

Biased Weak Polyform Achievement Games

Ian Norris*

Nándor Sieben†

*Northern Arizona University, Department of Mathematics and Statistics, Flagstaff, AZ, USA**received 21st July 2011, revised 20th May 2014, accepted 6th July 2014.*

In a biased weak (a, b) polyform achievement game, the maker and the breaker alternately mark a, b previously unmarked cells on an infinite board, respectively. The maker's goal is to mark a set of cells congruent to a polyform. The breaker tries to prevent the maker from achieving this goal. A winning maker strategy for the (a, b) game can be built from winning strategies for games involving fewer marks for the maker and the breaker. A new type of breaker strategy called the priority strategy is introduced. The winners are determined for all (a, b) pairs for polyiamonds and polyominoes up to size four.

Keywords: biased achievement games, priority strategy

MSC 2000: 91A46 (05B50)

1 Introduction

A *plane polyform* is a figure constructed by joining finitely many congruent basic polygons along their edges. If the basic polygons are *cells* of a regular tiling of the plane by squares, equilateral triangles or regular hexagons, then the polyform is called a *polyomino*, *polyiamond* or *polyhex* respectively. If the cells come from a regular tiling of the space by cubes, then the polyform is called a polycube. An *animal* is a polyomino, polyiamond, polyhex or polycube. We only consider animals up to congruence, that is, rotations and reflections of an animal are considered to be the same. The number of cells $s(A)$ of an animal A is called the *size* of A . The standard reference for polyominoes is [10].

In a *weak animal (a, b) achievement game* two players alternately mark a and b previously unmarked cells respectively using their own colors. The first player (the *maker*) tries to mark a copy of a given goal animal. The second player (the *breaker*) tries to prevent the maker from achieving his goal. An animal is an (a, b) -*winner* if the maker can win the (a, b) achievement game. Otherwise the animal is called a *loser*. Achievement games are studied, for example, in [1, 2, 3, 5, 9, 13].

In Section 2 we describe the pairing strategies and proof sequences which are the standard descriptions of breaker and maker strategies. We also prove some basic results.

Biased games [1, Sections 30–33] are more complex than the regular $(1, 1)$ game. In many cases, it is possible to decompose a biased game into simpler biased games involving fewer marks in each turn. In Section 3, we describe how an (a, b) maker strategy can be built from maker strategies for simpler games.

*Email: ian52n@gmail.com

†Email: nandor.sieben@nau.edu

The most important strategy for the breaker for an unbiased game is the pairing strategy. In fact, a long-standing difficulty is that the pairing strategy is almost the only tool we have for the breaker. The pairing strategy generalizes for $(1, b)$ games but the generalization does not seem straightforward for (a, b) games with $a \geq 2$. We remedy this problem in Section 4 with the introduction of the *priority strategy*.

Given an animal, our goal is to determine all the (a, b) pairs for which the animal is a winner. This information is collected in the *threshold sequence* for the animal described in Section 5. In Sections 6 and 7, we find the threshold sequence for each polyiamond and polyomino of size smaller than 5. One of the polyominoes requires a more sophisticated version of the priority strategy called *history dependent priority strategy*. In Section 8, we describe this strategy and we present an algorithm for verifying that a history dependent priority strategy works. The paper ends with an unsolved problems section.

The authors thank Ian Douglas and Steve Wilson for helpful discussions about the material.

2 Preliminaries

A strategy for the maker can be captured by a *proof sequence* (s_0, \dots, s_n) of situations [3, 19, 26, 27]. A situation $s_i = (C_{s_i}, N_{s_i})$ is an ordered pair of disjoint sets of cells. We think of the core C_{s_i} as a set of cells marked by the maker and the neighborhood N_{s_i} as a set of cells not marked by the breaker. A situation is the not necessarily connected part of the playing board that is important for the maker. A situation does not contain any of the breaker's marks. Those marks are not important as long as the situation contains enough empty cells in the neighborhood. Just like polyominoes, congruent situations are considered to be the same. In the situations of a proof sequence, it is always the breaker who is about to mark a cell. The game progresses from s_n towards s_0 . We require that C_{s_0} is the goal polyomino and $N_{s_0} = \emptyset$. This means that the maker already won by marking the cells in C_{s_0} and there is no need for any cells on the board in N_{s_0} . For each $i \in \{1, \dots, n\}$ we also require that if the breaker marks b or fewer cells in N_{s_i} , then the maker can mark a cells of N_{s_i} not marked by the breaker and reach a position s_j closer to his goal, that is, satisfying $j < i$. More precisely, for all $\{x_1, \dots, x_b\} \subseteq N_{s_i}$ there must be an $\{\tilde{x}_1, \dots, \tilde{x}_a\} \in N_{s_i} \setminus \{x_1, \dots, x_b\}$ and a $j \in \{0, \dots, i-1\}$ such that

$$C_{s_j} \subseteq C_{s_i} \cup \{\tilde{x}_1, \dots, \tilde{x}_a\} \text{ and } N_{s_j} \subseteq C_{s_i} \cup N_{s_i} \setminus \{x_1, \dots, x_b\}.$$

Figures 6a and 1b show examples of proof sequences. We present proof sequences graphically. In the figures, filled cells represent the marks of the maker. Cells with letters in them are the neighborhood cells that must be unmarked. Each letter represents a possible continuation for the maker. After the marks of the breaker, the maker picks a letter unaffected by the breaker marks. The maker marks the cells with the capital version of this letter. The cells with the lower case version of the chosen letter become the neighborhood cells of the new situation. Each situation is constructed to make sure that the breaker cannot mark b cells which together contain a lower case or capital copy of all the available letters. We include a flow chart for each proof sequence. The letter on the arrows of the flow chart is used to determine which situation the maker can reach by picking that letter.

Most of the known strategies for the breaker are based on pairings of the cells of the board. A *b-paving* of the board is a symmetric and irreflexive relation on the set of cells where each cell is related to at most b other cells. Figure 7 shows two examples of 1-pavings and one example of a 2-paving. A *b-paving* determines the following *paving strategy* for the breaker in the $(1, b)$ game. In each turn, the breaker marks the unmarked cells related to the cell last marked by the maker. If there are fewer than b such cells, then she uses her remaining marks randomly. If the breaker follows the paving strategy, then the maker

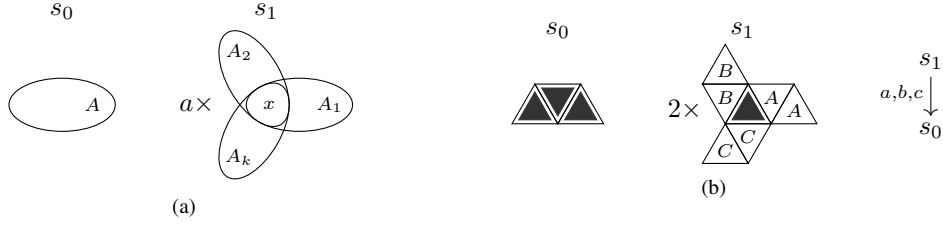


Figure 1: (a) Schematic (a, b) proof sequence for the animal A assuming $|A| = a + 1$ and $ak > b$. (b) A $(2, 5)$ -proof sequence for the polyiamond $T_{3,1}$.

cannot mark two related cells during a game. This allows the breaker to win if every placement of the goal animal on the board contains a pair of related cells.

Two cells of a regular tiling are *adjacent* if they share a common edge. The *exterior boundary* $E(A)$ of the animal A is the set of cells outside of A but adjacent to a cell of A . The *site-perimeter* of A is the number of cells $p(A) := |E(A)|$ in the exterior boundary (see [8, 25]). Let δ be the size of the site-perimeter of a single cell. Note that δ is 3, 4 and 6 on the triangular, rectangular and hexagonal board respectively.

Proposition 2.1. *Let A be an animal. If $a < |A|$ and $a\delta \leq b$ then A is an (a, b) -loser.*

Proof: Since $a < |A|$, the maker cannot build the goal animal using only the marks of a single turn. The size of the site-perimeter of the marks of the maker in a single turn is at most $a\delta$ so this whole site perimeter can be marked by the breaker. If the breaker marks every cell of the site-perimeter of the maker's mark in each turn, then the maker cannot build the goal animal using some marks from previous turns either. \square

Proposition 2.2. *Let A be an animal with $|A| = a + 1$. Assume A has k placements A_1, \dots, A_k on the board with a single common cell x , that is, $A_i \cap A_j = \{x\}$ for all $i, j \in \{1, \dots, k\}$ with $i \neq j$. If $ak > b$ then A is an (a, b) -winner.*

Proof: Figure 1a shows a schematic proof sequence. The maker can create situation s_1 in the first turn by marking a cells far from each other so that a copies of the placements of A_1, \dots, A_k are created. The common cell x of the k placements is marked by the maker in each of these copies. The breaker cannot mark a cell in each of the ak copies of the goal animal, so the maker can win in the second turn. \square

The previous result is often used with $k = \delta$ as the following example shows.

Example 2.3. Figure 1b shows that the polyiamond $T_{3,1}$ is a $(2, 5)$ -winner using the strategy of Proposition 2.2 with $k = \delta = 3$.

3 The $(a \rightarrow c, b)$ game

Now we introduce a variation of the regular (a, b) game that we call the $(a \rightarrow c, b)$ game. In this variation the maker marks a previously unmarked cells in each turn until the very last turn. In the last turn he is allowed to mark c cells. The breaker marks b previously unmarked cells in each turn.

An animal is called a *bounded winner* if the maker has a winning strategy consisting of at most a fixed number of moves on some finite subboard of the playing board. All the known winning polyforms are also bounded winners but see [1, Section 5.4] for examples of hypergraph games that are finite but unbounded in time or in space.

The reason for the study of the seemingly unnatural $(a \rightarrow c, b)$ games is the following result. The proof uses some ideas found in [1, Section 14, Section 30]. We use the notation $\mathbb{W} = \mathbb{N} \cup \{0\}$ for the set of non-negative integers.

Theorem 3.1. *Let $a = \sum_{i=1}^s a_i$ and $b = \sum_{i=1}^s b_i$ with $a_i, b_i \in \mathbb{W}$. If a goal animal is a bounded $(a_i \rightarrow a, b_i)$ -winner for all i , then it is also an $(a, b + s - 1)$ -winner.*

Proof: Let us call the $(a_i \rightarrow a, b_i)$ game the i -th game. The goal animal is a bounded winner in the i -th game for each i , that is, the maker can mark a copy of the goal animal after l_i turns on some sufficiently large but finite subset B_i of the original infinite board. Since s and the B_i are finite, we can find a finite subset B of the original infinite board that contains a copy of every B_i . The maker can win the i -th game for all i on any placement of B on the infinite board. Any breaker mark outside a placement of B has no effect on the outcome of the i -th game played on that placement. We are going to use subboards, which are disjoint placements of B . The main idea of the maker's $(a, b + s - 1)$ strategy is to mark s groups of cells containing (a_1, \dots, a_s) cells respectively far away from each other in subboards and to play distributed i -th games on the subboards that have at most b_i breaker marks in each turn.

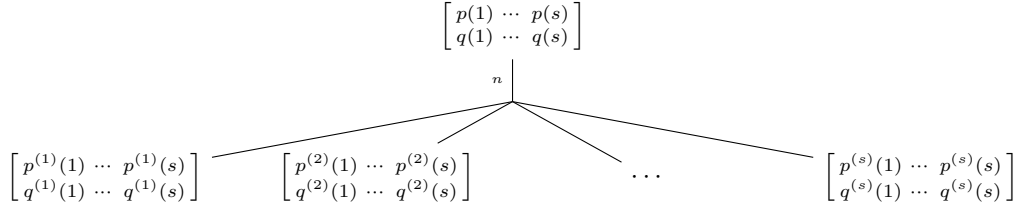
The strategy for the $(a, b + s - 1)$ game has stages consisting of several turns. In each stage, the maker tries to make a one-turn progress in one of the i -th games. We keep track of the progress using a progress vector p_j in \mathbb{W}^s . At the beginning of the game, the progress vector is $p_0 = (0, \dots, 0)$ indicating that no progress is made in any of the subgames.

The first stage contains n_1 turns in which the maker marks $n_1 a$ cells. In each turn, s new disjoint subboards congruent to B are created. The new subboards receive a_1, \dots, a_s marks respectively, according to the first moves in the corresponding $(a_i \rightarrow a, b_i)$ strategies. Let us call the subboards receiving maker moves according to the i -th strategy type i subboards. The total number of subboards at the end of the stage is $n_1 s$ since there are n_1 subboards for each type. At the end of the first stage, we also have $n_1(b + s - 1)$ breaker marks on the board. Let $k_{1,i}$ be the number of type i alive subboards containing at most b_i breaker marks. Then we have

$$n_1(b + s - 1) \geq \sum_{i=1}^s (n_1 - k_{1,i})(b_i + 1)$$

which implies $\sum_{i=1}^s k_{1,i}(b_i + 1) \geq n_1$. So a large enough n_1 guarantees that k_{1,i_1} is large enough for some i_1 , that is, the maker can create as many alive type i_1 subboards as he needs. Now the maker disregards all the subboards except the k_{1,i_1} alive subboards of type i_1 that have at most b_{i_1} breaker marks. The progress vector becomes $p_1 = (0, \dots, 0, 1, 0, \dots, 0)$ where the 1 is at the i_1 -th coordinate indicating that the maker made progress in the i_1 -th game.

Stage j contains n_j turns in which the maker marks $n_j a$ cells. The cells are played on s different subboards. For each $i \in \{1, \dots, s\}$ the maker marks a_i cells on a type i alive subboard according to the strategy for the $p_{j-1}(i) + 1$ -st move in the i -th game. If $p_{j-1}(i) = 0$ then the maker needs an alive subboard for a first move in the i -th game. This is an empty subboard, so the maker creates a new empty


 Figure 2: A vertex and its s descendants in the stage diagram.

subboard which is a disjoint placement of B . Repeating the counting argument above gives

$$\sum_{i=1}^s k_{j,i}(b_i + 1) \geq n_j, \quad (1)$$

that is, a large enough n_j guarantees that for at least one type i_j the number k_{j,i_j} of alive subboards of type i_j that have at most b_{i_j} breaker marks in the current stage is as large as needed. Now the maker disregards all the type i_j subboards except the k_{j,i_j} alive subboards. The progress vector p_{j+1} becomes p_j with the i_j -th coordinate incremented, indicating that the maker made progress in the i_j -th game.

The game continues in the same way. In each stage one coordinate of the progress vector is incremented. Eventually, say after at most $l - 1$ stages, the progress vector is going to have a coordinate i such that $p_{l-1}(i) = l_i - 1$. Then in stage l the maker can mark a cells in a type i alive subboard and win. So the last stage contains only $n_l = 1$ turn.

For this to work, the number of alive subboards of each type needs to be large enough, so that the maker can mark cells in an alive subboard of the type determined by the strategy. The number of required alive subboards of each type is potentially very large but is finite since the length l_i of the i -th game is finite for all i . So the maker can create sufficiently many subboards by playing each stage long enough, that is, picking a large enough n_j for all j .

The number n_j of required turns in each stage can be calculated using a *stage diagram*. Each vertex of the diagram is a $2 \times s$ matrix that represents a possible stage during the game play. The first row of the matrix is the progress vector p . The second row is the *supply vector* q . The i -th coordinate $q(i)$ of the supply vector is the minimum number of required type i alive subboards containing at most $p(i)b_i$ breaker marks and $p(i)a_i$ maker marks according to the winning strategy for the i -th game.

The game starts at the top of the diagram and progresses towards the bottom along the edges. Each vertex has s descendants. To get the progress vector of the i -th descendant, we increment the i -th coordinate of the progress vector of the parent vertex. The game ends when we reach a stage with $p(i) = l_i$ for some $i \in \{1, \dots, s\}$. The labels on the edges show the number of required turns between stages.

A dashed edge indicates a single turn in which the maker reaches a winning stage by putting all his a marks into a single type i subboard with progress $p(i) = l_i - 1$. In the winning stages the supply vector satisfies

$$q(i) = \begin{cases} 1 & \text{if } p(i) = l_i \\ 0 & \text{if } p(i) < l_i \end{cases}$$

since the maker only needs a single winning subboard of any type to finish the game. The supply vectors of the other stages and the labels on the edges can be calculated recursively from the bottom of the diagram

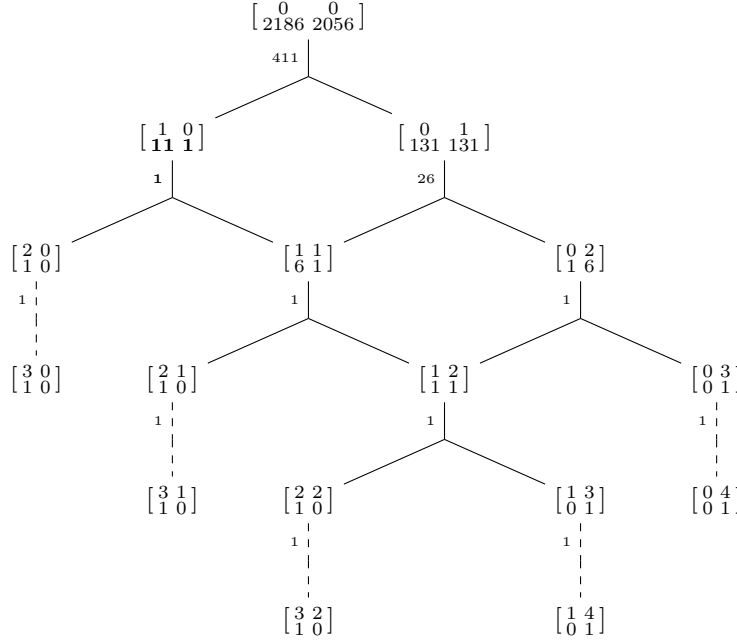


Figure 3: A stage diagram for $b_1 = 1, b_2 = 2, l_1 = 3$ and $l_2 = 4$.

to the top. Let

$$\begin{bmatrix} p^{(1)} \\ q^{(1)} \end{bmatrix}, \dots, \begin{bmatrix} p^{(s)} \\ q^{(s)} \end{bmatrix}$$

be the descendants of stage $v = \begin{bmatrix} p \\ q \end{bmatrix}$ as shown in Figure 2. Then the number of required turns after stage v is

$$n := 1 + \sum_{i=1}^s (q^{(i)}(i) - 1)(b_i + 1).$$

Equation 1 guarantees that after n turns the maker reaches the i -th descendant with $q^{(i)}(i)$ type i alive subboards for some i . We write this number as a label on the edge emanating from vertex v . During these n turns the maker uses up n subboards of each type and the breaker marks $(n - 1)(b + s - 1)$ cells between the maker's turns, each of which can ruin a subboard. So we must have $q(i) \geq n + (n - 1)(b + s - 1)$. If $q^{(j)}(i) > 0$ for some $j \neq i$, then the breaker can ruin $b + s - 1$ additional subboards in her n -th turn and so we must have $q(i) \geq q^{(j)}(i) + n + n(b + s - 1)$.

So to have enough supply of alive subboards we let

$$q(i) := \max(\{(n - 1)(b + s) + 1\} \cup \{q^{(j)}(i) + n(b + s) \mid q^{(j)}(i) > 0 \text{ and } j \neq i\})$$

for all i . We have infinitely many empty subboards with no progress so we never run out of subboards. \square

Example 3.2. Figure 3 shows a stage diagram for a game with $s = 2$, $b = b_1 + b_2 = 1 + 2$, $l_1 = 3$ and $l_2 = 4$. We show the details of the calculation at the stage with $p = (1, 0)$ which is the first vertex on the second row. We have

$$\begin{bmatrix} p^{(1)} \\ q^{(1)} \end{bmatrix} = \begin{bmatrix} 2 & 0 \\ 1 & 0 \end{bmatrix}, \quad \begin{bmatrix} p^{(2)} \\ q^{(2)} \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 6 & 1 \end{bmatrix},$$

$$n = 1 + (q^{(1)}(1) - 1)(b_1 + 1) + (q^{(2)}(2) - 1)(b_2 + 1) = 1.$$

Thus

$$q(1) = \max\{(n - 1)(b + s) + 1, q^{(2)}(1) + n(b + s)\} = \max\{1, 11\} = 11,$$

$$q(2) = \max\{(n - 1)(b + s) + 1\} = \max\{1\} = 1.$$

The computed values are shown as bold numbers on Figure 3. Note that the game finishes after at most 440 turns.

The number of required turns calculated using the stage diagram is potentially very large. Our calculation is a crude overestimate that could be improved with a more complicated argument but our goal was only to prove that the strategy works. Also note that the breaker can use her marks to ruin many subboards by ignoring some of the subboards completely. In this case, the maker has subboards with very few defensive marks on them and so he probably can win faster using a more refined strategy.

Corollary 3.3. *If an animal is a $(1 \rightarrow a, \lfloor \frac{b}{a} \rfloor)$ -winner, then it is also an (a, b) -winner.*

Proof: The result is a special case of Theorem 3.1 with $a_i = 1$ and $b_i = \lfloor \frac{b}{a} \rfloor$ for all $i \in \{1, \dots, a\}$. \square

Note that finding a proof sequence for the $(1 \rightarrow a, \lfloor \frac{b}{a} \rfloor)$ game is often much simpler than for the (a, b) game. The following is an easy consequence.

Corollary 3.4. *If $b < a$ then any animal is an (a, b) -winner.*

4 The priority strategy

One of the difficulties of the theory of achievement games is the lack of strategies for the breaker other than the pairing strategy based on pavings. In this section we describe a new strategy for the breaker.

Definition 4.1. An (a, b) *priority strategy* is a strategy for the breaker. Let (x_1, \dots, x_a) be an ordering of the current marks of the maker. The ordering can depend for example on the location of the marked cell or on the relative positions of the marks. By default we order them using the lexicographic order of the coordinates. The priority strategy assigns a *response set* R_{x_i} of *response cells* for each x_i . The priority among the response cells contained in R_{x_i} is determined by a *priority number*. A smaller number means higher priority. In the simplest case, the priority numbers of the response cells are the same for each mark of the breaker. In more complicated cases, the priority numbers of the response cells may depend on the location or on the ordering of the maker marks. The priority numbers in a response set can change after each breaker mark. The breaker tries to mark one of the highest priority unmarked cells in each of the response sets following the order $R_{x_1}, \dots, R_{x_a}, R_{x_1}, \dots$ until she runs out of marks. If there are no unmarked cells in a response set, then the breaker moves to the next response set. If the breaker is not able to use all her b marks, then she can use the remaining marks on random cells.

Note that every paving strategy is a priority strategy where the cells with priority one are guaranteed to be available.

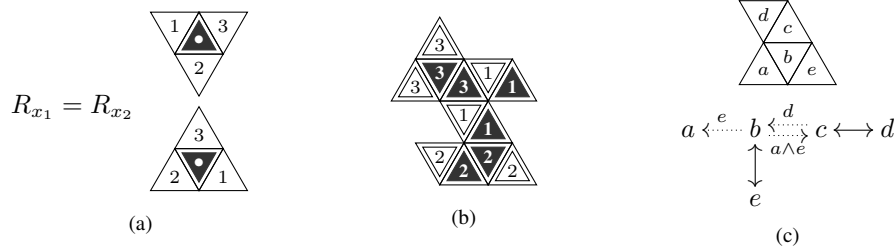


Figure 4: (a) A (2, 2) priority strategy. The bullets indicate the marks of the maker in the current turn. (b) A possible game play where the breaker follows the priority strategy. The marks of the maker are black while the marks of the breaker are white. (c) A dependency digraph of a placement of a goal animal.

Example 4.2. Figure 4a shows a graphical description of a (2, 2) priority strategy. For each mark of the maker, the breaker has three possible response cells. The priorities are the same for both maker marks but they depend on the type of the cell. The breaker first tries to mark the cells containing a 1. If any of those cells are not available because they are already marked, then she tries to mark the cells containing a 2 and so on. If none of these cells are available, then she marks random cells. Figure 4b shows a possible position as a result of the priority strategy. The breaker marks a priority 2 and a priority 3 cell in the last turn since no priority 1 cell is available.

Note that the graphical representations of priority strategies are generally not invariant under reflections or rotations.

Example 4.3. Figure 15a shows a graphical description of a (2, 4) priority strategy where the priority numbers in a response set are not constant. On her first mark in the response set, the breaker tries to mark cell α_1 . She marks α_2 if α_1 is already marked. On her second mark in the same response set, the breaker tries to mark cell β_1 and falls back on β_2 if β_1 is already marked.

Definition 4.4. Given a priority strategy and a placement of the goal animal we have a *dependency relation* on the set of cells of the goal animal. We write $a \leq b$ if the maker cannot achieve the placement of the goal animal if he tries to mark cell a in turn that comes later than the turn in which he marks cell b . We write $a \sim b$ if a and b have to be marked by the maker in the same turn, that is, $a \leq b$ and $b \leq a$.

It is clear that the dependency relation is transitive.

Definition 4.5. The dependency relation can be captured using a *dependency digraph*. The vertex set of the digraph is the set of cells of the goal animal. We use three types of arrows:

- Arrow $a \longrightarrow b$ is called an *unconditional arrow*. It indicates that cell b cannot be marked after cell a . We add this arrow when response cell b in R_a has highest possible priority. An unmarked cell b in this situation is going to be marked by the breaker right after the maker marks cell a .
- Arrow $a \xrightarrow{l_1, \dots, l_k} b$ is called a *conditional arrow*. It indicates that cell b cannot be marked after cell a if condition l_i is satisfied for some i . Each condition is of the form $l_i = x_1 \wedge \dots \wedge x_r$ where x_j is a logical variable corresponding to a cell of the goal animal for all j . The value of x_j is true if cell x_j is marked by either the maker or the breaker in a turn no later than the turn in which a is marked.

We add this arrow when response cell b in R_a does not have maximal priority but sufficiently many marked response cells x_1, \dots, x_r in R_a with higher priority makes the priority of b large enough to guarantee the marking of b . In this situation, cell b left unmarked by the maker is going to be marked by the breaker right after the maker marks cell a in spite of the lower priority.

- Arrow $a \overset{l_1, \dots, l_k}{\dashrightarrow} b$ is called a *secondary arrow*. This arrow is similar to the conditional arrow but condition $x_1 \wedge \dots \wedge x_r$ is interpreted differently. We add this arrow when response cell b in R_a does not have maximal priority but sufficiently many response cells in R_a with higher priority are already marked because these response cells are also maximal priority response cells for one of the cells x_1, \dots, x_r already marked by the maker. So the value of x_j is true if cell x_j is marked in a turn earlier than the turn in which a is marked. The value of x_j can be true even if cells x_j and a are marked at the same turn. For this to happen, x_j and a have to be ordered in the set of current marks by the priority strategy to make the common response cell z of x_j and a an earlier response cell for x_j than a response cell for a .

We might omit secondary arrows and even conditional arrows in our dependency digraphs if they are not needed.

Example 4.6. Figure 4c shows a goal animal and its dependency digraph based on the priority strategy presented in the same figure. The arrow $b \overset{a \wedge e}{\dashrightarrow} c$ has one label. This arrow indicates that if the maker marks cell b but leaves cell c unmarked, then the breaker is going to mark cell c assuming cells a and e are already marked.

If the maker wants to mark all the cells in $\{a, b, c, d, e\}$, then he needs to make sure that in each turn he marks all the cells that are dependent on other marks of his own. For example he can mark cell a without marking any other cells of the goal animal. On the other hand if he marks cell b , then he has to mark all the other cells in the same turn. To see this first note that there is a solid arrow from b to e so cell e needs to be marked. This implies that cell a needs to be marked since the label of the dotted arrow is satisfied. Now the label on the arrow from b to c is satisfied so cell c must be marked as well. Finally cell d needs to be marked since there is a solid arrow from c to d . Thus we have $a \leq b \sim c \sim d \sim e$.

It is easy to see that there are only two options for marking all the cells in $\{a, b, c, d, e\}$. The first option is to mark cell a in a turn and then mark the rest of the cells in a later turn. The other option is to mark all the cells in a single turn. Both of these options are impossible since the maker can only mark two cells in a turn.

Example 4.7. Figure 18b shows a dependency digraph containing secondary arrows. The secondary arrows exist because the response cell for a with priority β_1 is the same as the response cell for d with priority α_1 . This common response cell x is located above a and on the left of d . We clearly have $a, d \leq b \sim c$. If the maker marks a before d , then x is marked as a response cell for a and so the vertical secondary arrow from d to b is activated implying $a \leq b \sim c \sim d$. If the maker marks d before a , then x is marked as a response cell for d and so the horizontal secondary arrow from a to b is activated implying $d \leq a \sim b \sim c$.

The situation is a bit more delicate if the maker marks a and d in the same turn. In the lexicographic order a is considered smaller than d so the breaker first marks a response cell for a . This is the response cell on the left of a with priority α_1 . Next the breaker marks a response cell for d . This response cell is x with priority α_1 . Hence the horizontal secondary arrow from a to b is activated and we have $d \leq a \sim b \sim c$.







A						
$\tau(A)$	(∞)	$(2, \infty)$	$(1, 5, \infty)$	$(0, 3, 8, \infty)$	$(0, 3, 8, \infty)$	$(0, 3, 8, \infty)$

Figure 5: Polyiamonds and their threshold sequences up to size four.

In all three cases at least three cells have to be marked in a single turn.

5 The threshold sequence

Our main goal is to determine all the (a, b) pairs for which an animal is an (a, b) -winner. We use the following object to capture this information.

Definition 5.1. The *threshold sequence* $\tau(A) = (b_1, b_2, \dots)$ for a given animal A is a sequence of numbers in $\mathbb{W} \cup \{\infty\}$ such that b_n is the greatest value for which A is an (n, b_n) -winner.

The following is an easy consequence of the fact that marking more cells could never hurt the players.

Lemma 5.2. Let A be an animal and $n, k \in \mathbb{W}$. If A is an (a, b) -winner, then it is also an $(a + n, b - k)$ -winner. If A is an (a, b) -loser, then it is also an $(a - n, b + k)$ -loser.

Proposition 5.3. For every threshold sequence (b_1, b_2, \dots) there is an index $k \in \mathbb{W}$ such that b_n is finite for all $n < k$ and $b_n = \infty$ for all $n \geq k$.

Proof: If the animal A has l cells, then A is clearly an (l, b) -winner for all $b \in \mathbb{W}$. If A is an (k, ∞) -winner, then it is also an (n, b) -winner for all $n \geq k$ and $b \in \mathbb{W}$. \square

We are going to write the threshold sequence $(b_1, \dots, b_{k-1}, \infty, \dots)$ where $b_{k-1} < \infty$ simply as $(b_1, \dots, b_{k-1}, \infty)$.

Proposition 5.4. In a threshold sequence $(b_1, \dots, b_{k-1}, \infty)$ we have $b_i < b_{i+1}$ for all $i \in \{1, \dots, k-1\}$.

Proof: It is clear that every animal is a $(1, 0)$ -winner. Since the animal is also an (i, b_i) -winner, it must be a $(i + 1, b_i + 1)$ -winner by Theorem 3.1. \square

Proposition 5.5. If A is a subset of animal B , then $\tau(A)(i) \geq \tau(B)(i)$ for all $i \in \mathbb{N}$.

Proof: Any successful maker strategy for B is also successful for A . Hence if B is an (a, b) -winner, then so is A . \square

The following is an easy consequence of Corollary 3.4.

Proposition 5.6. For each animal A we have $\tau(A)(i) \geq i - 1$.

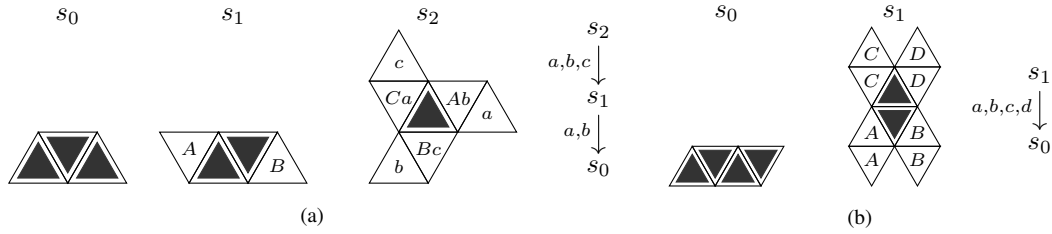


Figure 6: (a) A (1, 1) proof sequence for $T_{3,1}$. (b) A (2, 3) proof sequence for $T_{4,1}$.

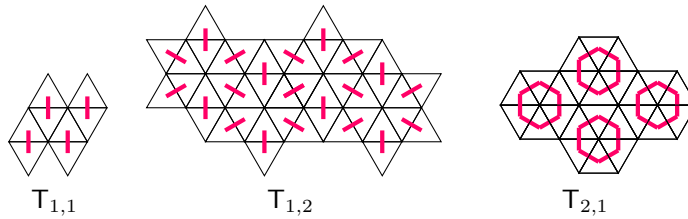


Figure 7: Triangular pavings.

6 Polyiamonds

Unbiased polyiamond games are studied in [4, 5, 8]. In this section we find the threshold sequences of polyiamonds up to size four. The results are summarized in Figure 5. The proof of the following result is left to the reader.

Proposition 6.1. *The threshold sequences of $T_{1,1}$ and $T_{2,1}$ are $\tau(T_{1,1}) = (\infty)$ and $\tau(T_{2,1}) = (2, \infty)$.*

We now consider the size-three polyiamond.

Proposition 6.2. *The threshold sequence of $T_{3,1}$ is $\tau(T_{3,1}) = (1, 5, \infty)$.*

Proof: The proof sequence of Figure 6a shows that $T_{3,1}$ is a (1, 1)-winner. The breaker strategy based on the 2-paving $T_{2,1}$ shown in Figure 7 makes $T_{3,1}$ a (1, 2)-loser.

We saw in Example 2.3 that $T_{3,1}$ is a (2, 5)-winner. Proposition 2.1 implies that $T_{3,1}$ is a (2, 6)-loser. \square

Proposition 6.3. *The threshold sequence of $T_{4,1}$ is $\tau(T_{4,1}) = (0, 3, 8, \infty)$.*

Proof: The breaker strategy based on paving $T_{1,2}$ shown in Figure 7 makes $T_{4,1}$ a (1, 1)-loser. It is easy to see that $T_{4,1}$ is a (3, 8)-winner by Proposition 2.2 and a (3, 9)-loser by Proposition 2.1. The proof sequence of Figure 6b shows that $T_{4,1}$ is a (2, 3)-winner.

It remains to show that $T_{4,1}$ is a (2, 4)-loser. We are going to show that the breaker wins following the priority strategy determined by Figure 8a. Figure 8b shows all possible orientations of the goal animal. For each of these orientations the dependency digraph of the cells show that cells a , b and c must be

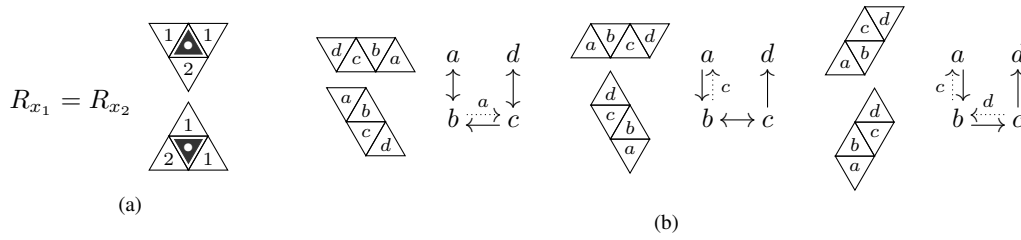


Figure 8: (a) A (2, 4) priority strategy for the breaker for $T_{4,1}$. (b) Dependency digraphs of the cells in the orientations of $T_{4,1}$.

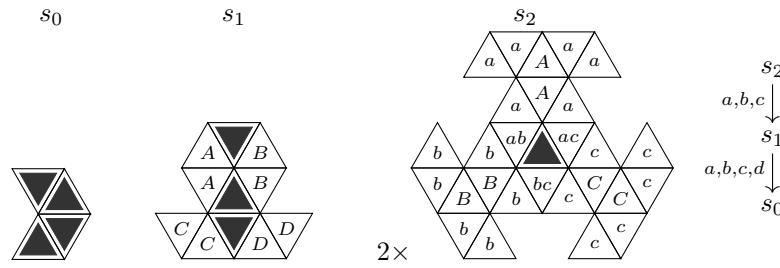


Figure 9: A (2, 3) proof sequence for $T_{4,2}$.

marked in the same turn by the maker to achieve the goal animal. The maker is not able to do so since he only has two marks in a turn. \square

Proposition 6.4. *The threshold sequence of $T_{4,2}$ is $\tau(T_{4,2}) = (0, 3, 8, \infty)$.*

Proof: The breaker strategy based on the double tiling $T_{1,1}$ shown in Figure 7 makes $T_{4,2}$ a (1, 1)-loser. It is easy to see that $T_{4,2}$ is a (3, 8)-winner by Proposition 2.2 and a (3, 9)-loser by Proposition 2.1. The proof sequence of Figure 9 shows that $T_{4,2}$ is a (2, 3)-winner.

It remains to show that $T_{4,2}$ is a (2, 4)-loser. We are going to show that the breaker wins following the

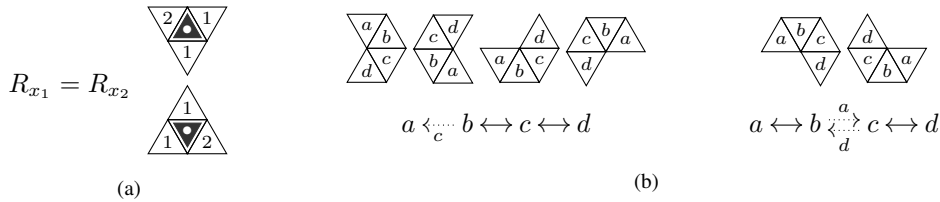


Figure 10: (a) A (2, 4) priority strategy for the breaker for $T_{4,2}$. (b) Dependency digraph of the cells in the orientations of $T_{4,2}$.

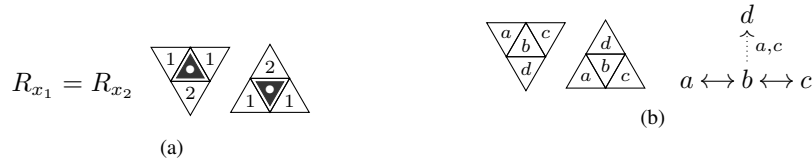


Figure 11: (a) A $(2, 4)$ priority strategy for the breaker for $T_{4,3}$. (b) Dependency digraph of the cells in the orientations of $T_{4,3}$.

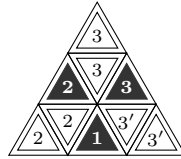


Figure 12: A forced game play of the $(1 \rightarrow 2, 1)$ game for the animal $T_{4,3}$.

priority strategy determined by Figure 10a. Figure 10b shows the possible orientations of the goal animal. In all of these orientations the dependency digraph of the cells show that cells b , c and d must be marked in the same turn by the maker to achieve the goal animal. The maker is not able to do so since he only has two marks in a turn. \square

Proposition 6.5. *The threshold sequence of $T_{4,3}$ is $\tau(T_{4,3}) = (0, 3, 8, \infty)$.*

Proof: The breaker strategy based on the double tiling $T_{1,1}$ shown in Figure 7 makes $T_{4,3}$ a $(1, 1)$ -loser. It is easy to see that $T_{4,3}$ is a $(3, 8)$ -winner by Proposition 2.2 and a $(3, 9)$ -loser by Proposition 2.1.

We show that the breaker wins the $(2, 4)$ game following the priority strategy determined by Figure 11a. Figure 11b shows the two possible orientations of the goal animal. For both of these orientations the dependency digraph of the cells show that cells a , b and c must be marked in the same turn by the maker to achieve the goal animal. The maker is not able to do so since he only has two marks in a turn.

It remains to show that $T_{4,3}$ is a $(2, 3)$ -winner. We are going verify that $T_{4,3}$ is a $(1 \rightarrow 2, 1)$ -winner and use Corollary 3.3. No matter how the breaker picks her first mark, the maker can force some rotation of the game shown in Figure 12. The breaker has to mark one of the cells labeled with 2 in the second turn otherwise the maker wins in the third turn. The breaker has to mark one of the cells labeled with 3 and one of the cells labeled with $3'$ in the third turn. This is impossible so the maker wins in the fourth turn. \square

7 Polyominoes

Polyomino achievement games were invented by Harary. The first proof sequences for the unbiased games appeared in [3]. Every known $(1, 1)$ -winner is a subset of one of the winning polyominoes shown in Figure 13a. The only undecided [11, 14, 17, 18, 23, 26] polyomino Snaky, shown in Figure 13b, is conjectured [2] to be a winner. Biased $(1, 2)$ polyomino set games were studied in [7, 24].

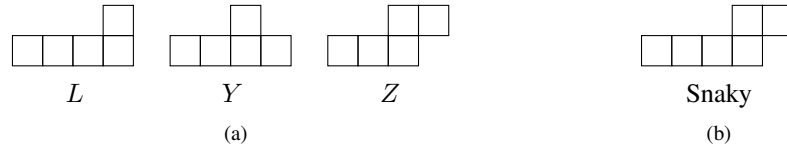






Figure 13: (a) Unbiased winners. (b) The only undecided polyomino.

				
A	$P_{1,1}$	$P_{2,1}$	$P_{3,1}$	$P_{3,2}$
$\tau(A)$	(∞)	$(3, \infty)$	$(1, 7, \infty)$	$(1, 7, \infty)$






					
A	$P_{4,1}$	$P_{4,2}$	$P_{4,3}$	$P_{4,4}$	$P_{4,5}$
$\tau(A)$	$(1, 3, 11, \infty)$	$(1, 5, 11, \infty)$	$(1, 3, 11, \infty)$	$(0, 3, 5, \infty)$	$(1, 3, 11, \infty)$

Figure 14: Polyominoes and their threshold sequences up to size four.

In this section, we find the threshold sequences of polyominoes up to size four. The results are summarized in Figure 14. The proof of the following result is an easy exercise.

Theorem 7.1. *The threshold sequences for $P_{1,1}$ and $P_{2,1}$ are $\tau(P_{1,1}) = (\infty)$ and $\tau(P_{2,1}) = (3, \infty)$.*

Theorem 7.2. *The threshold sequence for $P_{3,1}$ and $P_{3,2}$ is $\tau(P_{3,1}) = \tau(P_{3,2}) = (1, 7, \infty)$.*

Proof: Polyominoes $P_{3,1}$ and $P_{3,2}$ are $(1, 1)$ -winners since they are subsets of the winner L shown in Figure 13a. The breaker wins the $(1, 2)$ games for $P_{3,1}$ and $P_{3,2}$ using the strategy based on the double paving $T_{2,2}$ and $T_{2,1}$ respectively. It is easy to see that $P_{3,1}$ and $P_{3,2}$ are $(2, 7)$ -winners using Proposition 2.2 and $(2, 8)$ -losers using Proposition 2.1. □

Now we consider the size 4 animals.

Lemma 7.3. *Animals $P_{4,1}$, $P_{4,2}$, $P_{4,3}$ and $P_{4,5}$ are $(3, 11)$ -winners and $(3, 12)$ -losers.*

Proof: The result follows easily from Proposition 2.2 and Proposition 2.1. □

Note that $P_{4,4}$ is missing from the lemma since Proposition 2.2 does not apply for this animal.

Proposition 7.4. *The threshold sequence of $P_{4,1}$ is $\tau(P_{4,1}) = (1, 3, 11, \infty)$.*

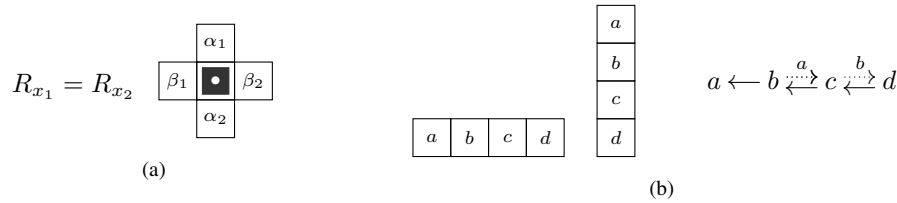


Figure 15: (a) A $(2, 4)$ priority strategy for the breaker for $P_{4,1}$ (b) Dependency digraph of the cells in the orientations of $P_{4,1}$.

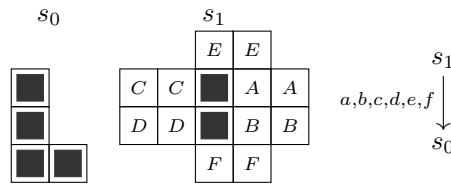


Figure 16: A proof sequence of the maker strategy for $(2, 5)$ -achieving $P_{4,2}$.

Proof: Polyomino $P_{4,1}$ is a subset of the winner L shown in Figure 13a so the maker wins the $(1, 1)$ game and therefore the $(2, 3)$ game by Corollary 3.3. The breaker wins the $(1, 2)$ game since $P_{4,1}$ contains the $(1, 2)$ -loser $P_{3,1}$.

We show that the breaker wins the $(2, 4)$ game following the priority strategy determined by Figure 15a. The breaker uses priorities (α_1, α_2) during her first mark in the response set and priorities (β_1, β_2) during her second mark in the response set.

Figure 15b shows the dependency digraph of the cells of the goal animal in both orientations. It is clear from the digraph that cells b, c and d must be marked in the same turn by the maker to achieve the goal animal. The maker is not able to do so since he only has two marks in a turn. \square

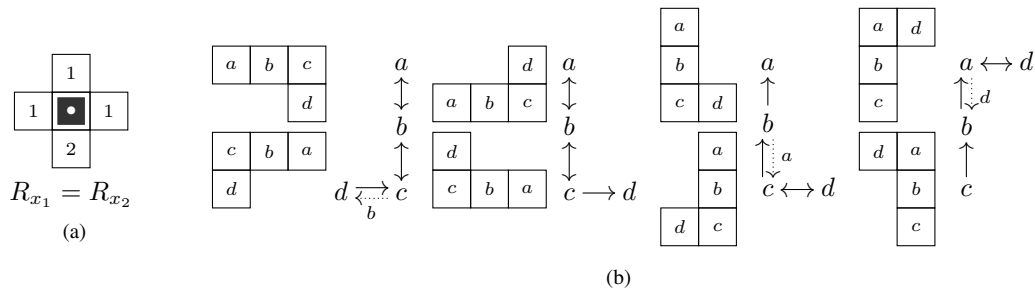


Figure 17: (a) A $(2, 6)$ priority strategy for the breaker for $P_{4,2}$ (b) Dependency digraph of the cells in the orientations of $P_{4,2}$ with secondary arrows omitted.

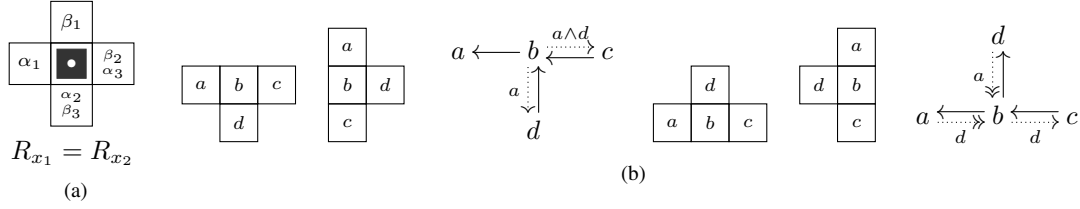


Figure 18: (a) A $(2, 4)$ priority strategy for the breaker for $P_{4,3}$ (b) Dependency digraph of the cells in the orientations of $P_{4,3}$.

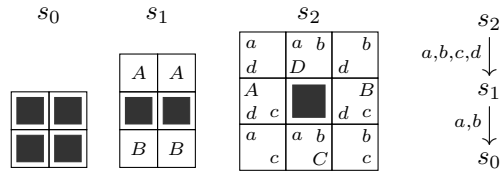


Figure 19: A proof sequence of the maker strategy for $(1 \rightarrow 2, 1)$ -achieving $P_{4,4}$.

Proposition 7.5. *The threshold sequence of $P_{4,2}$ is $\tau(P_{4,2}) = (1, 5, 11, \infty)$.*

Proof: Polyomino $P_{4,2}$ is a subset of the winner L shown in Figure 13a so maker wins the $(1, 1)$ game. The breaker wins the $(1, 2)$ game since $P_{4,2}$ contains the $(1, 2)$ -loser $P_{3,1}$.

The maker wins the $(2, 5)$ game using the proof sequence of Figure 16. It is easy to see that the breaker wins the $(2, 6)$ game following the priority strategy determined by Figure 17a. \square

Note that a somewhat more complicated proof sequence shows that $P_{4,2}$ is a $(1 \rightarrow 2, 2)$ -winner which also implies that $P_{4,2}$ is a $(2, 5)$ -winner.

Proposition 7.6. *The threshold sequence of $P_{4,3}$ is $\tau(P_{4,3}) = (1, 3, 11, \infty)$.*

Proof: Polyomino $P_{4,3}$ is a subset of the winner Y shown in Figure 13a so the maker wins the $(1, 1)$ game and therefore the $(2, 3)$ game. The breaker wins the $(1, 2)$ game since $P_{4,3}$ contains the $(1, 2)$ -loser $P_{3,1}$.

We show that the breaker wins the $(2, 4)$ game following the priority strategy determined by Figure 18a. The breaker uses priorities $(\alpha_1, \alpha_2, \alpha_3)$ during her first mark in a response set and priorities $(\beta_1, \beta_2, \beta_3)$ during her second mark in the response set.

Figure 18b shows the dependency digraph of the cells of the goal animal in all orientations. In the first two orientations, we have $b \sim c \sim d$ so cells b, c and d must be marked in the same turn by the maker to achieve the goal animal. In the last two orientations, either cells a, b and c or cells d, b and c must be marked in the same turn as explained in Example 4.7. The maker is not able to do so since he only has two marks in a turn. \square

Proposition 7.7. *The threshold sequence of $P_{4,4}$ is $\tau(P_{4,4}) = (0, 3, 5, \infty)$.*

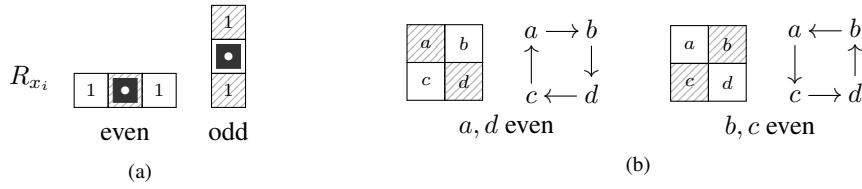


Figure 20: (a) A (2, 4) and (3, 6) priority strategy for the breaker for $P_{4,4}$. The priorities depend on the parity of the current maker mark. Even cells are shaded. (b) Dependency digraphs of the cells in the placements of $P_{4,4}$.

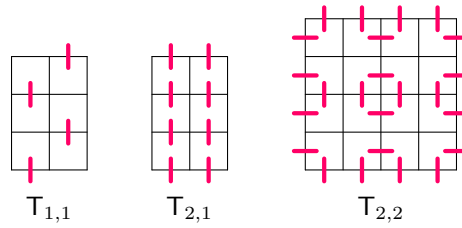


Figure 21: Rectangular pavings.

Proof: Polyomino $P_{4,4}$ is not a subset of any of the winners shown in Figure 13a so it is a (1, 1)-loser. Actually, the breaker wins using the paving strategy based on paving $T_{1,1}$ shown in Figure 21.

The maker wins the (1→2, 1) game using the proof sequence in Figure 19. So by Corollary 3.3 the maker wins the (2, 3) game. The maker also wins the (1→3, 1) and therefore the (3, 5) game since the (1→3, 1) game is easier for the maker than the (1→2, 1) game.

We show that the breaker wins the (2, 4) game and the (3, 6) game following the priority strategy determined by Figure 20a. We say that a cell on the board with coordinates (x, y) is even if $x + y$ is even. Otherwise the cell is called odd. The parities of the cells therefore form a checkerboard pattern. The priorities in the breaker strategy depend on the parity of the current cell marked by the maker. Figure 20b shows the dependency digraph of the cells of the goal animal. It is clear from the digraph that all four cells have to be marked in a single turn. The maker is not able to do so since he has fewer than four marks in a turn in both the (2, 4) and (3, 6) games. □

Proposition 7.8. *The threshold sequence of $P_{4,5}$ is $\tau(P_{4,5}) = (1, c, 11, \infty)$ where $c \geq 3$.*

Proof: Polyomino $P_{4,5}$ is a subset of the winner Z shown in Figure 13a so the maker wins the (1, 1) game and therefore the (2, 3) game. The breaker wins the (1, 2) game since $P_{4,5}$ contains the (1, 2)-loser $P_{3,2}$. □

It remains to show that $P_{4,5}$ is a (2, 4)-loser. For this we need a more complicated priority strategy for the breaker.

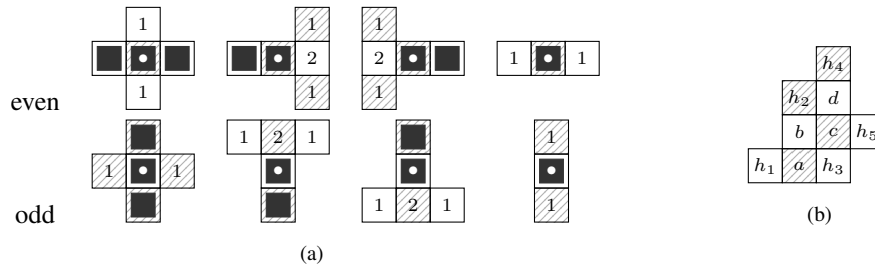


Figure 22: (a) A $(2, 4)$ history dependent priority breaker strategy for $P_{4,5}$. The priorities depend on the parity of the current maker mark. Even cells are shaded. (b) The goal cells $\{a, b, c, d\}$ and the considered history cells $\{h_1, \dots, h_5\}$.

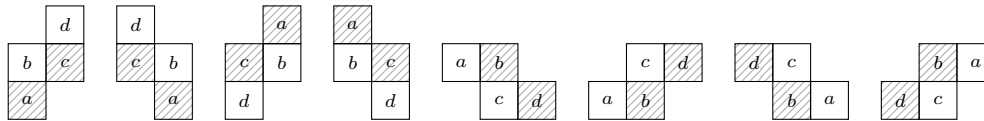


Figure 23: The eight different placements of $P_{4,5}$ on a checker board.

8 The history dependent priority strategy

We introduce a priority strategy where the priorities of the cells depend on the state of certain *history cells*. If some of the history cells are already marked by the maker, then the breaker uses different priorities. The priorities do not change if some the history cells are marked by the breaker. Figure 22a shows a history dependent priority strategy. The priorities depend on the parity of the current maker mark. The first row shows the priorities if this parity is even while the second row shows the priorities if the parity is odd. Both of these rows contain four rules. The breaker uses the first available rule for which the history cells have the correct state.

For example if the current maker mark is even, then the breaker uses the rules in the first row. The first rule requires that the history cells located on left and on the right of the current cell are both marked by the maker. If the condition is satisfied, then the breaker uses this first rule. If any of the history cells are unmarked or marked by the breaker, then the breaker jumps to the next rule. This second rule only requires that the history cell on the left of the current mark is marked by the maker. It is clear that the requirement for one of the rules is always satisfied. Note that the conditions for the second and the third rules are never satisfied together. Swapping these two rules in either row has no effect on the strategy.

Proposition 8.1. *Polyomino $P_{4,5}$ is a $(2, 4)$ -loser and so $\tau(P_{4,5}) = (1, 3, 11, \infty)$.*

Proof: We used a C++ computer program that implements Algorithms 1 and 2 to verify that the breaker wins using the strategy of Figure 22a. The program finished the calculation in less than a second on a Linux machine with a 3.33 GHz CPU.

The algorithm checks that the maker cannot mark all the cells of the goal animal in any placement on the board no matter what order he tries to mark the goal and history cells. First we partition the maker

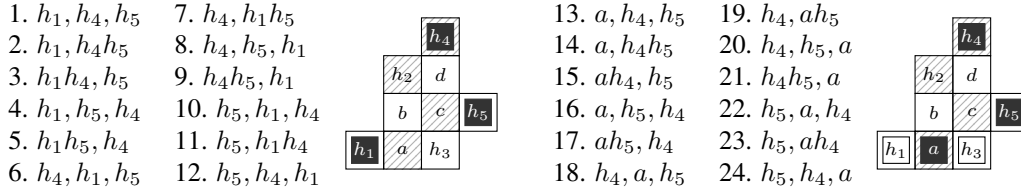


Figure 24: Terminal positions.

marks into classes such that every class contains one or two cells. Then we consider each permutation of the classes. The one-element classes represent turns where the maker used his second mark somewhere far away on the board. The two-element classes represent turns where the maker marks two cells and both of these cells are relevant to the position.

During the analysis of a specific permutation of the maker marks, we try to add the maker marks in the given order. We expect that this process eventually fails and we are not going to be able to mark all the goal cells. If the process succeeds, then we conclude that the history dependent priority strategy for the breaker fails.

We only add a maker mark on a history cell if at least one of the corresponding defensive move sets is not in contradiction with the position and the order of moves, as explained in line (14) of Algorithm 2. We do not add any defensive moves for a maker mark on a history cell. The missing breaker marks do not hurt the chances of the maker. For a maker mark on a goal cell, the defensive moves are determined since the mark order of the relevant history cells is determined by the permutation. We only add maker marks on goal cells if the defensive moves for these maker marks do not ruin the position for the maker. We call a position terminal if no subsequent set of maker marks can be added because the corresponding breaker marks would ruin the position.

The breaker strategy is invariant with respect to parity preserving horizontal and vertical reflections and parity changing rotations by 90 degrees. This implies that there is essentially one placement of the goal animal shown in Figure 22b that we need to consider. The labeling of the cells in Figure 23 shows how the 8 placements are isomorphic.

Our program produces 2 different terminal positions during the search shown in Figure 24. Every permutation fails after 2 or 3 turns because the breaker can spoil the position in the third or fourth turn. So we only show the beginning turns of the permutations. For example, in case 9 the maker tries to mark cells h_4 and h_5 during the first turn and then cell h_1 and another irrelevant cell during the second turn. This attempt results in a terminal position. \square

Algorithm 1. Analyze strategy

```

1.   for each  $G$  placement of the goal animal
2.       initialize the set  $H$  of history cells
3.        $E := G \cup H$ 
4.       push  $(E, M := \{\}, B := \{\})$  to  $Positions$ 
5.       while  $Positions$  is not empty
6.           pop  $(E, M, B)$  from  $Positions$ 
7.           if  $|G \cap E| \leq 2$ 
8.               breaker strategy fails, stop
9.           for each  $F \subseteq E$  such that  $|F| \leq 2$ 
10.               $\tilde{E} := E \setminus F$ 
11.               $\tilde{M} := M \cup F$ 
12.               $nPositions := \text{AddBreaker}(G, \tilde{E}, \tilde{M}, B, F)$ 
13.              push  $nPositions$  to  $Positions$ 
14.   strategy works

```

1. We need to verify that none of the placements of the goal animal can be marked by the maker in any order.
2. We collect the possible history cells in H . For an even goal cell we add the cell on the left and on the right of the goal cell. For an odd goal cell we add the cell above and below the goal cell.
3. The set E of empty cells contains the goal and the history cells. Although the maker only needs to mark the goal cells, marking the history cells as well may affect his success. So we test every permutation of the empty cells.
4. Variable $Positions$ is a stack that contains the set of game positions that are promising for the maker. A position is determined by the set E of empty cells, the set M of cells marked by the maker and the set B of cells marked by the breaker. At the beginning we only have one position in which every cell is empty.
5. The analysis continues while there are any promising positions left to consider.
6. We take one of the promising positions for further consideration.
7. If the position has fewer than 3 unmarked goal cells, then the maker can win since he is allowed to mark 2 cells.
8. This means the breaker strategy might fail, so we stop. Note that the breaker strategy might actually work but require a more sophisticated analysis.
9. We consider each possible one and two-element class F of the set of remaining empty cells E . Set F contains the cells that the maker is about to mark. We consider one-element classes since the maker may use one of his marks somewhere else on the playing board.
10. We remove the current maker marks from the set of empty cells.

11. We add the current maker marks to the set of maker marks.
12. We call the function in Algorithm 2 to add the defensive breaker marks corresponding to the current maker marks. The function returns a possibly empty set of new positions.
13. We add the new positions to the stack of positions.
14. We run out of positions promising for the maker. This means the breaker strategy worked.

Algorithm 2. AddBreaker(G, E, M, B, F)

```

1.  push ( $E, M, B$ ) to  $Positions$ 
2.  for each  $f \in F$ 
3.      if  $Positions$  is not empty
4.          if  $f \in G$ 
5.              pop ( $E, M, B$ ) from  $Positions$ 
6.              find rule  $R$  matching cell  $f$  in position ( $E, M, B$ )
7.              if  $R.breaker \cap (G \setminus M) = \emptyset$ 
8.                   $B := B \cup R.breaker$ 
9.                   $E := E \setminus R.breaker$ 
10.                 push ( $E, M, B$ ) to  $Positions$ 
11.             else
12.                 pop ( $E, M, B$ ) from  $Positions$ 
13.                 for each rule  $R$  matching cell  $f$ 
14.                     if  $R.history \cap E = \emptyset$  and  $R.breaker \cap (G \setminus M) = \emptyset$ 
15.                         push ( $E, M, B$ ) to  $Positions$ 
16.                     exit loop
17.  return  $Positions$ 

```

1. We fill the local variable $Positions$ with the unfinished position missing the defensive breaker marks.
2. We add defensive moves to every current maker mark.
3. The position may be ruined for the maker after the first set of defensive moves. In this case we do not need to try to add more defensive moves.
4. The defensive moves are handled differently for goal cells and history cells that are not goal cells. First we handle the goal cells.
5. We consider the unfinished position (E, M, B) and remove it from $Positions$.
6. We find the rule that matches the current cell f and the position. The defensive breaker moves are uniquely determined by this rule since the current cell is a goal cell and we consider every possible marking order of the relevant history cells.
7. If any of the breaker marks determined by the matching rule are amongst the unmarked goal cells, then the position is no longer promising for the maker. In this case $Positions$ is left empty.

8. We update the set of breaker marks with the current defensive marks.
9. We remove the current defensive marks from the set of empty cells. These cells are already marked by the breaker so the maker is not able to mark them in a later turn.
10. We store the position with the new defensive moves in *Positions*.
11. Now we handle the case when the current maker mark is a history cell.
12. We consider the unfinished position (E, M, B) and remove it from *Positions*.
13. Since we do not consider the history cells for the current maker mark, the defensive breaker marks are not uniquely determined. Hence we need to consider every possible rule that does not contradict the position.
14. A rule can contradict the position in two ways. A history cell of the rule that is required to be marked by the maker cannot be in E because the cells of E are scheduled to be marked by the maker at a later time. A response breaker move cannot be an unmarked goal cell since that ruins the position for the maker.
15. We store the position in *Positions*. We do not add any defensive breaker marks.
16. No more rules need to be considered since we already found a matching one.
17. We return the set containing the position updated with breaker marks or an empty set if the breaker marks ruined the position for the maker.

Example 8.2. Figure 25 shows why a move sequence starting with h_1, h_5, ac, \dots fails to achieve $P_{4,5}$. Since h_1 is not a goal cell, we consider all possible breaker responses. Two rules match the positions as shown in step 1.a. Since we have found a matching rule, the analysis continues at step 1.b. We do not add the defensive moves. Cell h_5 is again a history cell with two rules matching the position as shown in step 2.a. The analysis continues at step 2.b without any of the defensive moves. Note that h_1 and h_5 are considered alone, which means the maker places the corresponding second mark far away. The maker now tries to place cells a and c in a single turn. The defensive breaker marks are now determined because a and c are goal cells. The breaker now marks cell b that is a priority 2 cell. This ruins the position for the maker. Note that the move sequence h_1, h_5 does not produce a terminal position. It can be extended to h_1, h_5, h_4 shown as case 4 in Figure 24.

More sophisticated versions of our algorithm may be needed for checking more complicated history dependent priority strategies. One possibility is to include the defensive breaker marks for maker marks on history cells. These marks are not unique so we need to include all possibilities which result in a much larger search tree. Another possibility is to include n levels of history cells. The level 1 history cells are our usual history cells required for the goal cells. Level $k + 1$ history cells are induced by the level k history cells. During the analysis, the level n history cells would not produce breaker marks, but we would include the breaker response cells for the other history cells. Including more levels has a greater chance of success but it is more computationally demanding.

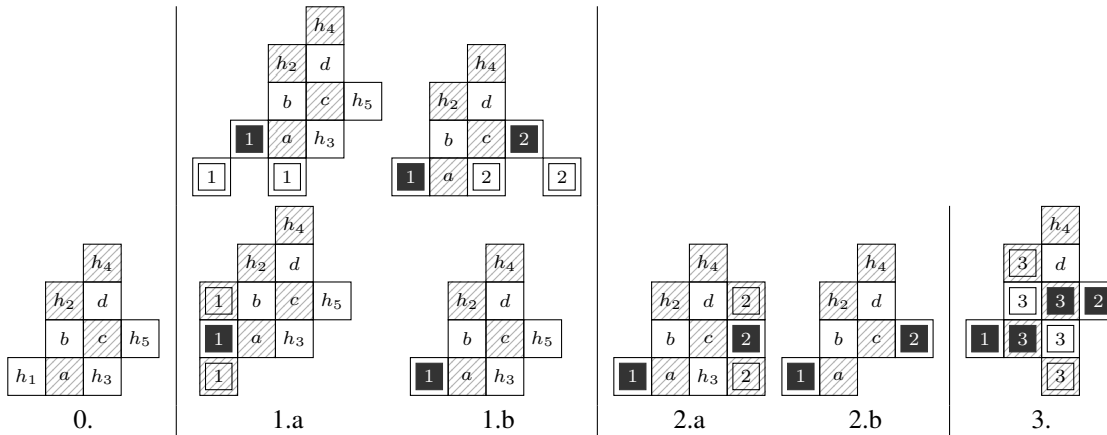


Figure 25: A failing attempt of the maker to use a move sequence starting with the moves h_1, h_5, ac, \dots in a placement of the goal animal.

9 Further directions

We list a few unanswered questions related to biased achievement games.

1. Polyhex achievement games are studied in [3, 20, 22]. What are the threshold sequences of small polyhexes?
2. Polycube achievement games are studied in [12, 16, 27]. What are the threshold sequences of small polycubes? Adding a dimension is a big advantage for the maker so the values in the two dimensional threshold sequences are lower bounds for three dimensional values.
3. There are some results about achievement games played on n -dimensional polycubes [12, 23]. How does the threshold sequence of a given polyomino change if the game is played on higher dimensional rectangular boards?
4. Biased animal set $(1, 2)$ games are studied in [5, 7]. In this version the maker wins if he marks any of the animals in a given goal set of animals. What can we say about the threshold sequences of goal sets of animals?
5. There are only finitely many $(1, 1)$ -winners in any animal achievement game [23]. Are there finitely many (a, b) -winners for a fixed a and b ? The answer is most likely yes. If the answer is in fact yes, what is the upper bound?
6. Are there two animals A and B with threshold sequences (a_1, a_2, \dots) and (b_1, b_2, \dots) respectively such that $a_i < b_i$ but $a_j > b_j$ for some i and j ?
7. What is the spectrum of the possible threshold sequences? For each threshold sequence (a_1, a_2, \dots) in Figures 5 and 14, i divides $a_i + 1$ for all i . Is this true for all threshold sequences? One interpretation of this property is that the (a, \tilde{b}) game is just as hard for the breaker as the (a, b) game if $\lceil \tilde{b}/a \rceil \leq \lfloor b/a \rfloor$. This seems reasonable considering Theorem 3.1.

8. In a handicap c game, the maker is allowed to mark c cells in the first turn and then play the usual $(1, 1)$ game [15]. It is known [11, 21, 26] that Snaky is a $(1, 1)$ -winner with handicap 1. Is there a connection between a handicap c game and a $(1 \rightarrow c, 1)$ game? Which game is easier for the maker?
9. Is there a way to use a dependency digraph to verify history dependent priority strategies? It seems likely that the history cells should be included in the digraph. The main difficulty is that the digraph usually does not have any unconditional arrows, only conditional and secondary arrows.
10. Although Snaky is conjectured to be a winner, it might actually be a loser [6, 17, 18]. Checking priority strategies by computers is a lot easier than finding winning strategies. So a systematic search for a history dependent priority strategy for the breaker using a multilevel version of our checking algorithm might not be hopeless.
11. We could slightly improve Algorithm 2. Currently, we do not add any defensive breaker moves for history cells. We only check (line 14) that at least one configuration of the earlier maker marks results in a set of breaker marks that does not ruin the position. If there is only one such configuration, then the defensive moves could be added together with the earlier maker marks that force this defensive response. The addition of these marks would increase the chances of a successful verification of the breaker strategy. It could also decrease the branching factor of the backtracking search which would make the search faster.
12. Polyominoes L , Y and Z are all $(1, 1)$ -winners but finding a proof sequence is relatively easy for Z and is quite challenging for L . This intuition is strengthened by the lengths of the known proof sequences for these animals. Can we use the threshold sequences to firmly confirm that Z is the easiest and L is the hardest five-cell animal to achieve?

References

- [1] József Beck, *Combinatorial games*, Encyclopedia of Mathematics and its Applications, vol. 114, Cambridge University Press, Cambridge, 2008, Tic-tac-toe theory.
- [2] Elwyn R. Berlekamp, John H. Conway, and Richard K. Guy, *Winning ways for your mathematical plays. Vol. 3*, second ed., A K Peters Ltd., Natick, MA, 2003.
- [3] Jens-P. Bode and Heiko Harborth, *Hexagonal polyomino achievement*, Discrete Math. **212** (2000), no. 1-2, 5–18, Graph theory (Dörnfeld, 1997).
- [4] ———, *Triangular mosaic polyomino achievement*, Proceedings of the Thirty-first Southeastern International Conference on Combinatorics, Graph Theory and Computing (Boca Raton, FL, 2000), vol. 144, 2000, pp. 143–152.
- [5] ———, *Triangle polyomino set achievement*, Proceedings of the Thirty-second Southeastern International Conference on Combinatorics, Graph Theory and Computing (Baton Rouge, LA, 2001), vol. 148, 2001, pp. 97–101.
- [6] Martin Csernenszky and András Pluhár, *On the complexity of chooser-picker games*, (preprint).

- [7] Edgar Fisher and Nándor Sieben, *Rectangular polyomino set weak $(1, 2)$ -achievement games*, Theoret. Comput. Sci. **409** (2008), no. 3, 333–340.
- [8] Gábor Fülep and Nándor Sieben, *Polyiamonds and polyhexes with minimum site-perimeter and achievement games*, Electron. J. Combin. **17** (2010), no. 1, Research Paper 65, 14. MR 2644851
- [9] Martin Gardner, *Mathematical games*, Sci. Amer. **240** (1979), 18–26.
- [10] Solomon G. Golomb, *Polyominoes: Puzzles, patterns, problem and packings*, Princeton University Press, 1965.
- [11] Immanuel Halupczok and Jan-Christoph Schlage-Puchta, *Achieving snaky*, Integers **7** (2007), G2, 28 pp. (electronic).
- [12] _____, *Some strategies for higher dimensional animal achievement games*, Discrete Math. **308** (2008), no. 16, 3470–3478.
- [13] Frank Harary, *Achievement and avoidance games for graphs*, Graph theory (Cambridge, 1981), Ann. Discrete Math., vol. 13, North-Holland, Amsterdam, 1982, pp. 111–119.
- [14] _____, *Is Snaky a winner?*, Geombinatorics **2** (1993), no. 4, 79–82.
- [15] Frank Harary, Heiko Harborth, and Markus Seemann, *Handicap achievement for polyominoes*, Proceedings of the Thirty-first Southeastern International Conference on Combinatorics, Graph Theory and Computing (Boca Raton, FL, 2000), vol. 145, 2000, pp. 65–80.
- [16] Frank Harary and M. Weisbach, *Polycube achievement games*, J. Recreational Math. **15** (1982–83), 241–246.
- [17] Heiko Harborth and Markus Seemann, *Snaky is an edge-to-edge looser*, Geombinatorics **5** (1996), no. 4, 132–136.
- [18] _____, *Snaky is a paving winner*, Bull. Inst. Combin. Appl. **19** (1997), 71–78.
- [19] _____, *Handicap achievement for squares*, J. Combin. Math. Combin. Comput. **46** (2003), 47–52, 15th MCCC (Las Vegas, NV, 2001).
- [20] Kazumine Inagaki and Akihiro Matsuura, *Winning strategies for hexagonal polyomino achievement*, Proceedings of the 12th WSEAS International Conference on Applied Mathematics (Stevens Point, Wisconsin, USA), World Scientific and Engineering Academy and Society (WSEAS), 2007, pp. 252–259.
- [21] Hiro Ito and Hiromitsu Miyagawa, *Snaky is a winner with one handicap*, 8th Hellenic European Conference on Computer Mathematics and its Applications (2007).
- [22] Nándor Sieben, *Hexagonal polyomino weak $(1, 2)$ -achievement games*, Acta Cybernet. **16** (2004), no. 4, 579–585.
- [23] _____, *Snaky is a 41-dimensional winner*, Integers **4** (2004), G5, 6 p. (electronic).

- [24] ———, *Wild polyomino weak $(1, 2)$ -achievement games.*, Geombinatorics **13(4)** (2004), 180–185.
- [25] ———, *Polyominoes with minimum site-perimeter and full set achievement games*, European Journal of Combinatorics **29** (2008), 108–117.
- [26] ———, *Proof trees for weak achievement games*, Integers **8** (2008), G07, 18.
- [27] Nándor Sieben and Elaina Deabay, *Polyomino weak achievement games on 3-dimensional rectangular boards*, Discrete Mathematics **290** (2005), 61–78.