# Who witnesses The Witness? Finding witnesses in The Witness is hard and sometimes impossible

**Zachary Abel**
MIT EECS Department, 50 Vassar St., Cambridge, MA 02139, USA
zabel@mit.edu

**Jeffrey Bosboom**
MIT CSAIL, 32 Vassar Street, Cambridge, MA 02139, USA
jbosboom@csail.mit.edu

**Erik D. Demaine**
MIT CSAIL, 32 Vassar Street, Cambridge, MA 02139, USA
edemaine@mit.edu

**Linus Hamilton**
MIT Mathematics Department, 77 Massachusetts Avenue, Cambridge, MA 02139, USA
luh@mit.edu

**Adam Hesterberg**
MIT Mathematics Department, 77 Massachusetts Avenue, Cambridge, MA 02139, USA
achester@mit.edu

**Justin Kopinsky**
MIT CSAIL, 32 Vassar Street, Cambridge, MA 02139, USA
jkopin@mit.edu

**Jayson Lynch**
MIT CSAIL, 32 Vassar Street, Cambridge, MA 02139, USA
jaysonl@mit.edu

**Mikhail Rudoy**[1]
MIT CSAIL, 32 Vassar Street, Cambridge, MA 02139, USA
mrudoy@gmail.com

## Abstract

We analyze the computational complexity of the many types of pencil-and-paper-style puzzles featured in the 2016 puzzle video game *The Witness*. In all puzzles, the goal is to draw a path in a rectangular grid graph from a start vertex to a destination vertex. The different puzzle types place different constraints on the path: preventing some edges from being visited (broken edges); forcing some edges or vertices to be visited (hexagons); forcing some cells to have certain numbers of incident path edges (triangles); or forcing the regions formed by the path to be partially monochromatic (squares), have exactly two special cells (stars), or be singly covered by given shapes (polyominoes) and/or negatively counting shapes (antipolyominoes). We show that any *one* of these clue types (except the first) is enough to make path finding NP-complete ("witnesses exist but are hard to find"), even for rectangular boards. Furthermore, we show that a final clue type (antibody), which necessarily "cancels" the effect of another clue in the same region, makes path finding $\Sigma_2$-complete ("witnesses do not exist"), even with a single antibody (combined with many anti/polyominoes), and the problem gets no harder with many antibodies.

[1] Now at Google Inc.

9th International Conference on Fun with Algorithms (FUN 2018).
Editors: Hiro Ito, Stefano Leonardi, Linda Pagli, and Giuseppe Prencipe; Article No. 3; pp. 3:1–3:21
LIPICS Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

**2012 ACM Subject Classification** Theory of computation → Problems, reductions and completeness

**Keywords and phrases** video games, puzzles, hardness

**Digital Object Identifier** 10.4230/LIPIcs.FUN.2018.3

**Related Version** https://arXiv.org/abs/1804.10193

**Acknowledgements** We thank Jason Ku and Quanquan Liu for helpful discussions related to this paper. Most figures were produced using SVG Tiler (https://github.com/edemaine/svgtiler), based on SVG source code from The Windmill (https://github.com/thefifthmatt/windmill-client/), a clone of Witness-style puzzles distributed under Apache License 2.0.

## 1 Introduction

*The Witness* [9] is an acclaimed 2016 puzzle video game designed by Jonathan Blow (who originally became famous for designing the 2008 platform puzzle game Braid, which is undecidable [5]). The Witness is a first-person adventure game, but the main mechanic of the game is solving 2D puzzles presented on flat panels (sometimes CRT monitors) within the game. The 2D puzzles are in a style similar to pencil-and-paper puzzles, such as Nikoli puzzles. Indeed, one clue type in Witness (triangles) is very similar to the Nikoli puzzle *Slitherlink* (which is NP-complete [10]).

In this paper, we perform a systematic study of the computational complexity of all single-panel puzzle types in The Witness, as well as some of the 3D "metapuzzles" embedded in the environment itself. Table 1 summarizes our single-panel results, which range from polynomial-time algorithms (as well as membership in L) to completeness in two complexity classes, NP (i.e., $\Sigma_1$) and the next level of the polynomial hierarchy, $\Sigma_2$. Table 3 summarizes our metapuzzle results, where PSPACE-completeness typically follows immediately.

For omitted proofs, see [1].

**Witness puzzles.** Single-panel puzzles in The Witness (which we refer to henceforth as *Witness puzzles*) consist of an $m \times n$ full rectangular grid;[2] one or more *start circles* (drawn as a large dot, ●); one or more *end caps* (drawn as half-edges leaving the rectangle boundary); and zero or more *clues* (detailed below) each drawn on a vertex, edge, or cell[3] of the rectangular grid. Figure 1 shows a small example and its solution. The goal of the puzzle is to find a path that starts at one of the start circles, ends at one of the end caps, and satisfies all the constraints imposed by the clues (again, detailed below). We focus on the case of a single start circle and single end cap, which makes our hardness proofs the most challenging.

We now describe the clue types and their corresponding constraints. Table 2 lists the clues by what they are drawn on — grid edge, vertex, or cell — which we refer to as *this* edge, vertex, or cell. While the last five clue types are drawn on a cell, their constraint

[2] While most Witness puzzles have a rectangular boundary, some lie on a general grid graph. This generalization is mostly equivalent to having broken-edge clues (defined below) on all the non-edges of the grid graph, but the change in boundary can affect the decomposition into regions. We focus here on the rectangular case because it is most common and makes our hardness proofs most challenging.

[3] We refer to the unit-square faces of the rectangular grid as *cells*, given that "squares" are a type of clue and "regions" are the connected components outlined by the solution path and rectangle boundary.

**Table 1** Our results for one-panel puzzles in The Witness: computational complexity with various sets of allowed clue types (marked by ✓). Allowed polyomino clues are either arbitrary (✓), or restricted to be monominoes (✓■), vertical dominoes (✓▮), or rotatable dominoes (✓▚).

| broken edge | hexagon | square | star | triangle | polyomino | antipolyomino | antibody | complexity |
|---|---|---|---|---|---|---|---|---|
| ▪▪ | ⬢ | ● | ✸ | ▲▲ | ▚ | ▞ | ⅄ | |
| ✓ | | | | | | | | $\in$ L |
| ✓ | ✓vertices | | | | | | | NP-complete |
| | ✓vertices | | | | | | | *OPEN* |
| | ✓edges | | | | | | | NP-complete |
| | | ✓1 color | | | | | | $\in$ P |
| | | ✓2 colors | | | | | | NP-complete |
| | | | ✓1 color | | | | | *OPEN* |
| | | | ✓$n$ colors | | | | | NP-complete |
| | | | | ✓▲ | | | | NP-complete |
| ✓ | | | | ✓▲▲ | | | | NP-complete |
| | | | | ✓▲▲ | | | | *OPEN* |
| | | | | ✓▲▲▲ | | | | NP-complete |
| ✓ | | | | | ✓■ | | | *OPEN* |
| | | | | | ✓■ | ✓□ | | NP-complete |
| | | | | | ✓▚ | | | NP-complete |
| | | | | | ✓▮ | | | NP-complete |
| ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | | $\in$ NP |
| ✓ | ✓ | ✓ | ✓ | ✓ | | | ✓$n$ | $\in$ NP |
| | | | | | ✓ | | ✓2 | $\Sigma_2$-complete |
| | | | | | ✓ | ✓ | ✓1 | $\Sigma_2$-complete |
| ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓$n$ | $\in \Sigma_2$ |

applies to the *region* that contains that cell (referred to as *this* region), where we consider the regions of cells in the rectangle as decomposed by the (hypothetical) solution path and the rectangle boundary.

The solution path must satisfy *all* the constraints given by all the clues. (The meaning of this statement in the presence of antibodies is complicated; see Section 8.) Note, however, that if a region has no clue constraining it in a particular way, then it is free of any such constraints. For example, a region without polyomino or antipolyomino clues has no packing constraint.

As summarized in Table 1, we prove that most clue types *by themselves* are enough to obtain NP-hardness. The exceptions are broken edges, which alone just define a graph search problem; and vertex hexagons, which are related to Hamiltonian path in rectangular grid graphs as solved in [6] but remain open. But vertex hexagons are NP-hard when we also add broken edges. For squares, we determine that exactly two colors are needed for hardness. For stars, we do not know whether one or any constant number of colors are hard. For triangles, we know that 1-triangles or 3-triangles alone suffice for hardness, but for 2-triangles the only hardness proof we know needs broken edges. For polyominoes, monominoes alone are easy to solve [8], but monominoes plus antimonominoes are hard, as are rotatable dominoes by themselves and vertical nonrotatable dominoes by themselves. All problems without antibodies or without (anti)polyominoes are in NP. Antibodies combined with (anti)polyominoes push the complexity up to $\Sigma_2$-completeness, but no further.

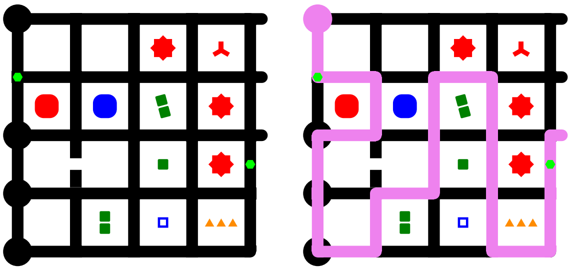


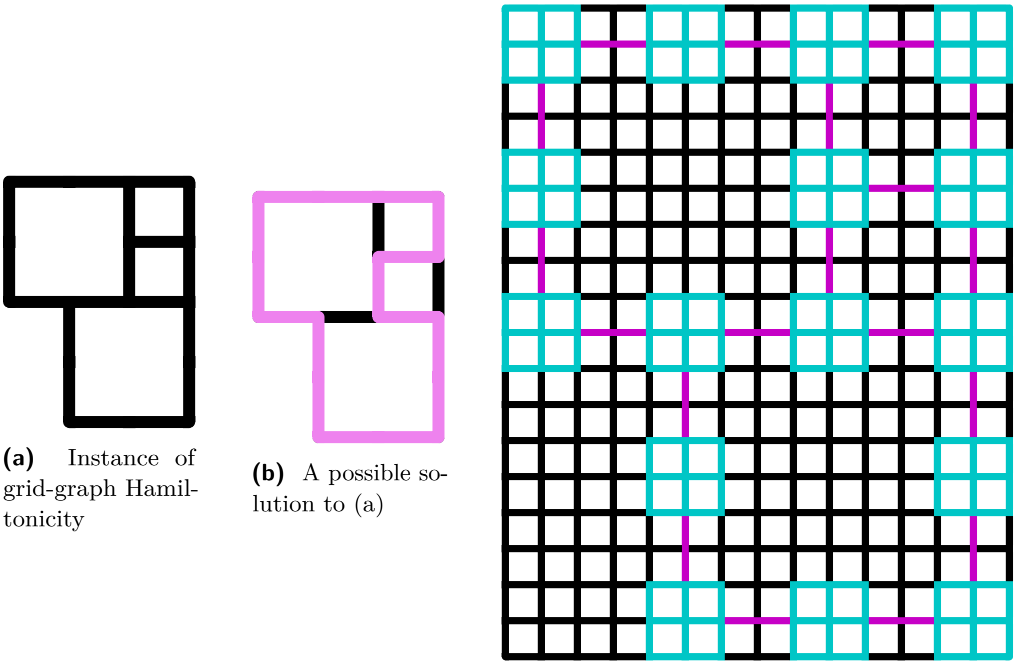
**Figure 1** A small Witness puzzle featuring all clue types (left) and its solution (right). (Not from the actual video game.)

**Table 2** Witness puzzle clue types and the definitions of their constraints.

| clue | drawn on | symbol | constraint |
|---|---|---|---|
| broken edge | edge | ▪ ▪ | The solution path cannot include this edge. |
| hexagon | edge | ⬢ | The solution path must include this edge. |
| hexagon | vertex | ⬢ | The solution path must visit this vertex. |
| triangle | cell | ▲▲ | There are three kinds of triangle clues ( ▲, ▲▲, ▲▲▲ ). For a clue with $i$ triangles, the path must include exactly $i$ of the four edges surrounding this cell. |
| square | cell | ■ | A square clue has a color. This region must not have any squares of a color different from this clue. |
| star | cell | ✹ | A star clue has a color. This region must have exactly one other star, exactly one square, or exactly one antibody of the same color as this clue. |
| polyomino | cell | ▞ | A polyomino clue has a specified polyomino shape, and is either nonrotatable (if drawn orthogonally, like ▞) or rotatable by any multiple of 90° (if drawn at 15°, like ▞). Assuming no antipolyominoes, this region must be perfectly packable by the polyomino clues within this region. |
| antipolyomino | cell | ◻ | Like polyomino clues, an antipolyomino clue has a specified polyomino shape and is either rotatable or not. For some $i \in \{0, 1\}$, each cell in this region must be coverable by exactly $i$ layers, where polyominoes count as +1 layer and antipolyominoes count as −1 layer (and thus must overlap), with no positive or negative layers of coverage spilling outside this region. |
| antibody | cell | ⅄ | Effectively "erases" itself and another clue in this region. This clue also must be necessary, meaning that the solution path should not otherwise satisfy all the other clues. See Section 8 for details. |

**Table 3** Our results for metapuzzles in The Witness: computational complexity with various sets of environmental features.

| features | complexity |
|---|---|
| sliding bridges | PSPACE-complete |
| elevators and ramps | PSPACE-complete |
| power cables and doors | PSPACE-complete |



**(a)** Instance of grid-graph Hamiltonicity

**(b)** A possible solution to (a)

**(c)** Corresponding chambers and hallways

**Figure 2** An example of the Hamiltonicity framework with $r = 1$ and $s = 4$.

**Witness metapuzzles.** We also consider some of the *metapuzzles* formed by the 3D environment in The Witness, which interact with the 2D single-panel puzzles. See Section 9 for details of these interaction models. Table 3 lists our metapuzzle results, which are all PSPACE-completeness proofs following the infrastructure of [2] (from FUN 2014).

## 2 Hamiltonicity Reduction Framework

We introduce a framework for proving NP-hardness of Witness puzzles by reduction from Hamiltonian cycle in a grid graph $G$ of maximum degree 3. Roughly speaking, we scale $G$ by a constant scale factor $s$, and replace each vertex by a block called a chamber; refer to Figure 2. Precisely, for each vertex $v$ of $G$ at coordinates $(x, y)$, we construct a $2r+1 \times 2r+1$ subgrid of vertices $\{sx - r, \ldots, sx + r\} \times \{sy - r, \ldots, sy + r\}$, and all induced edges between them, called a *chamber* $C_v$. This construction requires $2r < s$ for chambers not to overlap. For each edge $e = \{v, w\}$ of $G$, we construct a straight path in the grid from $sv$ to $sw$, and define the *hallway* $H_{v,w}$ to be the subpath connecting the boundaries of $v$'s and $w$'s chambers, which consists of $s - 2r$ edges. Figure 2 illustrates this construction on a sample graph $G$.

In each reduction, we define constraints to force the solution path to visit (some part of) each chamber at least once, to alternate between visiting chambers and traversing hallways that connect those chambers, and to traverse each hallway at most once. Because $G$ has maximum degree 3, these constraints imply that each chamber is entered exactly once and exited exactly once. Next to one chamber on the boundary of $G$, called the *start/end chamber*, we place the start circle and end cap of the Witness puzzle. Thus any solution to the Witness puzzle induces a Hamiltonian cycle in $G$. To show that any Hamiltonian cycle in $G$ induces a solution to the Witness puzzle, we simply need to show that a chamber can be traversed in each of the $\binom{3}{2}$ ways.

## 3 Hexagons and Broken Edges

Hexagons are placed on vertices or edges of the graph and require the path to pass through all of the hexagons. Broken edges are edges which cannot be included in the path. We show the positive result that puzzles with just broken edges are solvable in $L$, and the negative results that puzzles with just hexagons on edges are NP-complete and puzzles with just hexagons on vertices and broken edges are NP-complete. We leave open the question of puzzles with just hexagons on vertices (and no broken edges).

▶ **Lemma 1.** *Witness puzzles containing only broken edges, multiple start circles and multiple end caps are in $L$.*

**Proof.** We keep two pointers and a counter to track which pairs of starts and ends we have tried. For each start and end pair we run an $(s, t)$ path existence algorithm, which is in L. If any of these return yes, the answer is yes. Thus we've solved the problem with a quadratic number of calls to a log-space algorithm, a constant number of pointers, and a counter, all of which only require logarithmic space. ◀

▶ **Lemma 2.** *It is NP-complete to solve Witness puzzles containing only broken edges and hexagons on vertices.*

**Proof.** Hamiltonian path in grid graphs is a strict subproblem. ◀

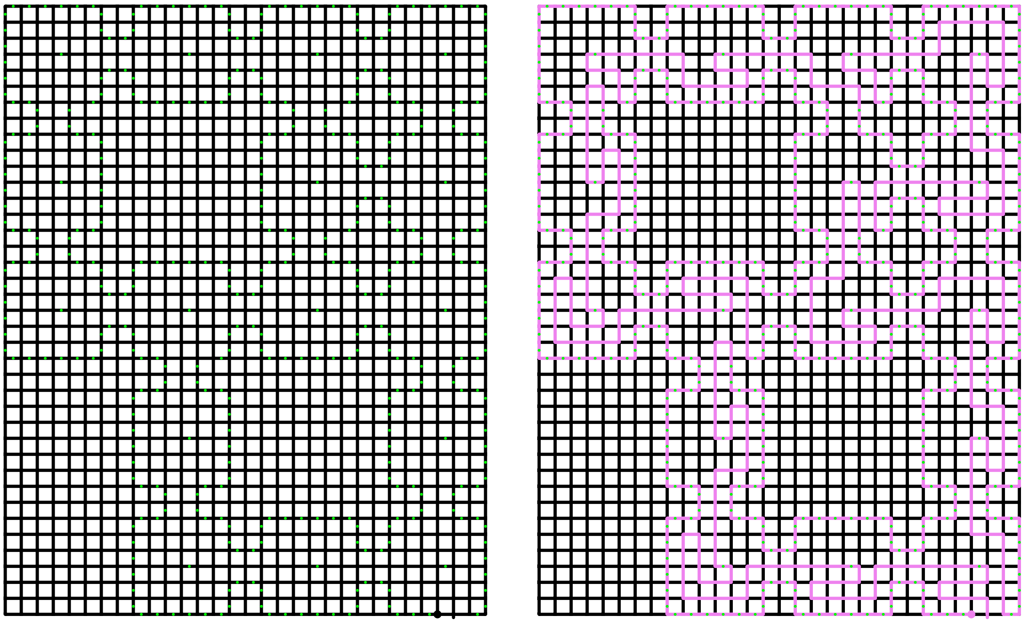
▶ **Theorem 3.** *It is NP-complete to solve Witness puzzles containing only hexagons on edges (and no broken edges).*

**Proof sketch.** We use the Hamiltonicity framework; refer to Figure 3. Noting that two edge hexagons incident to the same vertex must be consecutively traversed by the solution path, we carefully force the solution path to traverse the boundary of every chamber separate from the decision of which hallways to use. As with other Hamiltonicity framework reductions, we force each chamber to be visited with an edge hexagon in its center and can deduce the corresponding Hamiltonian cycle in the original grid graph from the set of used hallways. ◀

▶ **Open Problem 1.** *Is there a polynomial-time algorithm to solve Witness puzzles containing only hexagons on vertices?*

## 4 Squares

Each square clue has a color and is placed on a cell of the puzzle. Each region formed by the solution path and puzzle boundary must have at most one color of squares. If a puzzle has only a single color of squares, no non-trivial constraint is applied.



**(a)** Instance corresponding to Figure 2a

**(b)** Solution corresponding to Figure 2b

**Figure 3** Example of the Hamiltonicity framework applied to Witness with edge hexagons.

## 4.1 Tree-Residue Vertex Breaking

Our reduction is from *tree-residue vertex breaking* [4]. Define *breaking* a vertex of degree $d$ to be the operation of replacing that vertex with $d$ vertices, each of degree 1, with the neighbors of the vertex becoming neighbors of these replacement vertices in a one-to-one way. The input to the tree-residue vertex breaking problem is a planar multigraph in which each vertex is labeled as "breakable" or "unbreakable". The goal is to determine whether there exists a subset of the breakable vertices such that breaking those vertices (and no others) results in the graph becoming a tree (i.e., destroying all cycles without losing connectivity). This problem is NP-hard even if all vertices are degree-4 breakable vertices or degree-3 unbreakable vertices[4].

## 4.2 Squares with Squares of Two Colors

▶ **Theorem 4.** *It is NP-complete to solve Witness puzzles containing only squares of two colors.*

Concurrent work [8] also proves this theorem. However, we prove this by showing that the stronger *Restricted Squares Problem* is also hard, which will be useful to reduce from in Section 5.

▶ **Problem 1** (Restricted Squares Problem). *An instance of the* Restricted Squares Problem *is a Witness puzzle containing only squares of two colors (red and blue), where each cell in the leftmost and rightmost columns, and each cell in the topmost or bottommost rows, contains a square clue; and of these square clues, exactly one is blue, and that square clue is not in a corner cell; and the start vertex and end cap are the two boundary vertices incident to that blue square; see Figure 4.*

▶ **Theorem 5.** *The Restricted Squares Problem is NP-complete.*

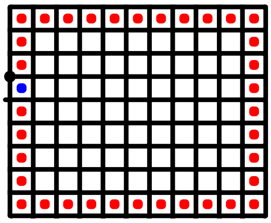
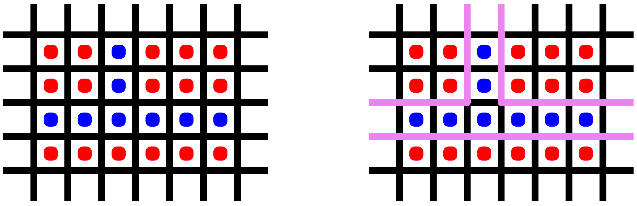
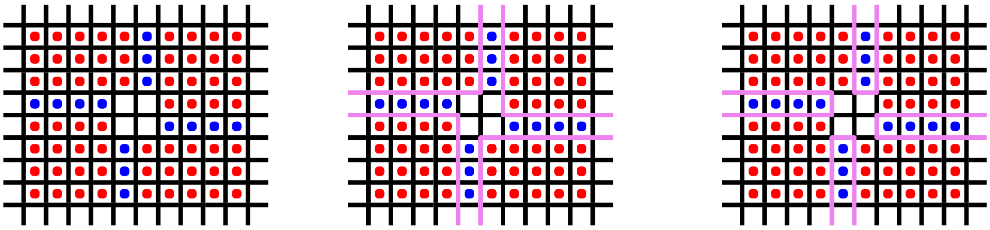


**Figure 4** Boundary of the Restricted Squares Problem.



**(a)** Unsolved gadget. **(b)** The unique solution path.

**Figure 5** Unbreakable degree-3 vertex gadget



**(a)** Unsolved gadget **(b)** Unbroken solution path. **(c)** Broken solution path.

**Figure 6** Breakable degree-4 vertex gadget

**Proof sketch.** We reduce from tree-residue vertex breaking and construct gadgets for an unbreakeable degree 3 vertex (Figure 5) and a breakable degree 4 vertex (Figure 6) out of squares. We force the solution path to take an Euler tour of these gadgets, which can only be done if the underlying tree-residue vertex breaking graph is a tree. ◀

## 5 Stars

Star clues are in cells of a puzzle. If a region formed by the solution path and boundary of a puzzle has a star of a given color, then the number of clues (stars, squares, or antibodies) of that color in that region must be exactly two. A star imposes no constraint on clues with colors different from that of the star.

▶ **Theorem 6.** *It is NP-complete to solve Witness puzzles containing only stars (of arbitrarily many colors).*

**Proof sketch.** We reduce from the Restricted Squares Problem. For every square in the source instance, $I$, we use exactly one pair of stars of a distinct color corresponding to that square, as well as ten auxiliary colors. Figure 7 shows the high level structure of the reduction. A subrectangle, $S$, of the puzzle is designated for recreating $I$. For each pair of stars corresponding to a square, we place one of the two stars on the boundary of the puzzle, and the other in $S$ in the same position as the corresponding square in $I$. The solution path will be forced to divide the overall puzzle into exactly two regions—an “inside” and an “outside”—such that all of the boundary stars corresponding to red squares are on the outside and all of the boundary stars corresponding to blue squares are on the inside. Then, inside of $S$, the solution path must ensure that all stars corresponding to red squares are in the outside region and all stars corresponding to blue squares are in the inside region, or else the star constraint will be violated. Then the solution path inside of $S$ must correspond exactly to a solution path in $I$. ◀

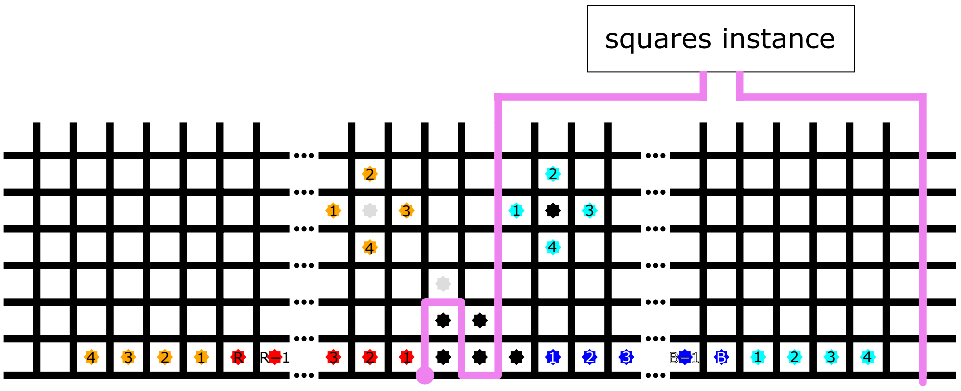


**Figure 7** The boundary of the reduction. Each visual (color, number) pair represents a distinct color in the constructed instance. All stars depicted as blue correspond to blue squares in the source instance and must be in the inside region. Stars depicted as red correspond to red squares and must be in the outside region. The other stars enforce this.

**Table 4** Summary of Slitherlink / Witness triangle constraints. New results are bold.

| *Clue types* | *Complexity* |
|---|---|
| 0 | P [10] |
| **1** | **NP-complete [Theorem 7]** |
| 2 | *Open* |
| **3** | **NP-complete [Theorem 8]** |
| 4 | P [trivial] |
| **0** and **2** | **NP-complete** |

▶ **Open Problem 2.** *Is it NP-complete to solve Witness puzzles containing only a constant number of colors of stars?*

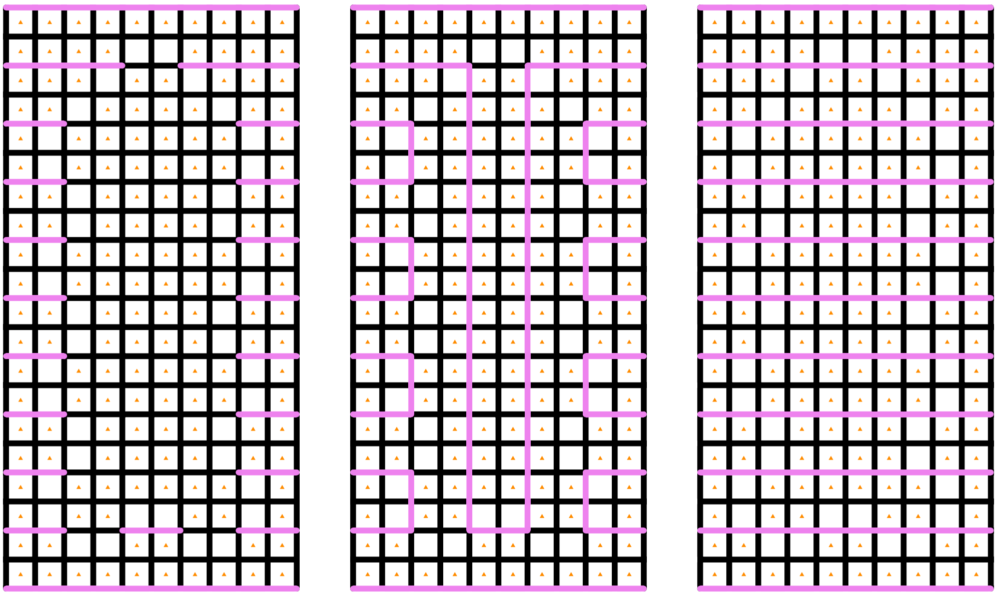
## 6 Triangles

Triangles are placed in cells. The number of solution path edges adjacent to that cell must match the number of triangles. This is similar to Slitherlink, which is known to be NP-complete [10]; however the proof in [10] relies critically on being able to force zero edges around a cell using 0-clues, which are not available in The Witness. We characterize all possibile combinations of constraints of these types for grid graphs. Table 4 summarizes what is known.

▶ **Open Problem 3.** *Is it NP-complete to solve Witness puzzles containing only 2-triangle clues (and no broken edges)?*

### 6.1 One Triangle Clues

Proving hardness of Witness puzzles containing only 1-triangle clues is made challenging by the fact that it is impossible to (locally) force turns on the interior of the puzzle. In particular, any rectangular interior region can be locally satisfied by a solution path which either traverses every second row of horizontal edges in the region or every second column of vertical edges in the region *regardless of the configuration of 1-triangle clues in the region.*



**(a)** Unsolved. **(b)** “All” or “active” solution. **(c)** “Nothing” or “inactive” solution.

**Figure 8** All-or-nothing gadget.

Therefore, any local arguments we want to make about gadgets on the interior of the puzzle will need to admit the possibility of local solutions which are comprised of just horizontal or vertical paths straight through.

▶ **Theorem 7.** *It is NP-complete to solve Witness puzzles containing only 1-triangle clues.*

**Proof sketch.** We reduce from positive 1-in-3SAT, making use of the fact that the solution path must be a single closed path. We force the solution path to traverse all horizontal edges except for on the interior of gadgets, in which the solution path is allowed to connect adjacent horizontal path segments in a controlled manner (see Figure 8 for one key gadget), such that doing so corresponds to a solution to the source 1-in-3SAT instance. ◀

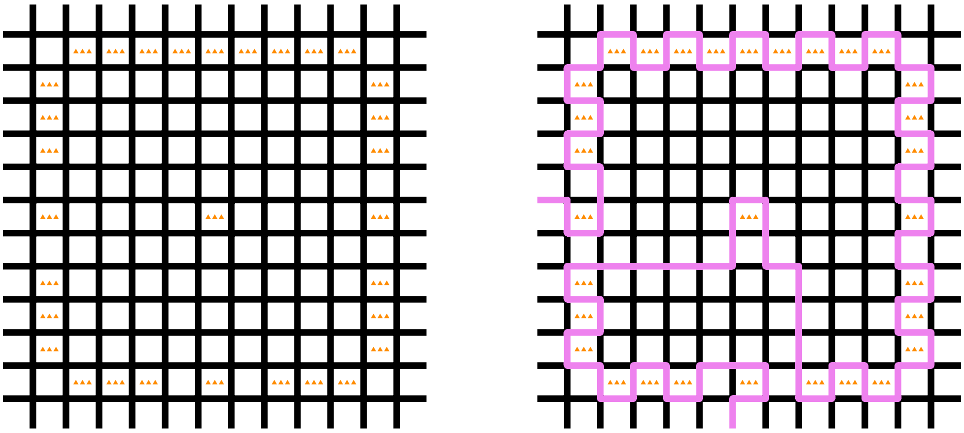
## 6.2 Three Triangle Clues

▶ **Theorem 8.** *It is NP-complete to solve Witness puzzles containing only 3-triangle clues.*

**Proof sketch.** We use the Hamiltonicity framework. Adjacent 3-triangle clues must be traversed consecutively by the solution path, so we can use them to for the solution path to trace the boundary of each chamber. Figure 9 shows the construction of a chamber. ◀

# 7 Polyominoes

This section covers various types of *polyomino* and *antipolyomino* clues. Polyomino clues can generally be characterized by the size and shape of the polyomino and whether or not they can be rotated (▮ vs. ▮). For each region, it must be possible to place all polyominoes and antipolyominoes depicted in that region’s clues (not necessarily within the region) so that



**(a)** Unsolved.

**(b)** One possible solution, using the left and bottom edges.

**Figure 9** A chamber with edges to the left, right, and below.

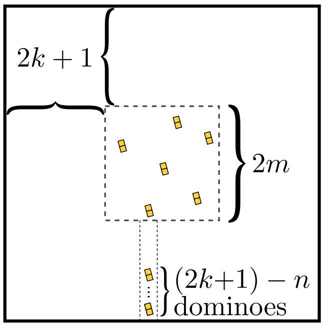


**Figure 10** Overview of the rotatable dominoes NP-completeness proof.

for some $i \in \{0,1\}$, each cell inside the region is covered by exactly $i$ more polyomino than antipolyomino and each cell outside the region is covered by the same number of polyominoes and antipolyominoes. We give several negative results showing that some of the simplest (anti)polyomino clues suffice for NP-completeness.

Concurrent work [8] shows that Witness puzzles with squares of two colors for which every cell contains a square clue can be solved in polynomial time. Interestingly, such puzzles are equivalent to puzzles with only monominoes, by replacing one color of square with monominoes and the other color with blank cells. The only constraint on the two puzzle types is that there can be no region with a mix of square colors or, equivalently, monomino clues and blank cells. However, the question of whether puzzles with only monominoes and broken edges can be solved in polynomial time is still open.

## 7.1 Rotatable Dominoes

▶ **Theorem 9.** *It is NP-complete to solve Witness puzzles containing only rotatable dominoes.*

**Proof sketch.** We reduce from Rectilinear Steiner Tree: given $n$ points with integer coordinates $(x'_i, y'_i)$ in the plane, $i \in \{1,2,\ldots,n\}$, and given an integer $k$, decide whether there exists a rectilinear tree connecting the $n$ points having total length at most $k$. As illustrated in Figure 10, we embed the tree in the cells of a Witness puzzle, putting a domino clue at each vertex of the tree, which the solution path must therefore visit. The total number of dominoes is proportional to $k$, such that with careful counting, the area enclosed by the solution path must "look like" a tree of length exactly $k$ in the original Steiner tree instance. ◀

## 7.2 Monominoes + Antimonominoes

▶ **Theorem 10.** *It is NP-complete to solve Witness puzzles containing only monominoes and antimonominoes.*

**Proof sketch.** The reduction is very similar to that of Theorem 9, except that the vertices of the Steiner tree contain antimonomino clues, and most of the other cells contain monomino clues. We force the solution path to partition the puzzle into two regions, an "outside" region which is entirely covered by monominoes, and an "inside" region which contains exactly as many antimonominoes as monominoes, thereby satisfying both. We show that doing this corresponds to a solution to the Steiner tree source instance. ◀

## 7.3 Nonrotatable Dominoes

▶ **Theorem 11.** *It is NP-complete to solve Witness puzzles containing only nonrotatable vertical dominoes.*

**Proof sketch.** We reduce from planar rectilinear monotone 3SAT [7]. Refer to Figure 11. We construct variable "wires" which are comprised of dominoes arranged on a diagonal which the solution path must enclose in one of two settings. Each clause needs to "connect" to at least one of its literals, but can only get close enough to do so if the corresponding variable is set appropriately. ◀

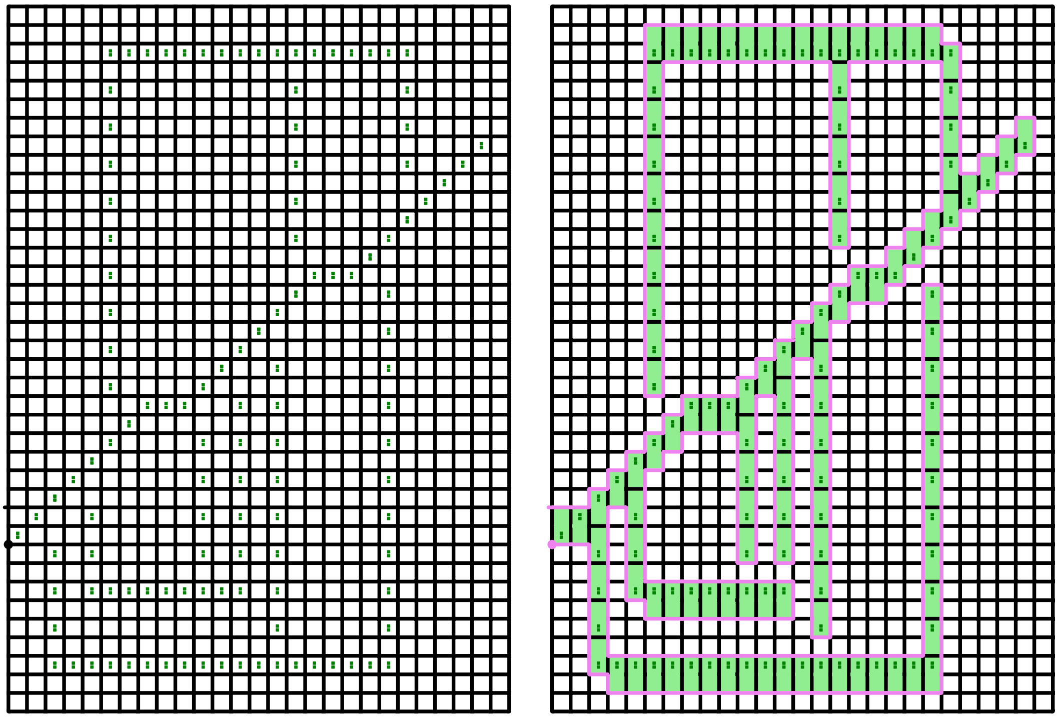
▶ **Open Problem 4.** *Is there a polynomial-time algorithm to solve Witness puzzles containing only monominoes and broken edges?*

# 8 Antibodies

An antibody (⅄) eliminates itself and one other clue in its region. For the antibody to be satisfied, this region must *not* be satisfied without eliminating a clue; that is, the antibody must be necessary. An antibody may be colored, but its color does not restrict which clues it can eliminate.[4] Very few Witness puzzles contain multiple antibodies, making the formal rules for the interactions between antibodies not fully determined by the in-game puzzles. We believe the following interpretation is a natural one: each antibody increments a count of clues that must necessarily be unsatisfied for their containing region to be satisfied. If there are $k$ antibodies in a region, then there must be $k$ clues which can be eliminated such that those $k$ clues were unsatisfied and all other clues were satisfied; furthermore, there must not have been a set of fewer than $k$ unsatisfied clues such that all other clues are satisfied[5]. Antibodies cannot eliminate other antibodies. The choice of clue to eliminate need not be

[4] Antibody color matters when checking if the antibody is necessary; a region containing only a star and an antibody of the same color is unsatisfied because the antibody is not necessary.

[5] Whether or not a clue is satisfied is usually determined only by the solution path; however, in the case of polyominoes and antipolyominoes, there might be several choices of packings which satisfy different sets of clues.



**(a)** The puzzle. **(b)** The solution.

**Figure 11** A Witness puzzle produced from $(x \vee y \vee z) \wedge (\neg x \vee \neg y \vee \neg z) \wedge (\neg x \vee \neg y \vee \neg y)$ and its solution ($x$ and $y$ are FALSE, $z$ is TRUE). Shaded cells show the domino tiling on the path's interior.

unique; for instance, a region with three white stars and one antibody is satisfied, even though the stars are not distinguished. Formally:

▶ **Definition 12** (Simultaneous Antibodies). A region with $k$ antibody clues is satisfied if and only if there exists a set $S$ of $k$ non-antibody clues such that eliminating all clues in $S$ and all $k$ antibodies leaves the region satisfied, and there does *not* exist a set $S'$ of non-antibody clues with $|S'| < k$ such that eliminating all clues in $S'$ and only $|S'|$ of the antibodies leaves the region satisfied.

▶ **Theorem 13.** *Witness puzzles containing all clue types except polyominoes and antipolyominoes are in NP.*

**Proof sketch.** Other than antibodies, polyominoes, and antipolyominoes, whether or not a clue is satisfied can be easily determined from the solution path. Thus, checking whether an antibody which eliminates such a clue is necessary is easy. ◀

▶ **Theorem 14.** *Witness puzzles containing all clue types except antipolyominoes and for which at least one solution eliminates at most one polyomino in each region are in NP.*

**Proof sketch.** If at least one polyomino is eliminated in a region containing at least two polyominoes and the region is satisfied as a result, then the region can't be satisfied without deleting at least one polyomino because the total area of the polyominoes is greater than that of the region, and therefore there is no packing. ◀

▶ **Theorem 15.** *Witness puzzles containing any set of clue types (including polyominoes, antipolyominoes, and antibodies) are in $\Sigma_2$.*

**Proof.** Solving this Witness puzzle requires picking clues for antibodies to eliminate and finding a path which respects the remaining clues, such that the regions cannot be satisfied if only a subset of antibodies are used to eliminate clues. Membership in $\Sigma_2$ requires an algorithm which accepts only when there exists a certificate of validity for which there is no certificate of invalidity (i.e., one alternation of $\exists x \forall y$). A certificate of invalidity allows a polynomial-time algorithm to check whether an instance of a given problem is false. Our certificate of validity is a solution path, a mapping from antibodies to eliminated clues, and a packing witness for any region with at least one uneliminated polyomino. Our certificate of *invalidity* is the solution path (from the certificate of validity), a mapping of a *subset* of the antibodies to eliminated clues, and a packing witness for any region with at least one uneliminated polyomino.

Our verification algorithm begins checking the certificate of validity by verifying the packing witnesses and checking that the antibody mapping specifies distinct eliminated clues in the same region as each antibody. Then we remove all antibodies, polyomino and antipolyomino clues, and eliminated clues from the Witness puzzle and run the algorithm given in the proof of Theorem 13 to verify that the remaining clues in each region are satisfied under the solution path.

To verify the certificate of invalidity, we again check its packing witnesses and its (partial) antibody mapping. Then we remove the used antibodies, polyomino and antipolyomino clues, and eliminated clues from the Witness puzzle. We replace any unused colored antibodies with stars of their color if they are in the same region as an (uneliminated) star of that color, then remove any remaining antibodies. We run the polynomial-time algorithm given in the proof of Theorem 13 on the resulting Witness puzzle. Our algorithm accepts if and only if the certificate of validity is valid and all certificates of invalidity are invalid. ◀

Finally, we will show that Witness puzzles in general are $\Sigma_2$-complete. We will proceed in two steps, first considering puzzles which have two (or more) antibodies which might be eliminating polyominoes in the same region, and then considering puzzles which have only one antibody but both polyominoes and antipolyominoes. In both cases, we will reduce from *Adversarial-Boundary Edge-Matching*, a one-round two-player game defined as follows:

▶ **Problem 2** (Adversarial-Boundary Edge-Matching). *A* signed color *is a sign (+ or −) together with an element of a set $C$ of colors. Two signed colors* match *if they have the same element of $C$ and the opposite sign. A* tile *is a unit square with a signed color on each of its edges.*

*An $n \times (2m)$* boundary-colored board *is an $n \times (2m)$ rectangle together with a signed color on each of the unit edges along its boundary. Given such a board and a multiset $T$ of $2nm$ tiles, a* tiling *is a placement of the tiles at integer locations within the rectangle such that two adjacent tiles have matching colors along their shared edge, and a tile adjacent to the boundary has a matching color along the shared edge. There are two types of tiling according to whether tiles can only be translated or can also be rotated.*

*The* adversarial-boundary edge-matching game *is a one-round two-player game played on a $2n \times m$ boundary-colored board $B$ and a multiset $T$ of $2nm$ tiles. Name the unit edges along $B$'s top boundary $e_0, e_1, \ldots, e_{2n}$ from left to right. During the first player's turn, for each even $i = 0, 2, 4, \ldots, 2n-2$, the first player chooses to leave alone or swap the signed colors on $e_i$ and $e_{i+1}$. During the second player's turn, the second player attempts to tile the resulting boundary-colored board $B'$ such that signed colors on coincident edges (whether on tiles or on the boundary of $B'$) match. If the second player succeeds in tiling, the second player wins; otherwise, the first player wins.*

*The* adversarial-boundary edge-matching problem *is to decide whether the first player has a winning strategy for a given adversarial-boundary edge-matching game; that is, whether there exists a choice of top-boundary swaps such that there does* not *exist an edge-matching tiling of the resulting boundary-colored board.*

▶ **Lemma 16.** *Adversarial-boundary edge-matching is $\Sigma_2$-hard, with or without tile rotation, even when the first player has a losing strategy.*

**Proof sketch.** We reduce from from QSAT$_2$, which is the $\Sigma_2$-complete problem of deciding a Boolean statement of the form $\exists x_1 : \exists x_2 : \cdots : \exists x_n : \forall y_1 : \forall y_2 : \cdots : \forall y_n : f(x_1, x_2, \ldots, x_n; y_1, y_2, \ldots, y_n)$ where $f$ is a Boolean formula using AND ($\wedge$), OR ($\vee$), and/or NOT ($\neg$). We convert this formula into a circuit, lay out the circuit on a square grid, and implement each circuit element as a set of tiles, one tile for each valid state (truth table row) of that element. The first player's boundary-edge swaps encode a setting of true or false for the first player's variables. Then, as part of solving the edge-matching problem, the second player must exhibit a setting of their variables that makes the formula false; otherwise the first player wins. ◀

▶ **Theorem 17.** *It is $\Sigma_2$-complete to solve Witness puzzles containing two antibodies and polyominoes.*

**Proof.** We reduce from adversarial-boundary edge-matching with the guarantee that the first player has a losing strategy. We create a Witness puzzle containing two antibodies. We will force the solution path to split the puzzle into two regions, with both antibodies in the same region and with part of the solution path encoding top-boundary swaps. In the construction, it will be easy to find a solution path satisfying all non-antibody clues when both antibodies are used to eliminate clues, but the antibodies themselves are only satisfied if they are necessary. When only one antibody is used, the remaining polyominoes in one of the regions, together with the solution path, simulate the adversarial-boundary edge-matching instance. The remaining polyominoes cannot pack the region (necessitating the second antibody and making the Witness solution valid) exactly when the adversarial-boundary edge-matching instance is a YES instance. (In the context of The Witness, the human player is the first player in an adversarial-boundary edge-matching game, and The Witness is the second player.)

**Encoding signed colors.** We encode signed colors on the edges of polyominoes in binary as unit-square tabs (for positive colors) or pockets (for negative colors) [3, Figure 7]. If the input adversarial-boundary edge-matching instance has $c$ colors, we need $\lceil \log_2(c+1) \rceil$ bits to encode the color[6]. To prevent pockets at the corners of a tile from overlapping, we do not use the $2 \times 2$ squares at each corner to encode colors, so tiles are built out of squares with side length $w = \lceil \log_2(c+1) \rceil + 4$[7].

**Clue sets.** We consider the clues in the Witness puzzle to be grouped into two clue sets, $A$ and $B$, which we place far apart on the board. We will argue that any valid solution path must partition the puzzle into two regions, such that each set is fully contained in one of the regions. Figure 12 shows (the intended packing of) most of the polyomino clues.

[6] We cannot use 0 as a color because we need at least one tab or pocket to determine the sign.

[7] At the cost of introducing disconnected polyomino clues, we could leave only one pixel at each corner out of the color encoding; that pixel is disconnected when the colors on its edges both have pockets next to it.

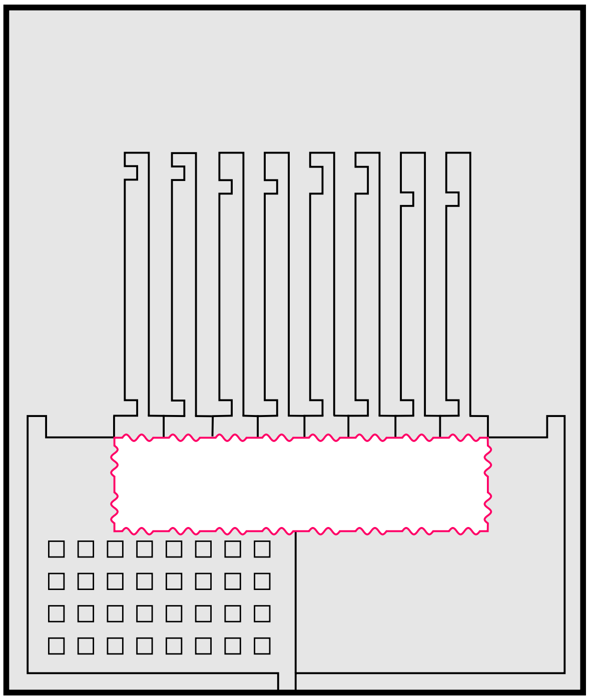


**Figure 12** The intended packing of the puzzle after eliminating the medium polyomino (not to scale). The left and right board-frame polyominoes slot inside the large polyomino, and the monominoes fill the holes in the left board-frame polyomino. The stamps fill in their matching handle slots in the large polyomino, leaving only the boundary-colored board for the simulated adversarial-boundary edge-matching instance.

Clue set $A$ contains:

- Two antibodies.
- $2nw - q$ monominoes, where $q$ is the total number of pockets minus the total number of tabs across the "dies" of the "stamps" in clue set $B$ (see below). There are $2n$ stamps each having up to $\lceil \log_2(c+1) \rceil$ tabs or pockets, so the total number of monominoes is between $2nw - 2n\lceil \log_2(c+1) \rceil = 8n$ and $2nw + 2n\lceil \log_2(c+1) \rceil = 4nw - 8n$ inclusive.
- A $w \times w$ square polyomino for each of the $2nm$ tiles in the adversarial-boundary edge-matching instance. The edges of each polyomino are modified with tabs and pockets encoding the signed colors on the corresponding edges of the corresponding tile. Call the upper-left corner of the $w \times w$ square the *key pixel* of that polyomino (even if tabs caused other pixels to be further up or to the left).
- A "medium" sized polyomino formed from a $2n(w+3) - 1 \times m(w+3) + 3$ rectangle polyomino; see Figure 13. Cut a hole out of this rectangle in the image of each tile polyomino, aligning the key pixel of each tile polyomino to a $2n \times m$ grid with upper-left point at the fourth row, second column of the rectangle and $w + 3$ intervals between rows and columns. Regardless of the pattern of tabs and pockets on each tile, this spacing ensures at least two rows of pixels above the top row of tile-shaped holes, at least one row on each other side, and at least one row between adjacent holes. Then add pixels above the upper-leftmost and upper-rightmost pixel of the rectangle (the *horns*) and below the middle-bottommost pixel of the rectangle (the *tail*). Finally, cut $2nw$ pixels out of the top row of the rectangle starting from the third pixel; this cutout is the *stamp accommodation zone*.

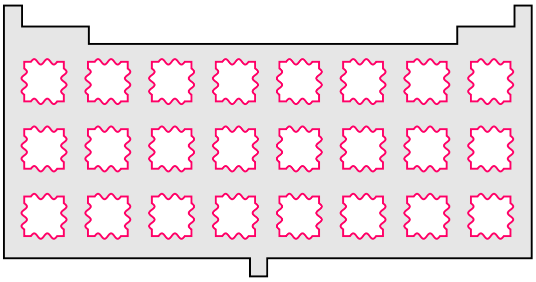


**Figure 13** The medium polyomino, with boundary-colored holes matching each tile polyomino.

- Two *board-frame* polyominoes. Again, starting from a $2n(w+3)-1 \times m(w+3)+3$ rectangle polyomino, add horns and tail pixels in the same locations. Then cut out a $2nw \times mw$ rectangle whose upper-left pixel is the third pixel in the top row of the rectangle. The left, right and bottom edges of this cutout are modified with tabs and pockets encoding the signed colors on the corresponding sides of the boundary-colored board in the adversarial-boundary edge-matching instance. Split the polyomino vertically along the column of edges immediately to the right of the tail pixel.

  Finally, for each monomino in this clue set, cut a pixel out of the left board-frame polyomino, starting from the second-bottommost pixel in the second column, continuing across every other column, then continuing with the fourth-bottomost pixel in the second column, and so on. The left board-frame polyomino has width $nw+3n$, we cut pixels out of every other column, and we do not cut holes in its left or right columns, so we cut pixels out of $\frac{nw+3n-2}{2}$ columns. Below the $mw$-tall cutout and allowing two rows to ensure cut pixels do not join with pockets encoding signed colors along the edges of the cutout, we can cut pixels out of $\frac{3w-1}{2}$ rows (or $\frac{3w}{2}$, depending on parity). This allows up to $(\frac{nw+3n-2}{2})(\frac{3w-1}{2}) = \frac{n(w-4)^2+2w(nw-3)+13n+2}{4} + 4nw - 8n$ pixels to be cut out, but there are at most $4nw-8n$ monominoes, so we can always cut enough pixels without interfering with any other cuts.

Clue set $B$ contains:

- A *stamp* polyomino for each of the $2n$ edge segments of the top edge of the boundary-colored board. Each stamp is composed of a $w \times 2$ rectangle modified to encode the signed color on the corresponding edge segment (called the *die*), a pixel centered above that rectangle, and a $2 \times h$ rectangular *handle* whose bottom-right pixel is immediately above that pixel, where $h = \max(m(w+3)+7, n)$. Stamps corresponding to 1-indexed edge segments $2i$ and $2i+1$ have pockets encoding $i$ in binary cut into the left edge of their handle, starting from the second-to-top row of the handle.
- A "large" sized polyomino built from a $2n(w+3)+1 \times t$ rectangular polyomino, where $t$ is the total area of all other polyominoes so far defined. Modify this polyomino by cutting out the middle pixel of the bottom row, the $2n(w+3)-1 \times m(w+3)+3$ horizontally-centered rectangle immediately above that removed pixel, and the pixels above the upper-left and upper-right removed pixels. (That is, cut out space for the medium polyomino, including the horns and tail but not including the stamp accommodation zone.) Then cut out the image of each stamp in the order of their corresponding edge segments in the adversarial-boundary edge-matching instance, aligning the leftmost-bottom pixel of the first stamp's die two pixels to the right of the upper-left removed pixel and aligning successive dies immediately adjacent to one another.

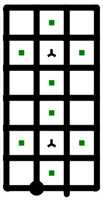


**Figure 14** Because both antibodies are surrounded by monominoes, any region containing an antibody also contains at least one monomino.

**Puzzle.** The Witness puzzle is a $2n(w+3)+1 \times t$ rectangle. The start circle and end cap are at the middle two vertices of the bottom row of vertices.

**Placement of $A$ clues.** We place a monomino from clue set $A$ in the cell having the start circle and end cap as vertices, then place an antibody above that monomino, surrounded by a monomino in each of its other three neighbors. We then place the other antibody, surrounded by monominoes in its neighboring cells, three cells above the first antibody. (See Figure 14.) It is always possible to surround the antibodies in this way because there are at least $8n$ monominoes. We place the remaining clues from clue set $A$ inside the $2n(w+3)-1 \times m(w+3)+3$ rectangle one row above the bottom of the puzzle; this is always possible because $|A| \leq 4nw - 8n + 2nm + 5$.

**Placement of $B$ clues.** We place the large polyomino clue in the upper-left cell of the board and the stamp clues in the $2n$ cells to its right.

**Argument.** In any valid solution to the resulting puzzle, the large polyomino is not eliminated. If it were, it must be in the same region as an antibody. Because each antibody is surrounded by monomino clues, the number of polyomino clues in this region is strictly greater than the number of antibodies, so the region must be packed by the non-eliminated polyomino clues. The nearest (upper) antibody is $t-4$ columns and $nw+3n$ rows away from the large polyomino clue, so this region has area at least $t$. Recall that $t$ is the total area of all polyomino clues except the large polyomino. If the large polyomino is eliminated, there is no way to pack this region, even if all other polyomino clues are used.

The large polyomino is as wide and as tall as the entire puzzle, so it has a unique placement. The large polyomino intersects its bounding box everywhere except one unit-length edge aligned with the start vertex and end cap, so any valid solution path can only touch the boundary at the start and end. Thus the solution path divides the puzzle into at most two regions (an inside and an outside).

Suppose the solution path places the entire puzzle into a single region; that is, suppose the solution path proceeds (in either direction) from the start vertex to the end cap without leaving the boundary. Then by the assumption that the first player has a losing strategy in the input adversarial-boundary edge-matching instance, we can pack the region while eliminating only one clue. The large polyomino's placement is fixed. We eliminate the medium polyomino, place the two board-frame polyominoes inside the large polyomino, and place the monominoes in the pixels cut out of the left board-frame polyomino. It remains to place the stamps and tiles. By the assumption, there is a losing set of top-boundary swaps; we swap the corresponding pairs of stamps when placing them into the cutouts in the large polyomino, and then place the tiles in the remaining uncovered area bordered by the board-frame polyominoes and stamp dies. Because we satisfied all non-antibody constraints after eliminating only one clue, the unused antibody is unsatisfied, so any solution path resulting in a single region is not a valid solution to the puzzle. Thus there are exactly two regions.

The cells containing the stamp clues are covered by the large polyomino, so any valid solution places the stamps in the same region as the large polyomino. The handles of the stamps are taller than the cutout in the bottom-middle of the large polyomino, so they must instead be placed in the stamp-shaped cutouts in the large polyomino. The pockets cut into the left edges of the handles ensure that stamps can only swap places corresponding to top-boundary swaps in the adversarial-boundary edge-matching instance.

All clues in set $A$ are in the other region. The monomino clue in the cell having both the start circle and end cap as vertices cannot be in the same region as the large polyomino (else the path could not divide the puzzle into two regions). Because each antibody is surrounded by monomino clues, the number of polyomino clues in this region is strictly greater than the number of antibodies, so the region must be packed by the non-eliminated polyomino clues. When both antibodies are used to eliminate clues, they must eliminate both board-frame polyominoes, and when only one is used, it must eliminate the medium polyomino; any other elimination leaves polyomino clues with too much or too little area to pack the area of the puzzle not yet covered by the large polyomino or the stamps. Thus either the medium polyomino or both board-frame polyominoes will not be eliminated. The medium polyomino and board-frame polyominoes have unique placements within the large polyomino determined by the horns and tail. The intersection of the outlines of these placements covers all the $A$ clues, so they are all in the same other region.

By this division of the clues into regions, any valid solution path traces the inner boundary of the large polyomino and the dies of the stamps (possibly after swapping some pairs). It remains to show that the solution path is valid exactly when the implied set of top-boundary swaps is a winning strategy in the adversarial-boundary edge-matching instance.

When using both antibodies to eliminate the board-frame polyominoes, the remaining polyominoes always pack their region. The medium polyomino's placement is fixed by the horns and tail; the stamp accommodation zone ensures this placement is legal regardless of the pattern of tabs on the dies of the stamps. The tile polyominoes fit directly into the cutouts in the medium polyomino and there are exactly enough monominoes to fill in the uncovered area in the stamp accommodation zone and the pockets of the dies.

The solution path is only valid if both antibodies are necessary. When using one antibody to eliminate the medium polyomino, the board-frame polyominoes' position is forced by the horns and tail. The monominoes are the only way to fill the single-pixel holes in the left board-frame polyomino and there are exactly enough monominoes to do so. Then the dies of the stamps and the edges of the rectangular cutout in the board-frame polyominoes models the boundary-colored board of the input adversarial-boundary edge-matching instance (see Figure 12). The tile polyominoes cannot pack this area, necessitating the second antibody and making the solution path valid, exactly when the set of top-boundary swaps is a winning strategy in the adversarial-boundary edge-matching instance. ◀

▶ **Theorem 18.** *It is $\Sigma_2$-complete to solve Witness puzzles containing one antibody, polyominoes and antipolyominoes.*

**Proof sketch.** As in the proof of Theorem 17, we reduce from adversarial-boundary edge-matching, and the reduction is similar. The primary difference is that the medium polyomino is also the singular board-frame polyomino. Besides the antibody and the tile polyominoes (same as before), clue set $A$ contains an antipolyomino called the *antikit* shaped like a 1-

pixel-wide tree with the tile polyominoes (as antipolyominoes) at the leaves and a polyomino shaped like the 1-pixel-wide tree (the *sprue*). The medium polyomino has the kit polyomino attached to its right side and a cutout for the sprue and for the boundary-colored board.

The stamps must be placed in the large polyomino as in the previous proof. When the antibody eliminates the medium polyomino, the antikit annihilates the sprue and tile polyominoes, leaving no (anti)polyominoes in the inner region (so it is trivially satisfied). When the antibody is not used, the antikit annihilates the kit-shaped part of the medium polyomino and the sprue fits in the cutout in the medium polyomino, leaving only a boundary-colored board for the tile polyominoes to be placed. Placing the tile polyominoes is impossible, necessitating the antibody and making the solution path valid, exactly when the top-boundary swaps are a winning strategy in the adversarial-boundary edge-matching instance. ◀

By Theorem 14, Theorem 17 and Theorem 18 are tight.

## 9 Metapuzzles

In this section, we analyze several of the *metapuzzles* that appear in The Witness. Metapuzzles are puzzles which have one or more puzzle panels as a sub-component of the puzzle, and in which solving the puzzle panel affects the surrounding world in a way that depends on the choice of solution that was used to solve the panel.

### 9.1 Sliding Bridges

The marsh area contains sliding bridges. In this metapuzzle, each bridge has a corresponding puzzle panel, and solving the puzzle causes the bridge to move into the position depicted by the outline of the solution path. The following theorem demonstrates that, regardless of the difficulty of the puzzle panels (i.e., even if it is easy to find all solutions of each individual panel), it is PSPACE-complete to solve sliding bridge metapuzzles.

▶ **Theorem 19.** *It is PSPACE-complete to solve Witness metapuzzles containing sliding bridges.*

**Proof sketch.** We straightforwardly construct the *one-way* and *door* gadgets of [2], which are known to be sufficient for PSPACE-completeness. ◀

### 9.2 Elevators and Ramps

Another metapuzzle which appears in The Witness consists of groups of platforms that move vertically at one or both ends to form an elevator or ramp, controlled by the path drawn on puzzle panels. Because the player cannot jump or fall in The Witness, the player can walk onto an elevator platform only if it is at the same height as the player. The player can adjust the height of the platforms from anywhere with line-of-sight to the controlling panel, including while on the platforms themselves. Besides the sawmill, the other building in the quarry contains a ramp and an elevator. The marsh contains a single puzzle with a $3 \times 3$ grid of elevators controlled by two identical panels; as a metapuzzle, our puzzle could be built out of multiple marsh puzzles with two platforms and one panel each.

▶ **Theorem 20.** *It is PSPACE-complete to solve Witness metapuzzles containing elevator reconfiguration, even when each panel controls at most one elevator.*

**Proof sketch.** We construct *one-way* and *door* gadgets similar to Theorem 19. ◀

## 9.3 Power Cables and Doors

In the introductory area of The Witness, there are panels with two solutions, each of which activates a power cable. Activated cables can power one other panel (allowing it to the solved) or one door (opening it). If a cable connected to a door is depowered, the door closes. Cables cannot be split and panels can power at most one cable at a time.

▶ **Theorem 21.** *It is PSPACE-complete to solve Witness metapuzzles containing power cables and doors.*

**Proof sketch.** Again we construct *one-way* and *door* gadgets, with the slight complication that all powered doors in The Witness are initially closed, so we need to give the player a way to open exactly the set of doors which are initially open in the source instance. ◀

### References

1 Zachary Abel, Jeffrey Bosboom, Erik D. Demaine, Linus Hamilton, Adam Hesterberg, Justin Kopinsky, Jayson Lynch, and Mikhail Rudoy. Who witnesses The Witness? Finding witnesses in The Witness is hard and sometimes impossible. arXiv:1804.10193, 2018.

2 Greg Aloupis, Erik D. Demaine, Alan Guo, and Giovanni Viglietta. Classic Nintendo games are (computationally) hard. *Theoretical Computer Science*, 586:135–160, 2015.

3 Erik D. Demaine and Martin L. Demaine. Jigsaw puzzles, edge matching, and polyomino packing: Connections and complexity. *Graphs and Combinatorics*, 23:195–208, June 2007.

4 Erik D. Demaine and Mikhail Rudoy. Tree-residue vertex-breaking: a new tool for proving hardness. In *Proceedings of the 16th Scandinavian Symposium and Workshops on Algorithm Theory*, 2018. arXiv:1706.07900.

5 Linus Hamilton. Braid is undecidable. arXiv:1412.0784, 2014.

6 Alon Itai, Christos H. Papadimitriou, and Jayme Luiz Szwarcfiter. Hamilton paths in grid graphs. *SIAM Journal on Computing*, 11(4):676–686, November 1982.

7 Donald E. Knuth and Arvind Raghunathan. The problem of compatible representatives. *SIAM Journal on Discrete Mathematics*, 5(3):422–427, 1992.

8 Irina Kostitsyna, Maarten Löffler, Max Sondag, Willem Sonke, and Jules Wulms. The hardness of Witness puzzles. In *Abstracts from the 34th European Workshop on Computational Geometry*, 2018.

9 Wikipedia. The Witness (2016 video game). https://en.wikipedia.org/wiki/The_Witness_(2016_video_game), 2018.

10 Takayuki Yato. On the NP-completeness of the Slither Link puzzle (in Japanese). *IPSJ SIG Notes*, AL-74:25–32, 2000.