but here the greatest fix-point characterization naturally leads us to reason about particular transfinite sets, yielding us a (transfinite) constructive herder strategy for capturing the cat.

We introduce the key formula  $\text{ECW}(x)$  (for “Exists Cat Winning”) that characterizes the nodes the cat can start at and win by only downwards moves in tree. We will then make use the formula  $\text{ECW}(x)$  (see formula  $\text{CatWin}(x)$  in Equation (3)) to describe the set of nodes the cat can start at and survive indefinitely.

$$\text{ECW}(x) := \exists X: X(x) \wedge \forall y: X(y) \Rightarrow \text{DBChildin}(X, y) \quad (2)$$



We can now state our first main theorem.

► **Theorem 15.** *Let  $T$  be a tree with a designated root. The following are equivalent.*

1. *The cat wins the game of Cat Herding on  $T$ .*
2.  $T \models \exists x: ECW(x)$ .
3.  *$T$  contains a complete infinite binary tree minor.*
4.  *$T$  has uncountably many distinct rays from the root.*

**Proof.** We show more directions than necessary, as some provide different clarity on the argument.

- (1)  $\Rightarrow$  (2). Argue by contrapositive as follows. Negating the logical statement of cat-win graphs, we obtain  $T \models \forall x: \forall X: X(x) \Rightarrow \exists y: X(y) \wedge \neg DBChildin(X, y)$ . This property in particular would hold for  $X \mapsto \nu V.f(V)$  (the greatest fix-point of  $f$ ) if non-empty, so that we must have  $T \models \exists y: X(y) \wedge \neg DBChildin(X, y)$ . The latter yields  $\nu V.f(V) \not\subseteq f(\nu V.f(V))$ , a contradiction for the fix-point. Therefore  $\nu V.f(V) = \emptyset$ . By Lemma 12,  $\bigcap_{\alpha \leq |N|} V_\alpha = \emptyset$ , so that that for every node  $v \in N$ , we have  $v \notin V_{\alpha_v}$  for some  $\alpha_v$ . Apply Lemma 14 with  $v_0$  the root of  $T$ , to conclude that herder can capture the cat. Observe that Lemma 14 provides an explicit herder strategy for capturing the cat.
- (2)  $\Rightarrow$  (3). We must construct some injective mapping  $h$  from the infinite binary tree on  $\{0, 1\}^*$  to a subtree of  $T$ . Let  $v \in N$  be a vertex such that  $T, [x \mapsto v] \models ECW(x)$ , and let  $V \subseteq N$  be such that  $T, [x \mapsto v, X \mapsto V] \models X(x) \wedge \exists y: X(y) \Rightarrow DBChildin(X, y)$ . We define the mapping  $h$  such that  $h(s) \in V$ , every  $s \in \{0, 1\}^*$ .

We proceed by induction over  $s \in \{0, 1\}^*$ .

- $h(\epsilon) := v \in V$  – the full binary tree minor we construct is rooted at  $v$ .
- Let node  $s \in \{0, 1\}^*$  be such that  $h(s)$  has already been defined and therefore  $h(s) \in V$ . Then, we know that  $T, [x \mapsto h(s), X \mapsto V] \models X(x) \wedge \exists y: X(y) \Rightarrow DBChildin(X, y)$ , which in particular entails  $T, [x \mapsto h(s), X \mapsto V] \models DBChildin(X, x)$ . By definition of formula  $DBChildin(X, x)$ , node  $h(s)$  has descendant nodes  $u_0, u_1$  in  $V$  on two different branches, that we naturally use to define  $h(s0)$  and  $h(s1)$  respectively.

Note that by construction  $h$  is injective, as  $h(s0)$  and  $h(s1)$  are disjoint and that we target vertices deeper and deeper in the tree. Say a node is *h-hit* if it is in the image of  $h$ . The complete infinite binary tree minor of  $T$  is obtained by contracting all edges with at most one *h-hit* node.

- (3)  $\Rightarrow$  (4). Immediate from the uncountability of the set of all infinite binary sequences, corresponding to rays.
- (4)  $\Rightarrow$  (1) by contrapositive. We must show that if the herder wins, then the rays  $\mathcal{R}$  are countable. Observe the  $i$ th cut may remove the cat from accessing infinitely much of a given ray  $R$ . Let  $\mathcal{R}_i$  to be the set of rays which the cat could access infinitely much of before the  $i$ th cut, but only finitely much after the  $i$ th cut. We argue  $\mathcal{R}_i$  is countable and  $\bigcup_i \mathcal{R}_i = \mathcal{R}$ . By contradiction, if there exists an  $i$ th cut that makes  $\mathcal{R}_i$  uncountable, then the cat made a bad move. Suppose the cat instead went to just on the other side of the herder cut. Then we get uncountably many rays available now, but the herder's cut may change. If iterating this process to get successively better cat moves happens to move the cat's choices upwards (towards the root), it must terminate, and  $\mathcal{R}_i$  is countable. If downwards (away from the root), get uncountably many rays with same prefix, for any prefix length. In the limit, obtain ray  $\pi$ . Then enumerate all rays starting with  $\pi$ , then 1 ray deviating at depth 1, then 1, 2, 1, 2, 3,  $\dots$ . A given ray  $R$  has  $R = \pi$  or  $R$  deviates from  $\pi$  at depth  $d$ . But depth  $d$  deviations are countable, and our enumeration of  $\mathcal{R}$  checks depth  $d$  infinitely many times, so  $R$  is counted. Thus  $\mathcal{R}$  is countable, as was to be

## 10:10 Cat Herding Game Played on Infinite Trees

shown. Alternately, the cat can make a strategy that never lets  $\mathcal{R}_i$  be uncountable, so that  $\mathcal{R}_i$  is always countable, and there are finitely many of them. Thus  $\bigcup_i \mathcal{R}_i$  is countable. But  $\mathcal{R} = \bigcup_i \mathcal{R}_i$  since clearly  $\mathcal{R}_i \subseteq \mathcal{R}$ , and if  $R \in \mathcal{R}$ , then since the herder won after  $i$  steps, the cat must have access to at most one vertex of  $R$ , and so the cat must have had access to infinitely many vertices of  $R$  removed at some point, say  $i$ . Thus  $R \in \mathcal{R}_i$  for some  $i$ . We have thus counted the set of all rays.

- (2)  $\Rightarrow$  (1). We include this direction to make the cat's winning strategy explicit. Since  $T \models \exists x: \text{ECW}(x)$ , we have a node  $v_0 \in N$  and a set  $V \subseteq N$  such that  $T, [x \mapsto v_0, X \mapsto V] \models X(x) \wedge \exists y: X(y) \Rightarrow \text{DBChildin}(X, y)$ . Thus  $V \neq \emptyset$ . The cat's strategy will be to always play to a vertex  $v \in V$  and has all herder cuts incomparable or ancestors of the cat's current position. This is possible, since on any herder cut when the cat is at  $v$  and  $T, [x \mapsto v, X \mapsto V] \models \text{DBChildin}(X, x)$ . By expanding formula  $\text{DBChildin}(X, x)$ , we get  $T, [x \mapsto v, X \mapsto V] \models \exists y \exists z: x < y \wedge x < z \wedge X(y) \wedge X(z) \wedge \text{DB}(x, y, z)$ , so that that there are  $u_0, u_1$  with  $T, [x \mapsto v, y \mapsto u_0, z \mapsto u_1, X \mapsto V] \models \text{DB}(x, y, z)$ . The latter implies that no single edge cut from the herder could remove the cat's path to both  $u_0$  and  $u_1$ , and the herder only gets one cut as  $u_0, u_1 > x$ , and all previous cuts were ancestors or incomparable. Thus the cat can move to  $u_0$  or  $u_1$ , and  $u_0, u_1 \in V$  are both promised, giving the invariant the cat seeks. Note this also corresponds to playing on a root of a complete infinite binary tree minor, so that (3)  $\Rightarrow$  (1) could have been shown instead similarly.

It follows that conditions (1)-(4) are equivalent. ◀

As a final matter of logical completeness, we provide our second main theorem (Theorem 16), by considering the following formula that characteres all nodes that the cat can start at and still win.

$$\text{CatWin}(x) := \exists y \exists z: \text{DB}(x, y, z) \wedge \text{ECW}(y) \wedge \text{ECW}(z) \quad (3)$$

It is clear that the formula  $\text{ECW}(x) \Rightarrow \text{CatWin}(x)$  always holds, but the reciprocal does not. To see this, consider a complete binary tree where the topmost edges are subdivided once; formally, the set of nodes is  $(0 + 00 + 1 + 11)(0 \cup 1)^*$ . The cat starting at node 0 can win by moving to vertex 00 or 11 (depending on herder's first cut), and because either if these two nodes satisfies  $\text{ECW}(x)$ , the cat can survive indefinitely. However, the node 0 the cat started from itself does not satisfy  $\text{ECW}(x)$ : indeed, any two branches from 0 both cross edge  $(0, 00)$  and therefore cannot be disjoint as required by Equation (2) for  $\text{ECW}(x)$ .

► **Theorem 16.**  $T, [x \mapsto v_0] \models \text{CatWin}(x)$  if, and only if, the cat can win when starting at  $v_0$ .

**Proof.** In this proof, we may abuse notation and conflate  $\text{ECW}, \text{CatWin} \subseteq N$  as a set of nodes with the predicate.

( $\Leftarrow$ ) By contrapositive, suppose  $T, [x \mapsto v_0] \models \neg \text{CatWin}(x)$ . Then  $T, [x \mapsto v_0] \models \forall y \forall z: \text{DB}(x, y, z) \Rightarrow (\neg \text{ECW}(y) \vee \neg \text{ECW}(z))$ . Since by assumption  $v_0 \notin \text{CatWin}$ , and because  $\text{ECW} \subseteq \text{CatWin}$ ,  $v_0 \notin \text{ECW}$  so that all vertices from  $\text{ECW}$  below  $v_0$  are on a same branch. The herder's strategy is to cut this branch from  $v_0$ . The resulting tree fails condition 2 of Theorem 15, and so the herder wins now.

( $\Rightarrow$ ) Suppose  $v_0 \in \text{CatWin}$ . Then on the cat starting at  $v_0$ , after any cut, the cat may move to a vertex satisfying  $\text{ECW}$  (using edge-disjointness of  $x, u$  and  $x, v$ -paths). Then the strategy of (2 implies 1) in Theorem 15 gives the cat a way to win once in  $\text{ECW}$ . ◀

### 3.3 Decidable cases for infinite tree arenas

Note that the case where the arena is some complete  $d$ -ary tree  $T_d$  is solved immediately: For the tree  $T_1$  is an infinite rooted path that herder in one initial cut can make finite, so that  $T_1$  is herder-win. For  $T_d$  where  $d \geq 2$ , there is an infinite binary tree minor and by Theorem 15, we necessarily have  $T_d \models \exists x: \text{ECW}(x)$ .

The setting becomes interesting for a tree arena  $A$  that is not a complete tree  $T_d$ , but that a regular set of nodes of some complete tree  $T_d$ : Suppose that the arena  $A$  is a regular subset of nodes of some tree  $T_d$ . In such a case,  $A$  is definable by some unary MSO-predicate  $A(\cdot)$ . Now, to decide whether the cat wins the game on  $A$  or not consists in evaluating the formula  $\exists x: \text{ECW}(x)$  “inside” the  $A$  part of the tree  $T_d$ . This is classically expressed by the *relativization* of  $\exists x: \text{ECW}(x)$  w.r.t.  $A(\cdot)$ .

Preliminary to formally defining it, we introduce convenient notations for first-order and second-order (existential) quantifiers. For every MSO-formula  $\varphi$ , we let:

- $\exists^{\psi(\cdot)} x \varphi := \exists x: (\psi(x) \wedge \varphi)$
- $\exists^{\psi(\cdot)} X \varphi := \exists X: ((\forall x: X(x) \Rightarrow \psi(x)) \wedge \varphi)$

Such quantifiers require the quantified object is confined within the set defined by the unary predicate  $\psi(\cdot)$ . Using such quantifiers, Formally, the *relativization* of an MSO-formula  $\varphi$  w.r.t. a unary MSO-predicate  $\psi(\cdot)$ , written  $\langle \varphi | \psi(\cdot) \rangle$ , can be inductively defined as follows.

$$\begin{aligned}
 \langle \text{Succ}(x, y) | \psi(\cdot) \rangle &:= \text{Succ}(x, y) \\
 \langle X(x) | \psi(\cdot) \rangle &:= X(x) \\
 \langle \neg \varphi | \psi(\cdot) \rangle &:= \neg \langle \varphi | \psi(\cdot) \rangle \\
 \langle \varphi \wedge \psi | \psi(\cdot) \rangle &:= \langle \varphi | \psi(\cdot) \rangle \wedge \langle \psi | \psi(\cdot) \rangle \\
 \langle \exists x: \varphi | \psi(\cdot) \rangle &:= \exists^{\psi(\cdot)} x: \langle \varphi | \psi(\cdot) \rangle \\
 \langle \exists X: \varphi | \psi(\cdot) \rangle &:= \exists^{\psi(\cdot)} X: \langle \varphi | \psi(\cdot) \rangle
 \end{aligned}$$

Notice that by the propagation of the formula  $\psi(\cdot)$  down the syntactic tree of the original formula  $\varphi$ , the quantifier alternation depth (q.a.d for short) of the formula  $\langle \varphi | \psi(\cdot) \rangle$  is at most the sum of the q.a.d of each formula, plus one since the outermost quantifier of the propagated formula might occur in the scope of a dual quantifier in the original formula.

We can now formally define the CAT HERDING PROBLEM we consider.

CAT HERDING PROBLEM

<b>Input:</b> A degree $d \in \mathbb{N}$ and an MSO-formula $A(\cdot)$ . <b>Output:</b> $T_d \models \langle (\exists x: \text{ECW}(x))   A(\cdot) \rangle$ ?
---

By Rabin’s Theorem [17] we know that the CAT HERDING PROBLEM is decidable. We below provide an upper bound for it.

Note that when resorting to alternating automata constructions [16], model checking an MSO-formula on the tree  $T_d$  requires some nested exponential-time operations that depend on its *quantifier alternation depth* (shortly written q.a.d in the remaining of the paper). The q.a.d of an MSO-formula is the maximum number of alternations between existential and universal quantifiers along any path in the formula’s parse tree.

In particular, when the MSO-formula with q.a.d  $k$  has an existential outermost quantifier and is expressed only with atomic formulas based on the binary symbol *Succ*, the model checking can be done in  $(k + 1)$ -EXPTIME. The original result comes from the logic *SdS*, the monadic second-order theory with  $d$ -successors, see [17].

## 10:12 Cat Herding Game Played on Infinite Trees

It therefore remains to determine the q.a.d of our formula  $\langle \exists x: \text{ECW}(x) | A(\cdot) \rangle$  to obtain an upper bound complexity. However, we preliminary need to translate every occurrence of atomic formulas of the form  $x \leq y$  in terms of the *Succ* predicate. This is a classic translation that we recall here.

A formula  $x \leq y$  holds of two nodes  $u$  and  $v$  in a tree  $T = (N, r, \text{succ})$  if  $v$  belongs to the descendants of  $u$  in  $T$ . We can characterize these descendants as the smallest set containing  $u$  and that is closed by the *succ* relation – recall that set  $V \subseteq N$  is *succ*-closed whenever  $\{w \in N \mid \exists v \in V, \text{succ}(v, w)\} \subseteq V$ . We can express the *succ*-closeness in MSO:

$$\text{isSuccClosed}(X) := \forall y: (\exists x: X(x) \wedge \text{Succ}(x, y)) \Rightarrow X(y) \quad (4)$$

We now define the descendants of  $x$ , namely the smallest *succ*-closed set that contains  $x$ :

$$\begin{aligned} \text{isDescendantsOf}(X, x) := & X(x) \wedge \text{isSuccClosed}(X) \wedge \\ & \forall Y: (Y(x) \wedge \text{isSuccClosed}(Y) \Rightarrow \forall y: (X(y) \Rightarrow Y(y))) \end{aligned} \quad (5)$$

The atomic formula  $x \leq y$  now translates as follows.

$$x \leq y := \exists X(\text{isDescendantsOf}(X, x) \wedge X(y))$$

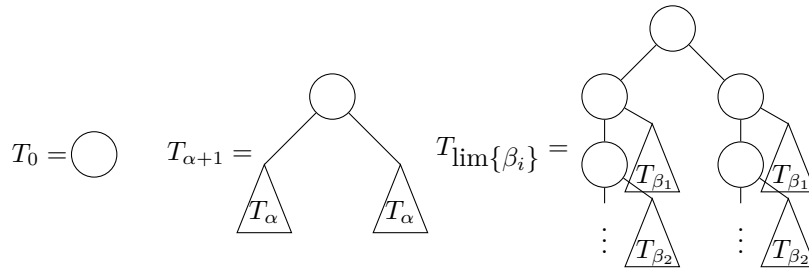
Note that the q.a.d of the formula  $x \leq y$  is 2 (this can be seen by drawing the its parse tree) with an existential outermost quantifier.

Noticing that formula  $\exists x: \text{ECW}(x)$  as written in (2) has q.a.d 3, it can be verified that by replacing in  $\exists x: \text{ECW}(x)$  every occurrence of an atomic formula of the  $y \leq z$  by its translation yields a formula with a q.a.d equal to 6. Assuming that the input formula  $A(\cdot)$  expressed only with predicate *Succ* (no occurrence of  $\leq$ ) has q.a.d equal to  $a$ , we get that the translation of the main formula  $\langle \exists x: \text{ECW}(x) | A(\cdot) \rangle$  has a q.a.d of at most  $6 + a + 1$ .

Because this MSO-formula has an existential outermost quantifier, the CAT HERDING PROBLEM can be solved in  $(a + 8)$ -EXPTIME.

### 4 Transfinite Cat Numbers

We introduce some transfinite structures including ordinals, which loosely correspond to the cat number, or the height of binary minors (binary minors of height  $\alpha$  rooted at vertices in  $V_\alpha$ ). We show with a brief construction that for any given countable ordinal  $\alpha$ , there exists an infinite tree with vertices in  $V_\alpha$ , but still herder-win.



■ **Figure 2** Binary tree constructions for every countable ordinal  $\alpha$ .

Consider the tree constructions as depicted in Figure 2. Note that all  $T_\alpha$  are herder-win, as every such tree allows the herder to have strategies to reduce the cat to a sub-tree  $T_{\alpha'}$  with  $\alpha' < \alpha$  within finitely many moves. Further, the root of every tree  $T_\alpha$  is in  $V_\alpha$ . This is

trivial for  $T_0$ . For successor ordinals, the root contains at least two edge-disjoint paths to vertices in  $V_{\alpha-1}$  inductively. For limit ordinals  $\alpha = \lim\{\beta_i\}$ , every  $\beta < \alpha$  has some  $\beta_i$  with  $\beta \leq \beta_i$ , and so the root having edge-disjoint paths to vertices in  $V_{\beta_i}$  is sufficient to show the root is in  $V_\beta$  for all  $\beta < \alpha$ , and thus the root is in  $V_\alpha$ . Because this argument does not rely on the particular  $\beta_i$  sequence chosen, limit ordinals may enjoy distinct trees, but we may still conclude that the root of any tree constructed this way has a root in  $V_\alpha$ .

Recall that the supremum of all countable ordinals is  $\omega_1$ . We observe that any tree construction from countable ordinal trees with no vertices of infinite degree must be countable (by breadth first search). Thus the construction involves a countable sequence of countable ordinals, which must be countable, and thus below  $\omega_1$ . The only hope for more is to allow infinite degree vertices. Indeed, once this is allowed, providing a branch from the root to tree  $T_\beta$  for every  $\beta < \alpha$  puts the root vertex in  $V_\alpha$ , for any limit ordinal  $\alpha$ . This allows for the construction of any given ordinal that is still herder-win (make a passing move, cut cat onto smaller ordinal tree), but we have to give up on having binary trees, so this construction is not viable for the automatic analysis we do later.

We may also investigate whether these constructions are regular. Write  $L(T_\alpha)$  for the language of addresses in tree  $T_\alpha$ , and write  $\text{Pref}(L)$  for the prefix language of  $L$ . Then, we observe that  $L(T_0) = \epsilon$ ,  $L(T_{\alpha+1}) = \text{Pref}((0 \cup 1)L(T_\alpha))$ , and  $L(T_{\lim\{\beta_i\}}) = \text{Pref}(0^i 1 L(T_{\beta_i}) \cup 1^i 0 L(T_{\beta_i}))$ . Thus this limit ordinal construction does not yield a regular language (as there are infinitely many non-isomorphic sub-trees).

We will push an interpretation of the set of nodes  $V_\alpha$  to a representation of the cat number. This requires generalizing our notion of cat number to ordinals. If  $T$  is an infinite tree where the cat is captured in exactly  $n$  cuts under optimal herder play, we say that  $\text{cat}(T) = n$ . We generalize this notion to transfinite ordinals, which to our knowledge has never been investigated in the literature.

► **Definition 17.** Let  $T = (N, r, \text{succ})$  be a tree for a game of Cat Herding,  $v \in N$  and  $\alpha$  be an ordinal. We define ordinal  $\text{cat}(T, v)$  as follows – we write  $T_c = (N_c, r, \text{succ})$  for the tree resulting from a single cut  $c$  in  $T$ .

If for every cut  $c$  in  $T$  and every  $\beta < \alpha$ , the cat can move to a vertex  $v' \in N_c$  such that  $\text{cat}(T_c, v') \geq \beta$ , then  $\text{cat}(T, v) \geq \alpha$ .

Conversely, if there exists a cut  $c$  and  $\beta < \alpha$  such that all reachable vertices  $v' \in N_c$  we have  $\text{cat}(T_c, v') < \beta$ , then  $\text{cat}(T, v) < \alpha$ .

We observe that the transfinite ordinal cat numbers are loosely connected to the transfinite sets  $V_\alpha$  (the  $V_\alpha$  are not sensitive to rays, providing a factor of error of  $\omega$ ).

► **Lemma 18.** Let  $T = (N, r, \text{succ})$  be a tree,  $v \in N$  and  $\alpha$  be an ordinal. If  $v \in V_\alpha$ , then  $\text{cat}(T, v) \geq \alpha$ , otherwise  $\text{cat}(T, v) \leq \alpha\omega$ .

**Proof.** If  $v \in V_\alpha$ , then there are edge-disjoint descendant paths to  $v' \in V_\beta$  for all  $\beta < \alpha$ . The herder cannot block both with their edge cut, so  $\text{cat}(T, v) \geq \alpha$  by Definition 17. Otherwise  $v \notin V_\alpha$ , then the herder's strategy is to cut above the cat – write  $T'$  for the remaining tree. If there exists  $\beta < \alpha$  such that for all  $v' \in V(T')$ ,  $\text{cat}(T, v') < \beta$ , then by definition  $\text{cat}(T, v) < \alpha \leq \alpha\omega$ . If instead for every  $\beta < \alpha$  there is a reachable vertex  $v_\beta$  such that  $\text{cat}(T', v_\beta) \geq \beta$ , we have seen (proof of Lemma 14) that the set  $\{v_\beta\}_{\beta < \alpha}$  of all those vertices must lie on a single ray from  $v$ . However, within finitely many moves ( $< \omega$  from the ray argument in Lemma 13), the herder can force the cat to a vertex  $v' \notin V_\beta$ , for some  $\beta < \alpha$ . The resulting play involves the cat being captured in at most  $\beta\omega$ . Given that  $\beta$  may be unbounded, or  $\alpha = \beta + 1$ , the overall score for is at most  $\beta\omega + \omega \leq \alpha\omega$ . ◀

Changing the construction of Figure 2's case of successor ordinal to the same as the limit ordinal with  $\beta_i = \alpha$ , we obtain a new family of trees witnessing the upper bound as tight, as all vertices along the two main rays are in  $V_{\alpha+1}$ , so the herder must take at least  $\omega$  moves (using the lower bound) to remove the cat's access to the  $V_{\alpha+1}$ , pushing them to the next lowest ordinal. The given construction also witnesses the lower bound, as the root of each  $T_\alpha$  is in  $V_\alpha$ , but no children vertices are.

► **Definition 19.** Let  $T = (N, r, \text{succ})$  be a tree. We define  $\text{cat}(T) = \sup_{v \in N} \{\text{cat}(T, v)\}$ .

► **Corollary 20.** If  $\alpha$  is a countable ordinal, then there exists a herder-win binary tree  $T$  with  $\text{cat}(T) \geq \alpha$ . If  $\alpha$  is any ordinal, then there exists a (possibly locally infinite) herder-win tree  $T$  with  $\text{cat}(T) \geq \alpha$ .

**Proof.** Consider building a binary tree  $T_\alpha$  as done in Figure 2 (we recall that for a limit ordinal  $\alpha$  the resulting tree is sensitive to the chosen  $\{\beta_i\}$  that reaches  $\alpha$ ). Still whichever  $T_\alpha$  is obtained, we have argued that its root belongs to  $V_\alpha$ , so that, by Lemma 18,  $\text{cat}(T) \geq \alpha$ . ◀

## 5 Conclusion and Discussion

We have approached the recently introduced pursuit-evasion game Cat Herding [2] through the lens of logic, and demonstrated how logic can help in many respects.

First, the logic allows to extend the setting to particular infinite arenas, namely trees, and is amenable to finding decision procedures for the infinite class of regular infinite tree arenas. We have introduced the CAT HERDING PROBLEM and shown an upper bound for it. To our knowledge, these results are new. They pave the way to follow-up questions such as the complexity lower bounds – we currently are not aware of any concrete approach to tackle it. Also, one may consider variants where the “the cat may not stay still” constraint is relaxed, and that do not reduce to the original problem.

Second, and interestingly, our construction involving the formula  $\text{DBChildin}(X, x)$  motivates a notion of transfinite cat numbers, and while we provided some rough bounds on the cat numbers of vertices, our arguments exhibit a factor of  $\omega$  on the bounds, leaving a natural future direction to tighten up a procedure for labeling all nodes with their cat ordinal.

Beyond immediate future direction, it remains to better understand how the logical setting could be used beyond infinite-tree arenas. In a large class of infinite graphs, such as the Caucal Hierarchy [7], our logical approach does not help to solve Cat Herding on arbitrary graphs of this hierarchy: playing Cat Herding on a given graph of the hierarchy obviously cannot reduce to playing Cat Herding in the complete binary tree – recall that in the complete binary tree, the cat can survive indefinitely, while single rays do belong to the hierarchy. Since, playing a cut in a graph of the hierarchy can be reflected as a “regular” set of cuts in the complete binary tree, there might be well-tuned new games to study on the complete binary tree that would provide the missing counterpart of playing Cat herding in those graphs. Also, it might be the case that by using the logic  $\text{MSO}_2$  that also quantifies over edge sets, one may find a way to accurately trace back herder's cut – note that links between  $\text{MSO}_2$  and  $\text{MSO}$  are tight for a large class of graphs [8].

---

## References

- 1 Julius Althoetmar, Jamico Schade, and Torben Schürenberg. Complexity of firefighting on graphs, 2025. doi:10.48550/arXiv.2505.11082.
- 2 Rylo Ashmore, Danny Dyer, Trent Marbach, and Rebecca Milley. Cuts, cats, and complete graphs. *Theoretical Computer Science*, page 115393, 2025. doi:10.1016/j.tcs.2025.115393.



- 3 Rylo Ashmore, Danny Dyer, and Rebecca Milley. Extremal cat herding, 2025. doi:10.48550/arXiv.2505.07588.
- 4 Anthony Bonato, Florian Lehner, Trent Marbach, and Jd Nir. The localization game on locally finite trees. In *European Conference on Combinatorics, Graph Theory and Applications*, pages 149–155, January 2023. doi:10.5817/CZ.MUNI.EUROCOMB23-021.
- 5 Anthony Bonato, Florian Lehner, Trent G. Marbach, and JD Nir. Locally finite graphs and their localization numbers, 2024. doi:10.48550/arXiv.2404.02409.
- 6 Richard Borie, Craig Tovey, and Sven Koenig. Algorithms and complexity results for pursuit-evasion problems. In *IJCAI International Joint Conference on Artificial Intelligence*, pages 59–66, January 2009. URL: <http://ijcai.org/Proceedings/09/Papers/021.pdf>.
- 7 D. Caucal. On infinite terms having a decidable monadic theory. In *International Symposium on Mathematical Foundations of Computer Science*, pages 165–176. Springer, 2002.
- 8 Bruno Courcelle and Joost Engelfriet. *Graph structure and monadic second-order logic: a language-theoretic approach*, volume 138. Cambridge University Press, 2012.
- 9 Alexander Dean, Sean English, Tongyun Huang, Robert A. Krueger, Andy Lee, Mose Mizrahi, and Casey Wheaton-Werle. Firefighting on the hexagonal grid and on infinite trees. *arXiv: Combinatorics*, 2020. URL: <https://api.semanticscholar.org/CorpusID:222291324>.
- 10 Erich Grädel, Wolfgang Thomas, and Thomas Wilke, editors. *Automata, Logics, and Infinite Games: A Guide to Current Research [outcome of a Dagstuhl seminar, February 2001]*, volume 2500 of *Lecture Notes in Computer Science*. Springer, 2002. doi:10.1007/3-540-36387-4.
- 11 William B. Kinnarsley. Cops and robbers is exptime-complete. *Journal of Combinatorial Theory, Series B*, 111:201–220, 2015. doi:10.1016/j.jctb.2014.11.002.
- 12 Oddvar Kloster. A solution to the angel problem. *Theoretical Computer Science*, 389:152–161, December 2007. doi:10.1016/j.tcs.2007.08.006.
- 13 D. Kozen. Results on the propositional mu-calculus. *Theoretical Computer Science*, 27(3):333–354, 1983. Special Issue Ninth International Colloquium on Automata, Languages and Programming (ICALP) Aarhus, Summer 1982. doi:10.1016/0304-3975(82)90125-6.
- 14 Florian Lehner. Cops, robbers, and infinite graphs, 2015. doi:10.48550/arXiv.1410.8412.
- 15 ANDRÁS Máthé. The angel of power 2 wins. *Combinatorics Probability and Computing.*, 16(3):363–374, May 2007. doi:10.1017/S0963548306008303.
- 16 D. Muller and P. Schupp. Simulating alternating tree automata by nondeterministic automata: New results and new proofs of the theorems of Rabin, McNaughton and Safra. *Theoretical Computer Science*, 141(1-2):69–107, 1995. doi:10.1016/0304-3975(94)00214-4.
- 17 Michael O. Rabin. Decidability of second-order theories and automata on infinite trees. *Transactions of the American Mathematical Society*, 141:1–35, 1969. URL: <http://www.jstor.org/stable/1995086>.
- 18 Alfred Tarski. A lattice-theoretical fixpoint theorem and its applications, 1955.
- 19 Igor Walukiewicz. Monadic second-order logic on tree-like structures. *Theoretical Computer Science*, 275(1):311–346, 2002. doi:10.1016/S0304-3975(01)00185-2.
- 20 Douglas B. West. *Introduction to Graph Theory*. Prentice Hall, 2nd edition, September 2000.