Games on Interval and Permutation graph representations

Jessica Enright* *Computing Science and Mathematics, University of Stirling*† and Lorna Stewart‡
Department of Computing Science, University of Alberta§

June 22, 2015

Abstract

We describe combinatorial games on graphs in which two players antagonistically build a representation of a subgraph of a given graph. We show that for a large class of these games, determining whether a given instance is a winning position for the next player is PSPACE-hard. In contrast, we give polynomial time algorithms for solving some versions of the games on trees.

Keywords: combinatorial games, interval graphs, permutation graphs, set representations of graphs

1 Introduction

Games on graphs and their representations combine two fascinating areas of study: combinatorial games on graphs, and intersection and other representations of graphs. These areas have been the subjects of numerous books and papers (see, for example, the references at the end of this paper). The study of such games provides a new context for graph representation problems, and a new collection of combinatorial games on graphs. Although some of the games are tractable for some inputs, many of them are PSPACE-complete in general.

We consider games played on graphs in which two players construct a representation of a subgraph of a given graph by taking turns choosing a vertex and adding a corresponding element to the representation. At each stage, the representation that has been constructed correctly represents the subgraph induced by the vertices that have been chosen so far. The game ends when all of the vertices have been played, or when the representation cannot be extended to include any of the unchosen vertices. The last player to make a move wins.

Many combinatorial games on graphs have been studied. Two such games that are closely related to the representation games of this paper are Kayles and Generalized Geography. In the game of Kayles, a graph is given and two players take turns choosing a vertex that has not been chosen before and that is not adjacent to any previously chosen vertex. The last player to make a move wins. Solving an arbitrary instance of Kayles is known to be PSPACE-complete [12], and the complexity of Kayles on trees remains an open problem. Exact exponential time algorithms are given for the game on general graphs and on trees in [6]. In contrast, Kayles is solvable in polynomial time on graphs of bounded asteroidal number, circular-arc graphs, and circular permutation graphs [5], as well as on trees with at most one vertex of degree greater than two [10]. In the two-person game of Generalized Geography, a directed graph and a specified starting vertex are given. The first player must play the starting vertex on his first turn and after that the players take turns choosing a vertex that has an edge directed to it from the previously chosen vertex. The last player able to move wins. While it is PSPACE-complete to determine whether there is a winning strategy for the first player in Generalized Geography [12], the problem is solvable in linear time for

*jae@maths.stir.ac.uk
†Stirling, United Kingdom, FK9 4LA
‡lorna.stewart@ualberta.ca
§Edmonton, Alberta, Canada T6G 2E8

graphs of bounded treewidth [4]. The results of [4] include PSPACE-completeness results for several other path-forming games on directed and undirected graphs, with and without specified starting vertices, and polynomial time algorithms for some of those games on graphs of bounded treewidth and other graph classes including trees. In contrast, Generalized Geography on undirected graphs without specified starting vertex is solvable in polynomial time: there is a winning strategy for the first player if and only if the given graph does not have a perfect matching (see Ex 5.1.4 of [7]). In addition, the triangulation games of [1] are related to Kayles and some of them may be thought of as graph representation games.

We begin by defining general representation games and, in Sections 3 and 4, we develop a general PSPACE-hardness proof. In Section 5, we give polynomial time algorithms for restricted versions of connected Interval and Permutation Games on trees and for a related game, the Caterpillar Subgraph Game, on trees that have at most one vertex of degree greater than two. In the next section, we define the games that we will consider and some related notions.

2 Definitions and Notation

Concepts that are needed to understand the paper are given in this section and throughout the paper. Combinatorial games and graph representations will be discussed in more detail in Section 5. For more information about those topics, we refer the reader to [2], [3], [9], and [13].

2.1 Graphs, Representations, and Sets

We consider finite, simple, undirected graphs. Let $G = (V, E)$. For $W \subseteq V$, $G[W]$ denotes the subgraph of G induced by W. For $X \subseteq V$ and $F \subseteq E$, we use $G - X$ and $G - F$ to stand for $G[V \backslash X]$ and the graph $(V, E \backslash F)$, respectively. For $v \in V$, $N_G(v) = \{w \mid vw \in E\}$, and $degree_G(v) = |N_G(v)|$. The subscript G will be omitted when the context is clear.

Let S be a set and let Φ be a symmetric binary relation on S. An (S, Φ) *representation* of a graph $G = (V, E)$ is a total function $f : V \mapsto S$ such that for all pairs of distinct vertices $u, v \in V$, $uv \in E$ if and only if $f(u)\Phi f(v)$. Let $X = \{x_1, x_2, \ldots, x_n\}$ be a finite multiset and let Φ be a binary relation on X. Then the Φ *graph* of X is the graph on n vertices $v_1, v_2, \ldots, v_n$ such that for all $1 \leq i, j \leq n$, $i \neq j$, $v_iv_j \in E$ if and only if $x_i \Phi x_j$. The Φ *graph* of a given multiset is unique up to isomorphism.

We consider the following sets, among others:

- Unit Interval: the set of all unit intervals on the real line
- Interval: the set of all intervals on the real line
- Permutation: the set of all straight line segments connecting two parallel lines in the plane

When S is a set of sets, some possibilities for Φ are:

- Intersection: for all $s_i, s_j \in S$, $s_i \Phi s_j$ if and only if $s_i \cap s_j \neq \emptyset$
- Overlapping: for all $s_i, s_j \in S$, $s_i \Phi s_j$ if and only if $s_i \cap s_j \neq \emptyset$ and $s_i \not\subseteq s_j$ and $s_j \not\subseteq s_i$
- Containment: for all $s_i, s_j \in S$, $s_i \Phi s_j$ if and only if $s_i \subseteq s_j$ or $s_j \subseteq s_i$
- Disjointness: for all $s_i, s_j \in S$, $s_i \Phi s_j$ if and only if $s_i \cap s_j = \emptyset$

We sometimes use diagrams to specify unit interval, interval and permutation representations. Note that it is only the relative orders of interval and line segment endpoints that are significant in these types of representations. This notion will be discussed further in the next subsection.

Unit interval graphs, *interval graphs*, and *permutation graphs* are the graphs that have (Unit Interval, Intersection), (Interval, Intersection), and (Permutation, Intersection) representations, respectively. A *tree* is a connected acyclic graph. A *caterpillar* is a tree from which the removal of all leaves results in a path; the resulting path is called the *spine* of the caterpillar. A *long star* is a tree that has at most one vertex of degree greater than two.

We use $\uplus$ to denote the multiset union operation that results in a multiset in which an element appears with multiplicity the sum of its multiplicities in the operands. $\mathbb{Z}^+$ denotes the set of positive integers and $\mathbb{N}$ denotes the set of natural numbers. The function *parity* on $\mathbb{N}$ returns 0 for even numbers and 1 for odd numbers. For a set S, $\mathcal{P}(S)$ denotes the power set of S.

2.2 Representation Games

The (S, Φ) *Game* is the game in which two players construct an (S, Φ) representation of a subgraph of a given graph by taking turns choosing a vertex and adding a corresponding element of S to the representation. At each stage, the representation that has been constructed correctly represents the subgraph induced by the vertices that have been chosen so far. The game ends when all of the vertices have been played, or when the representation cannot be extended to include any of the unchosen vertices. The last player to make a move wins. The *connected* (S, Φ) Game is the (S, Φ) Game with the added requirement that at each stage, the subgraph of G induced by the chosen vertices is connected.

A *position*, or *instance*, of the (S, Φ) Game is the triple $(G = (V, E), V', f)$ where G is the graph on which the game is played, $V' \subseteq V$ is the set of vertices that have been played, and f is an (S, Φ) representation of $G[V']$. The *initial position* of the games that we consider is the given graph with no played vertices and the empty representation: $(G, \emptyset, \emptyset)$. Given a position $(G = (V, E), V', f)$, a *legal move* in the (S, Φ) Game is the choice of a vertex $v \in V \backslash V'$ and an element $s \in S$ such that $f' = f \cup \langle v, s \rangle$ is a Φ representation of $G[V' \cup \{v\}]$ (and $G[V' \cup \{v\}]$ is connected in the connected version of the game). A *winning position* is one from which the next player can win no matter what moves the other player makes. A *losing position* is one from which the previous player can win no matter what moves the other player makes. Since all of the games described in this paper are impartial two-person games – games in which from a given position the same moves are available to either player – and end with one winner, each position is either a winning position or a losing position. An *end position* is one from which there are no legal moves. End positions are losing positions. *Solving* a position, or instance, means determining whether it is a winning or a losing position. The examples of Figure 1 illustrate some of these definitions.

Since we want to examine the complexities of solving positions of representation games, we have to restrict the elements used in (S, Φ) representations. Specifically, we require that every instance $(G = (V, E), V', f)$ of the (S, Φ) Games that we consider satisfy both of the following properties:

1. f is *polynomially sized*, that is, the binary representation of each element in the range of f has number of bits polynomial in terms of $|V|$.

2. Φ is computable in polynomial time.

These restrictions do not affect the outcomes of the games that we consider. For example, if S is the set of all intervals on a line, the first player can choose any vertex and any interval to represent it, so he can obviously choose an interval with endpoints whose binary representations are polynomial in length. Thereafter, a player can choose an interval each of whose endpoints is at the midpoint between two already chosen interval endpoints, at an already chosen endpoint, or at an already chosen endpoint plus or minus a constant in $\{1, 2\}$, without affecting the play of the game or the graph represented. The third option allows a new interval to be added that does not intersect any previous intervals or that overlaps an interval whose endpoint is leftmost or rightmost. The binary representation of the midpoints is at most one longer than the binary representations of the already chosen points, and the game ends after at most $|V|$ moves; therefore the resulting representation is polynomial sized. Similar comments apply to permutation representations.

With these restrictions in mind, we now state the representation game problems.

(S, Φ) **Game**

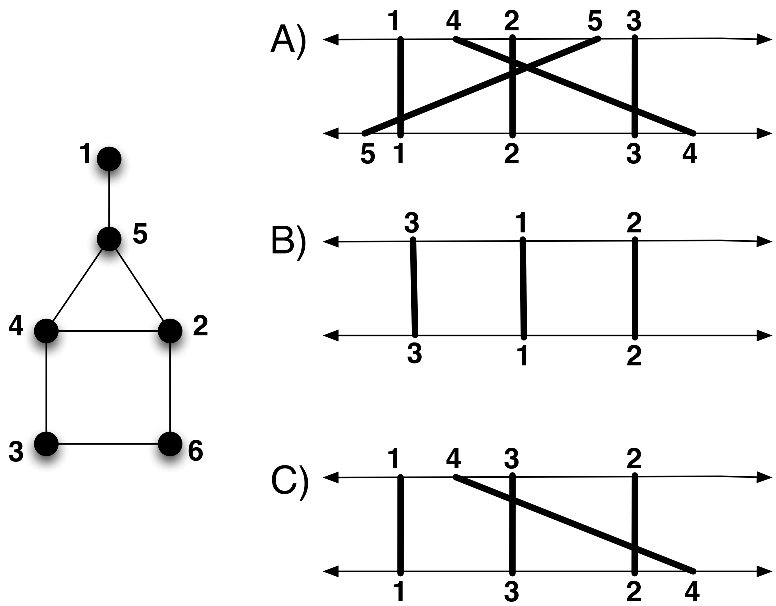

Figure 1: Three positions in the (Permutation, Intersection) Game on the given graph G. In each case, the set of played vertices and the representation are given in the form of a Permutation diagram where the label on each line segment indicates the represented vertex of G. The position in A) is a losing position; the positions in B) and C) are winning positions.

> Instance. Graph $G = (V, E)$, $V' \subseteq V$, and function $f : V' \mapsto S$, where f is an (S, Φ) representation of $G[V']$
>
> Question. Is the given position a winning position for the next player in the (S, Φ) Game?
>
> **Connected (S, Φ) Game**
>
> Instance. Graph $G = (V, E)$, $V' \subseteq V$, and function $f : V' \mapsto S$, where $G[V']$ is connected and f is an (S, Φ) representation for $G[V']$
>
> Question. Is the given position a winning position for the next player in the connected (S, Φ) Game?

In Kayles, given a graph $G = (V, E)$, two players take turns choosing a vertex of V that has not been chosen before and that is not adjacent to any previously chosen vertex. The last player to choose a vertex wins. An instance of Kayles is the graph $G = (V, E)$ on which the game is played and the set $U \subseteq V$ of vertices that have been chosen.

> **Kayles**
>
> Instance. Graph $G = (V, E)$ and subset $U \subseteq V$
>
> Question. Is the given position a winning position for the next player in Kayles?

As previously mentioned, solving an arbitrary Kayles position is known to be PSPACE-complete [12]. In Section 4, we prove hardness results for some representation games via a reduction from Kayles.

Later, we will equate some representation games with subgraph games, which are defined as follows. Let $\mathcal{H}$ be a class of graphs. In the $\mathcal{H}$ Subgraph Game, two players take turns choosing a vertex of a given graph G such that at each stage the subgraph of G induced by the chosen vertices is a member of the graph class $\mathcal{H}$. For example, when $\mathcal{H}$ is the class of graphs with no edges, the $\mathcal{H}$ Subgraph Game is Kayles. An instance of the $\mathcal{H}$ Subgraph Game is the graph $G = (V, E)$ on which the game is played and the set $U \subseteq V$ of vertices that have been chosen.

$\mathcal{H}$ Subgraph Game

Instance. Graph $G = (V, E)$ and subset $U \subseteq V$

Question. Is the given position a winning position for the next player in the $\mathcal{H}$ Subgraph Game?

3 Separability

In this section, we give some technical definitions and a lemma that will allow us to prove a general PSPACE-hardness result in Section 4. Examples illustrating the definitions of this section can be found in Example 5 and Figure 2.

Definition 1 *Let S be a set and Φ be a symmetric binary relation on S. For any $Q \subseteq R \subseteq S$ and $s \in S$, s is* consistent with (Q, R, S) *if for all $q \in Q$, $s\Phi q$ and for all $r \in R \backslash Q$, $\neg(s\Phi r)$. When the set S is clear from the context, we will write that s is consistent with (Q, R).*

Definition 2 *Let S be a set, Φ be a symmetric binary relation on S, n be a positive integer, $R_1, R_2, \dots, R_n$ be nonempty finite subsets of S, and $Q_1, Q_2, \dots, Q_n$ be subsets of S such that $Q_i \subseteq R_i$ for all $1 \leq i \leq n$. Then $(Q_1, R_1), (Q_2, R_2), \dots, (Q_n, R_n)$ are* separating set pairs for (S, Φ) *if for all $1 \leq i \leq n$, all of the following hold:*

1. *there exists $s_i \in S$ that is consistent with (Q_i, R_i)*
2. *for all $s_i, s_i' \in S$ consistent with (Q_i, R_i), for all $r \in R_1 \cup R_2 \cup \cdots \cup R_n$:*
   $s_i \Phi r$ *if and only if* $s_i' \Phi r$
3. *for all $1 \leq j \leq n$, $j \neq i$, for all s_i consistent with (Q_i, R_i) and all s_j consistent with (Q_j, R_j):*
   $\neg(s_i \Phi s_j)$

Definition 3 *Let S be a set and Φ be a symmetric binary relation on S. Then (S, Φ) is* separable *if for every positive integer n, there exist n* separating set pairs *for (S, Φ).*

Consider n separating set pairs $(Q_1, R_1), (Q_2, R_2), \dots, (Q_n, R_n)$ for (S, Φ) and let s_i be an element of S that is consistent with (Q_i, R_i) for $1 \leq i \leq n$. Observe that the Φ graph of the multiset $R_1 \uplus R_2 \uplus \cdots \uplus R_n \uplus \{s_1, s_2, \dots, s_n\}$ is the same, regardless of the choices of $s_1, \dots, s_n$, by virtue of Definition 2. If it is connected then $(Q_1, R_1), (Q_2, R_2), \dots, (Q_n, R_n)$ are called *c-separating set pairs* for (S, Φ). If c-separating set pairs exist for all $n \in \mathbb{Z}^+$ then (S, Φ) is said to be *c-separable*.

Finally, (S, Φ) is said to be *polynomial time* separable (respectively, c-separable) if (S, Φ) is separable (respectively, c-separable) and for each $n \in \mathbb{Z}^+$, n separable (respectively, c-separable) set pairs for (S, Φ) and an element consistent with each can be computed in time polynomial in n.

The next lemma shows how separable properties are inherited by isomorphic sets and supersets and relations. Before stating the lemma, we review the definition of isomorphism in this context. Let S and T be sets and let Φ and Ψ be symmetric binary relations on S and T, respectively. Then (S, Φ) and (T, Ψ) are *isomorphic* if there is a bijection $f : S \mapsto T$ such that for all $x, y \in S$, $x\Phi y$ if and only if $f(x)\Psi f(y)$. The bijection f is called an *isomorphism* from (S, Φ) to (T, Ψ).

Lemma 4 *For sets S, T and U where $T \subseteq U$, and symmetric binary relations Φ and Ψ on S and U, respectively, such that (S, Φ) and (T, Ψ) are isomorphic: if (S, Φ) is separable (respectively, c-separable) then so is (U, Ψ).*

Proof. Let $n \in \mathbb{Z}^+$. Let $(Q_1, R_1), (Q_2, R_2), \ldots, (Q_n, R_n)$ be separating set pairs for (S, Φ) and let f be an isomorphism from (S, Φ) to (T, Ψ). For $1 \le i \le n$, let $X_i = \{f(q) \mid q \in Q_i\}$ and $Y_i = \{f(r) \mid r \in R_i\}$. Since each of $R_1, \ldots, R_n$ is a nonempty finite subset of S, each of $Y_1, \ldots, Y_n$ is a nonempty finite subset of T and therefore of U. Since $Q_i \subseteq R_i$ for all $1 \le i \le n$ and f is a bijection, $X_i \subseteq Y_i$ for all $1 \le i \le n$. We show that $(X_1, Y_1), \ldots, (X_n, Y_n)$ are separating set pairs for (U, Ψ).

Let $1 \le i \le n$.

1. Consider an element $s_i \in S$ that is consistent with (Q_i, R_i). Combining the definitions of s_i and f leads to the following facts: for all $r \in Q_i$, $s_i \Phi r$ and $f(s_i) \Psi f(r)$, and for all $r \in R_i \backslash Q_i$, $\neg(s_i \Phi r)$ and $\neg(f(s_i) \Psi f(r))$. Therefore, by the definitions of X_i and Y_i, $f(s_i)$ is consistent with (X_i, Y_i).

2. Let z_i and z_i' be consistent with (X_i, Y_i) and let $y \in Y_j$ where $1 \le j \le n$. By the definitions of X_i and Y_i and because f is an isomorphism, $f^{-1}(z_i)$ and $f^{-1}(z_i')$ are consistent with (Q_i, R_i) and $f^{-1}(y)$ is in R_j. Therefore, $f^{-1}(z_i) \Phi f^{-1}(y)$ if and only if $f^{-1}(z_i') \Phi f^{-1}(y)$; therefore since f is an isomorphism, $z_i \Psi y$ if and only if $z_i' \Psi y$.

3. Let $1 \le j \le n$, $j \ne i$, and let z_i and z_j be consistent with (X_i, Y_i) and (X_j, Y_j) respectively. Then, by Definition 1 and since f is an isomorphism, for all $y \in X_i$, $z_i \Psi y$ and $f^{-1}(z_i) \Phi f^{-1}(y)$, and for all $y \in Y_i \backslash X_i$, $\neg(z_i \Psi y)$ and $\neg(f^{-1}(z_i) \Phi f^{-1}(y))$. Similarly, $f^{-1}(z_j) \Phi f^{-1}(y)$ for $y \in X_j$ and $\neg(f^{-1}(z_j) \Phi f^{-1}(y))$ for $y \in Y_j \backslash X_j$. Therefore, $f^{-1}(z_i)$ and $f^{-1}(z_j)$ are consistent with (Q_i, R_i) and (Q_j, R_j), respectively, which implies that $\neg(f^{-1}(z_i) \Phi f^{-1}(z_j))$ which in turn implies $\neg(z_i \Psi z_j)$.

Thus, if (S, Φ) is separable then so is (U, Ψ). To complete the proof, we need to consider c-separability. Since f is an isomorphism from (S, Φ) to (T, Ψ), the Φ graph of any finite multiset S', each element of which is an element of S, is isomorphic to the Ψ graph of the multiset $\uplus_{s' \in S'} \{f(s')\}$. Therefore, if $(Q_1, R_1), \ldots, (Q_n, R_n)$ are c-separating set pairs for (S, Φ) then $(X_1, Y_1), \ldots, (X_n, Y_n)$ are c-separating set pairs for (U, Ψ). $\square$

Example 5 *For each set and binary relation pair below, if the pair is c-separable (respectively separable) we give n c-separating (respectively separating) set pairs $(Q_1, R_1), \ldots, (Q_n, R_n)$ and consistent elements $s_1, \ldots, s_n$ for arbitrary $n \in \mathbb{Z}^+$, or provide some other justification. We provide demonstrating figures for parts 1, 2, 3, and 7 of the example in Figure 2.*

1. (Unit Interval, Intersection) is c-separable.

   For $1 \le i \le n$ let:

   $Q_i = \{[5i+1, 5i+2], [5i+2, 5i+3], [5i+3, 5i+4]\}$
   $R_i = \{[5i, 5i+1], [5i+1, 5i+2], [5i+2, 5i+3], [5i+3, 5i+4], [5i+4, 5i+5]\}$
   $s_i = [5i+2, 5i+3]$

2. (Unit Interval, Overlapping) is c-separable.

   For $1 \le i \le n$ let:

   $Q_i = \{[5i+1, 5i+2], [5i+3, 5i+4]\}$
   $R_i = \{[5i, 5i+1], [5i+1, 5i+2], [5i+2, 5i+3], [5i+3, 5i+4], [5i+4, 5i+5]\}$
   $s_i = [5i+2, 5i+3]$

3. (Unit Interval, Containment) is separable but not c-separable.

   For $1 \le i \le n$ let:

   $Q_i = \{[5i+2, 5i+3]\}$
   $R_i = \{[5i, 5i+1], [5i+1, 5i+2], [5i+2, 5i+3], [5i+3, 5i+4], [5i+4, 5i+5]\}$
   $s_i = [5i+2, 5i+3]$

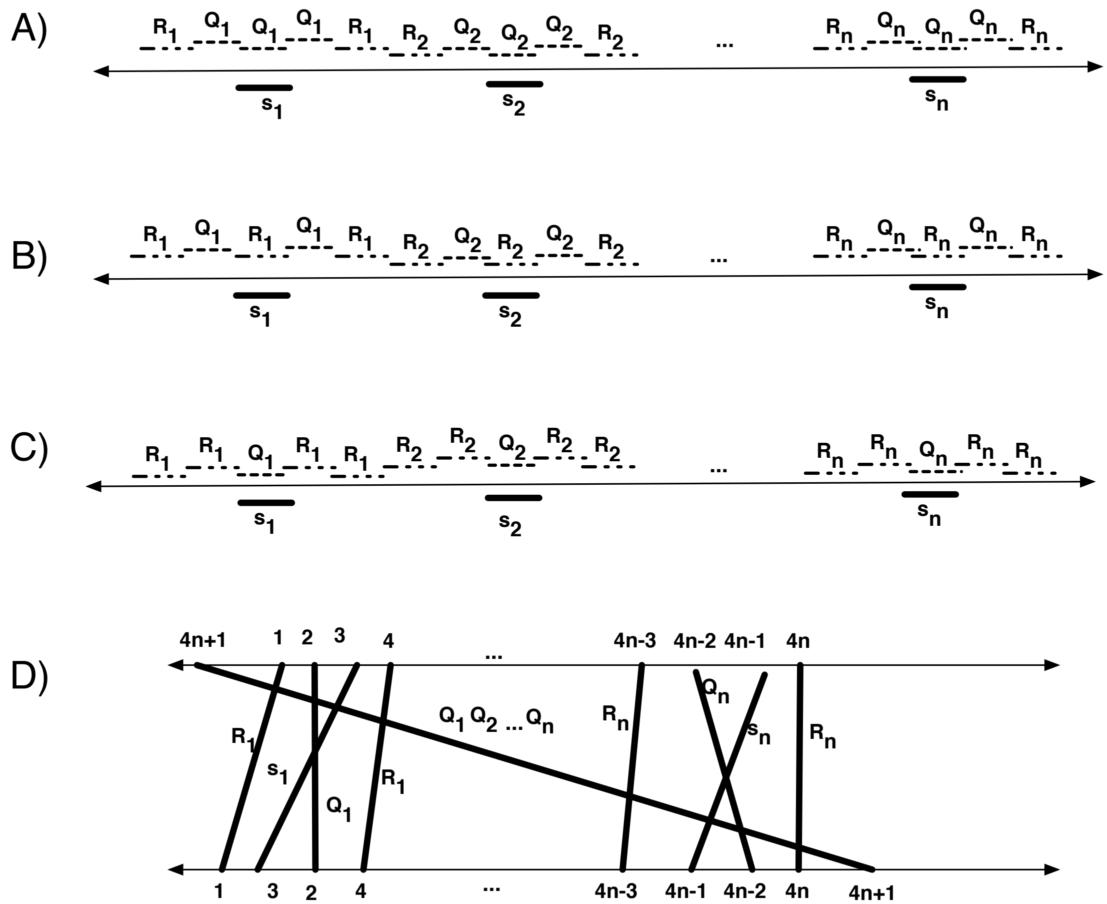

Figure 2: Figures showing some of the constructions described in parts 1, 2, 3, and 7 of Example 5. Note that the sets labeled as in Q_i are also in R_i. Separating set pairs (Q_i, R_i) and consistent elements s_i, for $1 \leq i \leq n$, where n is an arbitrary positive integer, for parts 1, 2, 3, and 7 of Example 5: **A**) (Unit Interval, Intersection) **B**) (Unit Interval, Overlapping) **C**) (Unit Interval, Containment) and **D**) (Permutation, Intersection). Note that the set pairs in parts A, B, and D are c-separating while the set pairs in part C are separating but not c-separating.

We now show that (Unit Interval, Containment) is not c-separable. Assume for contradiction that $(Q_1, R_1), \ldots, (Q_n, R_n)$ are c-separating set pairs with consistent elements $s_1, \ldots, s_n$ for arbitrary positive integer n. Note that if a unit interval is contained in another, then the intervals must be identical. Therefore, if the containment graph of $R_1 \uplus R_2 \uplus \cdots \uplus R_n \uplus \{s_1, s_2, \ldots, s_n\}$ is connected, then the sets $Q_1, \ldots, Q_n, R_1, \ldots, R_n, \{s_1, \ldots, s_n\}$ are all equal and contain a single interval. But then condition 3 of Definition 2 is not satisfied for $n > 1$.

4. (Unit Interval, Disjointness) is c-separable.

   For $1 \leq i \leq n$ let:

   $Q_i = \{[1, 2]\}$
   $R_i = \{[1, 2], [3, 4], [5, 6]\}$
   $s_i = [4, 5]$

5. (Interval, Containment) is c-separable.

   For $1 \leq i \leq n$ let:

   $Q_i = \{[6i + 1, 6i + 2], [6i + 4, 6i + 5], [6, 6(n + 1)]\}$
   $R_i = \{[6i, 6i + 3], [6i + 3, 6i + 6], [6i + 1, 6i + 2], [6i + 4, 6i + 5], [6, 6(n + 1)]\}$
   $s_i = [6i + 1, 6i + 5]$

6. (A, Intersection) is not separable, where A is any set of sets that all have an element in common.

   This follows from the fact that condition 3 of Definition 2 cannot be satisfied for $n > 1$.

7. (Permutation, Intersection) is c-separable.

   We give two permutations π_1 and π_2 and identify subsets of the lines in the resulting permutation diagram as c-separating set pairs and consistent elements.

   $\pi_1 = [4n + 1, 1, 2, 3, 4, \ldots, 4n - 3, 4n - 2, 4n - 1, 4n]$

   $\pi_2 = [1, 3, 2, 4, \ldots, 4n - 3, 4n - 1, 4n - 2, 4n, 4n + 1]$

   For $1 \leq i \leq n$ let:

   $Q_i = \{4i - 2, 4n + 1\}$
   $R_i = \{4i - 3, 4i - 2, 4i, 4n + 1\}$
   $s_i = 4i - 1$

8. (Permutation, Overlapping) is c-separable.

   This is implied by the permutations, separating set pairs, and consistent elements of 7.

9. (Permutation, Containment) is separable but not c-separable.

   Any pair of line segments connecting two parallel lines that satisfy containment must be identical. This case is similar to 3.

10. (Permutation, Disjointness) is c-separable.

    $\pi_1 = [1, 2, 3, 4, \ldots, 4n - 3, 4n - 2, 4n - 1, 4n, 4n + 1]$

    $\pi_2 = [4n - 3, 4n - 1, 4n - 2, 4n, \ldots, 1, 3, 2, 4, 4n + 1]$

    For $1 \leq i \leq n$ let:

    $Q_i = \{4i - 3, 4i, 4n + 1\}$
    $R_i = \{4i - 3, 4i - 2, 4i, 4n + 1\}$
    $s_i = 4i - 1$

11. $\mathcal{P}(\mathbb{N})$, the power set of natural numbers, and Intersection, Overlapping, Containment, Disjointness are not c-separable.

    The justifications are analogous to those of 1, 2, 5, and 4, respectively.

12. Unit disks in the plane and Intersection, Overlapping, Disjointness are c-separable.

    This follows from Lemma 4 and parts 1, 2, and 4.

13. Subtrees of a tree and Intersection, Overlapping, Containment, Disjointness are c-separable.

    This follows from Lemma 4 and parts 1, 2, 5, and 4, respectively.

Note that all of the separating set pairs and consistent elements mentioned in Example 5 are computable in polynomial time. In cases where Lemma 4 is invoked, the proof of polynomial time computability involves identification of a polynomial time computable isomorphism.

4 PSPACE-hardness

This section contains a general PSPACE-hardness reduction from Kayles to any polynomial time separable representation game. Recall that Φ is required to be polynomial time computable as discussed in Section 2.2.

Theorem 6 *Let S be a set and let Φ be a symmetric binary relation on S. If (S, Φ) is polynomial time separable then the (S, Φ) Game is PSPACE-hard.*

Proof. We give a polynomial time reduction from Kayles to the (S, Φ) Game. Let $G = (V, E)$ and $U \subseteq V$ be an arbitrary instance of Kayles. We construct a graph $G' = (V', E')$ and an (S, Φ) representation of a subgraph of G' that constitutes a next player win position in the (S, Φ) Game if and only if (G, U) is a next player win position of Kayles.

Let $V = \{v_1, v_2, \ldots, v_n\}$. Let $(Q_1, R_1), \ldots, (Q_n, R_n)$ be polynomial time computable separating set pairs for (S, Φ) and $s_1, \ldots, s_n$ be polynomial time computable elements of S that are consistent with the respective pairs. For each $1 \leq i \leq n$, we will refer to the elements of R_i as $r_i^1, r_i^2, \ldots, r_i^{|R_i|}$. First, we define the vertices of G': $V' = V \cup X$ where $X = \cup_{i=1}^n \{x_i^1, x_i^2, \ldots, x_i^{|R_i|}\}$ where the x_i^j vertices are distinct new vertices that do not appear in V. Next, we give an (S, Φ) representation for $G'[U \cup X]$: for all $1 \leq i \leq n$, $1 \leq j \leq |R_i|$, $f(x_i^j) = r_i^j$, and for all $v_i \in U$, $f(v_i) = s_i$. To complete the construction, the edges of G' are:

$$\begin{aligned} E' = E &\cup \{v_i x_i^k \mid 1 \leq i \leq n, 1 \leq k \leq |R_i|, \text{ where } r_i^k \in Q_i\} \\ &\cup \{x_i^k x_j^\ell \mid 1 \leq i, j \leq n, 1 \leq k \leq |R_i|, 1 \leq \ell \leq |R_j|, \text{ where } r_i^k \Phi r_j^\ell\} \\ &\cup \{v_i x_j^\ell \mid 1 \leq i, j \leq n, i \neq j, 1 \leq \ell \leq |R_j|, \text{ where } s_i \Phi r_j^\ell\}. \end{aligned}$$

Now we show that for all U' such that $U \subseteq U' \subseteq V$ where U' is an independent set in G, a vertex is a legal move in the Kayles instance (G, U') if and only if it is a legal move in the (S, Φ) instance $(G', U' \cup X, f')$, where $f' = f \cup \{\langle v_i, s_i \rangle \mid v_i \in U' \backslash U\}$.

For the only if part, suppose that v_i is a legal move in the Kayles instance (G, U'). The played vertices in instance $(G', U' \cup X, f')$ of the (S, Φ) Game are $U' \cup X$. Since v_i is a legal move in Kayles, $v_i \in V \backslash U'$ and therefore has not been played in the (S, Φ) Game, and v_i is not adjacent to any vertices of U' in G or in G'. We will show that in the (S, Φ) Game, for any previously played vertex v, s_i is related by Φ to $f'(v)$ if and only if $v_i v \in E'$ and thus playing v_i and setting $f'(v_i) = s_i$ is a legal move in the (S, Φ) Game. Let v be a previously played vertex; then $v \in U'$ or $v \in X$. If $v \in U'$ then $v = v_j \in V$ and $f'(v) = s_j$ for some $1 \leq j \leq n$, $j \neq i$. Since $v \in U'$, $v_i v \notin E'$ and by the definition of separating set pairs, $\neg(s_i \Phi s_j)$ as required. If $v \in X$ then either

1. $v = x_i^k$ for some $1 \leq k \leq |R_i|$ and $f'(v) = r_i^k$, or
2. $v = x_j^\ell$ for some $1 \leq j \leq n$, $j \neq i$, and some $1 \leq \ell \leq |R_j|$, and $f'(v) = r_j^\ell$.

In case 1, $v_i v \in E'$ if and only if $f'(v) \in Q_i$, which is the case if and only if $s_i \Phi f'(v)$ since s_i is consistent with (Q_i, R_i). In case 2, $v_i v \in E'$ if and only if $s_i \Phi f'(v)$. In both cases we have the required conclusion.

We now prove the if part. By the construction of G' and f', each vertex v_j of V that is played in the (S, Φ) Game must be represented by an element of S that is consistent with (Q_j, R_j). Therefore, since $(Q_1, R_1), \ldots, (Q_n, R_n)$ are separating set pairs, only vertices of V that induce an independent set in G can ever be played. Thus, the vertex played in a legal move in the (S, Φ) Game is independent from all previously played vertices of V and is therefore a legal move in the Kayles instance.

The previous argument implies that the set of all legal sequences of vertices that could be chosen in the Kayles instance (G, U) is identical to the set of legal sequences of vertices that could be chosen in the (S, Φ) Game $(G', U \cup X, f)$. Clearly, each sequence results in a win for the same player in both games since its length dictates the winner. We conclude that an arbitrary Kayles instance (G, U) is a next player win position if and only if the (S, Φ) Game $(G', U \cup X, f)$ obtained from (G, U) by the previously described construction, is a next player win position. The PSPACE-hardness result follows by the fact that the reduction can be computed in polynomial time and the fact that Kayles is PSPACE-complete [12]. □

Theorem 7 *Let S be a set and let Φ be a symmetric binary relation on S. If (S, Φ) is polynomial time c-separable then the connected (S, Φ) Game is PSPACE-hard.*

Proof. The proof is the same as the proof of Theorem 6, except that $(Q_1, R_1), \ldots, (Q_n, R_n)$ are c-separating set pairs for (S, Φ) and, as a result, the Φ graphs of all representations in the (S, Φ) Game are connected. □

Theorem 8 *The (Interval, Φ) Game and the connected (Interval, Φ) Game, where Φ is Intersection, Overlapping, Containment, or Disjointness, are PSPACE-complete.*

Proof. The games are PSPACE-hard by Example 5, Lemma 4, and Theorems 6 and 7. To complete the proof, we show that the games are in PSPACE. Since we consider only polynomial sized representations, each position of the game is representable in space polynomial in $|V|$. At each move, we only need to consider placing the new interval endpoints at an already-placed endpoint or between a pair of consecutive already-placed interval endpoints, that is, only a polynomial number of moves need to be considered from each position. Finally, the game ends after at most $|V|$ moves. Therefore, the games are in PSPACE since all move sequences can be examined recursively, one sequence at a time. The stack containing all of the information about one sequence of moves requires only polynomial space. □

Theorem 9 *The (Unit Interval, Φ) Game and the connected (Unit Interval, Φ) Game, where Φ is Intersection, Overlapping, Containment, or Disjointness, are PSPACE-complete.*

Proof. All of the games except the connected (Unit Interval, Containment Game) are PSPACE-hard by Example 5, Lemma 4, and Theorems 6 and 7.

The connected (Unit Interval, Containment) Game is PSPACE-hard by the following reasoning. As observed in Example 5, one unit interval is contained in another if and only if the two intervals are identical. Consequently, a connected graph has a (Unit Interval, Containment) representation if and only if it is a complete graph. To reduce Kayles to this game, we transform an arbitrary instance of Kayles, (G, U), to the instance $(\overline{G}, U, f)$ where f maps every vertex of U to the unit interval: $[1, 2]$. It is easy to see that the instance of Kayles is a winning position if and only if the constructed instance of the connected (Unit Interval, Containment) Game is a winning position. The connected (Unit Interval, Containment) Game is the same as the game of constructing a complete subgraph of G while Kayles is the game of constructing an independent set of G. To complete the proof, the games are in PSPACE by an argument similar to the one in the proof of Theorem 8. □

Theorem 10 *The (Permutation, Φ) Game and the connected (Permutation, Φ) Game, where Φ is Intersection, Overlapping, Containment, or Disjointness, are PSPACE-complete.*

Proof. All of the games except the connected (Permutation, Containment) Game are PSPACE-hard by Example 5, Lemma 4, and Theorems 6 and 7. The connected (Permutation, Containment) Game is PSPACE-hard by an argument similar to that of Theorem 9. To complete the proof, the games are in PSPACE by an argument similar to the one in the proof of Theorem 8. □

5 Efficient algorithms for some connected representation games on trees

In this section we consider the question: when can we solve initial positions of representation games efficiently? We show that the connected (Unit Interval, Intersection) Game is solvable in linear time on trees, and that the initial positions $(G, \emptyset, \emptyset)$ of some restricted versions of the connected (S, Intersection) Games where $S \in$ {Interval, Permutation} and G is a tree can be solved in polynomial time.

Note *Since all of the games in this section are connected games with set operation intersection, we often drop those qualifiers from our notation and refer to the connected (S, Intersection) Game as the S Game and an (S, Intersection) representation as an S representation, where $S \in$ {Unit Interval, Interval, Permutation}.*

Not all trees have unit interval, interval, or permutation representations. In particular, a tree is a unit interval graph if and only if it is a path, and a tree is an interval graph if and only if it is a permutation graph if and only if it is a caterpillar (see [8]). Therefore, when the input graph is a tree, the subgraphs represented at each stage are paths in the Unit Interval Game, and caterpillars in the Interval and Permutation Games.

We will refer to the following special elements of interval and permutation representations. The *leftmost interval* (respectively *rightmost interval*) of an interval representation is one whose left (respectively right) endpoint is less than (respectively greater than) all other endpoints of intervals in the representation. Recall that a permutation representation is composed of straight line segments connecting two parallel lines. Assume that the two parallel lines are L_1 and L_2, and imagine that they are horizontal in the plane. The *leftmost line* (respectively *rightmost line*) on $L \in \{L_1, L_2\}$ is the line in the representation whose endpoint on L is less than (respectively greater than) all other lines' points on L. Leftmost and rightmost intervals and lines are said to be *extreme*.

Two main ingredients of our algorithms are now discussed. First, we use the idea of a fixed starting vertex whose element in the representation restricts the placement of future elements, namely a *mandatory head*. This leads to games in which interval and permutation representations are allowed to grow in only one direction. The *Interval Game with mandatory head u* is the Interval Game where u is the first vertex played and u's interval remains extreme throughout the play. In a *Permutation Game with mandatory head u*, u is played first and u's line remains extreme throughout the whole game.

To solve the Interval Game, we rely on the second ingredient: the Sprague-Grundy theory of impartial combinatorial games (see [2], [3], and [9]), which applies to two-person games that are impartial (the same moves are available to both players), perfect information (both players know all information about available moves), deterministic (there are no random elements in the game, eg. dice), finite (at each move there are only a finite number of moves available and the game ends after a finite number of moves), and in which the last player to move wins. It is clear that our representation games satisfy all of these properties except possibly finiteness. The games that we discuss in this section satisfy finiteness as a result of the assumption that our representations are polynomially sized, and the strategy of picking endpoints of intervals and lines to be existing endpoints, midpoints between existing endpoints, or existing endpoints plus or minus a constant in $\{1, 2\}$, as discussed in Section 2.2. The games that we consider from now on satisfy all of the properties and we will not explicitly repeat that fact in what follows. This allows us to make use of the following definitions and theorems of the Sprague-Grundy theory, which are described in [2], [3] and [9]. The *minimum excluded element* of a set S of natural numbers, $\text{mex}(S)$, is the smallest natural number that is not in S.

Theorem 11 (see [2], [3] and [9]) *Every game position P has a unique natural number associated with it called its* nim-value *(or* Grundy-value*), denoted $\mathcal{G}(P)$, and*

$$\mathcal{G}(P) = \begin{cases} 0 & \text{if } P \text{ is an end position} \\ \text{mex}(\{\mathcal{G}(P') \mid P' \text{ can be reached from } P \text{ in one move}\}) & \text{otherwise} \end{cases}$$

Theorem 12 (see [2], [3], and [9])
Position P is a winning position if and only if $\mathcal{G}(P) > 0$.

The *nim-sum* of two natural numbers a and b, $a \oplus b$, is the natural number represented by the exclusive-or of the binary representations of a and b. Since $\oplus$ is commutative and associative, the result of applying the operation to more than two operands does not depend on the order of the operands or on the order in which the operations are applied. The *sum of two games* P and Q is the game $P + Q$ in which two players take turns choosing one of the games and making a move in it. The last player to move wins.

Theorem 13 (see [2], [3], and [9])
If P and Q are two game positions, then $\mathcal{G}(P + Q) = \mathcal{G}(P) \oplus \mathcal{G}(Q)$.

We consider the Unit Interval, Interval, and Permutation Games on an arbitrary tree T, starting from the initial position of $(T, \emptyset, \emptyset)$. Obviously, the first player wins if T has just one vertex and the second player wins if T has no vertices. Otherwise, if players are allowed to choose an interval or line that already appears in the representation, then the second player always wins since he can choose any neighbour of the first chosen vertex and the same interval or line segment as the one chosen by the first player, thereby ending the game.

Note *From now on we consider the more interesting games in which the endpoints of intervals and line segments in representations are required to be distinct.*

In the next subsection, we discuss the Unit Interval Game, and in the following two subsections, we solve a mandatory head game and a version of the game without mandatory head but in which a certain type of move is forbidden.

Definition 14 *Let uv be an edge of tree T. Then T_v^u denotes the connected component of $T - \{uv\}$ that contains v.*

5.1 Connected (Unit Interval, Intersection) Game on trees

The only trees that have unit interval representations are paths. Furthermore, the order of the interval endpoints in a unit interval representation of a given path is unique up to reversal. (Recall that interval endpoints are distinct.) Consequently, given a unit interval representation of a subpath of a tree, a vertex of the tree that is not already on the path can be added to the path and a corresponding interval can be added to the representation if and only if the vertex is adjacent to an endpoint of the represented path. Therefore, the representation is irrelevant to the outcome of the game and the following games are equivalent:

- the connected (Unit Interval, Intersection) Game on trees
- the $\mathcal{H}$ Subgraph Game on trees where $\mathcal{H}$ is the set of chordless paths
- Generalized Geography on undirected graphs that are trees without specified starting vertex, generalized to allow a vertex to be added to either end of the path

There is a winning strategy for the first player if and only if there is a first move such that for all possible second moves, the resulting position is a winning position. To solve the game, we consider all possible choices for the first two moves. After the adjacent vertices u and v are played in the first two moves, it can be determined whether the result is a winning position by computing the nim-value of the sum of two Generalized Geography instances on undirected graphs: T_u^v where u has been played, and T_v^u where v has been played. For each edge uv, this can be computed in linear time [4]. Therefore, it can be determined whether the first player has a winning strategy on input tree $T = (V, E)$ in $O(|V|^2)$ time.

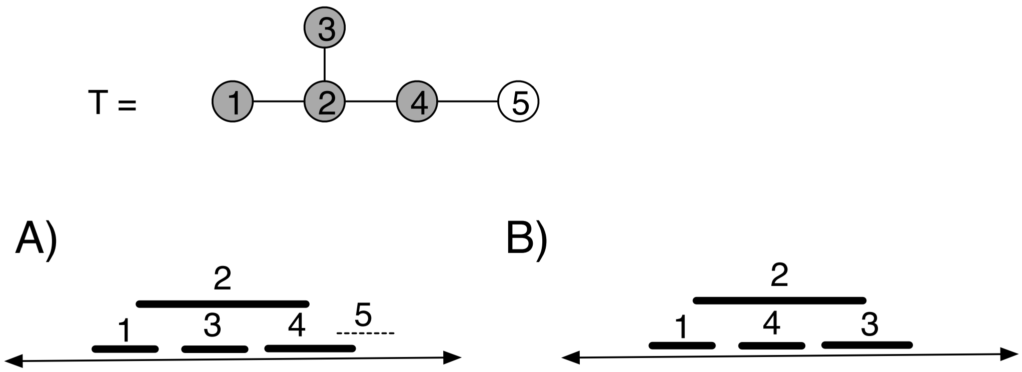

Figure 3: An example of a Caterpillar Subgraph Game on the top with chosen vertices shown in grey, and two different Interval Game positions on the bottom. Both of the positions on the bottom have the same vertices represented as are chosen on the top. However, while vertex 5 is a valid move on the top and in **A** on the bottom (as shown in thin dotted line), there is no way of representing 5 in **B**.

5.2 Connected (Interval, Intersection) Game on trees: two restricted versions

We have seen that the Unit Interval Game is nothing but the $\mathcal{H}$ Subgraph Game where $\mathcal{H}$ is the class of connected unit interval trees, i.e. paths. Are the Interval and Permutation Games the same as the Caterpillar Subgraph Game? The answer is no. For instance, if $T = (\{1, 2, 3, 4, 5\}, \{12, 23, 24, 45\})$ and the played vertices are $\{1, 2, 3, 4\}$, vertex 5 can be played in the Caterpillar Subgraph Game but whether or not it can be played in the Interval and Permutation Games depends on the representation (example: Figure 3).

Let $T = (V, E)$ be a tree, let T' be a caterpillar subtree of T, and consider an arbitrary interval representation of T'. The *leftmost* and *rightmost* vertices of T' (with respect to the representation) are the vertices represented by the leftmost and rightmost intervals of the representation, respectively. The unique path in T' between (and including) the leftmost and rightmost vertices is called the *placed-spine*. For example, in Figure 3, the placed spine in the representation of part A is $T[\{1, 2, 4\}]$ and the placed spine in the representation of part B is $T[\{1, 2, 3\}]$.

Lemma 15 *Let $T = (V, E)$ be a tree, let $T' = (V', E')$ be a caterpillar subtree of T, and consider an arbitrary interval representation of T'. For all $v \in V \backslash V'$, $T[V' \cup \{v\}]$ is connected and an interval can be added to the interval representation for T' to produce an interval representation for $T[V' \cup \{v\}]$ if and only if v is adjacent to a vertex of the placed-spine of the representation of T'.*

Proof. By the definition of the placed-spine and since T' is a caterpillar, every vertex of T' that is not on the placed-spine is adjacent to a vertex of the placed-spine, is a leaf of T', and is represented by an interval that is contained in a placed-spine interval. Every placed-spine interval contains a region that is not contained in any other interval of the representation; otherwise, it would have to be contained in the union of the intervals of its neighbours, contradicting that T' is a tree or that the endpoints of the placed-spine path are leftmost and rightmost.

If $v \in V \backslash V'$ is adjacent to a vertex of T' then it is adjacent to exactly one vertex of T'. Therefore, if an interval can be added to produce a representation for $T[V' \cup \{v\}]$, it must intersect an interval that contains a region that no other interval of T' contains, that is, an interval of the placed-spine.

Conversely, any $v \in V \backslash V'$ that is adjacent to a placed-spine vertex is adjacent to exactly one placed-spine vertex and an interval representing v can be added to the region of the corresponding placed-spine interval that is not contained in any other interval. $\square$

We now consider the Interval Game on a given tree T with mandatory head u. That is, vertex u is played first and remains leftmost (or rightmost) throughout the game. In this case, we assume without loss of generality that u remains leftmost and we call the rightmost interval and vertex the *spine-so-far* interval and vertex.

By Lemma 15, at any point in the Interval Game on tree T with mandatory head u, the next player must play an unplayed vertex v that is adjacent to a placed-spine vertex. This involves choosing such a vertex and adding an interval to the representation. The new interval cannot contain an already placed interval, as this would violate the mandatory head condition or the fact that T is a tree. In addition, the new interval must intersect exactly one already placed interval because of the connectedness requirement and the fact that T is a tree. Thus, the new interval must be contained in or overlap exactly one already played interval, and because of the mandatory head, the only interval it could overlap is the spine-so-far interval. Therefore each move (after u is played in the first move) is one of the following:

- a *leaf* move: v is adjacent to a placed-spine vertex other than the spine-so-far vertex and is represented by an interval that is contained in the corresponding placed-spine interval.
- a *foldunder* move: v is adjacent to the spine-so-far vertex and is represented by an interval that is contained in the spine-so-far interval.
- an *extension* move: v is adjacent to the spine-so-far vertex and is represented by an interval that overlaps the spine-so-far interval and becomes the new spine-so-far interval.

Mandatory head and no foldunders

We first consider the mandatory head game in which foldunders are not allowed. The second move must be an extension move, and each subsequent move either adds a vertex that is not adjacent to the spine-so-far and an interval that is contained in its neighbour's interval, or is an extension move that changes the spine-so-far vertex. Thus, the first player who plays a neighbour of the spine-so-far vertex decides which of its neighbours will be added to the placed-spine, and play continues until the spine-so-far vertex is a leaf of T.

The only aspect of the representation that is important in this game is the identity of the spine-so-far vertex. Let (T, u) denote the game on tree T with mandatory head u where the set of played vertices is exactly $\{u\}$. Clearly, u is the spine-so-far vertex in any interval representation of the graph $(\{u\}, \emptyset)$.

Theorem 16 *In the connected (Interval, Intersection) Game with no foldunders on tree T with mandatory head u:*

$$\mathcal{G}(T,u) = \begin{cases} 0 & \text{if } |N_T(u)| = 0 \\ \operatorname{mex}(\{\mathcal{G}(T_v^u, v) \oplus (1 - \operatorname{parity}(|N_T(u)|)) \mid v \in N_T(u)\}) & \text{otherwise} \end{cases}$$

Proof. The next player must select a neighbour v of the mandatory head and represent it with an interval that overlaps the mandatory head's interval (an *extension* move). After v is played, the game becomes the sum of independent games: (T_v^u, v), and $|N_T(u)| - 1$ games each containing one unplayed vertex and having nim-value 1. The latter games correspond to the neighbours of u in T other than v, which must all be played and represented by intervals contained in u's interval. No other vertices can ever be played. Therefore, by Theorems 11 and 13,

$$\mathcal{G}(T,u) = \begin{cases} 0 & \text{if } |N_T(u)| = 0 \\ \operatorname{mex}(\{\mathcal{G}(T_v^u, v) \oplus_{w \in N_T(u) \setminus \{v\}} 1 \mid v \in N_T(u)\}) & \text{otherwise} \end{cases}$$

Since $\mathcal{G}(T_v^u, v) \oplus 1 \oplus 1 \ldots \oplus 1$, where there are $(|N_T(u)| - 1)$ "$\oplus$ 1" terms, is equal to $\mathcal{G}(T_v^u, v) \oplus (1 - \operatorname{parity}(|N_T(u)|))$, the result follows. □

The previous theorem shows that the nim-value of (T, u) can be computed in linear time.

No foldunders

The method for solving the game with no foldunders with mandatory head can be extended to solve the game with no foldunders without mandatory head. In this case we have to consider the possibility that the second interval contains the first interval. We call this a *foldover* move. Note that it can only happen on the second move; after that, a new interval that contains another would contradict that T is a tree. Now, $T = (V, E)$ is a winning instance of the Interval Game with no foldunders if and only if there exists a vertex $u \in V$ (a winning move for the first player) such that for every $v \in N_T(u)$ (i.e. no matter what vertex is played by the second player), both of the following hold:

- $\mathcal{G}(T_u^v, u) \oplus \mathcal{G}(T_v^u, v) > 0$ (if the second player makes an extension move, the resulting position is a winning position for the first player).
- there exists a vertex $w \in N_T(v) \backslash \{u\}$ such that $\mathcal{G}((T_v^u)_v^w, v) \oplus \mathcal{G}(T_w^v, w) = 0$ (if the second player makes a foldover move, the first player can move such that the resulting position is a losing position for the second player).

For each edge uv, $\mathcal{G}(T_u^v, u)$ and $\mathcal{G}(T_v^u, v)$ can be computed in linear time and for each pair of edges uv and vw, $\mathcal{G}((T_v^u)_v^w, v) \oplus \mathcal{G}(T_w^v, w)$ can be computed in linear time; therefore the overall running time of the algorithm on tree $T = (V, E)$ is $O(|V|^3)$.

Mandatory head

In the next theorem, we examine the game with mandatory head when foldunders are allowed.

Theorem 17 *In the connected (Interval, Intersection) Game on tree T with mandatory head u:*

$$\mathcal{G}(T, u) = \begin{cases} 0 & \text{if } |N_T(u)| = 0 \\ \text{mex}\left(\begin{array}{c} \{\mathcal{G}(T_v^u, v) \oplus (1 - \text{parity}(|N_T(u)|)) \mid v \in N_T(u)\} \\ \cup \ \{\mathcal{G}(T_u^v, u) \mid v \in N_T(u)\} \end{array} \right) & \text{otherwise} \end{cases}$$

Proof. The next player must select a neighbour of the mandatory head and represent it either with an interval that overlaps the mandatory head's interval (an *extension* move) or an interval that is contained in the mandatory head's interval (a *foldunder* move). If vertex v is played in a foldunder move, no more vertices of the subtree T_v^u can ever be played and therefore that subtree can be removed from consideration. (This scenario is expressed by the second set in the mex expression above.) After v is played in an extension move, the game becomes the sum of independent games: (T_v^u, v), and $|N_T(u)| - 1$ games each containing one unplayed vertex and having nim-value 1. The latter games correspond to the neighbours of u in T other than v, all of which must be played in leaf moves. No other vertices of the corresponding subtrees can ever be played. Therefore, by Theorems 11 and 13,

$$\mathcal{G}(T, u) = \begin{cases} 0 & \text{if } |N_T(u)| = 0 \\ \text{mex}\left(\begin{array}{c} \{\mathcal{G}(T_v^u, v) \oplus_{w \in N_T(u) \backslash \{v\}} 1\} \\ \cup \ \{\mathcal{G}(T_u^v, u) \mid v \in N_T(u)\} \end{array} \right) & \text{otherwise} \end{cases}$$

Since $\mathcal{G}(T_v^u, v) \oplus 1 \oplus 1 \ldots \oplus 1$, where there are $|N_T(u)|$ terms and therefore $(|N_T(u)| - 1)$ "$\oplus$ 1" terms, is equal to $\mathcal{G}(T_v^u, v) \oplus (1 - \text{parity}(|N_T(u)|))$, the result follows. □

The previous theorem does not immediately suggest an efficient method for computing the nim-value of (T, u) but, as the next theorem shows, we can determine whether (T, u) is a winning position by looking only at the values of $\mathcal{G}(T_v^u, v)$ for all $v \in N_T(u)$.

Theorem 18 *In the connected (Interval, Intersection) Game on tree T with mandatory head u, where $p \in \{0, 1\}$ is the parity of $|N_T(u)|$, $n_0 = |\{v \in N_T(u) \mid \mathcal{G}(T_v^u, v) = 0\}|$, $n_1 = |\{v \in N_T(u) \mid \mathcal{G}(T_v^u, v) = 1\}|$,* *and* $n_2 = |\{v \in N_T(u) \mid \mathcal{G}(T_v^u, v) \geq 2\}|$*:*

$$\mathcal{G}(T,u) \text{ is } \begin{cases} = 0 & \text{if } |N_T(u)| = 0 & (1) \\ \geq 2 & \text{if } n_0 > 0 \text{ and } n_1 > 0 & (2) \\ = p & \text{if } n_1 = 0 \text{ and } n_0 + n_2 > 0 & (3) \\ \geq 2 & \text{if } n_0 = 0 \text{ and } n_1 = 1 & (4) \\ = 1-p & \text{if } n_0 = 0 \text{ and } n_1 > 1 & (5) \end{cases}$$

Proof. (1) is true by Theorem 11.

For (2), by Theorem 17, $\mathcal{G}(T,u) = \text{mex}(S)$ where $\{0 \oplus (1-p), 1 \oplus (1-p)\} = \{0,1\} \subseteq S$. Therefore, $\mathcal{G}(T,u) \geq 2$.

For (3), if $|N_T(u)| = 1$ then by Theorem 17, $\mathcal{G}(T,u)$ is equal to $\text{mex}(\{0\} \cup \{0\})$ or to $\text{mex}(\{2\} \cup \{0\})$. In either case, $\mathcal{G}(T,u) = 1$. We prove the statement for the case where $|N_T(u)| > 1$ by induction. Assume that $\mathcal{G}(R,z) = p'$ for all positions (R,z) such that $|N_R(z)| < |N_T(u)|$, $n'_1 = 0$, and $n'_0 + n'_2 > 0$, where p' is the parity of $|N_R(z)|$, $n'_0 = |\{v \in N_R(z) \mid \mathcal{G}(R_v^z, v) = 0\}|$, $n'_1 = |\{v \in N_R(z) \mid \mathcal{G}(R_v^z, v) = 1\}|$, and $n'_2 = |\{v \in N_R(z) \mid \mathcal{G}(R_v^z, v) \geq 2\}|$. By Theorem 17, $\mathcal{G}(T,u) = \text{mex}(A \cup B)$ where $A = \{\mathcal{G}(T_v^u, v) \oplus (1-p) \mid v \in N_T(u)\}$ and $B = \{\mathcal{G}(T_u^v, u) \mid v \in N_T(u)\}$. Recall that $p \in \{0,1\}$ and note that $0 \oplus (1-p) = 1-p$, $1 \oplus (1-p) = p$, and for any natural number $x \geq 2$, $x \oplus 0 \geq 2$ and $x \oplus 1 \geq 2$. Therefore, since $n_1 = 0$, $p \notin A$. By the inductive hypothesis, $B = \{\text{parity}(|N_{T_u^v}(u)|)\} = \{1-p\}$. Consequently, $\mathcal{G}(T,u) = p$.

For (4) and (5), we again use the fact that by Theorem 17, $\mathcal{G}(T,u) = \text{mex}(A \cup B)$ where $A = \{\mathcal{G}(T_v^u, v) \oplus (1-p) \mid v \in N_T(u)\}$ and $B = \{\mathcal{G}(T_u^v, u) \mid v \in N_T(u)\}$. Since $n_0 = 0$ and $n_1 \geq 1$, $(1-p) \notin A$ and $p \in A$. If $n_1 = 1$ then $(1-p) \in B$ by (3) and therefore $\mathcal{G}(T,u) \geq 2$ as required for (4).

For (5), we need to show that $(1-p) \notin B$. We prove this by induction on $|N_T(u)|$. If $|N_T(u)| = 2$ then B contains only nim-values greater than or equal to 2, by (4). If $|N_T(u)| > 2$, assume that $\mathcal{G}(R,z) = 1-p'$ for all positions (R,z) such that $|N_R(z)| < |N_T(u)|$, $n'_0 = 0$, and $n'_1 > 1$, where p' is the parity of $|N_R(z)|$, $n'_0 = |\{v \in N_R(z) \mid \mathcal{G}(R_v^z, v) = 0\}|$, $n'_1 = |\{v \in N_R(z) \mid \mathcal{G}(R_v^z, v) = 1\}|$, and $n'_2 = |\{v \in N_R(z) \mid \mathcal{G}(R_v^z, v) \geq 2\}|$. Then, for any $v \in N_T(u)$, $\mathcal{G}(T_u^v, u) = p$ by the inductive hypothesis or $\mathcal{G}(T_u^v, u) \geq 2$ by (4), and therefore $(1-p) \notin B$ and the result follows. □

Using the preceding theorem, we can solve the mandatory head game where all types of moves are allowed in linear time. But this does not necessarily lead to a polynomial time algorithm for the general problem without mandatory head since our strategy only determines whether the nim-sum of a given position is 0, 1, or greater than or equal to 2, and solving the problem without mandatory head seems to require exact nim-values for mandatory head games. However, for some trees it is easy to determine the winner, as the following observations show.

Observation 19 *In the connected (Interval, Intersection) Game on a tree, a leaf is never a winning first move. The second player can win by making a foldunder move.*

Observation 20 *In the connected (Interval, Intersection) Game on a tree, a vertex that is adjacent to a leaf is never a winning first move. The second player can win by playing a neighbour leaf in a foldover move.*

Combining Observations 19 and 20, we have:

Observation 21 *Trees in which every vertex is either a leaf or adjacent to a leaf are always second player wins in the connected (Interval, Intersection) Game.*

5.3 Connected (Permutation, Intersection) Game on trees: two restricted versions

In the Interval and Permutation Games, two players choose a caterpillar subgraph of a tree within constraints imposed by a representation. We have seen that the games are different from the Caterpillar Subgraph Game. The example in Figure 3 shows that the games are different from each other too. The tree in that example

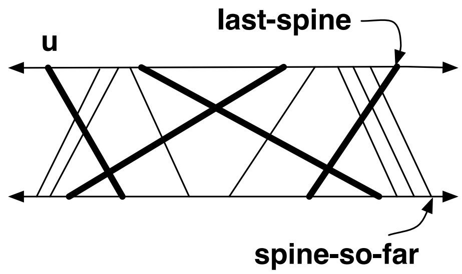

Figure 4: Illustrating the definitions of spine-so-far, last-spine, and placed-spine in a permutation representation. The placed-spine is shown in bold.

is a first player win in the Permutation Game if vertex 5 is played in the first move, but not in the Interval Game by Observation 21. It is now tempting to ask how similar the Interval and Permutation Games really are. As it turns out, our approaches to solving them are quite different.

Let T be a tree and let T' be a caterpillar subgraph of T. For a vertex u of T', a *permutation representation of T' with mandatory head u* is a permutation representation of T' in which u's line is extreme. We normally assume without loss of generality that the mandatory head's line is leftmost, and make the following definitions, all of which have analogous forms when the mandatory head is rightmost. Note that these definitions are slightly different from the interval definitions, as a result of the types of moves that are possible in the two contexts. The *spine-so-far* is the vertex represented by the rightmost line that crosses exactly one other line and is not u's line. The spine-so-far exists and is unique provided T' has at least two vertices. The *last-spine* is the neighbour of the spine-so-far vertex on the unique path in T from the mandatory head to the spine-so-far vertex. The *placed-spine* is the unique path between (and including) the mandatory head and the last-spine vertex. An example illustrating these definitions is given in Figure 4. We will sometimes blur the distinction between a line in the representation and the vertex that it represents, and use these notations to refer to both lines and vertices.

Lemma 22 *Let $T = (V, E)$ be a tree, let $T' = (V', E')$ be a caterpillar subtree of T, let u be a vertex of T', and consider an arbitrary permutation representation of T' with mandatory head u. Then for all $v \in V \backslash V'$, $T[V' \cup \{v\}]$ is connected and a line can be added to the representation to produce a permutation representation for $T[V' \cup \{v\}]$ with mandatory head u if and only if v is adjacent to a vertex of the placed-spine or to the spine-so-far vertex.*

Proof. It is easy to see that each vertex of T' that is not on the placed-spine is adjacent to exactly one placed-spine vertex and is a leaf of T'. In addition, since T is a tree, if $v \in V \backslash V'$ is adjacent to a vertex of T' then it is adjacent to exactly one vertex of T'. The line endpoint orderings in a permutation representation of a path are unique up to reversal and exchanging L_1 and L_2. Therefore, the line endpoint orderings in a permutation representation of a caterpillar are unique up to reversal and exchanging L_1 and L_2 and permuting degree one vertices that have the same neighbour. Taking this into account, it can be seen that a new line representing v that crosses exactly one line ℓ, can be added to a permutation representation of T' with mandatory head u, if and only if ℓ is on the placed-spine or ℓ is the spine-so-far vertex. $\square$

The Permutation Game on tree T with mandatory head u is the Permutation Game in which u is the first vertex played and, throughout the game, the representation under construction has mandatory head u. By Lemma 22, at any point in the Permutation Game on tree T with mandatory head u and spine-so-far vertex s, the next player must play an unplayed vertex v in one of the following types of moves:

- a *leaf* move: v is adjacent to a placed-spine vertex other than the spine-so-far vertex and is represented by a line that has both endpoints less than the endpoints of the spine-so-far vertex.

- an *overtake* move: v is adjacent to the last-spine, and is represented by a line that has both endpoints greater than the endpoints of the spine-so-far vertex, making the new vertex the new spine-so-far.
- an *extension* move: v is adjacent to the spine-so-far vertex and becomes the new spine-so-far.

Mandatory head and no overtakes

In the connected Permutation Game played with a mandatory head, if overtake moves are forbidden, then after the first point when a vertex v becomes the spine-so-far vertex, the next neighbour of v that is played becomes the new spine-so-far vertex and remains in the placed-spine throughout the rest of the game. Then, by Lemma 22, each unplayed neighbour w of v will eventually be played but no other vertex of the corresponding subtrees can ever be played. The game is identical to the Interval Game with mandatory head and no foldunders and can be solved by the method of the previous subsection.

No overtakes

As in the previous subsection, the method can be extended to solve the game with no overtakes without mandatory head.

Mandatory head

The remainder of this subsection is concerned with the Permutation Game in which all types of moves are allowed, played with a mandatory head. For vertices u and v of a tree T, we define $Cat_T(u,v)$ to be the subgraph of T induced by the u,v-path in T and all vertices of T that are adjacent to that path. Note that $Cat_T(u,v)$ is a caterpillar. As usual, we omit the subscript T when the context is clear. In light of the types of moves, we see that in the Permutation Game, play proceeds until the spine-so-far is a leaf of T. While the Interval Game does not share this property because of foldunder moves, the two games do share the following property: In every end position, all neighbours of the u,v-path are played, where v is the spine-so-far vertex.

Lemma 23 *Given the spine-so-far vertex v of an end position of the connected (S, Intersection) Game on tree T with mandatory head u, where $S \in \{Permutation, Interval\}$, the first player is the winner if and only if $|Cat(u,v)|$ is odd.*

Proof. By Lemmas 15 and 22, at the end of the game all vertices adjacent to the placed-spine will be represented. The first player has made the last move if and only if the number of played vertices is odd. □

In the Interval Game with foldunders, when a vertex is played in an extension move, it is guaranteed to be on the placed-spine for the remainder of the game. In the Permutation Game with overtakes, that is not the case. To handle the difference, we diverge from our previous point of view by solving positions of the Permutation Game with respect to the first or second player, not the next or previous player. This perspective simplifies matters because the next/previous player status of a position corresponding to a particular spine-so-far vertex depends on the number of neighbours of the placed-spine that have been played, whereas the first/second player status depends only on the number of vertices that will ultimately be played. Another difference in our approach is that, here, we examine only game positions that have a special property, not all positions of the game. In what follows, Player 1 and Player 2 refer to the first and second player to make a move in the game, respectively.

Definition 24 *Let $T = (V,E)$ be a tree with mandatory head u and, for each $v \in V$ let $N'(v)$ be the neighbours of v in T, each of whose distance from u is greater than the distance from u to v. For all $v \in V$, we define*

$$win(v) = \begin{cases} 1 & \text{if } |\{x \in N'(v) \mid win(x) = 1\}| > |\{x \in N'(v) \mid win(x) = 2\}| \\ & \text{or } (|\{x \in N'(v) \mid win(x) = 1\}| = |\{x \in N'(v) \mid win(x) = 2\}| \\ & \qquad \text{and } |Cat(u,v)| \text{ is odd}) \\ 2 & \text{otherwise} \end{cases}$$

We now show that certain positions containing v as the last-spine vertex are winning positions for Player i if $win(v) = i$ where $i \in \{1, 2\}$. The idea behind a winning strategy for Player 1 is to force Player 2 to make the first move of a vertex in $N'(v)$ when $|\{x \in N'(v) \mid win(x) = 1\}| = |\{x \in N'(v) \mid win(x) = 2\}|$. This leaves Player 1 enough vertices in $\{x \in N'(v) \mid win(x) = 1\}$ to overtake with such an element whenever Player 2 makes an extension or overtake move with an element of $\{x \in N'(v) \mid win(x) = 2\}$. As we will see, the following technical lemma implies a more general result.

Lemma 25 *Let P be a position of the connected (Permutation, Intersection) Game on a tree $T = (V, E)$ with mandatory head u in which v is the last-spine vertex, w is the spine-so-far vertex, and one of the following holds:*

- *$v = u$ and v and w are the only two played vertices*
- *$v \neq u$ and exactly two neighbours of v are played, namely, v's neighbour on the u, v-path, and w*

Let $\{i, j\} = \{1, 2\}$. Then P is a winning position for Player i if both of the following are true:

- *$win(v) = i$*
- *w was played by Player j if $|\{x \in N'(v) \mid win(x) = i\}| = |\{x \in N'(v) \mid win(x) = j\}|$*

Proof. Let T_u refer to the tree T rooted at u. The proof is by induction on $height(v)$, the height of v in T_u. For each $x \in V$ let $N'(x)$ and $win(x)$ be as defined in Definition 24. By the existence of w, $height(v) > 0$. If $height(v) = 1$ then all of v's children in T_u are leaves. This implies that $Cat(u, v) = Cat(u, x)$ and $|\{y \in N'(x) \mid win(y) = 1\}| = |\{y \in N'(x) \mid win(y) = 2\}| = 0$ for all $x \in N'(v)$. Therefore v and all of its children have the same win value. If $win(v) = 1$ then $|Cat(u, x)|$ is odd for all $x \in N'(v)$ and P is a winning position for Player 1 by Lemma 23.

If $height(v) > 1$, assume the lemma is true for positions having last-spine vertices of height less than $height(v)$. Let $W = N'(v)$, $W_1 = \{x \in N'(v) \mid win(x) = 1\}$, and $W_2 = \{x \in N'(v) \mid win(x) = 2\}$. By Definition 24, $W = W_1 \cup W_2$ and $W_1 \cap W_2 = \emptyset$.

We give the proof for $i = 1$ and omit the symmetric argument for $i = 2$. Assume that the conditions of the lemma for P to be a winning position for Player 1, hold. Note that, in P, exactly one vertex of W is played, and since it was the first vertex of W to be played, it was played in an extension move. Furthermore, if $|W_1| = |W_2|$ then it was played by Player 2. The following strategy for Player 1 leads to a position satisfying the conditions of the lemma for a winning position for Player 1 in which the placed-spine vertex has height less than $height(v)$. By the inductive hypothesis, this implies that P is a winning position for Player 1.

On each turn, Player 1 takes one of the following actions until a position P' is reached in which v is no longer the last-spine vertex and exactly two neighbours of the last-spine are played, or the game ends.

- If the spine-so-far vertex is an element of W_2 then Player 1 plays an element of W_1 in an overtake move.
- Otherwise, Player 1 plays a vertex of W_2 in a leaf move if there are any unplayed vertices of W_2.
- Otherwise, Player 1 plays a vertex of W_1 in an overtake move if there are any unplayed vertices of W_1.
- Otherwise, Player 1 plays a vertex adjacent to the placed-spine in a leaf move, if there are any such unplayed vertices.
- Otherwise, Player 1 plays a child of the spine-so-far that has win value 1 in an extension move, producing position P'.

Since $win(v) = 1$ we have $|W_1| \geq |W_2|$. If $|W_1| = |W_2|$ then Player 2 produces position P by playing w and, on the next turn Player 1 plays an element of the opposite set (W_1 or W_2) and leaves a member of W_1 in the spine-so-far position. If $|W_1| > |W_2|$ then no matter who played w or whether w is in W_1 or W_2, the position after Player 1's next move has a member of W_1 in the spine-so-far position. In both cases, we have the following invariant:

INV: after each of Player 1's moves, the number of unplayed vertices of W_1 is greater than or equal to the number of unplayed vertices of W_2, and a vertex of W_1 is in the spine-so-far position.

Next, we show that Player 1 can always make a move in the above strategy. Suppose that none of the first four types of moves is possible. Let w' be the spine-so-far vertex. Note that the first type of move would be possible if $w' \in W_2$ by the arguments of the previous paragraph. Therefore $w' \in W_1$, all vertices of W are played, and all vertices adjacent to the placed-spine are played. Thus, the set of played vertices is equal to the vertices of $Cat(u, v)$ and, since it is Player 1's turn, $|Cat(u, v)|$ is even. Suppose the fifth type of move is not possible, that is, that w' is a leaf or all children of w' have win value 2. If w' is a leaf then $Cat(u, w') = Cat(u, v)$ and therefore $Cat(u, w')$ has even size contradicting $win(w') = 1$. If all children of w' have win value 2, this again contradicts $win(w') = 1$.

If Player 1 makes the move resulting in position P', it is because none of the first four types of moves is possible. Suppose that w' is the spine-so-far vertex before Player 1's move that results in position P' with last-spine w' and new spine-so-far z. As we saw in the previous paragraph, this means that $win(w') = 1$ and $|Cat(u, v)|$ is even. Now $Cat(u, w')$ consists of the vertices of $Cat(u, v)$ and the children of w'. So if $|\{x \in N'(w') \mid win(x) = 1\}| = |\{x \in N'(w') \mid win(x) = 2\}|$, then $|Cat(u, w')|$ would be even, contradicting that $win(w') = 1$. Therefore, P' has last-spine vertex w' such that:

- $w' \neq u$ and exactly two neighbours of w' are played: v and z
- $win(w') = 1$
- $|\{x \in N'(w') \mid win(x) = 1\}| \neq |\{x \in N'(w') \mid win(x) = 2\}|$
- $height(w') < height(v)$

Therefore, since the conditions of the lemma are satisfied, P' is a winning position for Player 1 and, by the inductive hypothesis, so is P.

Suppose Player 2 makes the move that results in P' by playing z adjacent to w' in an extension move. Then $win(w') = 1$ by INV, $height(w') < height(v)$ since w' is a child of v, and P' satisfies all of the other conditions of the lemma by observations similar to the previous case. Therefore P', and by induction P, are winning positions for Player 1.

If the game ends without reaching P', Player 2 loses, since we have seen that Player 1 can always make a move. Therefore we again have that position P is a winning position for Player 1. □

Theorem 26 *Tree T with mandatory head u is a winning instance for Player 1 if and only if $win(u) = 1$.*

Proof. Player 1 plays u and then Player 2 plays a vertex w of $N'(u)$. If $win(u) = 1$ then this is a winning position for Player 1 by Lemma 25. Since the choice of w does not affect the outcome of the game, the initial position is also a winning position for Player 1. Suppose that $win(u) = 2$. If $|\{x \in N'(u) \mid win(x) = 1\}| = |\{x \in N'(u) \mid win(x) = 2\}|$, then $|Cat(u, u)|$ would be odd, contradicting that $win(u) = 2$. Therefore by Lemma 25, the position in which only u and w have been played is a winning position for Player 2 and since the choice of w does not affect the outcome, so is the initial position. □

Theorem 26 leads directly to an algorithm for solving the Permutation Game on a tree $T = (V, E)$ with mandatory head u. The first step is to compute $|Cat(u, v)|$ for each $v \in V$. This can be done in $O(|V|)$ time using a simple breadth-first search strategy. Then, the win values can be computed in $O(|V|)$ time in a single leaves-to-root traversal of T_u.

5.4 Connected Caterpillar Subgraph Game on long stars

Recall that a long star is a tree that has at most one vertex of degree greater than two. If a long star has a vertex of degree greater than two, then that unique vertex is called the *central vertex*. Let T be a long star with central vertex v_c. Then each connected component of $T - \{v_c\}$ is a path, and is called a *twig* of T. A twig that has more than one vertex is a *long twig* and the parity of a twig is the parity of the number of vertices of the twig. The *leaf of a twig* of T is the vertex of the twig that has degree one in T.

The theorem of this section characterizes all long stars T such that the initial position $(T, \emptyset)$ of the connected Caterpillar Subgraph Game is a winning position for the first player.

Theorem 27 *Let $T = (V, E)$ be a long star. The initial position $(T, \emptyset)$ of the connected Caterpillar Subgraph Game is a winning instance for the first player if and only if one of the following holds:*

- *T is a path or has at most two long twigs, and $|V|$ is odd.*
- *T has three or more long twigs, all long twigs have the same parity, and $degree(v_c)$ is even.*
- *T has three or more long twigs, at least one of each parity, and $degree(v_c)$ is odd.*
- *T has at least two long odd twigs and at least one long even twig, and $degree(v_c)$ is even.*

Proof. The first player wins if and only if, at the end of the game, an odd number of vertices have been played.

If T is a path or has at most two long twigs, then all vertices of T will be played during the game and therefore the outcome is given by the parity of $|V|$.

Otherwise T has at least three long twigs and, at the end of the game, the vertices that have been played are those of $Cat_T(u, v)$ where u and v are the leaves of two long twigs. In this case, the first player wins if and only if the two long twigs that are played have the same parity and $degree(v_c)$ is even, or have different parity and $degree(v_c)$ is odd.

If all long twigs have the same parity then the first player wins if and only if $degree(v_c)$ is even.

Otherwise, there are long twigs of each parity. If $degree(v_c)$ is odd, or $degree(v_c)$ is even and T has two or more long odd twigs, then the first player has the following winning strategy. On the first move, he plays a leaf of a long odd twig. The second player is eventually forced to play v_c. The first player then plays a vertex of a long odd twig if $degree(v_c)$ is even, or a vertex of a long even twig if $degree(v_c)$ is odd. No matter what the second player does, the first player can extend that twig, which allows him to decide whether the two long twigs that are eventually played are of the same parity or of different parity.

In the final scenario, $degree(v_c)$ is even and T has exactly one long odd twig. The following second player strategy ensures that the long odd twig is played, which guarantees that the second player wins. The second player need only play some vertex of the long odd twig that is not a neighbour of v_c to win. If the first move is on the long odd twig, the second player plays some vertex on that long odd twig that is not a neighbour of v_c. If the first play is v_c, then the second player plays the neighbour of v_c on the long odd twig and then, on his next turn, plays another vertex on the long odd twig. If the first play is on a long even twig, then the first player can be forced to eventually play v_c. Then the second player plays the neighbour of v_c on the long odd twig and on his next turn, plays another vertex on the long odd twig. $\Box$

6 Conclusion

We have considered two-person games on graphs and their representations, where the two players take turns choosing a vertex of a graph and adding an element to a representation. The games differ in the type of representation that is constructed – both in the elements used and the relationships among elements – and in possible restrictions imposed on the represented subgraphs (for example, the represented subgraph may be required to be connected after each move). We provided a general reduction from Kayles that proves the PSPACE-hardness of many representation games. We also showed that the following games can be solved efficiently:

- connected (Unit Interval, Intersection) Game on trees
- connected (Interval, Intersection) Game on trees with no foldunders

- connected (Interval, Intersection) Game on trees with mandatory head
- connected (Permutation, Intersection) Game on trees with no overtakes
- connected (Permutation, Intersection) Game on trees with mandatory head
- connected Caterpillar Subgraph Game on long stars

Our algorithms for the Interval and Permutation Games with mandatory head where all types of moves are allowed are quite different from one another. In the Interval Game, we focused on game positions from the perspectives of the next and previous players while, in the Permutation Game, it was more convenient to look at the problem in terms of the first and second players. This may be a consequence of the fact that, due to the properties of the two different representations, play always continues until a leaf is reached in the Permutation Game while that is not the case in the Interval Game. Our algorithms for the mandatory head games do not seem to generalize to the games without mandatory heads.

As a result of the previous work on Kayles and Generalized Geography mentioned in Section 1 and the results in this paper, the complexity of the Interval and Permutation Games (without mandatory head) and the Caterpillar Subgraph Game, on trees, graphs of bounded asteroidal number, and other graph classes may be interesting problems for further study.

7 Acknowledgements

The authors gratefully acknowledge support from an NSERC Discovery grant, an iCORE ICT Graduate Student Scholarship and a University of Alberta Dissertation Fellowship.

References

[1] Oswin Aichholzer, David Bremner, Erik D. Demaine, Ferran Hurtado, Evangelos Kranakis, Hannes Krasser, Suneeta Ramaswami, Saurabh Sethia, and Jorge Urrutia, Games on triangulations, Theoretical Computer Science 343, 42-71, 2005.

[2] Michael H. Albert, Richard J. Nowakowski, and David Wolfe, Lessons in Play: An Introduction to the Combinatorial Theory of Games, A K Peters, 2007.

[3] Elwyn R. Berlekamp, John H. Conway, and Richard K. Guy, Winning Ways for your Mathematical Plays, Academic Press, 1982, Second Edition, A K Peters, 2001.

[4] Hans L. Bodlaender, Complexity of path-forming games, Theoretical Computer Science 110, 215-245, 1993.

[5] Hans L. Bodlaender and Dieter Kratsch, Kayles and nimbers, Journal of Algorithms 43, 106-119, 2002.

[6] Hans L. Bodlaender, Dieter Kratsch, and Sjoerd T. Timmer, Exact algorithms for Kayles, Theoretical Computer Science 562, 165-176, 2015.

[7] J. A. Bondy and U. S. R. Murty, Graph Theory with Applications, North-Holland, 1976.

[8] Andreas Brandstädt, Van Bang Le and Jeremy P. Spinrad, Graph Classes: A Survey, SIAM Monographs on Discrete Mathematics and Applications, SIAM, 1999.

[9] John H. Conway, On Numbers and Games, Academic Press, 1976, Second Edition, A K Peters, 2001.

[10] Rudolf Fleischer and Gerhard Trippen, Kayles on the way to the stars, In H. Jaap van den Herik, Yngvi Björnsson, and Nathan S. Netanyahu, editors, Computers and Games, Lecture Notes in Computer Science, Volume 3846, 232-245, 2006.

[11] Udi Manber, Introduction to Algorithms: A Creative Approach, Addison-Wesley, 1989.

[12] Thomas J. Schaefer, On the complexity of some two-person perfect-information games, Journal of Computer and System Sciences 16, 185-225, 1978.

[13] Jeremy P. Spinrad, Efficient Graph Representations, Fields Institute Monographs, American Mathematical Society, 2003.