Tetris Is Not Competitive

Matthias Gehnen
RWTH Aachen University, Germany

Luca Venier
RWTH Aachen University, Germany

Abstract

In the video game Tetris, a player has to decide how to place pieces on a board that are revealed by the game one after another. We show that the missing information about the upcoming pieces is indeed crucial to a player's success. We present a construction for piece sequences that force (online) players without or with a finite preview of upcoming pieces to lose while (offline) players who know the entire piece sequence can clear the board and continue to play.

From a competitive analysis perspective, it follows that there cannot be any c-competitive online algorithm for various optimization goals in the context of playing Tetris. Furthermore, we improve existing results by providing a construction for piece sequences which force every player to lose for every possible board size with at least two columns. With this construction, we are also able to show that an instance with just 435 pieces is sufficient to force every player to lose on a standard-size board with ten columns and twenty rows.

2012 ACM Subject Classification Theory of computation → Online algorithms

Keywords and phrases Online Algorithms, Tetris

Digital Object Identifier 10.4230/LIPIcs.FUN.2024.16

Acknowledgements This article bases on Luca Venier's Mastes's Thesis [16].

1 Introduction

Tetris is a classical video game in which four-celled pieces (, , , , , ,) fall onto a rectangular board. While a piece drops from the top, the player can rotate and move it horizontally until it lands on a filled cell or the bottom of the board. Whenever every cell of a row is occupied by pieces it disappears (lineclear) and all filled cells above drop down by one row. The player continues to play as long as pieces can be placed on the board and loses if a piece cannot be placed on the board.

Apart from its fame as a video game, the underlying concept also leads to interesting combinatorial questions, like: Given a board, and a sequence of pieces to place, is there a way to place the pieces without losing? Are there piece sequences that always lead to a loss or always have a loss-avoiding strategy? These questions have been studied in the past, often with the assumption that the player knows all upcoming pieces. However, in the classical video game, the player needs to make irrevocable decisions with just a few pieces of lookahead. Therefore, it seems reasonable to consider the underlying combinatorial concept as an online-problem.

For standard-size boards (ten columns and twenty rows) Brzustowski [6] gives an instance that forces an online player with a lookahead of just one piece to lose. Later Burgiel [7] showed that there is no chance to play for more than 69600 pieces before losing in an instance with alternating and pieces, even if the entire instance is known in advance.

If an instance forces an offline player to lose, then an online player with less information about the piece sequence is naturally also forced to lose. However, the opposite case is not trivial. If there is an instance that forces an online player with a finite lookahead to lose, must an offline player who knows the entire instance also lose?

12th International Conference on Fun with Algorithms (FUN 2024).
Editors: Andrei Z. Broder and Tami Tamir; Article No. 16; pp. 16:1–16:16
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1.1 Related Work

After its development in the mid-1980s by Pajitnov, Pavlovsky, and Gerasimov [10], the history of analyzing the underlying theoretic concepts of Tetris started in 1992 with the already mentioned master's thesis by Brzustowski. Apart from showing that there are unwinnable instances for general Tetris, Brzustowski presents some winning strategies for restricted variants with only one or two different piece types. Even if there cannot be a general loss-avoiding strategy for classical Tetris, it is known that such strategies exist for some restricted piece-sets even on non-rectangular boards [15]. If the pieces of an instance cannot be chosen arbitrarily but have to be presented in a sequence of permutations of the seven different piece types (the so-called 7-bag randomizer) then some commercial game variants have loss-avoiding strategies [1]. For an overview of implemented piece selection algorithms see [12]. Here the different selection process has a crucial role since it is known that some specific sequences lead to a loss. If randomly selecting the next piece type determines the piece sequence then such a loss-forcing sequence appears for sure at some point in a resulting infinite random string. Burgiel therefore follows that every Tetris game (implying a random generation of the piece sequence) must end at some point [7].

Since it is known that the player has to lose for some piece sequences, it is natural to ask for the complexity to decide if a given instance is such a sequence. Demaine et al. found out in 2003 that this is indeed an NP-hard problem, and related optimization problems similar to the ones we investigate in Section 3.1 also cannot be approximated efficiently [9]. Later a simpler reduction was given [5, 4], and some modifications are shown to not be easier [2, 14].

For further related work about Tetris, also to non-theoretical results, we refer to the vast related work section of Dallant and Iacono's article [8].

In most of the mentioned results about Tetris, an algorithm or strategy is given the entire instance which must be solved in advance. In contrast, a Tetris player like an online algorithm has no advanced knowledge about the instance it needs to solve. Whenever a new element of the instance is given some irrevocable decision must be taken before the next piece is revealed. An online algorithm tries to optimize an objective function that is dependent on the solution set formed by its decisions. The *strict competitive ratio* of an algorithm, as defined by Sleator and Tarjan [13], is the worst-case ratio of the performance of an algorithm compared to that of an optimal solution computed by an offline algorithm for the given instance, over all instances. The competitive ratio of an online *problem* is then the best competitive ratio over all online algorithms. For a general introduction to online problems, we refer to the books of Borodin and Ran El-Yaniv [3] and of Komm [11].

1.2 Preliminaries and Notation

An instance of a Tetris game in the algorithmic sense consists of a board and a finite sequence of pieces (as piece types) that are played. The board has a width w, a height h, and consists of $w \times h$ many cells that are either filled or empty. We call each resulting pattern of filled and unfilled cells a board configuration. Each instance starts with an empty board configuration where all cells are empty. We divide the board in $\frac{w}{2}$ pairs of adjacent columns, which we call lanes like in [6].

The player immediately loses if a piece is placed in such a way that a cell outside the boundaries of the board would be filled. The player wins if each piece of the piece sequence has been successfully placed inside the board. A player can rotate a piece and select the column it is supposed to fall in. We call rotations of pieces where the piece is taller than wide (e.g.,  ,  , ...) vertical and the opposite rotations (e.g.,  ,  , ...) horizontal. Compared to implemented games and some mathematical models the player cannot manipulate the piece while it is falling as the additional complexity of the model is not needed for the claims made in this work.

In an online setting a player has a lookahead $l \in \mathbb{N}$. The game proceeds with the player placing the i-th piece of the piece sequence, and then the adversary chooses and appends the $i + l + 1$-th piece type to the piece sequence. In an offline setting the entire piece sequence is known before the game begins.

For some objective function, the ratio between the performance of the best possible online strategy and the optimal offline solution is called *(strict) competitive ratio*, usually denoted with c. Sometimes adding a constant α is sensible, making the analysis non-strict. Therefore, we call an online problem c-competitive if there is an online strategy that is not worse than $c{\cdot}\mathrm{OPT}{+}\alpha$ than an optimal solution. Here OPT denotes the quality of the optimal solution with respect to the objective function [3, 11].

The following sections use state diagrams where nodes are board configurations to depict game instances similarly to [6]. Transitions have labels depicting which piece types is presented to the player. Furthermore, transitions can have an optional index (e.g., , 2) to indicate that the player places the piece with its leftmost cells in the 0-indexed column. An optional multiplier (e.g., $2 \times$) denotes that the same piece type is presented multiple times.

1.3 Results

In Section 2 we construct instances without any loss-avoiding strategies for players with arbitrary (but finite) lookahead, but for which loss-avoiding strategies exist if the full instance is known to the player. A common question for online-problems is to ask how much worse an online algorithm (in our case the player with a finite preview of upcoming pieces) will perform on the same instance due to the lack of information.

In Section 3.1 the aforementioned construction is used to show that there are instances where the online player performs arbitrarily badly against an offline player for several optimization goals. Hence, for these goals, there cannot be any c-competitive online algorithms for any c. In Section 3.2 we use parts of the construction to improve the upper bound for instances that force offline players to lose from 69600 down to 435 pieces on a standard-size board. This adversarial strategy is also used in Section 3.3 to show that such loss-forcing instances exist for every board that is at least two columns wide, no matter its size or the length of the player's preview.

In Section 4 we give strictly 1-competitive online algorithms for some optimization goals. For other optimization goals, we give instances where competitiveness cannot be strict. Interestingly online players cannot play optimally for some optimization goals such as clearing as many lines as possible without the necessary lookahead, even in instances with just a single piece type.

2 Construction

This section shows a way to construct sequences in which online players necessarily lose, while offline players end with an empty board, and can continue to play. The construction works for even height and width ($w \geq 4$) of a rectangular board and is divided into the following three parts.

First, a sequence of and pieces is presented to ensure that the board is almost filled. This technique is similar to the one used by Brzustowski and Brugiel and described in Section 2.1. Second, given an almost filled board, we present two strategies an adversary can use to make online players lose, if they only have finite lookahead. Finally, we will see that

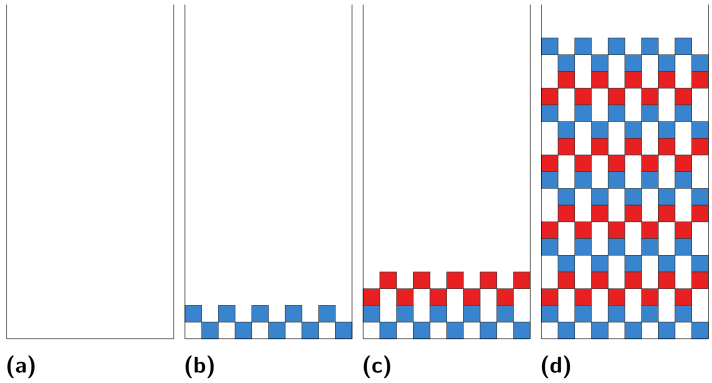


Figure 1 After the first iteration of the padding sequence a wave has been created (b). After another iteration a second wave has been placed on top (c). Repeating the procedure $\frac{h}{2} - 1$ times leads to a padded board (d).

the offline player (who chose the other option and therefore has not lost yet) can also clear the whole board. This ensures that the loss of an offline player is not just delayed by a few pieces.

2.1 Padding Sequence

In this part, we will see that playing with a sequence of and pieces leads to a board configuration, containing *waves* alternately.

▶ **Definition 1** (Wave). *A wave is a pattern of cells consisting of two rows where every second cell is filled horizontally and vertically.*

▶ **Example 2** (Wave). Placing a piece vertically in each lane of a 10 columns wide board gives a board configuration wave as depicted in Figure 1b. Repeating the same with pieces stacks a wave on top of the existing one as depicted in Figure 1c.

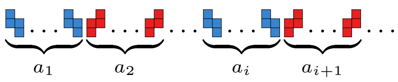
Two waves of the same orientation cannot be stacked on top of each other, as the resulting lineclears would remove one of them. Stacking several waves on each other ultimately leads to a padded board.

▶ **Definition 3** (Padded Board). *A padded board is an even-width even-height board configuration which consists of $\frac{h}{2} - 1$ stacked waves (of alternating orientation) and where the two topmost rows are completely empty.*

▶ **Example 4** (Padded Board). A padded standard-size board can be seen in Figure 1d.

Brzustowski [6] and Burgiel [7] use a similar approach, by presenting iterations of repeated and pieces as their piece sequence. We call piece sequences that switch between finite subsequences of and pieces SZ-Sequences.

▶ **Definition 5** (SZ-Sequence). *An SZ-Sequence is a piece sequence of alternating and pieces, with the amount of pieces in each iteration given by integers $a_1, ..., a_n$.*



▶ **Example 6** (SZ-Sequence). The sequence of 69.600 alternating and pieces that force an offline player to lose on a standard size board given by [7] can be characterized as an SZ-Sequence with $1 \le i \le 69.600$ and $a_i = 1$.

This section proposes an adapted construction that we call *padding sequence* intending to force the creation of a padded board by stacking waves on an empty board. The goal is that the competitive analysis performed in Section 3.1 is independent of the board size and does only have to regard a constant number of rows on top of the board. Thus, the goal of the padding sequence is not to force a player to lose, nonetheless Section 3.2 shows how it can be expanded to force offline players to lose and gives some results that follow. The following padding sequence is defined only for even-height boards, as this simplifies notation, and the bifurcation sequence presented in Section 2.2 is limited to even-height boards. The key to the following lemma is to prove that, given a padding sequence, a player has no other choice than to create a padded board as they would otherwise lose.

▶ **Lemma 7** (Padding Sequence)**.** *Given an empty even-width even-height board, there exists an SZ-Sequence with a set of integers $a_1, ..., a_{\frac{h}{2}-1} \in \mathbb{N}$ such that, after all pieces have been played, the resulting board configuration is a padded board or the player has lost.*

Proof. The padding sequence consists of $\frac{h}{2} - 1$ iterations, each with a multiple of $\frac{w}{2}$ pieces and enough pieces to force the creation of a wave. Given these constraints, the player has four different options on how to play an iteration. The key of this proof is to show, that any other option than to create a wave (without residual pieces) in each iteration, will lead to a loss of the player.

1. **Playing all pieces vertically in lanes and distributed equally.** Given that the iteration consists of a multiple of $\frac{w}{2}$ pieces, this adds a wave to the board configuration. All following options will add additional unremovable rows to the board configuration and thus lead to a loss of the game.
2. **Playing some pieces horizontally (i.e., rotated as /).** According to [6] the horizontal piece can not be removed by additional or pieces and thus two more rows have been added to the board configuration that cannot be removed. Given that each iteration consists of at least $\frac{w}{2}$ pieces and there can only fit $\frac{w}{2} - 1$ horizontal pieces in two rows, the resulting board configuration after placing pieces horizontally must be at least two rows taller than if Option 1 had been chosen.
3. **Playing some pieces vertically in between lanes.** If a piece has been played vertically in between lanes, the adjacent cells to the right of the piece (or left of the piece) must be filled. If they are not filled, then two additional rows have been created that cannot be removed anymore compared to Option 1. To fill the adjacent cells another in-between piece must be played. This can be repeated until the adjacent cells to be filled reach the outermost column of the board. There they cannot be filled anymore and if enough pieces are played a piece must be placed to irrevocably cover the unfilled cells thus creating two additional unremovable rows compared to Option 1.
4. **Playing all pieces vertically in lanes, but not distributed equally.** The resulting board configuration has a wave and some residual pieces stacked in some lanes. When switching to the following iteration, the player has two options.
   - **(a)** Play all pieces of the following iteration vertically in lanes. For each piece in a lane that is not part of a wave two unremovable rows would be created by the switch from to or vice-versa.
   - **(b)** Play some pieces of the following iteration in-between lanes to avoid multiple pairs of unremovable rows like in Option 4a. However, playing in-between lanes will also lead to additional unremovable rows as described in Option 3 and depicted in Figure 2.

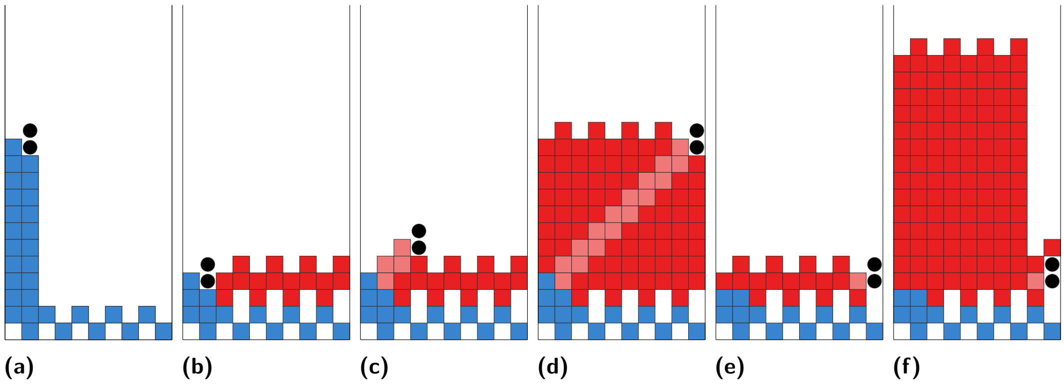


Figure 2 Example that distributing pieces unequally during one iteration of the padding sequence leads to additional unremovable rows. If the player plays the pieces exclusively in lanes, the two black circles will not be filled thus creating two unremovable rows. Also filling them with a piece in-between lanes (marked with lighter shade in (c)) just moves the cells to be filled two columns to the right.
This can be repeated until the marked cells reach the side of the board. Latest here there is no way to fill the two black circles with pieces. This however must happen since the other lanes just allow a finite amount of pieces until they are full, thus creating two more unremovable rows.

Playing all pieces vertically in lanes and distributing them equally creates a wave in each iteration. After $\frac{h}{2} - 1$ many iterations that yields $\frac{h}{2} - 1$ many stacked waves which equals a padded board. If however, the player chooses to deviate from the procedure and place some piece in another way, then at least two additional unremovable rows would be added to the board configuration as detailed in Options 2 - 4. This would lead to a loss of the player as there would not be enough space left on the board to place the remaining pieces of the padding sequence. Hence, given a padding sequence, a player has no other option than to play according to Option 1 and create a padded board. ◀

2.2 Bifurcation Sequence

We use the notion of an adversary to construct a piece sequence where an online player is forced to lose while an offline player can continue to play. Therefore, the piece sequence requires an element where offline and online players make different decisions. If the lookahead of an online player enables them to see a sufficient part of the piece sequence at the time of the decision, they would be able to avoid loss by choosing to play the item the same way as the offline player. Since we want to show that no finite amount of lookahead is sufficient, we construct the piece sequence in a way that the correct decision is dependent on pieces arbitrarily far in the future.

As such we construct two bifurcation sequences, both consisting of three parts: the decision, the consequence, and a looping part in between to push the two further away than the lookahead size. The first sequence can be used for an even width greater or equal than six, the second for a width of four.

▶ **Lemma 8** (Bifurcation Sequence)**.** *Given an even-width ($w \geq 6$) even-height ($h \geq 2$) padded board, then there is a piece sequence that forces an online player to lose while an offline player can continue to play.*

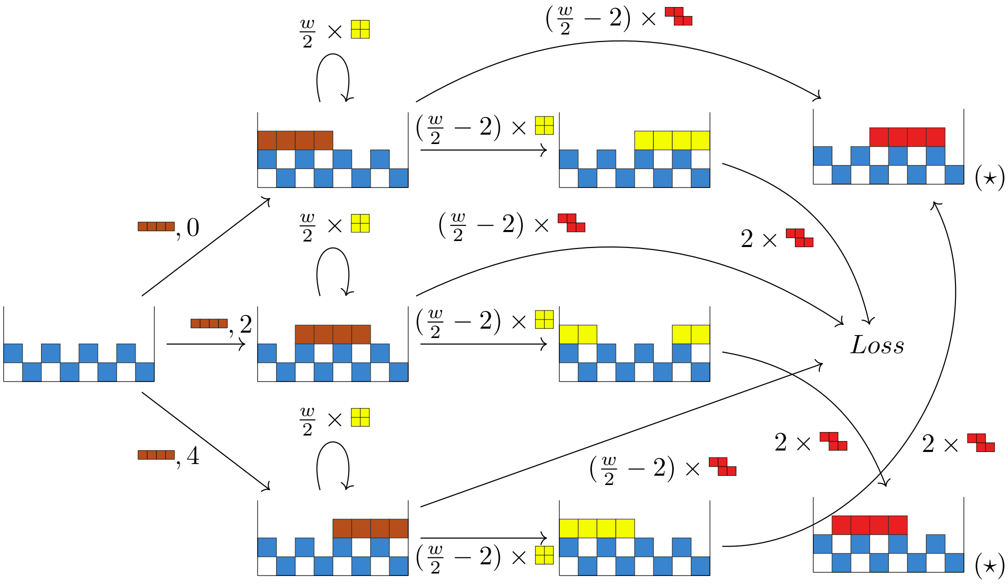
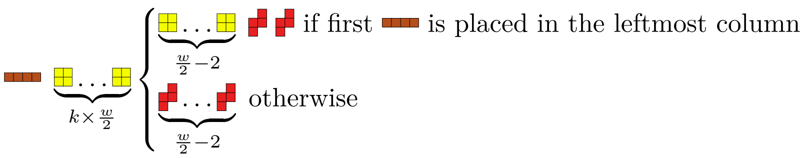


Figure 3 Game graph for bifurcation sequence presented in Lemma 8 with board width $w = 8$. The board configurations marked with $(\star)$ are only reachable for offline players.

Proof. Without loss of generality assume that the topmost wave is induced by pieces.

Given the player has a lookahead of l, we choose an integer k with $k \times \frac{w}{2} > l$ and construct the following piece sequence:



The can only be placed horizontally in every second column such that it covers just two lanes. Otherwise, the following pieces would not fit on the board and the player would lose.

The following set of repeating $\frac{w}{2}$ many pieces always returns to the same board configuration. Once the set has been repeated often enough to surpass the online player's lookahead the adversary can choose the appropriate response to force the online player to lose according to the case distinction above. Hence, the adversary forces the online player to lose with this sequence. An example for a board width of eight is given in Figure 3. As an offline player know the full piece sequence of the entire game beforehand, they can choose the correct way to place the initial piece, as they know the response that will come later. Thus, an offline player is not forced to lose with this sequence and can continue to play. ◀

▶ **Lemma 9.** *Given an even-width ($w \geq 4$) even-height ($h \geq 4$) padded board then there is a piece sequence that forces an online player with arbitrary but finite lookahead to lose while an offline player can continue to play.*

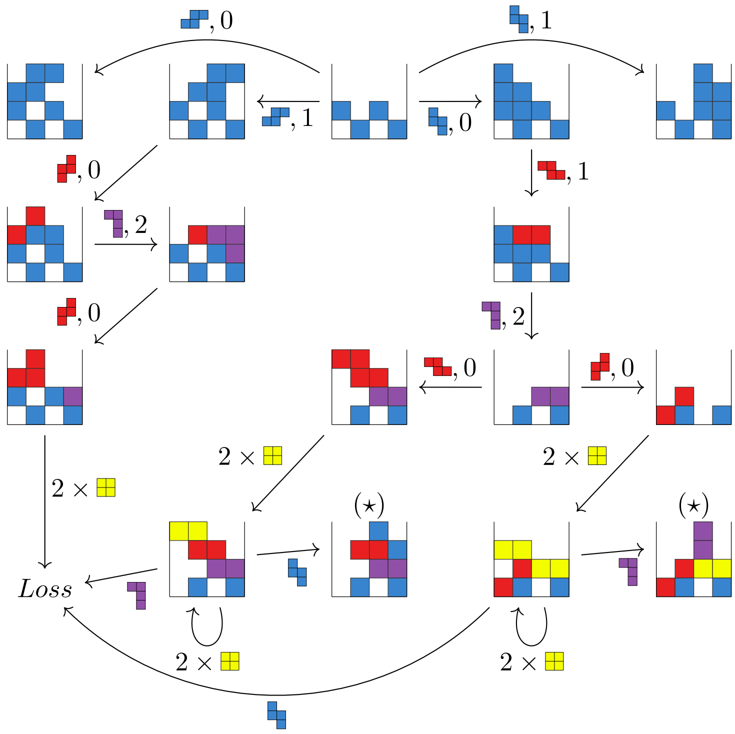


Figure 4 Game graph for bifurcation sequence described in Lemma 9 for boards with $w = 4$, even h, and $h \geq 4$. The board configurations marked with $(\star)$ are only reachable for offline players.

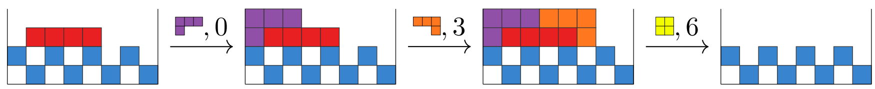
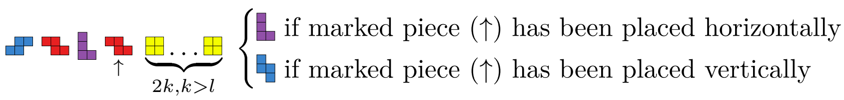


Figure 5 Example of the first part of a clearing sequence for bifurcation sequence presented in Figure 3.

Proof. Without loss of generality assume that the topmost wave is induced by pieces. There is a bifurcation sequence for padded boards with width $w = 4$ consisting of the following pieces.



As can be seen from Figure 4 the online player has to choose how to place the second piece and loses either way. An offline player however will choose the opposite way and therefore reaches one of two possible board states marked with $(\star)$ in Figure 4. ◀

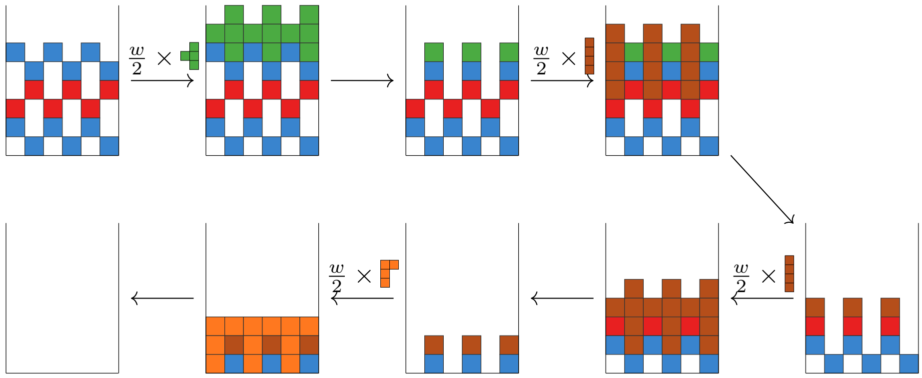


Figure 6 Example of the second part of a clearing sequence for a padded board with width $w = 6$ and height $h = 8$.

2.3 Clearing Sequence

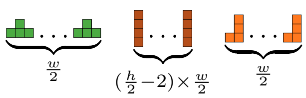
After the bifurcation sequence has been played, the online player must have lost, while there are still some board configurations the offline player can reach without having lost. It is important to show that the loss of the offline player has not been delayed by a finite number of pieces but instead that the offline player can continue to play forever if they are given an appropriate piece sequence. The following lemma shows that every board configuration resulting from the bifurcation sequence can be transformed into an empty board by a piece sequence we call *clearing sequence*.

▶ **Lemma 10** (Clearing Sequence). *There is a piece sequence that can be played after the bifurcation sequence from Lemma 8 or Lemma 9 to clear the board.*

Proof. Given the bifurcation sequence for a board with a width of four there are two board configurations only reachable for offline players (marked with $(\star)$ in Figure 4). For both cases, there is a piece sequence that removes the four topmost rows of the board given by and respectively. This either leaves us with an empty or padded board, which can also be removed as shown below.

If the bifurcation sequence for a board of width six or more was used, there are four filled cells in the penultimate row in (0-indexed) columns $x, x+1, x+2, x+3$ for $x \in \{1, 3, \ldots, w-5\}$. As the four filled cells on the penultimate row always start in an odd column, by placing a and a around it, the remaining space can be filled by $\frac{w}{2} - 3$ many pieces and the induced line clear removes everything from the top two rows. The result is a return to the padded board as can be seen in Figure 5.

Both cases leave us with a padded board, which can be deconstructed as seen in Figure 6 using the following piece sequence:



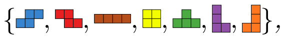
The rotation of the / piece should be chosen based on whether the topmost wave was induced by or pieces. The pieces are played in $\frac{h}{2} - 2$ iterations with each $\frac{w}{2}$ pieces to remove all but the last wave. The last wave is removed by $\frac{w}{2}$ many pieces and the result is an empty board. ◀

3 Consequences

The strategy presented in the previous chapter is useful for different results, that follow immediately as a corollary and are presented in this section. First, online players cannot perform competitively against offline players for different optimization goals. Second, we compute the number of pieces required for the padding sequence on a standard-size board and show that there is a piece sequence that forces any player to lose in just 435 pieces. Third, as the padding sequence applies to any even-width board we can generalize the known results by Brzustowski [6] and Burgiel [7], and show that there exists a piece sequence that forces an offline player to lose on any board that is at least two columns wide.

3.1 Competitive Analysis of Online Tetris

The padding, bifurcation, and clearing sequences of the previous section can be combined to show that an online player with any lookahead of $l \in \mathbb{N}$ pieces will perform arbitrarily badly in comparison to an offline player who knows the entire piece sequence from the beginning of the game. However, for Tetris, it is not immediately clear in which ways the performance is measured. This observation was already made by Demaine et al. when they analyzed approximation algorithms [9]. We are using similar optimization goals to [9] to show that the restriction to finite lookahead is crucial for any strategy. While the first three problems are identical to the ones analyzed by Demaine et al. [9] and known to be NP-hard, the fourth is a variation since we are interested in the highest filled cell at the end and not during the game.

▶ **Definition 11.** *Given an instance I consisting of an initial board configuration and a piece sequence $p_1 p_2 \dots p_n$ with $p_i \in$*  *we define the following four optimization goals:*

1. *Survival Maximization: Number of pieces played in Instance I before losing*
2. *Lineclear Maximization: Number of lineclears performed in Instance I before losing*
3. *Tetris Maximization: Number of tetrises (four lineclears at once) performed in Instance I before losing*
4. *Highest Filled Cell Minimization: Highest row with filled cells at the end of the game*

The first three problems can be understood as a proxy for the decision problem of survival (*Can a player place all given pieces without loosing?*), and the fourth as a proxy for the decision problem of clearing (*Can a player place all piece in such a way that the resulting board configuration only contains empty cells?*). The bifurcation sequences presented in the previous chapter apply to even-width even-height boards with at least four columns. It follows as a corollary from the construction that for each of the four problems there cannot be any c-competitive online algorithm for any possible value of c.

▶ **Theorem 12.** *There cannot be any c-competitive online algorithm for Online Survival Maximization, Online Lineclear Maximization, Online Tetris Maximization, and Highest Filled Cell Minimization for any c for even-width even-height boards with $w \geq 6, h \geq 2$ and $w = 4, h \geq 4$.*

It is easy to see that this result also stays true for the maximization goals when allowing non-strict competitiveness. Note that for Tetris Maximization the height of the board must be at least 4.

Proof. For a given size of the board, construct an instance I that starts with the padding, bifurcation, and clearing instance from the previous chapter. This ensures that the online player loses with some finite amount of played pieces and cleared lines, and without having performed a Tetris.

However, the offline player can continue playing on an empty board, thus the instance continues with the necessary pieces for the optimization goals. The number of lineclears and played pieces for the offline player can easily be increased by filling up the piece sequence with arbitrarily many pieces, which can be played forever without difficulties. Analogously the number of tetrises can be increased arbitrarily by playing pieces if the board has a height of at least four. Since the online player could only achieve a finite amount of played pieces, cleared lines, or performed tetrises, the ratio can get arbitrarily bad.

For Highest Filled Cell Minimization, it is sufficient to see that the online player leaves the game when the full height of the board is covered, while the offline player can decrease the size down to zero with the clearing sequence. ◀

3.2 A computed result for 10×20 boards

As seen in Lemma 7 and previously in the works by Brzustowski [6] and Burgiel [7], playing sequences of and alternately is an efficient way to force a player to fill the board. By adding an iteration to the padding sequence the player can be forced to lose.

▶ **Proposition 13.** *Given a padded board where the topmost wave has been induced by pieces (or), then $\frac{w}{2}$ pieces (or) force the player to lose.*

Proof. The $\frac{w}{2}$ pieces do not fit vertically in lanes on top of a padded board. Horizontally or vertically in-between lanes there is only enough room left for $\frac{w}{2} - 1$ pieces. As the player cannot perform a lineclear by playing the pieces horizontally or vertically in between lanes, the last piece necessarily loses, since the next piece might not fit onto the board anymore. ◀

The goal of this section is to find lower bounds on how many pieces each iteration of the padding sequence must contain. As seen in Lemma 7, there are several ways a player can play the piece of an iteration, but finally at least two unremovable rows will be created. Similar to [6, 7], playing pieces horizontally or vertically in-between lanes immediately creates bumps that cannot be filled with subsequent pieces. This creates additional unremovable rows faster than Option 4 (playing all pieces in lanes but not distributing them equally) in the proof of Lemma 7. Therefore, we restrict the algorithm to play new pieces vertically, and compute how many it takes to create additional unremovable rows if the pieces of the previous iteration have not been distributed equally. If the previous iteration ended with a wave without any deviations, the amount of pieces required for the following iteration is given by completely stacking pieces in all lanes except of one, plus one piece. While it is obvious to see that distributing the pieces equally is a rather fast way to create a wave and the corresponding unremovable rows, it is not so clear which kind of deviation is able to prolong the construction of the wave most. Therefore, Algorithm 1 computes the number of pieces that suffices to force the creation of additional unremovable rows, given any possible way of unequally distributing the pieces in lanes of the previous iteration.

Since it is also possible to play pieces in-between lines like in Figure 2 to avoid new unremovable rows for some finite time, Algorithm 1 tries to prolong this for as many rows (and therefore pieces) as possible. In fact, the result pictured in Example 2 turns out to be the worst case. There, the rightmost piece in picture f is the first which covers the cells (with the black dots) irrevocably. The algorithm avoids covering the cells and instead loses the game, as there is not any other option. Hence, the number of pieces played by the algorithm until loss is the same as the maximum number of pieces that the player can play until they have to have created additional unremovable rows.

Algorithm 1 Compute iteration sizes for padding sequence.

for all iterations $a_2, \ldots, a_{\frac{h}{2}-1}$ **do**
  $result \leftarrow 0$
  for all deviations playable in the previous iteration **do**
    $counter \leftarrow 0$
    for all lanes l without residual pieces **do**
      Play piece in lane l
      $counter \leftarrow counter + 1$
    end for
    repeat
      Play piece in lowest leftmost (for  , rightmost for ) possible placement which does not cover a cell,
      $counter \leftarrow counter + 1$
    until game over
    if $counter > result$ **then**
      $result \leftarrow counter$
    end if
  end for
  ceil result to next multiple of $\frac{w}{2}$
end for

Applying the algorithm to a board with ten columns and twenty rows leads to the results in Table 1. This yields the following result.

▶ **Theorem 14.** *Given an empty* 10×20 *board, there exists an SZ-Sequence with a set of integers* $a_1, \ldots, a_{10} \in \mathbb{N}$ *that forces an offline player to lose that is* 435 *pieces long.*

Proof. The first nine iterations are the padding sequence as described in Lemma 7, and the final iteration in Proposition 13.

- The first iteration requires $\lceil (\frac{w}{2}-1) \times (\frac{h}{2}-1)+1 \rceil_{\frac{w}{2}} = 40$ pieces. The first $(\frac{w}{2}-1) \times (\frac{h}{2}-1)$ pieces may be filled into all but one lane, the next piece must necessarily placed into the last lane to induce one line clear. The number of pieces must be ceiled to the next multiple of $\frac{w}{2}$ to enable the creation of a wave without residual pieces.
- Iterations a_2 to a_9 can be computed with Algorithm 1. See Table 1 for the results.
- In the final iteration $\frac{w}{2} = 5$ force the player to lose according to Proposition 13.

Adding the sizes of each iteration yields a sum of 435 pieces. ◀

The results of 435 pieces to force a player to lose on a standard-size board improves on the previously known upper bound of 69600 pieces by Burgiel [7]. Since ensuring that waves are created is not a necessary requirement for losing, it is plausible that even faster ways exist that do not enforce waves.

3.3 Offline players lose on every board of width at least two

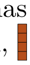
We use the padding sequence with Proposition 13 together with existing results to show that every board that is at least two columns wide has a losing piece sequence. Since we expect every piece to physically fit on the empty board,  is the only allowed piece type for a single-column instance, and a player cannot lose.

Table 1 Results for applying Algorithm 1 on a standard-size board. Deviations are given as a sequence of integers representing the number of pieces played per lane. The number $5, 0, 0, 0, 0$ describes 5 pieces in the first lane and 0 pieces in the following ones. Iteration a_2 is depicted in Figure 2.

Iteration	Best Deviations			Unceiled Result	Ceiled Result
a_2	5,0,0,0,0			69	70
a_3	5,0,0,0,0			65	65
a_4	5,0,0,0,0			61	65
a_5	5,0,0,0,0			57	60
a_6	5,0,0,0,0			53	55
a_7	2,0,3,0,0	2,3,0,0,0		30	30
a_8	2,0,3,0,0			26	30
a_9	1,0,2,2,0	2,1,0,2,0	2,2,1,0,0	13	15

While it is known that there are losing piece sequences for offline players on odd-width boards ($w \geq 3$) [6] and for boards with a width of 2 mod 4 [7], the case for boards with width 0 mod 4 remained open. Brzustowski also proved that online players cannot win on boards of even width [6].

▶ **Theorem 15.** *For any empty board with a width of at least two, there is a piece sequence that forces an offline player to lose.*

Proof. For $w \geq 3, w \mod 2 = 1$ Brzustowski gives loss-forcing piece sequences [6]. Combining the padding sequence (Lemma 7) and Proposition 13 gives a piece sequence that forces any player to lose on even-width boards. ◀

4 Competitiveness of Single Piece Tetris

This section looks at the competitiveness of a constrained version of Tetris. Brzustowski has shown that limiting the piece sequences of the game to a single piece type allows strategies to play forever, at least if the board size allows it [6]. We show that for each optimization goal analyzed in the previous section, there is an online algorithm that is 1-competitive if the piece sequence is constrained to contain only a single piece type. However, for some optimization goals, there are instances that do not allow strict 1-competitive algorithms. The constrained online problems are defined as follows.

▶ **Definition 16** (Single Piece Online Tetris)**.** *The Online Tetris Problems defined in Definition 11 where every piece sequence can only contain elements of the same piece type are prefixed with* SINGLE PIECE.

4.1 Single Piece Survival Maximization and Single Piece Tetris Maximization have strictly optimal online algorithms

The following theorems show that there is a strategy to play optimally for SINGLE PIECE SURVIVAL MAXIMIZATION and SINGLE PIECE TETRIS MAXIMIZATION.

▶ **Theorem 17.** *There is a strictly* 1*-competitive online algorithm for* SINGLE PIECE SURVIVAL MAXIMIZATION.

Proof. If there is no perpetual loss-avoiding strategy for the given board size and piece type, then online and offline players will play such that the highest achievable number of moves is played before losing. If there is a perpetual loss-avoiding strategy both players will place every single piece of the piece sequence without losing. In both cases, $gain_{\text{ALG}}(I) = gain_{\text{OPT}}(I)$ holds, and therefore, the strategy employed by the online player is strictly 1-competitive. ◀

▶ **Theorem 18.** *There is a strictly* 1*-competitive online algorithm for* Single Piece Tetris Maximization.

Proof. If the chosen piece type is not or $h < 4$ then no tetrises can be played and hence $gain_{\text{ALG}}(I) = gain_{\text{OPT}}(I) = 0$. If the chosen piece type is and the board has at least four rows then tetrises can be created by placing pieces in columns next to each other until the last column is filled and a Tetris is performed. This leads to $gain_{\text{ALG}}(I) = gain_{\text{OPT}}(I) = \lfloor \frac{|P|}{w} \rfloor$ with $|P|$ the length of the piece sequence, which cannot be beaten by an offline algorithm. ◀

4.2 Online Players cannot perform optimally for Single Piece Lineclear Maximization and Single Piece Highest Filled Cell Minimization

Compared to the two optimization goals above we can construct instances for Single Piece Lineclear Maximization and Single Piece Highest Filled Cell Minimization where online players are not able to perform optimally. Nonetheless, there are 1-competitive online algorithms for the problems.

The key to creating such instances is that an offline player can choose to play the final piece in such a way that improves the optimization function but would lead to a loss if the piece sequences did not end. As the offline player chooses to deviate from a perpetual loss-avoiding strategy at some point since they know when the instance ends, which an online player can not.

▶ **Theorem 19.** *There cannot be a strictly* 1*-competitive online algorithm for* Single Piece Lineclear Maximization *without lookahead.*

Proof. Assume there is an online strategy to achieve optimality. On a 6×4 board and with items, it, therefore, has to put the first two pieces horizontally to get the first lineclear to play optimal, since the instance could end after those two items and the optimal solution would also have a lineclear. Since an optimal algorithm can get two lineclears with four pieces, an online player has to place the next two pieces horizontally as well. This continues with the fifth piece and the sixth piece for the third lineclear with the next two horizontally placed pieces. Therefore, the online player is left with a configuration with three filled cells in the second and fifth column, where it is not able to make any further lineclears with the following items. Therefore, a strategy that guarantees optimality for the first items will necessarily lose after finitely many steps, while there is a strategy to play infinitely long. ◀

Algorithms with a non-strict competitive ratio of $c = 1$ exist, using the same strategies as in Theorem 17. The instance above requires an online player that does not have any lookahead. If there is an instance that is non-strictly competitive for Single Piece Lineclear Maximization even if the online player is granted some lookahead remains an open question. However, we believe that an algorithm needs arbitrarily much lookahead if the instance gets wider.

▶ **Theorem 20.** *There cannot be a strictly* 1*-competitive online algorithm for* Single Piece Highest Filled Cell Minimization.

Proof. Given an instance I with a $4x4$ board and a single piece the optimal strategy is to place it horizontally, resulting in a maximum height of 2. However, an online player without lookahead cannot place the item horizontally, as an adversary could play additional pieces which force a loss. Since it is possible to continue playing by placing the pieces vertically, it is crucial to know if more than one item is coming or not to decide whether placing the first item vertically or horizontally is optimal. ◀

Note that a single piece strategy needs at most eight rows [6], hence 1-competitive online algorithms with additive constant $\alpha \leq 8$ exist. Here, the instance given above can be extended to cover any available lookahead l by making the board $4 + 2 \times l$ columns wide. Again, the online player must either start by playing the pieces vertically or horizontally, depending on their beliefs about what would be optimal. This however is still hidden behind the lookahead pieces.

5 Conclusion and Open Problems

In this article, we were able to see that for some instances knowing the whole instance is crucial if a player wants to avoid losing. However, this only works if the adversary is allowed to use the presented construction which leads to some open problems discussed in this section. Our construction works on any even-width, even-height board with $w = 4, h \geq 4$ and $w \geq 6, h \geq 2$ leaving out two classes of boards: boards with a width of just two columns and boards of odd width.

We analyzed a very restricted class of Tetris games limited to a single piece type, and have shown that online players (even without lookahead) can play optimally. Interestingly, for some optimization goals, the competitiveness is not strict. Another known restricted class of Tetris games can be found in some implemented video games with a standard-size board, a 7-bag randomizer, three pieces of lookahead, and one piece of hold. A perpetual loss-avoiding strategy for this class is given in [1].

Somewhere between those results must be the point, when knowing the future gets crucial. How can the game be restricted such that a preview of one, three, or infinite many pieces starts making a difference? One main feature of Tetris in favor of a player is the option to hold a piece, which can be taken at later steps. Similar to lookahead, this feature relaxes the need to play immediately in an online way. Therefore, the influence of the option to hold would be interesting: does Tetris get competitive if the player is allowed to hold one or more items?

On the other hand, lookahead and hold are two features to relax the strict online setting of Tetris: for online problems, many more relaxations such as advice, randomization, reservations, or predictions are known and part of current research. For future work it would be interesting to see how such relaxations applied to Tetris behave, and how large their impact is.

Last, we believe that 435 pieces are still just an upper bound for an instance size that is sufficient to force an (offline) player to lose a game of Tetris on a standard-size board. Since finding the actual number was neither a primary goal for Burgiel [7] nor for us, we believe that the bound can be decreased even further with a dedicated approach.

References

1 Playing forever - Hard Drop Tetris Wiki. https://harddrop.com/wiki/Playing_forever. Accessed: 2023-10-22.

2 Sualeh Asif, Michael Coulombe, Erik D. Demaine, Martin L. Demaine, Adam Hesterberg, Jayson Lynch, and Mihir Singhal. Tetris is NP-hard even with $O(1)$ Rows or Columns. *Journal of Information Processing*, 28:942–958, 2020. doi:10.2197/ipsjjip.28.942.

3 Allan Borodin and Ran El-Yaniv. *Online computation and competitive analysis*. Cambridge University Press, 1998.

4 Ron Breukelaar, Erik D. Demaine, Susan Hohenberger, Hendrik Jan Hoogeboom, Walter A. Kosters, and David Liben-Nowell. TETRIS IS HARD, EVEN TO APPROXIMATE. *International Journal of Computational Geometry & Applications*, 14(01n02):41–68, April 2004. doi:10.1142/S0218195904001354.

5 Ron Breukelaar, Hendrik Jan Hoogeboom, and Walter A Kosters. Tetris is Hard, Made Easy. *Technical Report 2003-9, Leiden Institute of Advanced Computer Science*, 2003.

6 John Brzustowski. Can You Win at Tetris? Master's thesis, The University of British Columbia, March 1992. doi:10.14288/1.0079748.

7 Heidi Burgiel. How to lose at Tetris. *The Mathematical Gazette*, 81(491):194–200, July 1997. doi:10.2307/3619195.

8 Justin Dallant and John Iacono. How fast can we play tetris greedily with rectangular pieces? *Theoretical Computer Science*, 992:114405, 2024. doi:10.1016/j.tcs.2024.114405.

9 Erik D. Demaine, Susan Hohenberger, and David Liben-Nowell. Tetris is Hard, Even to Approximate, October 2002. arXiv:cs/0210020.

10 Vadim Gerasimov. Original Tetris: Story and Download. https://vadim.oversigma.com/Tetris.htm. Accessed: 2023-10-23.

11 Dennis Komm. *An Introduction to Online Computation – Determinism, Randomization, Advice*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 2016. doi:10.1007/978-3-319-42749-2.

12 Simon Laroche. The history of Tetris randomizers. https://simon.lc/the-history-of-tetris-randomizers, November 2018. Accessed: 2023-10-01.

13 Daniel Dominic Sleator and Robert Endre Tarjan. Amortized efficiency of list update and paging rules. *Commun. ACM*, 28(2):202–208, 1985. doi:10.1145/2786.2793.

14 Oscar Temprano. Complexity of a Tetris variant. *CoRR*, June 2015. arXiv:1506.07204.

15 Kaitlyn Tsuruda. A closer look at TETRIS: Analysis of a variant game. *Atlantic Electronic Journal of Mathematics [electronic only]*, 4, January 2011.

16 Luca Venier. Online Tetris. Master's thesis, RWTH Aachen University, November 2023.