Tetris is Hard, Even to Approximate

Erik D. Demaine* Susan Hohenberger* David Liben-Nowell*

The game of Tetris. In the popular computer game of Tetris, we are given an initial m-by-n *gameboard*, which is a partially filled rectangular grid. The player is given, one by one, a sequence of p *tetrominoes*; see Figure 1. Each piece begins in the middle of the top row of the gameboard, and falls downward at a constant speed. As it falls, the player can rotate the piece and slide it horizontally. If, when the piece comes to rest, all gridsquares in an entire row h of the gameboard are filled, row h is *cleared*: all rows above h fall one row lower, and the top row of the gameboard is replaced by an entirely unfilled row. As soon as a piece is fixed in place, the next piece appears at the top of the gameboard. Typically a one-piece *lookahead* is provided: when the ith piece begins falling, the identity of the $(i+1)$st piece is revealed. A player *loses* when a new piece is blocked by filled gridsquares from entirely entering the gameboard. The player's objective is to maximize his or her score (which increases as pieces are placed and as rows are cleared).

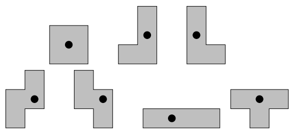

Figure 1: *The tetrominoes* Sq *("square"),* LG *("left gun"),* RG *("right gun"),* LS *("left snake"),* RS *("right snake"),* I*, and* T*, with each piece's* center *marked.*

Offline Tetris. We introduce the natural full-information (offline) version of Tetris: there is a *deterministic, finite* piece sequence, and the player knows the identity and order of all pieces that will be presented. We study the offline version because its hardness captures much of the difficulty of playing Tetris; intuitively, it is only easier to play Tetris with complete knowledge of the future, so the difficulty of playing the offline version suggests the difficulty of playing the online version. It is also natural to let m and n grow, since a relatively simple dynamic program solves the case of $m \cdot n = O(1)$ in time poly(p).

NP-hardness of maximizing rows cleared. We first show that maximizing the number of rows cleared while playing the given sequence is NP-complete. Our proof proceeds by a reduction from 3-Partition, in which we are given a set $S = \{a_1, \ldots, a_{3s}\}$ of integers and a bound T, and asked to partition S into s sets of three numbers each, so that the sum of the numbers in each set is exactly T. (3-Partition is NP-complete even when the a_i's and T are given in unary [2].) The initial gameboard is shown in Figure 2. The piece sequence consists of the following sequence of pieces for each i: one *initiator* $\langle \mathsf{I}, \mathsf{LG}, \mathsf{Sq} \rangle$, then a_i repetitions of the *filler* $\langle \mathsf{LG}, \mathsf{LS}, \mathsf{LG}, \mathsf{LG}, \mathsf{Sq} \rangle$, and then one *terminator* $\langle \mathsf{Sq}, \mathsf{Sq} \rangle$. After the pieces associated with $a_1, \ldots, a_{3s}$, we have the following additional pieces: s successive I's, one RG, and $3T/2 + 5$ successive I's. (Without loss of generality, we force T to be even by doubling all input numbers.) The last three columns of the gameboard form a *lock* which prevents any rows from being cleared using only the pieces corresponding to $a_1, \ldots, a_{3s}$. If all buckets are filled exactly to the same height, then the entire board can be cleared using the last portion of our piece sequence.

The piece sequence is chosen carefully so that all pieces corresponding to each a_i must be placed into the same bucket. The bulk of our proof of correctness is devoted to showing that, despite the decoupled nature of a sequence of Tetris pieces, the only way to possibly clear the entire gameboard is to place into a single bucket all pieces associated with each integer. We can then prove that there is a legal 3-partition for $\{a_1, \ldots, a_{3s}\}$ if and only if the gameboard can be entirely cleared using the given piece sequence. The NP-completeness of Tetris follows.

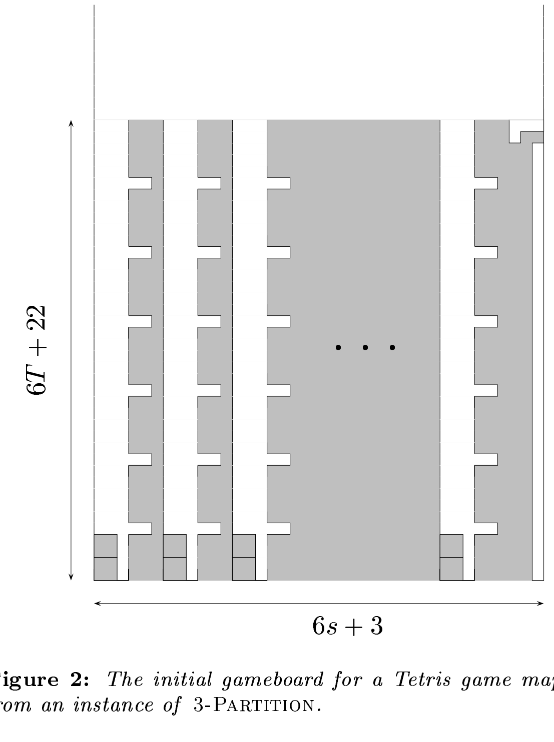

Figure 2: *The initial gameboard for a Tetris game mapped from an instance of* 3-Partition.

*Laboratory for Computer Science; Massachusetts Institute of Technology; 200 Technology Square; Cambridge, MA 02139, USA. {edemaine,srhohen,dln}@theory.lcs.mit.edu.

Inapproximability and NP-hardness for other objectives. We extend the above proof to show NP-hardness of optimizing several other natural objective functions: (1) maximizing the number of pieces placed before a loss occurs; (2) maximizing the number of *tetrises*—the simultaneous clearing of four rows; (3) minimizing the height of the highest filled gridsquare over the course of the sequence. We also prove the extreme inapproximability of the most natural objective functions: for any constant $\varepsilon > 0$, it is NP-hard to approximate to within a factor of $p^{1-\varepsilon}$ the maximum number of pieces that can be placed without a loss, or the maximum number of rows that can be cleared. We also show $(2-\varepsilon)$-inapproximability of the minimum height of a filled gridsquare.

To show these results, we use the extension in Figure 3. (The extension for the maximization of the number of tetrises is somewhat different.) Beneath the original construction, we have created a large *reservoir* of r rows filled only in the first column, and a second *lock* in four new columns, which prevents access to the reservoir until all the top rows are cleared. We append to our piece sequence a single RG (to open the second lock) and enough Sq's to exactly fill the reservoir.

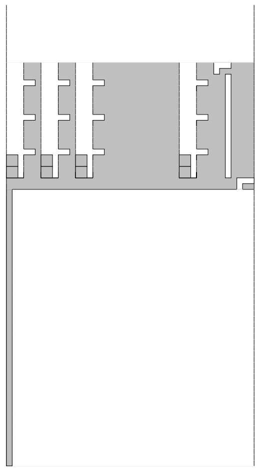

Figure 3: *Gameboard for inapproximability and hardness results for other objectives.*

In the "yes" case of 3-Partition, we can clear the top of the gameboard as before, and then clear the second lock and reservoir with the appended pieces. In the "no" case, we cannot entirely clear the top part, and thus cannot unlock the reservoir with the RG. No number of Sq's can subsequently clear the lock row. Because the gameboard has odd width, the rows above the second lock can be cleared at most once by Sq's (and then only if an odd number of gridsquares in the row were initially filled).

If r is chosen so that the unfilled area of the reservoir is more than twice the total area of the remainder of the gameboard, a loss must occur before all the Sq's have been placed. Choosing a much larger $r = \mathrm{poly}(s, T)$ yields the inapproximability results.

Variants on Tetris. Our reduction is robust to a several modifications to the rules of Tetris, including (1) with restricted player agility—allowing only two rotation/translation moves before each piece drops in height, (2) without any losses—i.e., with an infinitely tall gameboard, and (3) when the piece set is restricted to $\{\mathsf{LG}, \mathsf{LS}, \mathsf{I}, \mathsf{Sq}\}$ or $\{\mathsf{RG}, \mathsf{RS}, \mathsf{I}, \mathsf{Sq}\}$, plus at least one other piece.

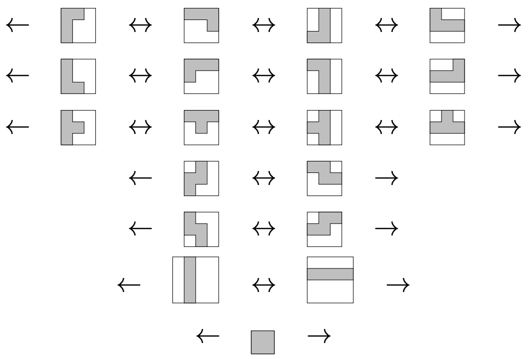

Figure 4: *The Tetris model of rotation. The pictured k-by-k bounding box is in the same position in each configuration; the piece can be rotated clockwise (respectively, counterclockwise) to yield the configuration on its right (respectively, left).*

The most interesting variants on the Tetris game, however, arise from altering the definition of the way in which pieces rotate. In our proof, we make use of only a few properties of the rotation model; thus our results hold under any model that satisfies these requirements. There are three important rotation models which meet these requirements: (1) the *instantaneous* model, in which the piece rotates $\pm 90°$ around its center, as shown in Figure 1; (2) the *continuous* (or *Euclidean*) model, which differs from instantaneous in that we require that all gridsquares that the piece *passes through* during its rotation must be unoccupied—this is the natural model if one pictures Tetris pieces as physically rotating in space; and (3) the somewhat unintuitive *Tetris* model (illustrated in Figure 4), which we have observed to be the one used in a number of actual Tetris implementations.

Conclusion. We have shown that it is NP-hard to optimize—or even approximate—a number of natural objectives for offline Tetris. (See [1] for details.) A number of interesting open questions remain. What is the complexity of Tetris if the initial gameboard is empty? For what piecesets is Tetris polynomially solveable? What is the complexity for $m = O(1)$ or $n = O(1)$? It would also be interesting to analyze theoretically the online version of Tetris, perhaps considering a probabilistically generated piece sequence. (Some possible directions for this question have been considered by Papadimitriou [3].)

References

[1] E. D. Demaine, S. Hohenberger, D. Liben-Nowell. Tetris is hard, even to approximate. Tech report, LCS, MIT, 2002.

[2] M. R. Garey, D. S. Johnson. *Computers and Intractability.*

[3] C. Papadimitriou. Games against nature. *Journal of Computer and System Sciences*, 31:288–301, 1985.