Games of No Chance 6
MSRI Publications
Volume **71**, 2025

Keeping your distance is hard

KYLE BURKE, SILVIA HEUBACH,
MELISSA A. HUGGAN AND SVENJA HUNTEMANN

We study the computational complexity of distance games, a class of combinatorial games played on graphs. A move consists of placing a colored token on an unoccupied vertex subject to it not being at certain distances to already occupied vertices. The last player to move wins. Well-known examples of distance games are NODE-KAYLES, SNORT, and COL, whose complexities were shown to be PSPACE-hard. We show that many more distance games are also PSPACE-hard.

1. Introduction

We begin by introducing distance games and defining specific combinatorial games needed in the remainder of the paper, then give an introduction to computational complexity and explain our proof strategy, as well as a table summarizing our results. At the end of the section we will give an overview of the organization of the paper.

Distance games. Huntemann and Nowakowski introduced distance games [9]; they are part of a larger class of combinatorial games called *placement games* studied in [3] and [6]. Distance games are played by placing game pieces (tokens) on empty vertices of a graph (= game board) according to rules that forbid game pieces to be placed at certain distances to already occupied vertices. The *distance* between two game pieces is defined as the graph distance between the respective vertices they occupy. We make the rules of the game precise in the following definition.

Thanks to Richard Nowakowski and AARMS for making Games at Dal 2015 at Dalhousie University possible, where most of this work came from. Burke was affiliated with Plymouth State University and his visit at this conference was supported by the Offices of the Provost and the Dean of Arts and Sciences. Huggan and Huntemann's work was completed while affiliated with Dalhousie University and was supported in part by the Natural Sciences and Engineering Research Council of Canada and the Killam Trust.

MSC2020: primary 91A46; secondary 03D15.

Keywords: combinatorial game, distance game, computational complexity.

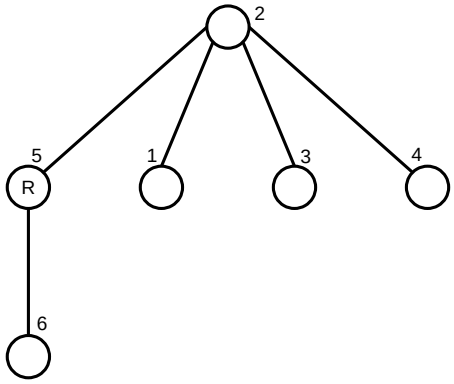

Figure 1. A position in a graph distance game. A red piece has been placed on vertex five, while all other vertices do not yet have a piece placed on them.

Definition 1.1. GRAPHDISTANCE(D, S) is the combinatorial game played on an arbitrary graph $G = (V, E)$, where S (stands for "same") and D (stands for "different") are subsets of $\{1, 2, \ldots, n\}$, where $n = |V|$. Each graph vertex may be empty or contain a blue or red piece. Two players, Blue and Red, take their turn by placing a single piece of their color on an empty vertex according to these rules:

(1) A blue piece and a red piece are not allowed to have distance $d \in D$.

(2) Two pieces of the same color are not allowed to have distance $s \in S$.

Pieces may neither be moved to another vertex nor be removed once placed. The game ends when one player can no longer place a piece of their color.

For ease of readability, we will sometimes refer to the action of placing a colored piece on an unoccupied vertex as "coloring the vertex" and we will call an unoccupied vertex "uncolored". Also note that, since we play on a finite graph of n vertices, the diameter of the graph is at most $n-1$, so S and D can be restricted to be finite without any loss of generality.

By definition, GRAPHDISTANCE(D, S) is a *partizan* combinatorial game, that is, the two players have different moves available. This results from the fact that their game pieces have different colors. For example, in the game GRAPHDISTANCE($\{1\}$, $\varnothing$) played from the position shown in Figure 1, Blue cannot play on vertices 2 or 6, while Red is allowed to play on all vertices.

On the other hand, if $D = S$, then the color of the game pieces becomes irrelevant, allowing both players to play on the same set of vertices (though with different colored pieces). Consider again the position shown in Figure 1.

If $D = S = \{1\}$, then Blue cannot play on vertices 2 or 6 because $D = \{1\}$, while Red cannot play on these vertices because $S = \{1\}$. Thus, when $D = S$, GRAPHDISTANCE(D, S) is equivalent to an *impartial* combinatorial game, one where both players can make the same moves and the pieces have only one color.

The games listed below are well-known combinatorial games that will play a key role in the derivation of the complexity of distance games. We define them here for the reader unfamiliar with these games and express them as graph distance games.

Definition 1.2. In the game of NODE-KAYLES, players alternate placing tokens of a common color on empty vertices, all of whose neighbors are also empty. The game ends when no tokens can be placed.

Thus the impartial game NODE-KAYLES corresponds to GRAPHDISTANCE-($\{1\}$, $\{1\}$).

Definition 1.3. The game BIGRAPH-NODE-KAYLES is a partizan version of NODE-KAYLES played on a bipartite graph with vertex partition $V = V_B \cup V_R$, with the additional restriction that Blue can only play on V_B and Red can only play on V_R.

Even though the game pieces in BIGRAPH-NODE-KAYLES have the same color, the fact that all vertices within V_B and V_R are at even distances, and those from different vertex sets are at odd distances, BIGRAPH-NODE-KAYLES corresponds to GRAPHDISTANCE(D, S), where S is the set of odd integers and $D = \{1\} \cup \{2, 4, 6, \dots\}$, if at least one vertex is colored. (If no vertices are colored initially, then there is no way to enforce that the first player plays on the appropriate side of the bipartite graph.)

Definition 1.4. In the partizan games of SNORT and COL, players place red and blue tokens on vertices of a graph on any empty vertex with the restriction that adjacent vertices cannot have tokens of different or the same color, respectively. The game ends when no tokens can be placed.

Thus SNORT is the game GRAPHDISTANCE($\{1\}$, $\varnothing$) and COL is the game GRAPHDISTANCE($\varnothing$, $\{1\}$). More information, such as simple winning strategies and other properties of NODE-KAYLES, SNORT, and COL can be found in [1] and [12].

In our discussion about the complexity of determining winnability, we need to take into account all the possible positions of the game, together with the rules for each player. We will refer to this larger structure as a *ruleset*. Since S and D completely determine the available moves, the triple (Q, S, D), where Q is the set of positions of the game, specifies the ruleset.

Computational complexity of games. Computational complexity can be applied to combinatorial games to measure how hard it is to determine whether the next player has a winning strategy. We will give a short introduction to the needed concepts in this section. Readers interested further in the application of computational complexity to combinatorial games should reference [5].

An algorithm is considered *efficient* if the running time can be expressed as a polynomial of the size of the input (i.e., the game to be analyzed). The computational class P is the set of all computational problems with these polynomial-time solutions. EXPTIME consists of all problems that can be solved in exponential time or less. Problems that *require* an exponential amount of time make up the class EXPTIME-hard.

For the ruleset of any combinatorial game, the problem to be answered is always: "For a given position, G, that has *some* or *no* colored vertices, does Blue, moving next, have a winning strategy?" Note that this does not mean we need to find the winning strategy, just determine whether one exists. For some rulesets, such as NIM [2], this problem is in P. Some other games, such as a version of CHESS generalized to an $n \times n$ board, are instead EXPTIME-hard [8], meaning the problems are *intractable*.

PSPACE is the set of decision problems that can be solved using a polynomial amount of storage with no restrictions on time. It is known that $\mathrm{P} \subseteq \mathrm{PSPACE} \subseteq \mathrm{EXPTIME}$ and $\mathrm{P} \subsetneq \mathrm{EXPTIME}$, but it is not known whether either of the first two inclusions are strict or not [10]. Many decision problems are PSPACE-*hard*, meaning they are at least as hard as the most difficult problems in PSPACE. A decision problem is PSPACE-*complete* if it is both PSPACE-hard and in PSPACE.

The input for the decision problem is the triple (Q, S, D) which specifies the ruleset. Since we play on a graph with n vertices, each position has at most n move options for the current player. Furthermore, since each move places a token on an unoccupied vertex, the game ends in at most n moves, so the game tree for a given position has height at most n. To analyze the decision tree, we only need to keep track of the options of any given position that is being analyzed. So at worst we need to store n positions (maximal length of the game) with at most n options each, so the storage space is of order n^2. Therefore, distance games on graphs are at worst in PSPACE. As a result, any PSPACE-hard distance game is also PSPACE-complete: it is among the hardest problems included in PSPACE.

A ruleset T' can be shown to be PSPACE-hard with the help of another ruleset, say T, already known to be PSPACE-hard. T' is PSPACE-hard if a function f exists where

- $f : \mathrm{positions}(T) \to \mathrm{positions}(T')$,
- f can be computed in polynomial time, and

- f preserves winnability (e.g., for $t \in \text{positions}(T)$, Blue has a winning move going next on $f(t)$ exactly when Blue has a winning move going next on t) [5].

Such a function f is called a *reduction* (from T to T'). Finding reductions from PSPACE-hard games to new games is common practice for showing the PSPACE-hardness of these new games. Sometimes these reductions have a stronger property: each move from any position $t \in T$ corresponds to exactly one move in $f(t)$, that is, the game trees have exactly the same shape. Due to this injective homomorphism, we can refer to these reductions as *play-for-play* reductions.

There are two different approaches to complexity of combinatorial games: one can consider only starting positions or consider all positions that occur during play. For some placement games, such as NODE-KAYLES and BIGRAPH-NODE-KAYLES [12], these two approaches are equivalent as every midgame position is equal to a starting position on a smaller board. If this is not the case, it is common practice to consider all positions. This includes for example HEX [11], SNORT [12], planar SNORT [4], NOGO [4], and planar COL [4]. One interesting exception is COL [7], where hardness has been shown for starting positions. We will be taking the more common approach of considering all positions, implying that our reductions can use partially colored graphs as boards, as long as the colored vertices satisfy the distance rules for the games. One property of distance games is that such graphs are positions that can occur during play from the empty board.

Reduction strategy. In what follows, we will describe each reduction as a transformation of the graph G, on which a game T is played, to a graph G', on which game T' is played, via the insertion of subgraphs called *gadgets*. All reductions to be used will be play-for-play, as we will enforce the following two properties in all of our constructions:

Vertex condition: No vertex added to G to form G' is playable. No vertex of the original graph G is deleted.

Edge condition: None of the additional edges will result in any restrictions on the play on any of the vertices $v \in V$ from the original graph G. That is, for any $v \in V$, a Blue/Red piece can be played at v under ruleset T on G exactly when it can be played on v using ruleset T' on G'.

We will use the fact that NODE-KAYLES, BIGRAPH-NODE-KAYLES, SNORT, and COL (all placement games played on graphs) are PSPACE-hard [4; 7; 12] to create reductions showing that the games GRAPHDISTANCE(D, S) are PSPACE-hard for many pairs D and S. Table 1 gives a summary of these results. In

D	S	reference	planar result
$\{1, 2, \ldots, r\}$	$\varnothing$	Proposition 4.1	Proposition 6.1
$\{1, 2, \ldots, r\}$	$\max(S) < r$	Corollary 4.2	Proposition 6.1
$\varnothing$	$\{1, 2, \ldots, r\}$	Proposition 4.3	Proposition 6.1
$\max(D) < r$	$\{1, 2, \ldots, r\}$	Proposition 4.3	Proposition 6.1
$\{1, 2, \ldots, r\}$	$S \subseteq D, \max(S) = r$	Proposition 5.1	?
$D \subseteq S, \max(D) = r$	$\{1, 2, \ldots, r\}$	Proposition 5.1	?

Table 1. Pairs (D, S) for which GRAPHDISTANCE(D, S) is PSPACE-hard ($r \geq 2$).

all the reduction proofs, the goal is to create a graph G' via a play-for-play reduction so that playing GRAPHDISTANCE(D, S) on G' is equivalent to playing the game from which we reduce on G. We let V' be the union of all inserted vertices, E' be the union of all inserted edges, E_1 the set of edges of G not replaced, and $G' = (V \cup V', E_1 \cup E')$ be the desired graph on which to play GRAPHDISTANCE(D, S). In our diagrams, we will use B (blue) and R (red) for the pieces of the respective players.

Outline. Even though the simplest case, namely $D = \{1, 2\}$, is covered by the more general cases to follow, we will use it to give an introduction to the construction of gadgets by reducing from NODE-KAYLES using very simple gadgets in Section 2. We develop a more complex gadget in Section 3 that will be used for more general sets S and D in Section 4 ($\max(S) \neq \max(D)$) and Section 5 ($\max(S) = \max(D)$). We apply results on PSPACE-hardness of planar COL and planar SNORT to our results in Section 6. We conclude the paper with open problems for future work.

2. $D = \{1, 2\}$ and either $S = \varnothing$ or $S = \{1\}$

Proposition 2.1. The games GRAPHDISTANCE$(\{1, 2\}, S)$ are PSPACE-hard for $S = \varnothing$ and $S = \{1\}$.

Proof. Since BIGRAPH-NODE-KAYLES is PSPACE-hard, we will construct a reduction from a bipartite graph $G = (V_B \cup V_R, E)$ on which BIGRAPH-NODE-KAYLES is played to a graph G' on which GRAPHDISTANCE$(\{1, 2\}, S)$ is played, where $S = \varnothing$ or $S = \{1\}$.

We first look at the reduction from G to G' when $S = \{1\}$, which is illustrated in Figure 2. Note that we do not show the edges connecting the sets V_B and V_R to better focus on the reduction, which preserves G as a subgraph.

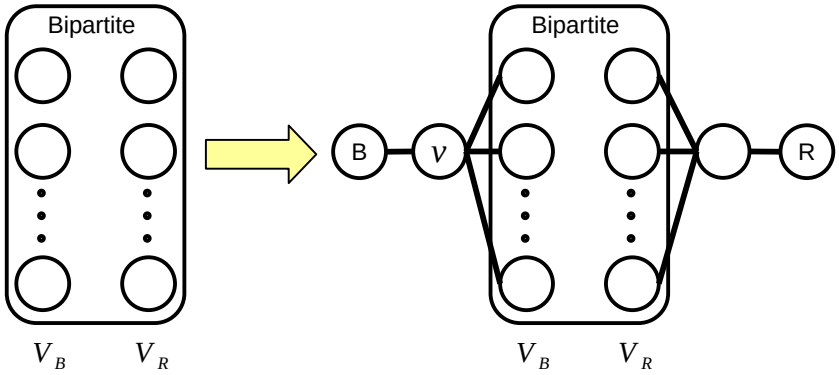

Figure 2. The reduction from BIGRAPH-NODE-KAYLES on the left to GRAPHDISTANCE($\{1, 2\}$, $\{1\}$) on the right.

In BIGRAPH-NODE-KAYLES, V_B is restricted to only be playable by Blue and V_R to only be playable by Red. Our goal is to create the graph G' via a play-for-play reduction so that playing GRAPHDISTANCE(D, S) on G' is equivalent to playing BIGRAPH-NODE-KAYLES on G. We describe the reduction as it relates to the vertices in V_B.

The first goal is to ensure that V_B cannot be played by Red, hence we connect all vertices from V_B to an uncolored vertex (labeled v) that is connected to an external vertex colored blue. Since this vertex is distance two from all vertices in V_B, no vertex in V_B can be colored red. Also, the vertex v is at distance one from B, so it can be colored neither red nor blue, so the vertex condition is satisfied. Next we check the edge condition. We need to be careful because vertex v now connects vertices in V_B with a path of length two. However, all vertices in V_B can only be colored in blue, and since $S = \{1\}$, these new paths do not impose restrictions on vertices in the graph G. Replicating the gadget on the right-hand side of the bipartite graph with B replaced by R completes the reduction for $S = \{1\}$.

We now consider the case $S = \varnothing$, which removes the constraint on labeling the vertex v blue, so we need to put additional vertices into the gadget to create this restriction. We replace v by a path of length two consisting of two uncolored vertices v_1 and v_2 and a terminal vertex colored red, as shown in Figure 3. The red vertex now keeps vertex v_2 from being colored blue.

In addition, the intermediate vertex v_1 is unplayable by both players as it is distance two from B and hence cannot be colored red, and is adjacent to R and so cannot be colored blue. Therefore the gadget consisting of these four vertices satisfies the vertex condition. Since the restrictions on coloring vertices

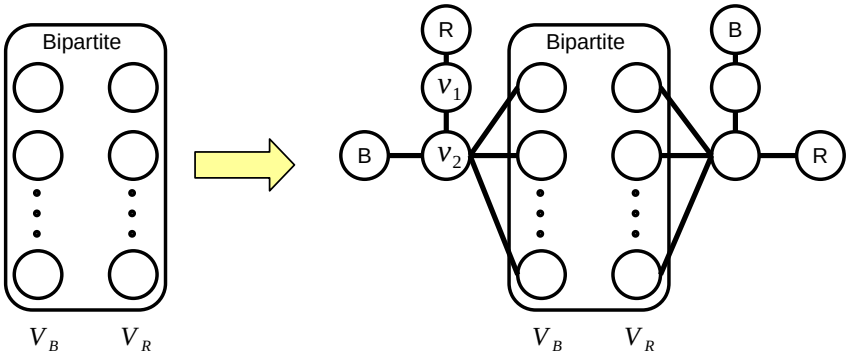

Figure 3. The reduction from BIGRAPH-NODE-KAYLES on the left to GRAPHDISTANCE($\{1, 2\}, \varnothing$) on the right.

in the same color have been removed completely, none of the paths of length two created by v_1 has any impact, so the edge condition is satisfied as well. We replicate this gadget on the right-hand side, switching the roles of R and B for V_R, which completes the reduction. □

Note that in these reductions for $D = \{1, 2\}$ we inserted a fixed number of vertices (four and eight, respectively), but that the number of additional edges is a function of the number of vertices of the original graph G. Specifically, we added a total of $n + 2$ and $n + 6$ edges, respectively. The number of edges of G does not play any role in the construction of the gadget. This is quite different from the general case discussed later, where the number of the gadgets (and hence the number of vertices and edges) inserted also depends on the number of edges of the original graph G.

3. Construction of the forbidden path gadget

For the remaining sets S and D to be considered in this paper, we will utilize a common construction for the various reductions. So far we have only added vertices and edges, but in the general case, we will also replace edges by subgraphs. As before, the concern is to make any vertex that is added into the graph unplayable by each of the two players in such a way that the vertices in the original graph G from which we reduce are not affected. We will achieve this by creating a forbidden vertex gadget and a forbidden path gadget.

Lemma 3.1. If D or S equals the set $\{1, 2, \ldots, r\}$ for some r, and the other is a subset of $\{1, 2, \ldots, r\}$, then we can create a *forbidden vertex gadget* $F(r)$ *of size* r which creates a vertex v such that v is uncolored, but neither player may choose to play at v, and the playability of any vertex connected to v is not

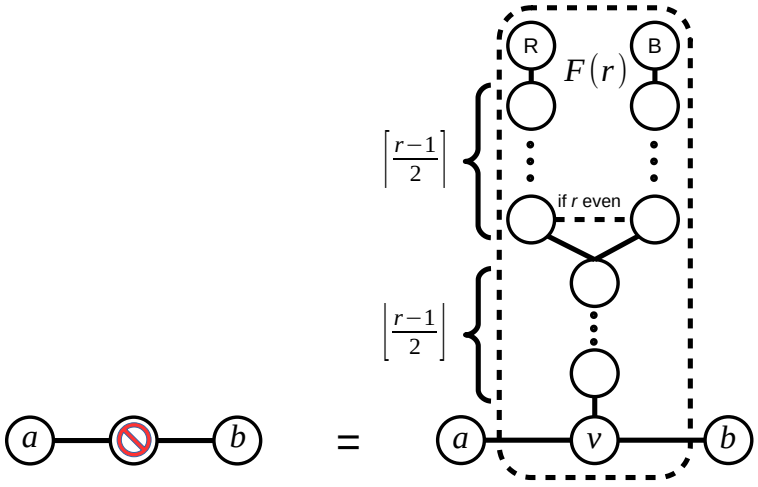

Figure 4. Given $D = \{1, 2, \ldots, r\}$, $F(r)$ creates a forbidden vertex (between a and b) without affecting the playability of a or b.

affected by the vertices in the gadget. Furthermore, all vertices in the gadget are either colored, or uncolored and unplayable.

Proof. Consider the gadget $F(r)$ shown in Figure 4, which is connected to vertices a and b. We now prove that any uncolored vertex in the gadget is unplayable by either player. Since the gadget is symmetric, we assume without loss of generality that $D = \{1, 2, \ldots, r\}$. For any r, the paths from the vertices labeled R and B, respectively, to vertex v are of length r, so each of the vertices on these paths cannot be colored blue and red, respectively. This means the vertices common to both paths cannot be colored with either red or blue. In addition, any vertex connected to v is not affected by the vertices labeled R and B as their distance from such a vertex is at least $r + 1$. For the upper portions of the two paths we now need to ensure that these vertices also cannot be colored with either color. When r is even, the shortest path from R to B using the dashed edge has length $2\lceil \frac{1}{2}(r-1) \rceil + 1 = r + 1$, so each of those vertices is within distance r of the R and B vertices and cannot be colored in either color. When r is odd, then the shortest path from R to B has length $2\lceil \frac{1}{2}(r-1) \rceil + 2 = r + 1$. Overall, all the unlabeled vertices in $F(r)$ cannot be played by either player, as stated. □

We will use a path of an appropriate length made up of forbidden vertex gadgets in the various reductions, where the size of the forbidden vertex gadget

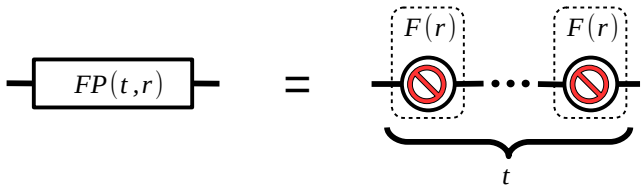

Figure 5. $FP(t, r)$ represents a path of t forbidden vertices of size r with edges leaving either end.

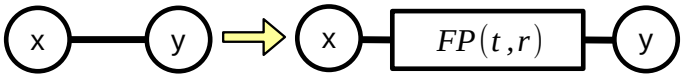

Figure 6. The edge replacement operation for edges (x, y) in the original graph G for the reduction to graph G' for GRAPHDISTANCE-(D, S), where t and r depend on the particular reduction.

is equal to the maximal element in the distance set that consists of consecutive integers. We refer to a path consisting of t forbidden vertices $F(r)$ as $FP(t, r)$, as shown in Figure 5.

Such a path will be used to replace an edge between two vertices of the original graph, as shown in Figure 6. We will refer to this operation as *edge replacement*. Note that inserting either one of these gadgets into the graph G automatically satisfies the vertex condition of the play-for-play reduction by Lemma 3.1.

Note that when we replace an edge with a forbidden path $FP(t, r)$, we add a total of

$$t\left(1 + \left\lfloor \tfrac{1}{2}(r-1) \right\rfloor + 2\left\lceil \tfrac{1}{2}(r-1) \right\rceil + 2\right) \approx t\tfrac{3}{2}r$$

vertices. The number of edges added is of the same order (there is a difference of 1 edge depending on whether r is odd or even). Overall, replacement of an edge by a forbidden path $FP(t, r)$ or addition of $FP(t, r)$ to any vertex adds $O(tr)$ edges and vertices to the graph. We will use this fact when looking at specific reductions in the next few sections.

We are now ready to prove that GRAPHDISTANCE(D, S) is PSPACE-hard for more general sets D and S.

4. D or S equals $\{1, 2, \ldots, r\}$, $\max(D) \neq \max(S)$

Let us first assume that $D = \{1, 2, \ldots, r\}$ with $r \geq 2$ and consider the case $S = \varnothing$. These distance games are generalizations of SNORT. We define ENSNORT(r) to be the game GRAPHDISTANCE$(\{1, 2, \ldots, r\}, \varnothing)$ for $r \geq 2$.

Proposition 4.1. ENSNORT(r) is PSPACE-hard.

Proof. ENSNORT(r) is a generalization of SNORT, which is PSPACE-hard [12], so we will use a reduction from SNORT to prove the result. Let $G = (V, E)$ be any graph. Since coloring a vertex in ENSNORT(r) affects vertices up to distance r from a colored vertex, we need to create a reduction that allows us to increase the distance between the vertices in G in such a way that any vertex that is inserted is not playable by either player. This can be achieved by replacing each edge in G with a forbidden path $FP(r-1, r)$. Now playing ENSNORT(r) on G' is exactly the same as playing SNORT on G. □

For nonempty sets S with $\max(S) < r = \max(D)$, the same reduction as in the case $S = \varnothing$ works because we only used properties of D, specifically the maximal reach, in the construction of the forbidden vertices $F(r)$ and paths $FP(r-1, r)$. As long as $\max(S) < r$, coloring restrictions from the set S do not impact any of the uncolored vertices from V in G', as all vertices from V are now at distance r from any other vertex in V.

Corollary 4.2. For $r \geq 2$, GRAPHDISTANCE$(\{1, 2, \ldots, r\}, S)$ is PSPACE-hard when $\max(S) < r$.

The case $S = \{1, 2, \ldots, r\}$ is very similar to the one treated above, with the roles of S and D interchanged, with reduction from COL, which is PSPACE-hard as well [7].

Proposition 4.3. For $r \geq 2$, GRAPHDISTANCE$(D, \{1, 2, \ldots, r\})$ is PSPACE-hard when $D = \varnothing$ or $\max(D) < r$.

To measure the impact of the gadget insertion on the graph in both cases, we let $m = |E|$ denote the number of edges in the original graph G. Since we have replaced each edge by a forbidden path $FP(r-1, r)$, we have added $O(mr^2)$ edges and vertices. Note that the maximal value of m is n^2 for a complete graph, and that $r \leq n$. So in the worst case scenario, this reduction adds $O(n^4)$ edges and vertices.

We now turn to the question of why we have to exclude the case $\max(S) = \max(D)$. If $\max(S) = \max(D) = r$, then for a vertex x colored in one color, a vertex y such that $(x, y) \in E$ would now be uncolorable in either color, not just the other color. This is why we need a reduction from a game that has the feature that a vertex adjacent to a colored vertex in V cannot be colored in either color. This suggests a reduction from NODE-KAYLES.

5. D or S equals $\{1, 2, \ldots, r\}$ and $r = \max(D) = \max(S)$

In this section, we consider distance games in which the maximum distances not playable by the same and different colors are identical.

Proposition 5.1. GRAPHDISTANCE(D, S) is PSPACE-hard when either D or S equals $\{1, 2, \ldots, r\}$ and the other set is a subset of $\{1, 2, \ldots, r\}$ with $2 \leq r = \max(D) = \max(S)$.

Proof. Let $G = (V, E)$ be any graph. We reduce from NODE-KAYLES, which is PSPACE-hard [12], and start with the extreme case where both D and S consist of the full set $\{1, 2 \ldots, r\}$. Since all vertices at distances less than or equal to r are unplayable by either player, we replace each edge $(x, y) \in E$ by the path gadget $FP(r-1, r)$. Then playing GRAPHDISTANCE(D, S) on G' is exactly the same as playing NODE-KAYLES on G.

For the more general case where $S \neq D$, the same construction works as any vertex inserted through the path gadget is unplayable as long as one of the two sets equals $\{1, 2, \ldots, r\}$ by Lemma 3.1. The conclusion follows as in the case $S = D$ since the only relevant distances for play are the maximal distances. □

Since we used the same edge replacement as in Proposition 4.1, we add at most $O(n^4)$ edges and vertices in this case as well.

6. Distance games on planar graphs

So far we have considered the computational complexity of distance games on any graph. A game on a more specialized (potentially simpler) graph may be easier to solve, and therefore might not be PSPACE-hard even though the game played on a general graph is PSPACE-hard.

In [4] the authors show that SNORT and COL are PSPACE-hard on planar graphs. The edge replacement operation we have used in our various reductions results in a planar graph G' when starting from a planar graph G. Thus our constructions show that the corresponding planar GRAPHDISTANCE(D, S) games are also PSPACE-hard. The stronger results are listed below:

Proposition 6.1. Both planar GRAPHDISTANCE($\{1, 2, \ldots, r\}$, S) and planar GRAPHDISTANCE(D, $\{1, 2, \ldots, r\}$) are PSPACE-hard when $S = \varnothing$ or $\max(S) < r$ and $D = \varnothing$ or $\max(D) < r$, respectively.

7. Conclusion and future work

To summarize, we used various play-for-play reductions from known PSPACE-hard games to show that GRAPHDISTANCE(D, S) is PSPACE-hard when D or S is $\{1, 2, \ldots, r\}$ and the other one is a subset. The games from which we reduced

were in most cases the natural choices based on the properties of the distance sets D and S. At the heart of the reductions was the forbidden vertex gadget. To obtain a play-for-play reduction required that the larger of the distance sets consists of consecutive integers. This leads to the following question:

Open Problem 7.1. Is GRAPHDISTANCE(D, S) PSPACE-hard for cases not covered by our results?

As illustrated in the previous section, planar SNORT and planar COL being PSPACE-hard implies that many of the distance games we considered are also PSPACE-hard on planar graphs because our reduction preserves planarity. If this is also the case for NODE-KAYLES, then these results can be further extended. Thus we are interested in:

Open Problem 7.2. Is planar GRAPHDISTANCE(D, S) PSPACE-hard for other cases?

The case where $\max(S) = \max(D)$ and at least one of S or D is equal to $\{1, \ldots, r\}$ would be covered using our reduction if NODE-KAYLES were PSPACE-hard on planar graphs.

References

[1] E. R. Berlekamp, J. H. Conway, and R. K. Guy, *Winning ways for your mathematical plays*, vol. 1, Academic Press, London, 1982. MR Zbl

[2] C. L. Bouton, "Nim, a game with a complete mathematical theory", *Ann. of Math.* (2) **3**:1-4 (1901/02), 35–39. MR Zbl

[3] J. I. Brown, D. Cox, A. H. Hoefel, N. McKay, R. Milley, R. J. Nowakowski, and A. A. Siegel, "A note on polynomial profiles of placement games", pp. 243–258 in *Games of no chance* 5, edited by U. Larsson, Math. Sci. Res. Inst. Publ. **70**, Cambridge University Press, 2019. MR Zbl

[4] K. Burke and R. A. Hearn, "PSPACE-complete two-color planar placement games", *Internat. J. Game Theory* **48**:2 (2019), 393–410. MR Zbl

[5] E. Demaine and R. Hearn, "Playing games with algorithms: algorithmic combinatorial game theory", pp. x+575 in *Games of no chance* 3, edited by M. H. Albert and R. J. Nowakowski, Mathematical Sciences Research Institute Publications **56**, Cambridge University Press, 2009. MR Zbl

[6] S. Faridi, S. Huntemann, and R. J. Nowakowski, "Games and complexes I: transformation via ideals", pp. 285–296 in *Games of no chance* 5, edited by U. Larsson, Math. Sci. Res. Inst. Publ. **70**, Cambridge University Press, 2019. MR Zbl

[7] S. A. Fenner, D. Grier, J. Messner, L. Schaeffer, and T. Thierauf, "Game values and computational complexity: an analysis via black-white combinatorial games", pp. 689–699 in *Algorithms and computation*, edited by K. Elbassioni and K. Makino, Lecture Notes in Comput. Sci. **9472**, Springer, 2015. MR Zbl

[8] A. S. Fraenkel and D. Lichtenstein, "Computing a perfect strategy for $n \times n$ chess requires time exponential in n", *J. Combin. Theory Ser. A* **31**:2 (1981), 199–214. MR Zbl

[9] S. Huntemann and R. J. Nowakowski, "Doppelgänger placement games", *Recreat. Math. Mag.* 1 (2014), 55–61. MR Zbl

[10] C. H. Papadimitriou, *Computational complexity*, Addison-Wesley, Reading, MA, 1994. MR Zbl

[11] S. Reisch, "Hex ist PSPACE-vollständig", *Acta Inform.* **15**:2 (1981), 167–191. MR

[12] T. J. Schaefer, "On the complexity of some two-person perfect-information games", *J. Comput. System Sci.* **16**:2 (1978), 185–225. MR Zbl

kburke@flsouthern.edu *Department of Computer Science, Florida Southern College, Lakeland, FL, United States*

sheubac@calstatela.edu *Department of Mathematics, California State University Los Angeles, Los Angeles, CA, United States*

melissa.huggan@viu.ca *Department of Mathematics, Vancouver Island University, Nanaimo BC, Canada*

svenja.huntemann@msvu.ca *Department of Mathematics and Statistics, Mount Saint Vincent University, Halifax NS, Canada*