Journal of Information Processing Vol.0 1–23 (??? 1992)

[DOI: 10.2197/ipsjjip.0.1]

Regular Paper

Edge Matching with Inequalities, Triangles, Unknown Shape, and Two Players

Jeffrey Bosboom[1,a)] Charlotte Chen[1,b)] Lily Chung[1,c)]
Spencer Compton[1,d)] Michael Coulombe[1,e)] Erik D. Demaine[1,f)]
Martin L. Demaine[1,g)] Ivan Tadeu Ferreira Antunes Filho[1,h)]
Dylan Hendrickson[1,i)] Adam Hesterberg[1,j)] Calvin Hsu[1,k)] William Hu[1,l)]
Oliver Korten[2,m)] Zhezheng Luo[1,n)] Lillian Zhang[1,o)]

Received: xx xx, xxxx, Accepted: xx xx, xxxx

Abstract: We analyze the computational complexity of several new variants of edge-matching puzzles. First we analyze inequality (instead of equality) constraints between adjacent tiles, proving the problem NP-complete for strict inequalities but polynomial-time solvable for nonstrict inequalities. Second we analyze three types of triangular edge matching, of which one is polynomial-time solvable and the other two are NP-complete; all three are #P-complete. Third we analyze the case where no target shape is specified and we merely want to place the (square) tiles so that edges match exactly; this problem is NP-complete. Fourth we consider four 2-player games based on $1 \times n$ edge matching, all four of which are PSPACE-complete. Most of our NP-hardness reductions are parsimonious, newly proving #P and ASP-completeness for, e.g., $1 \times n$ edge matching. Along the way, we prove #P- and ASP-completeness of planar 3-regular directed Hamiltonicity; we provide linear-time algorithms to find antidirected and forbidden-transition Eulerian paths; and we characterize the complexity of new partizan variants of the Geography game on graphs.

Keywords: complexity, puzzles and games, satisfiability, Hamiltonicity, Eulerian paths, Geography

1. Introduction

In an ***edge-matching puzzle***, we are given several tiles (usually identical in shape), where each tile has a label on each edge, and the goal is to place all the tiles (usually via translation and rotation) into a given shape such that shared edges between adjacent tiles have compatible labels. In ***unsigned*** edge matching, labels are compatible if they are identical (a matches a and nothing else). In ***signed*** edge matching the labels have signs (e.g., $+a$ and $-a$), and two labels are compatible if they are negations of each other ($+a$ matches $-a$ and nothing else, and vice versa). Physical edge-matching puzzles date back to the 1890s [26]; perhaps the most famous example is *Eternity II* which offered a US$2,000,000 prize for a solution before 2011 [29].

1.1 Previous Work

The complexity of edge-matching puzzles has been studied since 1966 [3]. The most relevant work to this paper is from two past JCDCG conferences. In 2007, Demaine and Demaine [10] proved that signed and unsigned edge-matching square-tile puzzles are NP-complete and equivalent to both jigsaw puzzles and polyomino packing puzzles. In 2016, Bosboom et al. [7] proved that signed and unsigned edge-matching square-tile puzzles are NP-complete even when the target shape is a $1 \times n$ rectangle, and furthermore hard to approximate within some constant factor. Our work on $1 \times n$ triangle edge-matching puzzles is inspired by an open problem proposed in the latter paper.

1.2 Our Results: Edge Matching

Table 1 summarizes our results in edge matching, described in more detail below.

Inequality edge matching.

Our most complex result is an NP-hardness proof for a new "<" compatibility condition, where edge labels are

1 MIT Computer Science and Artificial Intelligence Laboratory, 32 Vassar St., Cambridge, MA 02139, USA
2 Department of Computer Science, Tufts University, Medford, MA, USA
a) jbosboom@csail.mit.edu
b) charlotte_z_chen@yahoo.com
c) ikdc@mit.edu
d) scompton@mit.edu
e) coulomb@mit.edu
f) edemaine@mit.edu
g) mdemaine@mit.edu
h) ivanaf@mit.edu
i) dylanhen@mit.edu
j) achesterberg@gmail.com
k) ineq@mit.edu
l) whu2704@mit.edu
m) oliverkorten123@gmail.com
n) ezzluo@mit.edu
o) lillianz@mit.edu

Compatibility	Board	Tiles	Players	Complexity
$<$	$1 \times n$	square	1-player	NP-complete
$\leq$	$m \times n$	square	1-player	P
Signed/unsigned	$1 \times n$	square	1-player	NP/#P-complete, (2-)ASP-hard*
Signed/unsigned	$1 \times n$	equilateral triangle	1-player	NP/#P-complete, (2-)ASP-hard*
Signed/unsigned	$1 \times n$	right triangle (hypotenuse contact)	1-player	NP/#P-complete, (2-)ASP-hard*
Signed/unsigned	$\frac{\sqrt{2}}{2} \times n$	right triangle (leg contact)	1-player	$\in$ P, #P-complete
Signed/unsigned	$O(1) \times n$	square/triangular with $O(1)$ colors	1-player	$\in$ P
Signed/unsigned	shapeless	square	1-player	NP/#P/ASP-complete
Signed/unsigned	$1 \times n$	square	impartial 2-player	PSPACE-complete
Signed/unsigned	$1 \times n$	square	partizan 2-player	PSPACE-complete

Table 1: Our results on edge-matching puzzles. *Our proof gives ASP-completeness for $1 \times n$ edge matching only when at least one boundary edge is colored; otherwise, each solution can be rotated 180 degrees to form another valid solution, so we get 2-ASP-hardness (NP-hard to find a third solution given two).

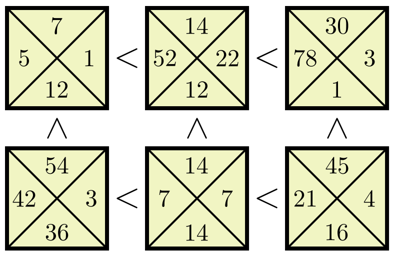


Fig. 1: A solved 2×3 $<$-compatible edge-matching puzzle. This solution is valid because $1 < 52$ and $22 < 78$ in the top row, $3 < 7$ and $7 < 21$ in the bottom row, and $12 < 54$, $12 < 14$, and $1 < 45$ in the columns.

numbers, horizontally adjacent edges match if the left edge's number is less than the right edge's number, and vertically adjacent edges match if the top edge's number is less than the bottom edge's number. Fig. 1 shows an example. In Section 2, we prove NP-hardness of $<$-compatible $1 \times n$ edge matching by reduction from another new NP-hard problem, Interval-Pair Cover. The $\leq$-compatibility condition (where equal numbers also match, or we assume all labels are distinct) turns out to be substantially easier: even rectangular puzzles turn out to be always solvable and we give a polynomial-time algorithm.

ASP/#P-completeness for $1 \times n$ edge matching.

In Section 3, we analyze edge matching for the first time from the perspective of the number of solutions to an instance, which is relevant to constructing puzzles with unique solutions. Specifically, we prove ASP-completeness for signed and unsigned $1 \times n$ edge-matching puzzles when the left boundary edge is colored (to prevent trivial 180° rotation of solutions), and 2-ASP-hardness and #P-completeness even if the boundary is colorless.

Recall the following definitions of FNP, ASP-complete, and #P-complete. ***FNP*** is a variant of NP that actually specifies the valid certificates/solutions for an instance (instead of just requiring that they exist); that is, an FNP problem is a relation between instances and polynomial-length certificates/solutions that can be checked in polynomial time. For edge matching problems, the certificate we consider is a valid placement of the given tiles within the given shape. An FNP problem Π is ***ASP-complete*** [30] if every FNP problem has a polynomial-time ***parsimonious reduction*** (preserving the number of solutions) to Π along with a polynomial-time bijection between solutions of the two problems. ASP-completeness implies that the ***k-ASP*** version of the FNP problem — given an instance and k solutions to it, determine whether there is another solution — is NP-hard [30]. An FNP problem is ***#P-complete*** [27] if counting the number of solutions is as hard as counting the number of solutions to any FNP problem, which is implied by a reduction that is ***c-monious***, meaning that it multiplies the number of solutions by a computable consistent factor $c \geq 1$.[*1] Our reductions to $1 \times n$ edge matching are the first to be parsimonious or, when global 180° rotation is allowed, 2-monious.

Triangular edge matching.

The conclusion of [7] claimed that the paper's results extended to equilateral-triangle edge matching, but the proposed simulation of squares by triangles is incorrect because it constrains the orientation of the simulated squares. In Section 4.1, we extend our $1 \times n$ parsimonious proof to obtain NP/#P/ASP-completeness for signed and unsigned edge matching with equilateral triangles, with or without reflection.

For right isosceles triangles, there are two natural "$1 \times n$" arrangements. For clarity, we assume the legs of the triangles have length 1. If we still want a height-1 tiling, then length-$\sqrt{2}$ hypotenuses are forced to match, so matching is NP/#P/ASP-complete by simulation of squares. But if we ask for a height-$\frac{\sqrt{2}}{2}$ tiling, so only legs match, we show in Section 4.2 that, surprisingly, both signed and unsigned edge matching can be solved in polynomial time using an algorithm based on Eulerian paths. Nonetheless, the latter problems are still #P-complete.

Shapeless edge matching.

In Section 5, we prove that square-tile edge-matching puzzles remain NP/#P/ASP-complete when the goal is to con-

[*1] This terminology naturally generalizes "parsimonious" ($c = 1$), and was introduced in an MIT class in 2014 [9].

nect all tiles in any (unspecified) single connected shape, with either signed or unsigned compatibility. For #P- and ASP-completeness, we need to give one tile a fixed position in the plane (translation and rotation) to make the number of solutions finite. The proof builds a unique spiral frame that effectively forces a $1 \times n$ edge-matching puzzle with a fixed left boundary color.

2-player edge matching.

In Section 6, we consider natural 2-player variants of $1 \times n$ edge-matching puzzles, where the left boundary edge of the rectangle has a prespecified color, players alternate placing a tile in the leftmost empty cell that matches the edge color to the left, and the first player unable to move loses (***normal play***). We prove PSPACE-completeness for four variants of this problem: both signed and unsigned square-tile edge matching, and both when players can play any remaining tile from a shared pool (***impartial***) and when players play from separate pools of tiles (***partizan***).

1.3 Our Results: Not Edge Matching

Along the way to proving our results on edge matching, we derive other results of possible independent interest in graph algorithms/complexity.

Hamiltonicity parsimony.

In Section 3.1, we prove #P- and ASP-completeness of the Hamiltonian cycle problem in planar 3-regular *directed* graphs, by modifying the clause gadget of Plesník's *NP*-hardness proof [23] and parsimoniously reducing from 1-in-3SAT instead of 3SAT. Previous work showed the analogous *undirected* problem ASP-complete (and #P-complete) in planar graphs of maximum degree 3 [25]. We also prove #P- and ASP-completeness of the Hamiltonian path problem with specified start and end vertices in planar 3-regular directed graphs.

Antidirected Eulerian paths.

In Section 4.2.1, we characterize when a directed graph admits an ***antidirected Eulerian path*** [4], [13], [31], that is, a path[*2] that alternates between going forward and going backward along directed edges and visits every edge (in either direction) exactly once. (Such directed graphs are called ***aneulerian*** [4], [13], [31].) Specifically, we show how to reduce this problem to finding an Eulerian path in a modified graph, enabling solution in linear time. Although antidirected Eulerian paths were introduced over 50 years ago [4], their existence does not seem to have been characterized before our work and a recent independent discovery [1].

Forbidden-transition Eulerian paths.

In Section 4.2.2, we give linear-time algorithms to find Eulerian paths or antidirected Eulerian paths when certain monochromatic edge-to-edge transitions are forbidden, extending past work by Kotzig [20] to be algorithmic (and to the antidirected case). Specifically, each vertex can define a partition of its incident edges into groups, and the problem forbids the Eulerian path from passing through the vertex via two edges from the same group.

[*2] Throughout this paper, we follow the half-standard terminology that paths and cycles are allowed to repeat vertices and/or edges (though we will rarely allow repeated edges). In a different half-standard terminology, these notions are called "walks/trails" and "circuits". If a path or cycle makes no such repetitions, it is called ***simple***.

Graph	Partizan	Geography	Complexity
undirected	vertex	vertex	polynomial
undirected	vertex	edge	polynomial
undirected	edge	vertex	PSPACE-complete
undirected	edge	edge	PSPACE-complete
directed	vertex	vertex	PSPACE-complete
directed	vertex	edge	PSPACE-complete
directed	edge	vertex	PSPACE-complete
directed	edge	edge	PSPACE-complete

Table 2: Partizan geography results

Partizan Geography game.

We introduce eight new *partizan* variants of Geography where the two players have different available moves, and characterize their complexity. Specifically, in vertex-partizan geography, vertices have two different colors, and each player can only move to vertices of their color; while in edge-partizan geography, edges have two different colors, and each player can only move along edges of their color. We can consider either variant for both Vertex and Edge Geography (where respectively vertices and edges cannot be repeated by either player), and in directed or undirected graphs, resulting in eight possible variants. Table 2 summarizes our results from Section 6.1, which prove every variant either polynomial or PSPACE-complete.

2. Edge Matching with Inequalities

In this section, we analyze the complexity of the following problems:

Definition 2.1. ***$m \times n$ <-compatible edge matching*** is the following problem:

Instance: mn unit-square tiles, where each tile is defined by four numbers, one for each side. We use ${}_{a}\overset{b}{\underset{d}{\square}}{}_{c}$ to represent a unit-square tile with numbers a, b, c, d.

Question: Can the mn tiles cover an $m \times n$ rectangle such that

- for every two horizontally adjacent tiles, the left tile's right number is strictly less than the right tile's left number; and
- for every two vertically adjacent tiles, the top tile's bottom number is strictly less than the bottom tile's top number?

The related problem ***≤-compatible edge matching*** is defined similarly, except that we do not require strict inequalities among the numbers.

2.1 Polynomial-Time Algorithm for ≤-Compatible Edge Matching

Theorem 2.1. *$m \times n$ ≤-compatible edge-matching puzzles are always solvable and a solution can be found in $O(mn \log(mn))$ time.*

Proof. Rotate each tile $A\overset{B}{\underset{D}{\square}}C$ such that $A \geq C$ and $B \geq D$.

Then sort the tiles in ascending order by D and place them in the board in row-major order. Because $B \geq D$, sorting by D ensures all tiles are vertically $\leq$-compatible. Then sort the tiles in each row in ascending order by C. Because $A \geq C$, sorting by C ensures all tiles in the row are horizontally $\leq$-compatible. Being both vertically and horizontally $\leq$-compatible, this is a compatible tiling. This algorithm runs in $O(mn \log(mn))$ time from the sorting. □

The following special cases of the $m \times n$ $<$-compatible edge-matching puzzles are tractable:

Corollary 2.2. *$m \times n$ $<$-compatible edge-matching puzzles in which all edge labels are distinct are always solvable and a solution can be found in polynomial time.*

Theorem 2.3. *$1 \times n$ $<$-compatible edge-matching puzzles in which every tile has at least one pair of parallel sides with unequal labels are always solvable and a solution can be found in polynomial time.*

Proof. Rotate each tile $\overset{B}{\underset{D}{A\square C}}$ such that $A > C$. If there are two pairs of unequal parallel sides, then choose arbitrarily. Now sort all tiles in ascending order by A, breaking ties arbitrarily, and place them in the board in row-major order. Let A_i and C_i be the left and right numbers of tile i. From sorting, we know that $A_i \leq A_{i+1}$, and from our rotation of the tiles, we know that $C_i < A_i$. Composing the inequalities gives $C_i < A_{i+1}$, which is the $<$-compatibility condition, so this is a compatible tiling. □

2.2 NP-hardness of $1 \times n$ $<$-Compatible Edge Matching

To show NP-hardness of $<$-compatible edge matching, we start from the known NP-hard problem N3P-3SAT-2P-E1N [11] defined in Section 2.2.1. In Section 2.2.2, we reduce N3P-3SAT-2P-E1N to a novel variant literal-matching N3P-3SAT-2P-E1N. In Section 2.2.3, we reduce literal-matching N3P-3SAT-2P-E1N to a new problem called Interval-Pair Cover, which implies NP-hardness of $1 \times n$ $<$-compatible edge matching.

2.2.1 N3P-3SAT-2P-E1N

Our starting point is the following variant of SAT (named to follow notation from [12]):

Definition 2.2. An instance of ***N3P-3SAT-2P-E1N*** is an instance of 3SAT, consisting of n variables $x_1, x_2, \ldots, x_n$ and m clauses each with at most three literals, where each literal is of the form x_i (positive) or $\neg x_i$ (negative), satisfying the following constraints:

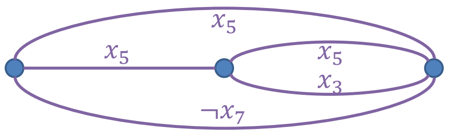
(1) ***N3P***: Every clause has at least one negative literal (i.e., no clause has three positive literals).

(2) ***2P***: Every variable x_i appears in at most two positive literals x_i.

(3) ***E1N***: Every variable x_i appears in exactly one negative literal $\neg x_i$.

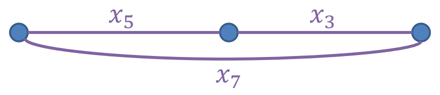
Ding et al. [11] proved that N3P-3SAT-2P-E1N is NP-complete. In fact, they proved NP-completeness of a slightly more general problem, N3P-3SAT-3-1N, which constrains each variable to appear in at most three literals, at most one of which is negative. But any variable with zero negative occurrences can be eliminated (setting it to true), so by repeated application of this process, we attain the E1N property. Because each variable appears in at most three literals, at most two of them are positive, so we also have the 2P property. Thus we reduce N3P-3SAT-3-1N to N3P-3SAT-2P-E1N.

$(x_1 \vee x_5 \vee \neg x_7) \wedge (\neg x_1 \vee x_3 \vee x_5) \wedge (x_3 \vee x_5 \vee \neg x_7)$



(a) 3SAT formula

$(\neg x_2 \vee x_5 \vee x_7) \wedge (\neg x_1 \vee x_3 \vee x_5) \wedge (x_3 \vee \neg x_4 \vee x_7)$



(b) N3P-3SAT-2P-1N formula

Fig. 2: Shared-literal graph: two examples.

2.2.2 Reduction from N3P-3SAT-2P-E1N to literal-matching N3P-3SAT-2P-E1N

Define the ***shared-literal graph*** of a 3SAT instance to have one vertex for each clause, and connect two clauses by an edge for each literal these share, as shown in Fig. 2. For a N3P-3SAT-2P-E1N instance, the shared-literal graph has two additional properties. By the E1N constraint, every edge corresponds to a shared *positive* literal. By the 2P property, the shared-literal graph has maximum degree 2. We will show that we can in fact reduce the shared-literal graph to maximum degree 1.

Definition 2.3. A ***literal-matching N3P-3SAT-2P-E1N*** instance is an instance of N3P-3SAT-2P-E1N whose shared-literal graph is a matching.

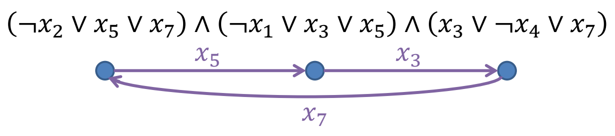
Theorem 2.4. *Literal-matching N3P-3SAT-2P-E1N is NP-complete.*

Proof. Trivially, literal-matching N3P-3SAT-2P-E1N $\in$ NP. We reduce N3P-3SAT-2P-E1N to literal-matching N3P-3SAT-2P-E1N to show literal-matching N3P-3SAT-2P-E1N is NP-hard. Refer to Fig. 3.

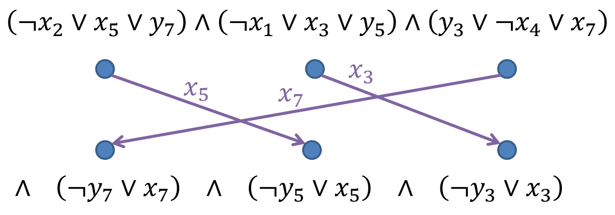
First we orient the shared-literal graph to have maximum indegree and maximum outdegree 1. Because the shared-literal graph is maximum degree 2, every connected component is either a path or a cycle. Direct each path from one end to the other, and direct each cycle cyclically.

Reduction: For each edge (c, d) in the directed shared-literal graph, corresponding to a shared literal x_i, replace the occurrence of x_i in d with a new helper variable h_i. Additionally, create a new helper clause $\neg h_i \vee x_i$, i.e., $h_i \Rightarrow x_i$.

This reduction conserves occurrences of the original (non-helper) variables, and each helper variable appears positively


$(\neg x_2 \vee x_5 \vee x_7) \wedge (\neg x_1 \vee x_3 \vee x_5) \wedge (x_3 \vee \neg x_4 \vee x_7)$
x_5
x_3
x_7

(a) Oriented N3P-3SAT-2P-E1N instance from Fig. 2b


$(\neg x_2 \vee x_5 \vee y_7) \wedge (\neg x_1 \vee x_3 \vee y_5) \wedge (y_3 \vee \neg x_4 \vee x_7)$
x_5
x_7
x_3
$\wedge \quad (\neg y_7 \vee x_7) \quad \wedge \quad (\neg y_5 \vee x_5) \quad \wedge \quad (\neg y_3 \vee x_3)$

(b) Reduced literal-matching N3P-3SAT-2P-E1N instance

Fig. 3: Reduction from N3P-3SAT-2P-E1N to literal-matching N3P-3SAT-2P-E1N of Theorem 2.4.

once (replacing some x_i in an original clause) and negatively once (in the helper clause), so the transformed formula is still N3P-3SAT-2P-E1N.

The transformed formula is satisfiable under an augmented truth assignment $\sigma_{X,H} = \sigma_X \cup \sigma_H$ if and only if the original formula is satisfiable under σ_X. If h_i satisfies an original clause (by being true), the helper clause ensures that x_i is also true. If x_i is false, the helper clause ensures that h_i is also false, and so cannot satisfy the original clause it is a member of. Thus if $\sigma_{X,H}$ satisfies the transformed formula, σ_X satisfies the original formula. Variable h_i can be false when x_i is true, but as x_i already satisfies h_i's helper clause and h_i always appears positively in its original clause, such an assignment cannot satisfy more clauses than if h_i were true. Thus if σ_X satisfies the original formula, $\sigma_{X,H} = \sigma_X \cup \{h_i = \sigma_X(x_i)\}$ satisfies the transformed formula.

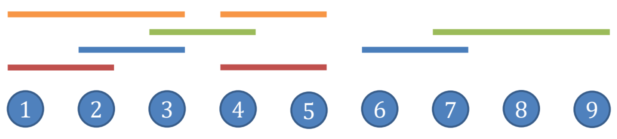
After replacing the occurrence of x_i in clause d, each edge (c, d) in the original formula's directed shared-literal graph corresponds to an edge between c and the helper clause containing x_i in the transformed formula's shared-literal graph, so original clauses have a degree of at most 1. Each helper variable appears only once in each polarity, so helper variables do not give rise to edges in the shared-literal graph. Thus all helper clauses have degree 1. The transformed formula's shared-literal graph thus has maximum degree 1, meaning that it is a matching. □

2.2.3 Reduction from literal-matching N3P-3SAT-2P-E1N to Interval-Pair Cover

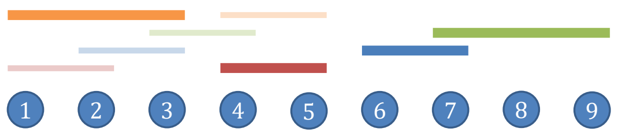
To begin, we define a new problem Interval-Pair Cover; refer to Fig. 4.

Definition 2.4. ***Interval-pair cover*** is the following problem:

Instance: A universe $U = \{1, 2, \dots, n\}$ and m pairs of closed intervals $([a_i, b_i], [c_i, d_i])$ for $i = 1, 2, \dots, m$. Here $a_i, b_i, c_i, d_i \in U$, $a_i \le b_i$, and $c_i \le d_i$.



(a) Input



(b) Solution

Fig. 4: Interval-Pair Cover: sample input and solution. The two intervals in the same pair are colored the same and share the same y coordinate.

Question: Is there a choice of one interval from each pair such that every $i \in U$ is covered by some chosen interval?

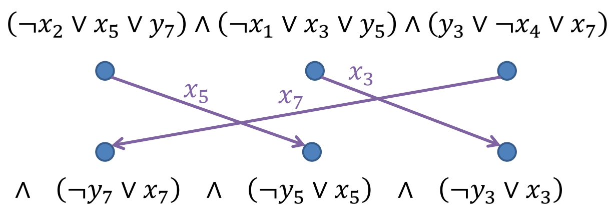
Theorem 2.5. *Interval-Pair Cover is NP-complete, even when every interval pair $([a_j, b_j], [c_j, d_j])$ satisfies $a_j = b_j$ and $d_j - c_j \in \{0, 1\}$.*

Proof. We reduce from literal-matching N3P-3SAT-2P-E1N; refer to Fig. 5. We draw the shared-literal graph on the integer line from 1 through n, placing the vertices at integer coordinates and using unit-length edges. This is always possible because the shared-literal graph is a matching. Then we create an interval pair for each variable x_i. The pair's first interval contains only the coordinate of the vertex representing the clause where x_i appears negatively; by the E1N property, there is exactly one such vertex. The pair's second interval contains only the coordinate(s) of the vertex or vertices representing the clause(s) where x_i appears positively; by the 2P property, there are at most two, and they are adjacent on the line because they share an edge in the shared-literal graph. If x_i does not appear positively, we set the second interval equal to the first interval.

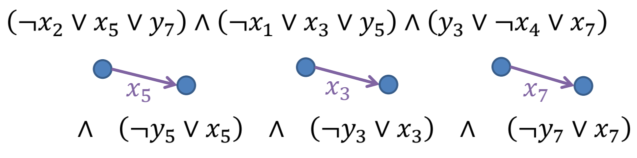
The produced Interval-Pair Cover instance has a solution if and only if the input literal-matching N3P-3SAT-2P-E1N instance is satisfiable. Given a satisfying truth assignment, from the interval pair corresponding to variable x_i, we choose the first interval if x_i is assigned false and the second interval if x_i is assigned true. Each chosen interval covers the coordinate(s) of the clause vertices satisfied by x_i, so if the truth assignment satisfies the formula, the chosen intervals cover all integers in the universe. Given a complete interval cover, we assign true to x_i if the second interval was chosen from its corresponding pair and false if the first interval was chosen. By the same interval-variable correspondence, if the intervals cover all integers in the universe, the constructed truth assignment satisfies the formula. □

2.2.4 Reduction from Interval-Pair Cover to $1 \times n$ <-Compatible Edge Matching

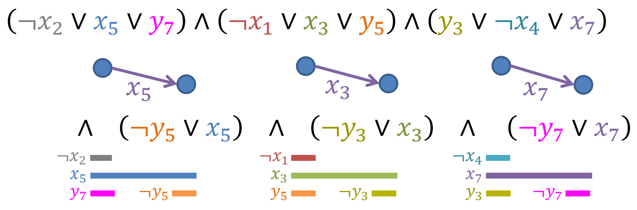
Theorem 2.6. *$1 \times n$ $<$-compatible edge matching is NP-complete.*


$(\neg x_2 \vee x_5 \vee y_7) \wedge (\neg x_1 \vee x_3 \vee y_5) \wedge (y_3 \vee \neg x_4 \vee x_7)$
x_5
x_7
x_3
$\wedge \ (\neg y_7 \vee x_7) \ \wedge \ (\neg y_5 \vee x_5) \ \wedge \ (\neg y_3 \vee x_3)$

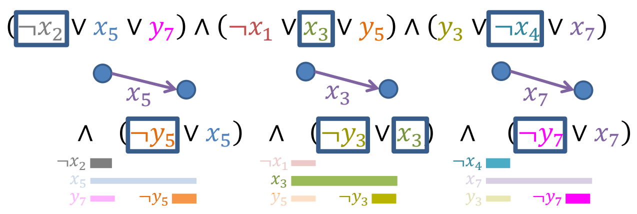
(a) Literal-matching N3P-3SAT-2P-E1N instance


$(\neg x_2 \vee x_5 \vee y_7) \wedge (\neg x_1 \vee x_3 \vee y_5) \wedge (y_3 \vee \neg x_4 \vee x_7)$
x_5
x_3
x_7
$\wedge \ (\neg y_5 \vee x_5) \ \wedge \ (\neg y_3 \vee x_3) \ \wedge \ (\neg y_7 \vee x_7)$

(b) Short drawing


$(\neg x_2 \vee x_5 \vee y_7) \wedge (\neg x_1 \vee x_3 \vee y_5) \wedge (y_3 \vee \neg x_4 \vee x_7)$
x_5
x_3
x_7
$\wedge \ (\neg y_5 \vee x_5) \ \wedge \ (\neg y_3 \vee x_3) \ \wedge \ (\neg y_7 \vee x_7)$
$\neg x_2$
x_5
y_7
$\neg y_5$
$\neg x_1$
x_3
y_5
$\neg y_3$
$\neg x_4$
x_7
y_3
$\neg y_7$

(c) Interval-pair cover instance


$(\boxed{\neg x_2} \vee x_5 \vee y_7) \wedge (\neg x_1 \vee \boxed{x_3} \vee y_5) \wedge (y_3 \vee \boxed{\neg x_4} \vee x_7)$
x_5
x_3
x_7
$\wedge \ (\boxed{\neg y_5} \vee x_5) \ \wedge \ (\boxed{\neg y_3} \vee \boxed{x_3}) \ \wedge \ (\boxed{\neg y_7} \vee x_7)$
$\neg x_2$
x_5
y_7
$\neg y_5$
$\neg x_1$
x_3
y_5
$\neg y_3$
$\neg x_4$
x_7
y_3
$\neg y_7$

(d) Solution

Fig. 5: Reduction from literal-matching N3P-3SAT-2P-E1N to Interval-Pair Cover of Theorem 2.5.

Proof. We reduce from Interval-Pair Cover. For each integer i in the Interval-Pair Cover universe $\{1, 2, \ldots, n\}$, we create two copies of the element tile ${}_{i}\overset{i}{\underset{i}{\square}}{}_{i}$. For each interval pair $([a_j, b_j], [c_j, d_j])$, we create an interval-pair tile ${}_{a_j-1}\overset{c_j-1}{\underset{d_j+1}{\square}}{}_{b_j+1}$. The edge-matching board is $1 \times (2n + m)$, where n is the size of the universe and m is the number of interval pairs.

Given a solution to the produced edge-matching instance, we can construct a solution to Interval-Pair Cover by choosing each interval tile's horizontally-oriented interval (e.g., the interval $[a_j, b_j]$ for a tile oriented as ${}_{a_j-1}\overset{c_j-1}{\underset{d_j+1}{\square}}{}_{b_j+1}$ or as ${}_{b_j+1}\overset{d_j+1}{\underset{c_j-1}{\square}}{}_{a_j-1}$). Suppose for contradiction that an element i is uncovered by every chosen interval. Then in every placed tile whose left edge is at least $i + 1$, its right edge is at least i, so the left edge of the next tile is at least $i + 1$. In the sequence of left edges of tiles, the left edge of the tile after the first copy of ${}_{i}\overset{i}{\underset{i}{\square}}{}_{i}$ is at least $i + 1$, so every following left edge is at least $i + 1$, leaving no place for the second copy of ${}_{i}\overset{i}{\underset{i}{\square}}{}_{i}$.

Given a solution to Interval-Pair Cover, we can construct a solution to the produced edge-matching instance. We will first describe a solution that uses extra copies of ${}_{i}\overset{i}{\underset{i}{\square}}{}_{i}$. For each chosen interval $[a_j, b_j]$, orient the tile as ${}_{b_j+1}\overset{d_j+1}{\underset{c_j-1}{\square}}{}_{a_j-1}$, and attach to its right ${}_{a_j}\overset{a_j}{\underset{a_j}{\square}}{}_{a_j}, \ldots, {}_{b_j}\overset{b_j}{\underset{b_j}{\square}}{}_{b_j}$ to get a sequence of tiles with left edge $b_j + 1$ and right edge b_j. For each $i \in \{1, 2, \ldots, n\}$, place the tile ${}_{i}\overset{i}{\underset{i}{\square}}{}_{i}$ followed by any of the above sequences of tiles with left edge $i + 1$ and right edge i. This solution uses as many copies of ${}_{i}\overset{i}{\underset{i}{\square}}{}_{i}$ as the number of intervals that cover i, plus 1, which is at least two. We can remove any ${}_{i}\overset{i}{\underset{i}{\square}}{}_{i}$ and leave a valid solution, so arbitrarily removing copies until there are two copies of each ${}_{i}\overset{i}{\underset{i}{\square}}{}_{i}$ left leaves a solution to the edge-matching instance. □

3. 1 × *n* Edge Matching ASP/#P-completeness

In this section, we adapt the work of [7] to show that $1 \times n$ edge-matching puzzles are ASP- and #P-complete. Like [7], we reduce from Hamiltonian path in planar 3-regular directed graphs, which we newly prove ASP- and #P-complete.

3.1 Directed Hamiltonicity ASP/#P-completeness

Seta's thesis [25] proves ASP-completeness for Hamiltonicity in planar maximum-degree-3 *undirected* graphs. Here we prove the analogous result for *directed* graphs:

Theorem 3.1. *Finding Hamiltonian cycles in a planar 3-regular directed graph with maximum indegree 2 and maximum outdegree 2 is ASP-complete, and counting Hamiltonian cycles in those graphs is #P-complete.*

Proof. These problems are clearly in FNP and #P respectively. To prove hardness, we give a parsimonious reduction from (planar) positive 1-in-3SAT, which is known to be ASP-complete and #P-complete [19].[*3] Our reduction is patterned after Plesník's NP-hardness reduction from 3SAT for Hamiltonian cycle in this class of graphs [23]. Plesník's reduction does not conserve the number of solutions because the clause gadget admits multiple solutions when multiple literals in the clause are satisfied (Fig. 9a). Reducing from 1-in-3SAT and simplifying Plesník's clause gadget allows us to conserve the number of solutions, and reducing from *positive* 1-in-3SAT (no negated literals) allows us to simplify the clause gadget. Plesník's exclusive-or gadget and exclusive-or crossover gadget do not give rise to additional solutions,

[*3] Our proof does not actually use the planarity of the 1-in-3SAT instance. To avoid the exclusive-or crossover gadget, we would need the variable-clause graph to remain planar with a line through all of the variables and all of the clauses, a variant not known hard [12].

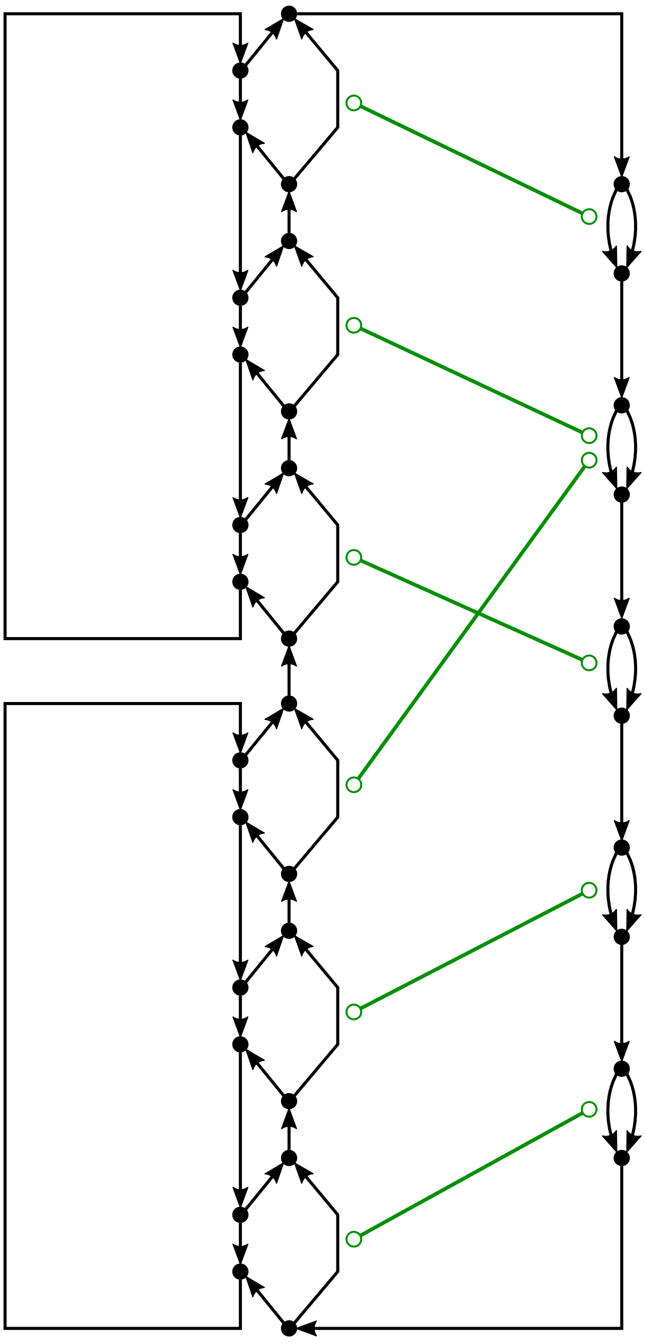


Fig. 6: A full Hamiltonicity instance produced by our reduction, with variable gadgets on the right (heading down) and clause gadgets on the left (heading up). Variables and clauses are connected by exclusive-or lines (the green lines with hollow endpoints) as defined in Fig. 7, with crossings expanded as in Fig. 8.

so they can be used without modification.

Fig. 6 shows a full Hamiltonicity instance produced by our reduction, with variable gadgets on the right (heading down) and clause gadgets on the left (heading up), and variables and clauses connected by exclusive-or lines (the green lines with hollow endpoints) which may cross. (Compare [23], Figure 1, in which Plesník has abbreviated the clause gadgets.)

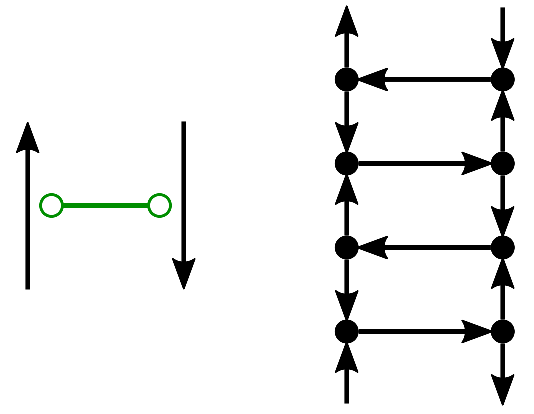


Fig. 7: Our notation for an exclusive-or line between two edges and its expansion into additional vertices and edges. (Redrawing of [23], Figure 4.)

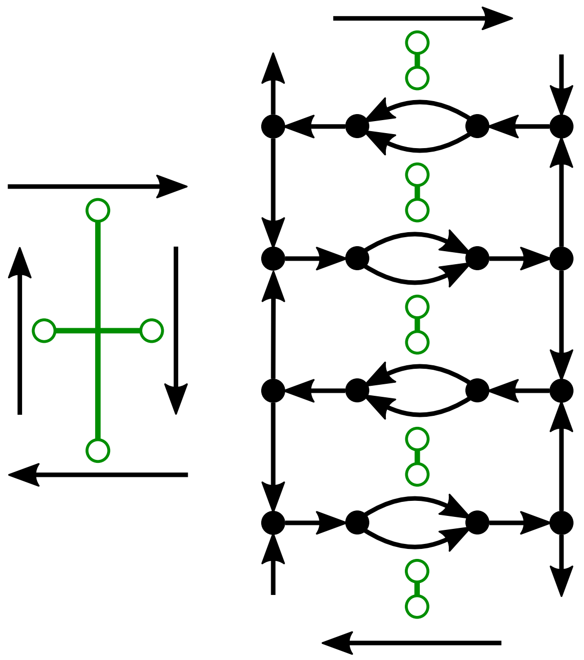


Fig. 8: Expansion of an exclusive-or line that crosses another exclusive-or line. (Based on [23], Figure 5, simplified to show only two lines crossing.)

Exclusive-or line.

An exclusive-or line between two edges abbreviates the pattern of additional vertices and edges shown in Fig. 7. Traversing either of the two edges covers all of the additional vertices in exactly one way, excluding the other original edge from the cycle. Traversing a path not corresponding to one of the original edges (e.g., from the bottom left to bottom right in Fig. 7) prevents the center four vertices from being part of any cycle (either they are uncovered, or they are the last four vertices in the path, so the path is not a cycle). If neither of the two original edges is used, all of the additional vertices are uncovered.

Exclusive-or crossover.

Exclusive-or lines connecting variable gadgets to clause gadgets may cross, necessitating the exclusive-or crossover shown in Fig. 8. The crossover works by splitting each crossed-over edge between one pair of original edges into two edges and adding new exclusive-or lines that guarantee the parity of these paired edges is the same throughout the gadget. For example, in Fig. 8, if the top edge is in the cycle, then the top edge of each pair is also in the cycle and the bottom edge is not in the cycle, regardless of which of the left or right edges are in the cycle. As before, the expansion can be traversed in exactly one way for each pair of original edges traversed, and a traversal not corresponding to an original edge leaves some vertices uncovered.

Variable gadget.

The variable gadget is a pair of vertices connected by a pair of parallel edges.[*4] The edge on the interior face of the variable-clause cycle is connected by exclusive-or lines to each clause in which the variable appears. Including this edge in the Hamiltonian cycle represents setting the variable to true. The other edge of the variable gadget is not connected to anything and represents setting the variable to false. The variable gadgets are connected in sequence.

Plesník's variable gadget used two pairs of parallel edges, connected on the exterior by an exclusive-or line such that they have opposite settings, with the second pair connected to clauses where the variable appeared as a negative literal. We reduce from planar *positive* 1-in-3SAT, so all literals in our clauses are positive, making the second pair unnecessary.

Clause gadget.

Our clause gadget and its three Hamiltonian paths are shown in Fig. 9b. The three rightmost edges in the clause gadget are connected by exclusive-or lines to the variable gadgets corresponding to the variables appearing in this clause. If a variable is set to true, then the rightmost edge connected to that variable gadget cannot be in the cycle; otherwise, the rightmost edge *must* be in the cycle. If exactly one of the three variables is true, then the clause gadget can be covered in exactly one way (using one of the paths shown in Fig. 9b). If a variable is true, the path must go to the left of that hexagon, where it must enter the left loop. If the path leaves the left loop before visiting all vertices in it, it cannot visit the top vertex of the hexagon where it entered the loop, so the left loop must be covered in its entirety. But then the path cannot go left in any other hexagon, so the other variable must be false. If all variables are false, the left loop is uncovered. Thus this gadget simulates a 1-in-3SAT clause.

Our clause gadget differs from Plesník's by the deletion of the "bridges" between the hexagons and the left loop. The bridges allowed multiple literals to be simultaneously true, which was necessary for Plesník's reduction (from 3SAT), but not desired for our reduction from 1-in-3SAT.

Conclusion.

Fig. 6 shows a full instance produced by our reduction. For each satisfying assignment of the variables, there is one corresponding Hamiltonian cycle using the corresponding configuration of the variable gadgets and the unique satisfying path through each clause gadget. Conversely, a satisfying assignment can be uniquely read off from each Hamiltonian cycle based on the configuration of the variable gadgets. □

[*4] The graph is a simple graph, not a multigraph: If we remove any variables not used in any clauses, then for each variable, one of these edges will be replaced by an exclusive-or gadget, leaving no parallel edges.

Theorem 3.2. *Finding Hamiltonian paths, with or without given start vertex s and/or end vertex t, in planar 3-regular directed graphs with maximum indegree 2 and maximum outdegree 2 is ASP-complete, and counting Hamiltonian paths in those graphs is #P-complete. The same result holds when the given vertex s has outdegree 1 and the given vertex t has indegree 1.*

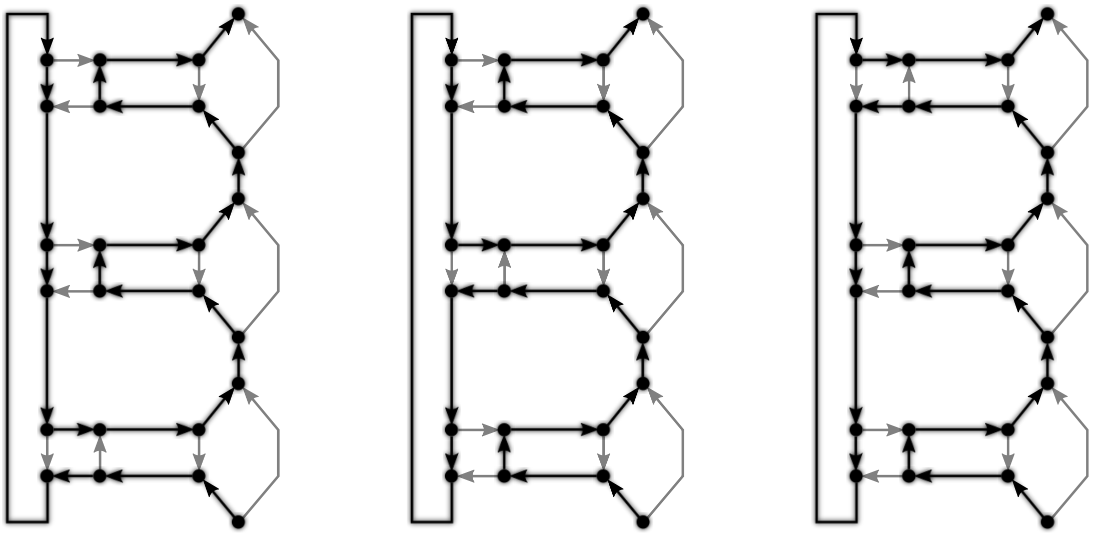
Proof. We prove this result via a parsimonious reduction from Hamiltonian cycle in planar 3-regular graphs with maximum indegree and outdegree 2. Given a 3-regular directed graph, we find an edge uv that must be in every Hamiltonian cycle (an outgoing edge from a vertex with indegree 2, or an incoming edge to a vertex with outdegree 2). We split uv, introducing two degree-1 vertices but otherwise leaving the graph 3-regular.

To restore 3-regularity we replace the degree-1 vertices with the graphs shown in Fig. 10. The unique longest (simple) path entering the graph in Fig. 10b ends at the vertex labeled t, because the first three vertices have outdegree 1 and the other successor of the fourth vertex is already in the path. By a similar argument working backwards from the outgoing edge of the graph in Fig. 10a, the unique longest path leaving the graph starts at the vertex labeled s. Thus, whether or not s and t are specified as the start and end vertices in the Hamiltonian path instance, all Hamiltonian paths in the transformed graph start at s and end at t. Vertex s has outdegree 1 and t has indegree 1, as claimed in the theorem statement. Because uv occurs in every Hamiltonian cycle of the input graph, there is a bijection between Hamiltonian cycles in the input instance and Hamiltonian paths in the output instance, and this bijection can be computed in polynomial time by replacing uv with the unique paths in the start/end gadgets or vice versa. □

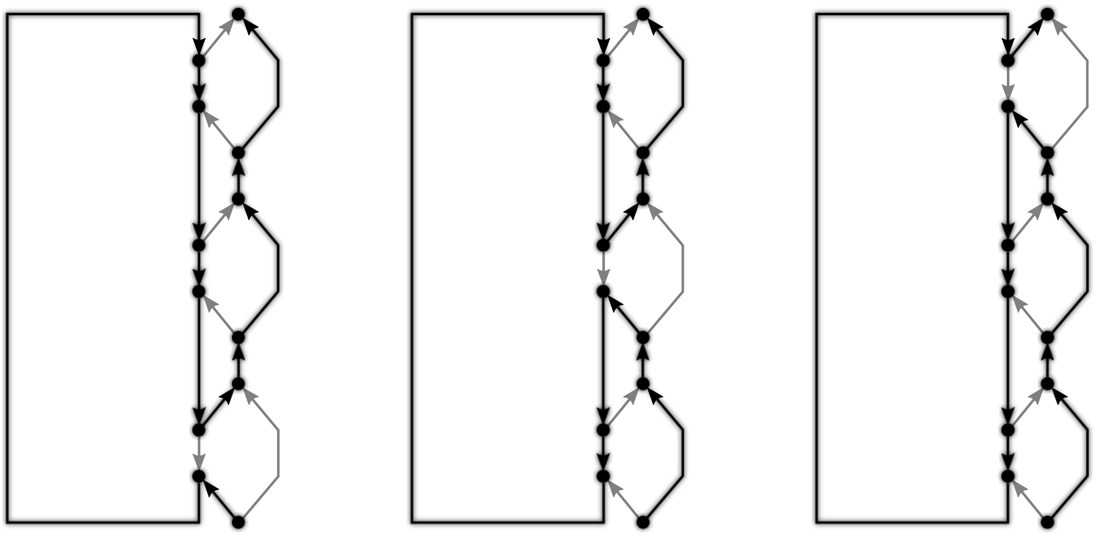
3.2 Reduction from Hamiltonicity to $1 \times n$ Edge Matching

The symmetry of $1 \times n$ edge-matching puzzles is problematic for ASP-hardness. Because rotating any solution by 180° will give another solution, the answer to the ASP problem is always 'yes'. To avoid this trivial additional solution, we consider the version of $1 \times n$ edge-matching puzzles where the left boundary edge's color is specified. This breaks the rotational symmetry, and we will show that this problem is ASP-complete through a parsimonious reduction. Without this restriction, our reduction is 2-monious, so we show #P-hardness even for $1 \times n$ edge-matching puzzles without any such restriction.

The reduction in [7] that establishes NP-hardness of $1 \times n$ edge-matching puzzles is not parsimonious because of

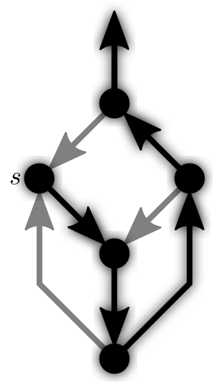


(a) The three Hamiltonian paths through Plesník's clause gadget [23], Fig. 2 when all three literals are true. (The right edge of each hexagon is covered via the exclusive-or line from the variable gadget.)

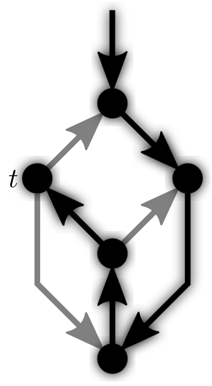


(b) The three Hamiltonian paths through our modified clause gadget. The right edges of two of the three hexagons are always used, so this is a 1-in-3SAT clause.

Fig. 9: Comparison of Plesník's clause gadget and our modified clause gadget.



(a) 3-regular start vertex gadget



(b) 3-regular end vertex gadget

Fig. 10: Gadgets that replace degree-1 start or end vertices to restore 3-regularity to the overall graph while maintaining a unique Hamiltonian path. Vertices s and t are the new start and end vertices.

garbage collection. Namely, the tiles corresponding to edges which are not part of the Hamiltonian path are placed at the end of the row of tiles in an arbitrary order. Our reduction will instead place these unused tiles near the corresponding vertex tiles so that there is only one tile sequence corresponding to each Hamiltonian path.

Theorem 3.3. *$1 \times n$ signed and unsigned edge-matching puzzles with the left boundary edge color specified are ASP-complete and #P-complete.*

Proof. Clearly this problem is in FNP and its counting problem is in #P. To show hardness, we present a parsimonious reduction from Hamiltonian path in 3-regular directed graphs, adapted from the reduction in [7].

Given a 3-regular directed graph G with specified vertices s and t, we construct a $1 \times n$ signed edge-matching puzzle as follows. (For the unsigned case, we will simply remove all signs.) For each edge e in G, we have a color e, and for each vertex v we have three colors v_I, v_O, and v_X. For each vertex v, we build three tiles; refer to Fig. 11. In one case, v has one edge e_1 coming in and two edges e_2 and e_3 going out. Then we construct the following tiles:

$$+e_1 \overset{-v_X}{\underset{-v_X}{\square}} -v_I, \quad +v_O \overset{-v_O}{\underset{+v_I}{\square}} -e_2, \quad \text{and} \quad +v_O \overset{-v_O}{\underset{+v_I}{\square}} -e_3.$$

In the other case, v has two edges e_1 and e_2 coming in and one edge e_3 going out. Then we construct the following tiles:

$$+e_1 \overset{+v_I}{\underset{-v_O}{\square}} -v_I, \quad +e_2 \overset{+v_I}{\underset{-v_O}{\square}} -v_I, \quad \text{and} \quad +v_O \overset{-v_X}{\underset{-v_X}{\square}} -e_3.$$

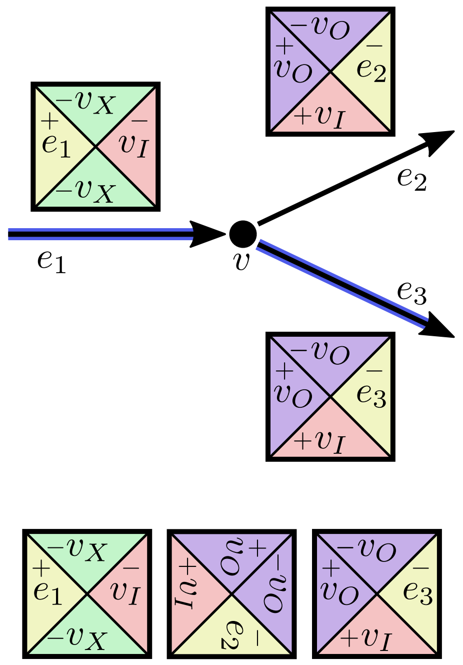
Each of these tiles corresponds to one of the half-edges incident to v. (Overall, each edge is represented by two half-edge tiles.) We use the properties that s has outdegree 1 and t has indegree 1, as provided by Theorem 3.2. We remove the tiles corresponding to the half-edges entering s and the tiles corresponding to half-edges leaving t, so s and t each have only one corresponding tile. Finally, we specify that the left boundary edge has color $-s_O$.

We claim that the number of solutions to this edge-matching puzzle is the same as the number of Hamiltonian paths in G from s to t.

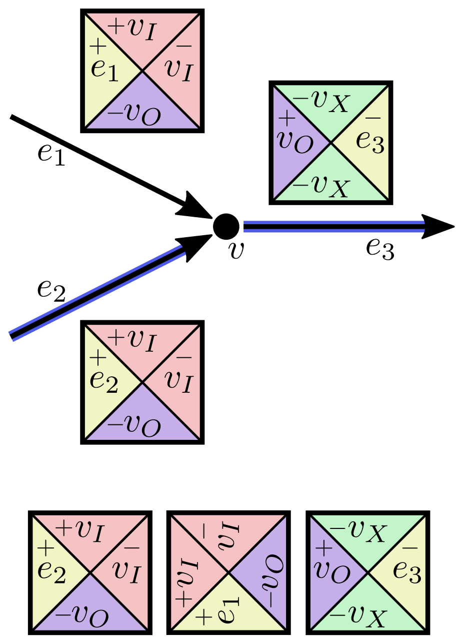
First suppose that we have such a Hamiltonian path $s = v_1, v_2, \ldots, v_{|V|} = t$. We can construct a solution to the edge-matching puzzle by placing the three tiles for each vertex v_i consecutively, in the order $i = 1, 2, \ldots, |V|$ that the vertices appear in the path. As in the bottom of Fig. 11, we place the three tiles for each vertex v_i so that the tiles corresponding to the edges $e_i = (v_{i-1}, v_i)$ and $e_{i+1} = (v_i, v_{i+1})$ that the path uses to enter and exit v are respectively first and last so that the sequence of colors is $e_i, v_{i,I}, v_{i,O}, e_{i+1}$. The exposed colors are $+e_i$ on the left and $-e_{i+1}$ on the right, so the these placed triples of tiles match up at their ends (because the sequence of vertices is a path). There is only one tile for each of s and t, which we place at the beginning and end. The left boundary color is then $+s_O$, as required, and the rightmost boundary color is $+t_I$.

Next we show that every solution to the edge-matching puzzle has this form, and thus corresponds to a Hamiltonian path. Suppose we have a solution to the edge-matching puzzle. Because the left boundary color is $-s_O$, the tile corresponding to s must be placed on the left oriented with $+s_O$ on the left and the outgoing edge color on the right. The only tile corresponding to t is $+e \overset{-t_X}{\underset{-t_X}{\square}} -t_I$, where e is the incoming edge. Because colors t_X and t_I do not appear on any other tiles, this tile must be placed rightmost with color $+e$ on the left.

Consider a vertex v other than s and t. None of the tiles corresponding to v can be at either end of the solution, because those spaces are claimed by s and t. Suppose v has indegree 1 and outdegree 2; the other case is similar. Because $+e_1 \overset{-v_X}{\underset{-v_X}{\square}} -v_I$ is the only tile with the color v_X, it must be adjacent to other tiles on the other two sides. The tile adjacent on the side with color $-v_I$ must be one of the two



(a) A vertex with indegree 1 and outdegree 2.



(b) A vertex with indegree 2 and outdegree 1.

Fig. 11: The tiles in the reduction showing ASP- and #P-hardness of $1 \times n$ edge-matching puzzles. At the bottom we show one possible edge-matching solution corresponding to one (blue) path through v.

other tiles corresponding to v. Whichever tile it is, its orientation is fixed by matching color v_I, so the opposite side must have color $-v_O$, and therefore the following tile must be the third tile corresponding to v, with the color of another edge incident to v on the side touching the next tile. In summary, the three tiles corresponding to v must be consecutive, and the two colors they expose to other tiles are two edges incident to v with different orientations relative to v, with the local configuration of the three tiles determined by those exposed colors.

Suppose the sequence of tiles corresponding to vertex u are adjacent to the sequence corresponding to vertex v. Then the side where these sequences touch must have color e, where e is either (u, v) outgoing from u and incoming to v or (v, u) outgoing from v and incoming to u. The other left and right edges of these tiles must also have edge colors corresponding to edges incident to u and v. By induction, if the solution has several consecutive sequences of tiles corresponding to vertices, the sequence of vertices must form a path in G in either direction. The entire solution must therefore be a concatenation of sequences corresponding to vertices starting with s and ending with t, such that adjacent vertices share an edge from left to right, and using each tile exactly once. Hence the solution must correspond to a Hamiltonian path.

For each Hamiltonian path, there is exactly one corresponding solution to the edge-matching puzzle, because there is only one way to connect the tiles corresponding to a vertex for each pair of edges used at that vertex. So there are the same number of Hamiltonian paths in G from s to t and solutions to the edge-matching puzzle. Because this reduction is parsimonious, it shows that $1 \times n$ signed edge-matching puzzles with the color of the left boundary edge specified is ASP- and #P-complete. The same reduction with all the signs removed proves the same result for unsigned edge-matching puzzles. □

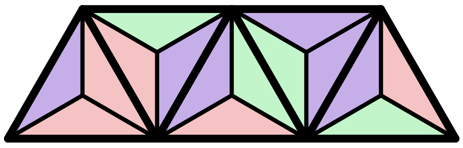
Corollary 3.4. *$1 \times n$ signed and unsigned edge-matching puzzles are #P-complete and their 2-ASP problem is NP-hard.*

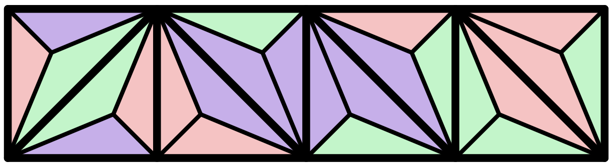
Proof. Without a specified left boundary color, we cannot guarantee that the tile corresponding to the start vertex s is on the left and the tile corresponding to the end vertex t is on the right; we only know that they are at the ends. Thus each solution to the edge-matching puzzle can be rotated 180° to form another solution, so the reduction is 2-monious. □

4. Triangular Edge Matching

In this section, we study $1 \times n$ edge-matching puzzles with triangular tiles, specifically, equilateral and right isosceles triangles. There is one natural interpretation of "$1 \times n$" for equilateral triangles, as shown in Fig. 12a. However, for right isosceles triangles, there are two natural interpretations. If the triangles have legs of length 1, then to pack a $1 \times n$ box they must have alternating hypotenuse/leg contact, which we will simply refer to as hypotenuse contact, as shown in Fig. 12b. On the other hand, if the triangles have a height of 1, then they must be packed using only leg-to-leg contacts, as shown in Fig. 12c.

Hypotenuse-contact right triangles can directly and parsimoniously simulate square tiles: for each square create two triangles whose hypotenuses have a matching, unique color. (This idea is mentioned in another context in the conclusion of [7].) Thus NP-completeness, ASP-completeness (with left


(a) Equilateral triangles


(b) Right triangles, hypotenuse contact

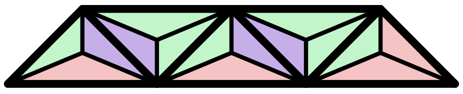

(c) Right triangles, leg contact

Fig. 12: Three types of triangular tiles.

boundary specified), and #P-completeness of these puzzles follows directly from results on square tiles. We devote the rest of this section to equilateral triangles (Section 4.1) and right triangles with leg contact (Section 4.2).

4.1 Equilateral-Triangle Edge Matching

In this section, we prove NP/#P/ASP-completeness of $1 \times n$ equilateral triangular edge-matching puzzles. We start with an NP-completeness proof, then augment it and analyze it further to prove #P/ASP-completeness.

Theorem 4.1. *$1 \times n$ signed and unsigned equilateral-triangle edge-matching puzzles are NP-complete. The same results hold if we allow tile reflection.*

Proof. Clearly these problems are in NP. To show NP-hardness, we reduce from Hamiltonian path in 3-regular *undirected* graphs [17] (in contrast to Section 3 which considered directed graphs). We describe signed tiles resulting from our reduction to signed edge matching; for the unsigned puzzle we will just drop the signs. Similar to the proof of Theorem 3.3, we will create exactly two tiles per edge; refer to Fig. 13. To assign complementary signs to the edge colors, arbitrarily orient each edge e (but paths need not follow this orientation). For every vertex v with incident edges e_1, e_2, e_3, construct three corresponding triangular tiles:

$$^{+v}\underset{\pm e_1}{\triangle}^{-v}, \quad ^{+v}\underset{\pm e_2}{\triangle}^{-v}, \quad \text{and} \quad ^{+v}\underset{\pm e_3}{\triangle}^{-v},$$

where the sign of each e_i color is positive if e_i was arbitrarily oriented to be incoming to v and negative otherwise. We claim that these tiles have a signed or unsigned edge-matching solution if and only if the graph has a Hamiltonian path.

First suppose that there is a Hamiltonian path $v_1, v_2, \ldots, v_n$. We can construct an edge-matching solution by arranging the three corresponding tiles for each vertex v_i so that $\{v_{i-1}, v_i\}$ is on the left boundary edge and $\{v_i, v_{i+1}\}$ is on the right boundary edge, as in Fig. 13

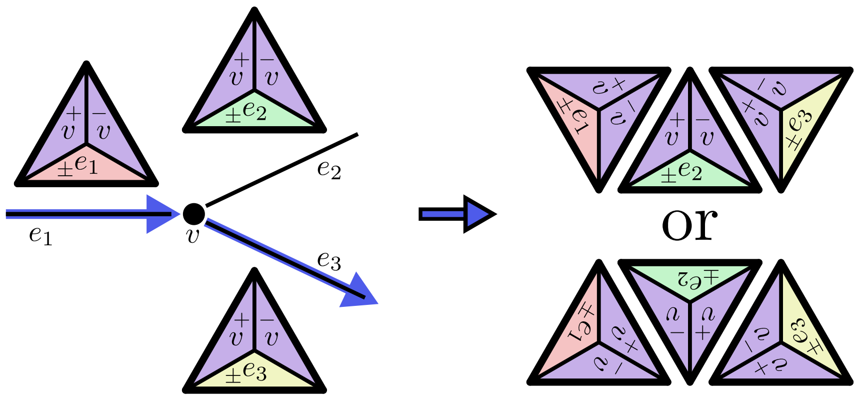


Fig. 13: NP-hardness of $1 \times n$ equilateral-triangle edge matching, showing one possible (blue) path through v and the corresponding edge-matching solutions (depending on parity up to this point).

(right, top or bottom according to parity of i as required by the tiling). The figure illustrates that the vertex colors match with opposite signs, and that tiles do not need to be reflected. By the arbitrary orientation of the edges, every edge color will match with its negated color.

Now suppose that there is an edge-matching solution, even without the color signs. Without the color signs, the tiles are reflectionally symmetric, so the following argument works also when we allow tile reflection. Each vertex color v appears in exactly three tiles, so the three vertex tiles can match only with each other, or some of them can appear as the extreme left or extreme right tile. If any of the three tiles for v are extreme, then none of the tiles can be placed in the middle of an edge-matching solution (lacking the three tiles required to form an 180° angle), so in this case, all three tiles for v appear at the left and right extremes of the solution, effectively "wrapping around" the $1 \times n$ board. For every other vertex, the three corresponding tiles must appear together. Listing all of the vertices in the order in which their color appears in the solution yields a Hamiltonian path of the original graph. (If the tiles corresponding to one vertex wrap around, then this process in fact yields a Hamiltonian cycle.) □

The proof above suggests an alternate approach to proving Theorem 3.3 about squares: unify the v_I, v_O, v_x colors into a single color, and reduce from *undirected* Hamiltonian path. However, for unsigned colors, the change would make this reduction nonparsimonious, because it enables the middle tile to rotate by 180° in the two arrangements on the bottom of Fig. 11. But equilateral triangles lack this ambiguity, and we are able to obtain parsimony by a more careful handling of the start and end.

First we need a slightly different form of undirected Hamiltonicity:

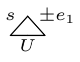
Lemma 4.2. *Finding Hamiltonian paths, with or without a specified start vertex s and/or end vertex t, in maximum-degree-3 planar undirected graphs is ASP-complete, and counting Hamiltonian paths in those graphs is #P-complete. The same result holds when the given vertices s, t have degree 1.*

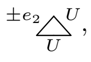
Proof. We present a parsimonious reduction from Hamiltonian cycle in maximum-degree-3 planar undirected graphs (the same graphs) having at least one vertex of degree 2, proved ASP-complete by Seta [25]. Our reduction is similar to the first step in the proof of Theorem 3.2.

Let G be a maximum-degree-3 undirected graph with a degree-2 vertex v. Let $\{u, v\}$ be one of v's incident edges, which must be in every Hamiltonian cycle. Construct G' by adding two new vertices s and t, and replacing the edge $\{u, v\}$ with edges $\{s, u\}$ and $\{t, v\}$. Because s and t have degree 1, they are in every Hamiltonian path of G'. Because edge $\{u, v\}$ is contained in every Hamiltonian cycle in G', there is a direct bijection between Hamiltonian cycles in G and Hamiltonian (s-t) paths in G'. □

Theorem 4.3. *$1 \times n$ signed and unsigned equilateral-triangle edge-matching puzzles with the left boundary edge color specified are ASP-complete and #P-complete.*

Proof. Clearly this problem is in FNP and its counting problem is in #P. To show hardness, we present a parsimonious reduction from Hamiltonian s-t paths in maximum-degree-3 undirected graphs where s and t have degree 1, from Lemma 4.2. Our reduction is a modification of the NP-hardness reduction in Theorem 4.1 that differs only for the new case of vertices with degree < 3. For each degree-2 vertex v, we attach a half-edge $\{v\}$ (with no other endpoint), and then apply the degree-3 construction from Fig. 13. We can assume that the only degree-1 vertices are s and t because no other degree-1 vertices could possibly be reached by an s-t path (so if there were such a vertex we could parsimoniously reduce by constructing any unsolvable edge-matching instance). For the degree-1 vertices s and t, we construct two corresponding tiles:



and


,

where e_1 and e_2 represent the unique edges incident to s and t respectively, with signs chosen for these edge colors based on our arbitrary orientation of the original graph, in the same fashion as for all other tiles. (As before, for the unsigned problem, we just drop the signs.) Each occurrence of

U represents a unique color not occurring in any other tile. Finally, we specify the left boundary color to be s, which is another unique color.

Because the tile corresponding to vertex s is the only one with color s, it must be placed as the leftmost tile. Because the tile corresponding to t has two sides with unique colors, it must be placed as the rightmost tile. As argued in Theorem 4.1, every triplet of tiles corresponding to a degree-3 (or degree-2) vertex must occur consecutively, because the s and t tiles prevent "wrapping around". Therefore every edge-matching solution induces an ordering of the vertex tile triplets between the leftmost s tile and the rightmost t tile. To guarantee a bijection between edge-matching solutions and Hamiltonian s-t paths solutions, it only remains to show that, given an ordering of the tile triplets, there is a unique arrangement of the three tiles within each triplet.

Suppose the tile triplet for vertex v occurs between the triplets for vertices u and w. The only edge colors that v's triplet have in common with u's and w's triplets are the colors representing edges $\{u, v\}$ and $\{v, w\}$ in the original graph, so the two tiles in v's triplet containing the $\{u, v\}$ and $\{v, w\}$ colors must be on the left and right respectively, with those edges exposed. The remaining tile in the triplet has no choice but to be oriented between them with its v-colored edges facing the two other tiles in the triplet, and its third edge facing the $1 \times n$ boundary. Thus the arrangement of tiles within each triplet is uniquely defined by the ordering of tile triplets along the box, completing the proof that our reduction is parsimonious. □

Corollary 4.4. *$1 \times n$ signed and unsigned equilateral-triangle edge-matching puzzles are #P-complete and their 2-ASP problem is NP-hard.*

Proof. As in Corollary 3.4. □

4.2 Leg-Contact Right-Isosceles-Triangle Edge Matching

In this section, we show that edge matching with right isosceles triangles that tile a $1 \times n$ box by leg contact (as in Fig. 12c) is closely related to finding an Eulerian path in a graph. More precisely, we show relations to two variants, called antidirected and forbidden-transition Eulerian paths, which we define and analyze in Sections 4.2.1 and 4.2.2 respectively. We use this connection to show that these puzzles can be solved in polynomial time (Section 4.2.3), and then to show that counting solutions to these puzzles is #P-complete (Section 4.2.4).

4.2.1 Antidirected Eulerian Path Characterization

Consider a directed graph G. Recall that a ***(directed) Eulerian path*** is a directed path in G (respecting the edge directions in G) that visits every edge in G exactly once. It is well-known that a connected graph has such a path if and only if it has zero or two vertices of odd degree [6], Corollary 4.1, and in this case the path can be constructed in linear time [14].

Here we analyze the variant where the edge directions of G must alternate. Precisely, an ***antidirected path*** [2], [4], [18] is a sequence of edges where every pair of consecutive edges share an endpoint (an undirected path) and furthermore those edges either both point toward or both point away from that shared endpoint. In other words, an antidirected path alternates between following an edge of G in the "forwards" direction and following an edge of G in the "backwards" direction, with an arbitrary starting parity. An ***antidirected Eulerian path*** [4], [13], [31] of G is an antidirected path of G that visits every edge (either forwards or backwards) exactly once. Examples of past results on this topic include that a directed graph without degree-2 vertices has an odd number of Eulerian paths if and only if it is 4-regular and has an antidirected Eulerian path [4], while not every connected 4-regular undirected graph with an odd cycle has an orientation admitting an antidirected Eulerian path [31].

In the ***[antidirected] Eulerian path problem***, we are given a directed graph G, and want to know whether G has an [antidirected] Eulerian path, and if it does, to find one. We relate these two problems as follows:

Theorem 4.5. *The antidirected Eulerian path problem can be reduced in linear time to the Eulerian path problem.*

Proof. Let G be a directed graph input for the antidirected Eulerian path problem. Construct an undirected bipartite graph G' (called the "split" of G by West [28], Definition 1.4.20) as follows; refer to Fig. 14. For each vertex $v \in G$, construct two vertices v^+ and v^- in G'. For every directed edge $e = (u, v) \in G$, add the undirected edge $e' = \{u^+, v^-\}$ to G'. Because every edge in G' connects a plus vertex to a minus vertex, G' is bipartite.

We claim that paths in G' correspond to antidirected paths in G. For any path $p' = (v_1^\pm, v_2^\mp, v_3^\pm, v_4^\mp, \dots)$ in G' (where signs alternate by bipartiteness), consider mapping each edge of the form $\{v_i^+, v_{i+1}^-\}$ in p' to the corresponding edge (v_i, v_{i+1}) of G, and mapping each edge of the form $\{v_i^-, v_{i+1}^+\}$ in p' to the (backwards traversal of) the corresponding edge (v_{i+1}, v_i) of G. Then we obtain an antidirected path in G. Because the mapping between edges of G and G' is a bijection, so is this transformation. By the same bijectivity, if p' is Eulerian, then so is p. Therefore Eulerian paths in G' correspond to antidirected Eulerian paths in G. □

A similar result was obtained independently in [1].

For our application to edge matching, we will need to solve a slightly restricted version of the problem:

Corollary 4.6. *The antidirected Eulerian path problem can be solved in linear time. The same result holds if the path is further restricted to start and/or end with a specified direction (forwards or backwards).*

Proof. The first sentence follows from the reduction of Theorem 4.5 combined with linear-time algorithms for finding

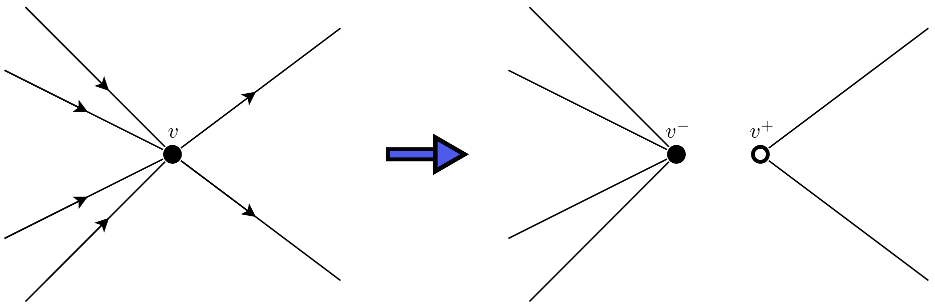


Fig. 14: Reduction from antidirected Eulerian path to Eulerian path.

Eulerian paths [14].

Now suppose we are given the starting and ending directions $s, t \in \{\text{forwards}, \text{backwards}\}$ for an antidirected Eulerian path. Applying the previous algorithm, we can detect whether G has any antidirected Eulerian path, i.e., whether G' from the proof of Theorem 4.5 has any Eulerian path. If the answer is "no", then we know there is no path. Otherwise, by the characterization of Eulerian paths [6], Corollary 4.1, either (1) every vertex of G' has even degree, or (2) exactly two vertices of G' have odd degree.

In the first case, every Eulerian path p' of G' is also a cycle, so when we translate to an antidirected Eulerian path/cycle p of G, the starting orientation is the same as the ending orientation if and only if G has an odd number e of edges. Thus we can answer the restricted antidirected Eulerian path problem by checking whether $(s = t) \leftrightarrow (e \text{ odd})$. If $s = t$ and e is odd, then we find an antidirected Eulerian cycle and choose the starting parity for a path to match $s = t$. If $s \neq t$ and e is even, then we find any antidirected Eulerian cycle and any starting point, and reverse the path if s and t mismatch. Otherwise, no satisfying antidirected Eulerian path exists.

In the second case, every Eulerian path p' of G' has its endpoints at the two odd-degree vertices o_1, o_2 of G', so every antidirected Eulerian path p in G has its extreme edge orientations determined by whether o_1 and o_2 are plus or minus vertices (and which of o_1 and o_2 is chosen to be the start versus end of the path). If o_1 and o_2 are both plus vertices, then s = forwards and t = backwards is the only possibility. If o_1 and o_2 are both minus vertices, then s = backwards and t = forwards is the only possibility. If o_1 and o_2 are plus and minus vertices, then $s = t$ is the only constraint: if $s = t$ = forwards, then we start at the plus vertex; and if $s = t$ = backwards, then we start at the minus vertex. Otherwise, no satisfying antidirected Eulerian path exists.

Therefore we can solve the restricted form of the antidirected Eulerian path problem. □

4.2.2 Forbidden-Transition Eulerian Path Characterization

In the ***forbidden-transition Eulerian path problem*** [20], we are given an undirected graph $G = (V, E)$ and, for every vertex $v \in V$, a partition of the edges E_v incident to v into groups $P_{v,1}, P_{v,2}, \dots, P_{v,k_v}$. The goal is to find an Eulerian path $v_0, v_1, \dots, v_{|E|}$ of G such that, for every vertex visit v_i where $0 < i < |E|$, the incident edges (v_{i-1}, v_i) and (v_i, v_{i+1}) belong to different groups among $P_{v_i,1}, P_{v_i,2}, \dots, P_{v_i,k_{v_i}}$. In other words, we forbid use of the subpath (v_{i-1}, v_i, v_{i+1}) when (v_{i-1}, v_i) and (v_i, v_{i+1}) belong to a common group $P_{v_i,j}$.[*5] In a ***forbidden-transition Eulerian cycle***, we similarly restrict the subpath $(v_{|E|-1}, v_{|E|} = v_0, v_1)$.

Kotzig [20] showed (in a slightly more general scenario) that the natural necessary conditions for this problem are in fact sufficient. We repeat Kotzig's mathematical argument here in order to verify that it also yields an efficient algorithm.

Theorem 4.7 ([20]). *An undirected graph G and partition system P has a forbidden-transition Eulerian path if and only if G has an Eulerian path and every group $P_{v,i}$ has $|P_{v,i}| \leq \lceil \text{degree}(v)/2 \rceil$. If furthermore G has an Eulerian cycle, then (G, P) has a forbidden-transition Eulerian cycle. When such a path/cycle exists, it can be found in linear time.*

Proof. By the characterization of Eulerian paths [6], Corollary 4.1, G must have exactly zero or two vertices of odd degree. We can reduce to the case of zero odd-degree vertices as follows. If G has two odd-degree vertices, then add an edge between them, which increases their degrees to even but does not change $\lceil \text{degree}(v_i)/2 \rceil$. Now apply the zero-odd-degree-vertices case of the present theorem (proved below) to obtain an Eulerian cycle with the desired property. Removing the added edge results in an Eulerian path with the desired property. Therefore we can assume every vertex has even degree, so we can ignore the ceilings.

Next we prove that the conditions are necessary. Clearly G having an Eulerian path is necessary for it to have a forbidden-transition Eulerian path. If any $|P_{v,i}| >$ degree$(v)/2$, then we claim that (G, P) cannot have a forbidden-transition Eulerian path. Any Eulerian path in G is a cycle, and thus its traversal order pairs up the edges E_v incident to v into degree$(v)/2$ pairs. By the Pigeonhole

[*5] It is tempting to think that antidirected Eulerian path in a directed graph is a special case of forbidden-transition Eulerian path in an undirected graph, using two groups at each vertex to represent the outgoing vs. incoming edges. However, the antidirected constraint *requires repeating* the incoming/outgoing nature at each vertex, while the forbidden-transition constraint *prevents repeating* the incoming/outgoing nature at each vertex.

Principle, some pair has both its edges in $P_{v,i}$, which is a forbidden transition.

Now suppose G has an Eulerian path and every group $P_{v,i}$ satisfies $|P_{v,i}| \leq \text{degree}(v)/2$. For each vertex v, order its incident edges $E_v = \{e_1, e_2, \ldots, e_{\text{degree}(v)}\}$ so that all edges from group $P_{v,i}$ appear consecutively in the ordering, for all $1 \leq i \leq k_v$. Now pair each edge e_j with $e_{j+\text{degree}(v)/2}$, for $1 \leq j \leq \text{degree}(v)/2$. Because each $|P_{v,i}| \leq \text{degree}(v)/2$, this pairing has no forbidden pairs. The perfect pairing at each vertex partitions the graph's edges into edge-disjoint cycles.

To merge these cycles into one Eulerian cycle, take any two cycles C, C' that share a vertex v (which exist because G has an Eulerian path so its edges are connected). Suppose one cycle pairs edges (e_1, e_2) at v, while the other cycle pairs edges (e'_1, e'_2) at v. Suppose e_1, e_2, e'_1, e'_2 are in groups i_1, i_2, i'_1, i'_2. If we change the local pairing to (e_1, e'_2) and (e'_1, e_2), then we merge the cycles, and avoid forbidden pairs provided $i_1 \neq i'_2$ and $i'_1 \neq i_2$. If we change the local pairing to (e_1, e'_1) and (e'_2, e_2) (and reverse one of the cycles), then we again merge the cycles, this time avoiding forbidden pairs provided $i_1 \neq i'_1$ and $i'_2 \neq i_2$. Because the cycles have no forbidden pairs, $i_1 \neq i_2$ and $i'_1 \neq i'_2$. Thus we can have at most two equalities among the four possible comparisons between $\{i_1, i_2\}$ and $\{i'_1, i'_2\}$. Therefore one of the two merging strategies works.

We can implement this algorithm in linear time by constructing the pairing locally as linked pointers, representing each cycle as a doubly linked list on its edges, where each edge stores its two neighboring edges in the cycle in no particular order. Number the cycles $1, 2, \ldots, k$, and iterate over the cycles to mark each vertex with each of the cycles it belongs to, along with one edge pairing from that cycle. Label cycle 1 as "merged" and the rest as "unmerged". Perform a depth-first search in G from any vertex that is in cycle 1. At each vertex v visited, iterate through the cycles that v belongs to (via v's marks); if any cycle i has not yet been merged, then merge it into cycle 1 by adjusting $O(1)$ pointers among v's marked edge pairings for cycles 1 and i, labeling cycle i as "merged". By induction, every vertex visited by the depth-first search will have already been merged into cycle 1. The running time beyond the linear cost of depth-first search is proportional to the number of marks, which (by the Handshaking Lemma) is twice the number of edges. This algorithm is essentially the efficient implementation of Hierholzer's Algorithm for Eulerian tours from [14]. □

Next we combine this result with the results of Section 4.2.1 about antidirected Eulerian paths. For a *directed* graph G and a partition system P, define a ***forbidden-transition antidirected Eulerian path*** in (G, P) to be an antidirected Eulerian path $e_1, e_2, \ldots, e_{|E|}$ of G such that no two edges e_i and e_{i+1} belong to a common group $P_{v,j}$ where v is the shared vertex of e_i and e_{i+1}.

Corollary 4.8. *The forbidden-transition antidirected Eulerian path problem can be solved in linear time. The same result holds if the path is further restricted to start and/or end with a specified direction (forwards or backwards).*

Proof. Apply the reduction of Theorem 4.5 to obtain an undirected graph G' with the property that Eulerian paths in G' correspond to antidirected Eulerian paths in G. For each vertex $v^{\pm}$ of G' and each $1 \leq i \leq k_v$, define $P'_{v^{\pm},i}$ to be the set of edges of G' incident to $v^{\pm}$ that correspond to edges of G in $P_{v,i}$. Then apply Theorem 4.7 to decide whether (G', P') has a forbidden-transition Eulerian path, which is equivalent to whether (G, P) has a forbidden-transition antidirected Eulerian path. To handle the start/end direction constraints, we can apply the same post-analysis as in Corollary 4.6. □

4.2.3 Linear-Time Algorithm for Leg-Contact Right-Isosceles-Triangle Edge Matching

Now we use the algorithms we have built for antidirected and forbidden-transition Eulerian paths to solve leg-contact right-isosceles-triangle edge matching. The unsigned case reduces to antidirected Eulerian paths, while the signed case reduces to forbidden-transition antidirected Eulerian paths.

Theorem 4.9. *$1 \times n$ signed and unsigned leg-contact right-isosceles-triangle edge-matching puzzles can be solved in linear time.*

Proof. First note that tile hypotenuses can never touch in a $1 \times n$ box by leg contact, so we can ignore those edges' colors completely. We treat the signed and unsigned cases separately:

Unsigned case: Our algorithm reduces unsigned edge matching to the antidirected Eulerian path problem in a directed graph, as solved in Section 4.2.1. Given an instance of unsigned $1 \times n$ leg-contact isosceles-right-triangle edge matching, we construct a directed graph G as follows. Create a vertex for each unique color that occurs on the legs of the tiles. For every triangle ${}^{u}\underset{H}{\triangle}{}^{v}$, create a directed edge (u, v).

Any edge-matching solution consists of some ordering of the triangles that they pack into the $1 \times n$ box, with triangles alternating between being oriented with its hypotenuse on the top or bottom (see Fig. 12c), and consecutive triangles matching on their shared edges. We claim that such an edge-matching solution corresponds, by replacing each tile with its corresponding edge in G, to an antidirected Eulerian path in G. First, the path must be antidirected: following an edge (u, v) in the forwards direction corresponds to placing ${}^{u}\underset{H}{\triangle}{}^{v}$ with its hypotenuse on the bottom (so colors u and v are on the left and right, respectively), while following edge (u, v) in the reverse direction (v, u) corresponds to placing the tile rotated 180° with its hypotenuse on the top (so colors v and u are on the left and right, respectively). Second, the path must be Eulerian, because an edge-matching solution must use every tile exactly once.

The last constraint to handle is the left and right boundary conditions. If the left edge of the box has an acute angle at the bottom [top], then the first tile must be placed with its hypotenuse on the bottom [top], so the first edge of the antidirected Eulerian path must be forwards [backwards]. Similarly, if the right edge of the box has an acute angle at the bottom [top], then the last tile must be placed with its hypotenuse on the bottom [top], so the first edge of the antidirected Eulerian path must be forwards [backwards]. These constraints are exactly what Corollary 4.6 handles in polynomial time. By deciding whether G has an appropriate antidirected Eulerian path, we decide whether the edge-matching puzzle has a solution, and an actual solution can be converted by the tile–edge correspondence.

Signed case: Our algorithm reduces signed edge matching to the forbidden-transition antidirected Eulerian path problem in a directed graph, as solved in Section 4.2.2. Given an instance of signed $1 \times n$ leg-contact isosceles-right-triangle edge matching, we construct the same directed graph G as the unsigned case. To capture the color sign constraint on adjacent tiles, we define forbidden transitions for the antidirected Eulerian path in G. Specifically, for each vertex corresponding to an unsigned color c, define four groups:

(1) $P_{c,1}$ consists of all edges incoming to c corresponding to tiles of the form $\triangle^{+c}$;

(2) $P_{c,2}$ consists of all edges incoming to c corresponding to tiles of the form $\triangle^{-c}$;

(3) $P_{c,3}$ consists of all edges outgoing from c corresponding to tiles of the form ${}^{+c}\triangle$; and

(4) $P_{c,4}$ consists of all edges outgoing from c corresponding to tiles of the form ${}^{-c}\triangle$.

We claim that edge-matching solutions correspond to forbidden-transition antidirected Eulerian paths in (G, P). Any antidirected path, when visiting a vertex c not as a path endpoint, will use either two incoming edges (groups 1 and 2) or two outgoing edges (groups 3 and 4). The forbidden transitions thus exactly prevent matching together two instances of c of the same sign. Therefore Corollary 4.8, with the same start/end conditions as the unsigned case, solves the problem. □

4.2.4 #P-completeness of Leg-Contact Right-Isosceles-Triangle Edge Matching

Even though leg-contact right-isosceles-triangle edge-matching puzzles are not hard to solve, counting their solutions remains hard.

Theorem 4.10. *$1 \times n$ signed and unsigned leg-contact right-isosceles-triangle edge-matching puzzles are #P-complete.*

Proof. We reduce from counting the number of Eulerian cycles in an undirected graph, proved #P-complete in [8]. Given such an undirected graph G, we first add two vertices s, t and attach them to an arbitrary vertex v of G, forming an undirected graph G'. The number of Eulerian cycles in G is exactly twice the number of Eulerian paths in G' (whose endpoints are necessarily s and t — choosing which endpoint is the start of the path incurs the factor of 2). Thus we can reduce from counting the number of Eulerian paths in a graph G' with two degree-1 vertices s, t.

Unsigned case: For the endpoint vertices s, t, construct two corresponding triangles

$${}^{U_1}\underset{H}{\triangle}{}^{s} \quad \text{and} \quad {}^{U_2}\underset{H}{\triangle}{}^{t},$$

where s and t are colors representing those vertices, H is an arbitrary hypotenuse color, and U_1 and U_2 are globally unique colors. Because U_1 and U_2 appear only in these tiles, the tiles must be placed leftmost and rightmost in the puzzle (where the rightmost tile is rotated 180°).

For each edge $e = \{u, v\}$ in G', construct two corresponding triangles

$${}^{e}\underset{H}{\triangle}{}^{u} \quad \text{and} \quad {}^{e}\underset{H}{\triangle}{}^{v},$$

where u, v are colors representing these vertices and e is a color representing this edge. Because color e appears only in these two tiles, these tiles must be placed together (with one of them rotated 180°), resulting in a parallelogram with left color u and right color v or, rotating by 180°, the same shape with left color v and right color u. Thus these two tiles (or the resulting parallelogram) simulates the edge $\{u, v\}$ that can be used in either direction.

It follows that edge-matching solutions correspond bijectively to Eulerian paths in G'.

Signed case: For the endpoint vertices s, t, construct two corresponding triangles:

$${}^{U_1}\underset{H}{\triangle}{}^{+s} \quad \text{and} \quad {}^{U_2}\underset{H}{\triangle}{}^{+t},$$

where s and t are colors representing those vertices, H is an arbitrary hypotenuse color, and U_1 and U_2 are globally unique colors, forcing these tiles to be placed leftmost and rightmost in the puzzle.

For each vertex $v \notin \{s, t\}$ in G', which has even degree k, construct $k/2$ copies of two corresponding triangles:

$${}^{-v_X}\underset{H}{\triangle}{}^{+v} \quad \text{and} \quad {}^{+v_X}\underset{H}{\triangle}{}^{+v},$$

where v, v_X are two colors corresponding to vertex v. Because v_X appears only in these two triangles, they must be placed together (with one of them rotated 180°) to match up the v_X-color edges, resulting in a parallelogram with end colors $+v$ and $+v$.

For each edge $e = \{u, v\}$ in G', construct two corresponding triangles:

$${}^{-e}\underset{H}{\triangle}{}^{-u} \quad \text{and} \quad {}^{+e}\underset{H}{\triangle}{}^{-v},$$

where u, v are colors representing these vertices and e is a color representing this edge. (This construction depends slightly on how we distinguish the endpoints of e as u and v, but the choice can be made arbitrarily for each edge without affecting the rest of the construction.) Because color e appears only in these two tiles, these tiles must be placed together (with one of them rotated $180°$), resulting in a parallelogram with left color $-u$ and right color $-v$ or, rotating by $180°$, the same shape with left color $-v$ and right color $-u$.

By the signs of the colors, any edge-matching solution must alternate between edge parallelograms and vertex parallelograms, starting and ending with edge parallelograms, surrounded by the s and t triangles. It follows that edge-matching solutions correspond to Eulerian paths in G'.

This reduction is not parsimonious. Each vertex parallelogram (with the same external colors of $+v$) can be formed in two ways, blowing up the number of solutions by a factor of 2. If G' has m edges, then there are $m-1=\sum_{v\notin\{s,t\}}\text{degree}(v)/2$ such vertex parallelograms, for a total blowup of 2^{m-1}. Furthermore, if we do not treat copies of the vertex tiles as identical, then the $k/2$ copies of each degree-k vertex tile can be permuted arbitrarily, blowing up the number of solutions by a factor of $(k/2)!^2$. The total blowup is thus $c=2^{m-1}\prod_v(\text{degree}(v)/2)!^2$, an easy-to-compute constant, making the reduction c-monious. □

5. Shapeless Edge Matching

In this section, we analyze the complexity of the following problems:

Definition 5.1. ***Signed/unsigned shapeless edge matching*** is the following problem: given a set of n unit square tiles where each edge of each tile is given a color (and a sign in the signed case), can the tiles be laid out in any configuration in the plane such that the overall arrangement is connected via edges, and all edge-to-edge contacts between tiles are compatible? In the ***rooted*** variant, the problem specifies a single tile to be fixed at the origin in a specified orientation.

The distinguishing feature of this problem, compared to the rectangular edge-matching problems for which hardness is already known, is that the target shape is not specified, so there is no constraint on the spatial footprint of a solution. We will show that shapeless edge matching is NP-complete and rooted shapeless edge matching is ASP-complete and #P-complete, by reduction from $1\times n$ edge matching with specified left boundary color, which was proved NP-complete by [7] and proved ASP/#P-complete in Section 3 of this paper (for both the signed and unsigned cases).

5.1 Shapeless Edge Matching NP-completeness

Theorem 5.1. *Signed and unsigned shapeless edge-matching puzzles are NP-complete.*

Proof. A shapeless edge-matching solution can clearly be checked in polynomial time, so shapeless edge matching is in NP.

To prove NP-hardness, we reduce from $1\times n$ edge matching with specified left boundary color. Suppose we are given an instance consisting of a set T of n tiles (signed or unsigned) and a single color L denoting the color of the left boundary edge of the $1\times n$ target box. We will produce a shapeless edge-matching instance consisting of tile set $T\cup T'$, where $|T'|=O(|T|)=O(n)$.

We design tile set T' to force these tiles into a rectangular frame structure that simulates a $1\times n$ box. Fig. 15 lists the tiles, and Fig. 16 shows their intended placement. We use four new colors $\{TW, RW, BW, LW\}$ that appear positively and negatively (or in the unsigned case, without signs); each instance of U represents a globally unique (and hence unmatchable) color.

Next we show that the frame tiles in T' must be positioned to form the frame shown in Fig. 16. Our proof mentions signed tiles, but does not depend on these signs, and thus works equally well in the unsigned case by dropping signs from all tiles. Consider the outer cap $U\overset{U}{\underset{U}{\square}}{-TW}$. Because the overall arrangement of tiles must be connected but edges colored U are unmatchable, the outer cap's edge colored TW must be adjacent to either a top-wall tile $+TW\overset{U}{\underset{U}{\square}}{-TW}$ or the top-right corner $+TW\overset{U}{\underset{-RW}{\square}}U$, as they are the only other tiles with edges colored TW. If the top-right corner were adjacent to the outer cap, it would be impossible to connect any of the $n+1$ top-wall tiles, as there would be no further way to expose an edge colored TW (of either sign). By induction, all top-wall tiles are forced to be placed in a row adjacent to the outer cap before the top-right corner is placed, being the only remaining tile with an edge colored TW. By the same argument, the right-wall tiles $U\overset{+RW}{\underset{-RW}{\square}}U$ and bottom-right corner $-BW\overset{+RW}{\underset{U}{\square}}U$ are the only tiles with edges colored RW. Therefore, following the top-right corner must be the three right-wall tiles and then the bottom-right corner, and similarly along the bottom wall and left wall, terminating with the left boundary tile $U\overset{U}{\underset{+LW}{\square}}L$ as the final frame tile. Thus any solution must form the frame as desired.

Finally, we show that the shapeless edge-matching puzzle $T\cup T'$ has a solution if and only if the corresponding $1\times n$ edge-matching instance T has a solution. The forced arrangement of frame tiles only exposes edges colored with an unmatchable U color, except for the single exposed edge colored L. Thus the input tiles of T must connect to the frame through that single edge. Fig. 16 shows that the only available region in which to arrange the tiles of T is within a $1\times n$ box with its leftmost boundary colored L. □

5.2 Shapeless Edge Matching ASP/#P-completeness

Corollary 5.2. *Signed and unsigned rooted shapeless edge-matching puzzles are ASP-complete and #P-complete.*

Proof. For ASP/#P-completeness, we reduce from the rooted variant of shapeless edge matching (which specifies the position and orientation of one tile) to avoid the infinite number of translations as well as global rotations. We

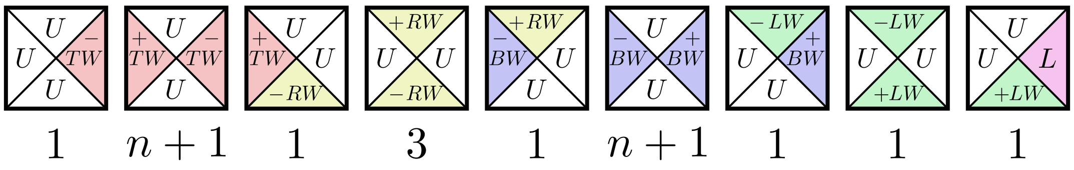


Fig. 15: Frame tile set, each labeled by their multiplicity.

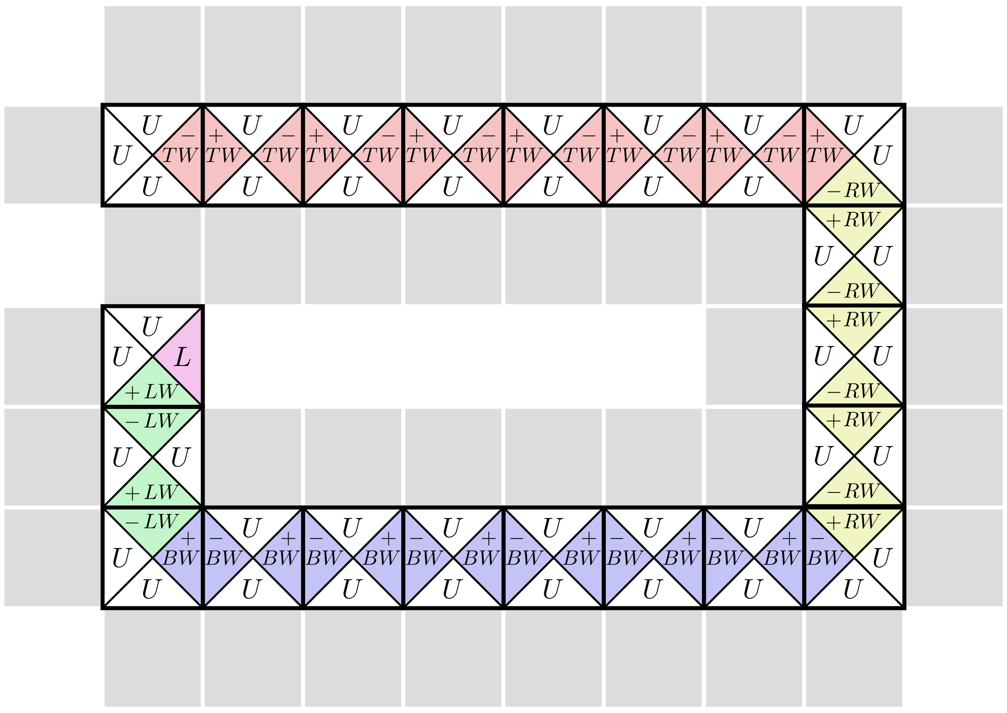


Fig. 16: Frame tiles laid out in their forced positions for $n = 5$. Grey squares show regions which cannot be occupied by further tiles because they are adjacent to U-colored edges.

follow a similar reduction as the proof of Theorem 5.1, but modified so that the frame has a unique construction, making the reduction parsimonious. By Theorem 3.3, $1 \times n$ signed/unsigned edge matching with specified left boundary color is ASP/#P-completeness, so this parsimonious reduction gives us ASP/#P-completeness for signed/unsigned shapeless edge matching.

The only degree of freedom in Theorem 5.1's frame construction is the ordering of the wall tiles along each wall. In order to fix their order, we create distinct tiles for each position along the wall, and give them each unique colors only shared with their neighbors in that ordering. For example, we modify the upper wall to consist of $n + 1$ unique upper-wall tiles and a suitably modified outer cap and upper-right corner as follows:

$$U\underset{U}{\overset{U}{\square}}{-TW_1} \quad +TW_1\underset{U}{\overset{U}{\square}}{-TW_2} \quad \cdots \quad +TW_i\underset{U}{\overset{U}{\square}}{-TW_{i+1}} \quad \cdots$$
$$+TW_{n+1}\underset{U}{\overset{U}{\square}}{-TW_{n+2}} \quad +TW_{n+2}\underset{-RW_1}{\overset{U}{\square}}U.$$

Applying the same modification to the other walls and corners gives us a frame that has a unique construction, and thus the number of solutions to the shapeless edge-matching instance corresponds exactly to the number of solutions to the original $1 \times n$ edge-matching puzzle with specified left boundary. □

6. 2-player 1 × *n* Edge Matching

In this section, we prove PSPACE-hardness for 2-player variants of $1 \times n$ edge matching. In Section 6.1, we introduce and analyze a new variant of geography called partizan geography. Then in Section 6.2, we reduce from geography and our new variant to 2-player $1 \times n$ edge matching.

6.1 Partizan Geography

Geography (also called generalized geography) is a game played on a directed or undirected graph with a designated start vertex. In ***vertex geography*** [15], [21], players take turns moving from the current vertex to a neighboring vertex that has not been visited, with the player who can no longer move losing the game. In ***edge geography*** [15], [24], revisiting vertices is allowed, but each edge can be used only once. In all four variants, directed/undirected vertex/edge geography, the decision question is whether the first player has a winning strategy. Undirected vertex geography can be solved in polynomial time [15], while all three other versions are PSPACE-complete [15], [21], [24].

We introduce partizan versions of geography, where the available moves depend on which player is moving

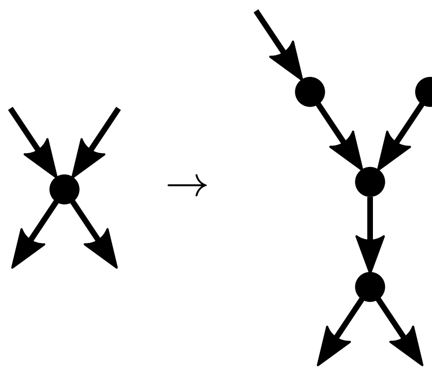


Fig. 17: Gadget simulating vertex geography in edge geography

next. In ***X* *Y*-partizan *Z* geography**, with $X \in \{\text{directed}, \text{undirected}\}$ and $Y, Z \in \{\text{vertex}, \text{edge}\}$, players take turns in an X graph extending a shared path,[*6] playing only Ys of their color while not repeating any Z already visited. For example, in edge-partizan vertex geography, players can play only edges of their color that lead to a vertex not already visited. We give a complete characterization for X Y-partizan Z geography for all combinations of X, Y, Z, as summarized in Table 2.

First we need a result about (impartial) geography that has been widely assumed, but to the best of our knowledge, not explicitly proved in the literature:

Theorem 6.1. *Directed edge geography remains PSPACE-hard even when restricted to bipartite planar graphs with maximum degree 3 and maximum in/outdegree 2.*

Problem GP2 in Garey and Johnson [17] is called simply "Generalized Geography", but its decision question describes directed edge geography, and they cite Schaefer's paper [24] which gives a PSPACE-hardness proof. But Garey and Johnson also cite Lichtenstein and Sipser [21] to add the bipartite, planar, and degree restrictions on the graph, apparently overlooking the fact that the latter paper is about vertex geography. This claim and citation pair have been repeated in works such as Fraenkel et al.'s paper on undirected geography [15], though Bodlaender [5] correctly distinguishes between vetex and edge geography.

Proof. Directed vertex geography is PSPACE-hard on bipartite planar graphs with maximum degree 3 and maximum in/outdegree 2 [21]. We reduce from vertex to edge geography by replacing each vertex (with any number of incoming and outgoing edges) with the gadget shown in Fig. 17. This gadget is bipartite, planar, and has the same maximum indegree and outdegree as the vertex it replaces.

If player 1 plays any of the incoming edges to this gadget, the next two moves are forced; then it is player 2's turn to play one of the outgoing edges. Once the gadget has been traversed, playing any of the remaining incoming edges loses the game (because the central edge has already been played). Thus this gadget correctly simulates a vertex in the vertex geography instance. □

[*6] Fraenkel and Simonson [16] analyze "path-construction games" with two paths, with partizan and impartial variants that specify which paths each player is allowed to extend. Tron [22] is another PSPACE-complete two-player two-path game. By contrast, partizan geography is about two players building a single path (like geography).

Theorem 6.2. *Vertex-partizan geography is equivalent to geography in bipartite graphs. Specifically:*

- *Directed vertex-partizan vertex geography and directed vertex-partizan edge geography are PSPACE-complete even when restricted to bipartite planar graphs with maximum degree 3 and maximum in/outdegree 2.*
- *Undirected vertex-partizan vertex geography and undirected vertex-partizan edge geography can be solved in polynomial time.*

Proof. Given a bipartite geography instance, coloring the vertices according to the bipartition produces a vertex-partizan game with the same winner. Conversely, no monochromatic edges in a vertex-partizan instance can be played because the players alternate moves, so those edges can be deleted without changing the winner. The resulting graph is bipartite, with each partition containing only vertices of a single player's color. Thus the problems are equivalent.

Directed vertex geography in bipartite planar maximum-degree-3 maximum-in/outdegree-2 graphs is proved PSPACE-complete in [21] and Theorem 6.1 extends this to directed edge geography in the same class of graphs. Undirected vertex geography (in all graphs) and bipartite undirected edge geography are both polynomial [15]. All of these results carry over directly to vertex-partizan geography. □

Theorem 6.3. *Edge-partizan geography (of all kinds) is PSPACE-complete even when restricted to bipartite planar graphs with maximum degree 3 and maximum in/outdegree 2.*

Proof. Given an (impartial) bipartite directed vertex/edge geography instance, we can color the vertices red and blue, so (by bipartiteness) every edge is from red to blue or from blue to red. Color the first type of edge red and the second type of edge blue. Because every path alternates vertex colors, every path also alternates edge colors, so adding the edge-partizan constraint does not prohibit any path. Thus bipartite directed geography reduces to directed edge-partizan geography.

We can reduce directed edge-partizan geography to undirected edge-partizan geography using the directed-edge-simulation gadget in Fig. 18. When the blue player plays the left edge, the red and blue player's next moves are forced; then it is the red player's turn at the right vertex. If blue tries to play the simulated edge backwards (starting at the right vertex), then red can immediately win using the leaf.

Thus all edge-partizan geography games are PSPACE-complete even when restricted to bipartite planar graphs with maximum degree 3 and maximum in/outdegree 2,

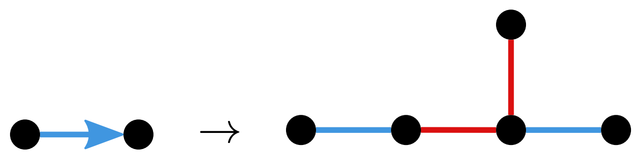


Fig. 18: A gadget simulating a directed edge with undirected edges. (Exchange colors to simulate a red edge.)

again carrying through the results in [21] and Theorem 6.1. □

6.2 Reduction from Geography to 2-player 1 × *n* Edge Matching

In this section, we analyze the complexity all four variants of the following 2-player edge-matching game:

Definition 6.1. In the ***2-player signed/unsigned edge-matching game***, two players play on a $1 \times n$ board where the left boundary edge has a specified (possibly signed) color. Also given are n square tiles, where each tile $T_i = {}_{a_i}\overset{b_i}{\underset{d_i}{\square}}{}_{c_i}$ consists of four (possibly signed) edge colors. In two variants, the players draw from a ***shared pool*** (any player can choose any tile) or from their ***own pools*** (each player can choose a tile only from their own pool). The players take turns making the following type of moves: choosing an unused tile from the available pool, choosing one of the four rotations of the tile, and placing the rotated tile in the leftmost unoccupied position of the board. A move is valid only if the tile's left edge is compatible with the edge to its left (on the right of the previously played tile or the edge of the board). If a player has no valid move, then that player loses and the other player wins. The decision problem is to determine whether the first player can force a win.

First we present a proof similar to the proof of Theorem 3.3, although its results are subsumed by the following theorem.

Theorem 6.4. *If players draw from a shared pool of tiles, which can be signed or unsigned, the 2-player edge-matching game is PSPACE-complete.*

Proof. We reduce from directed vertex geography in graphs with maximum degree 3, which was proved PSPACE-hard in [21]. Our reduction is the same as the reduction used in the proof of 1-player ASP-completeness in Theorem 3.3, whose tiles are shown in Fig. 11. In the proof of Theorem 3.3, three tiles are placed for each vertex, so if two players alternate placing tiles, then they alternate placing the first tile for each vertex, which corresponds to taking that vertex in the geography game. In the same proof, the only choices are which tile to place second for each vertex of outdegree 2 (the first tile is fixed, and the unchosen tile must be placed third), a choice which the player who did not place the first tile for that vertex can make and which determines the next visited vertex in the tile-placing game. Correspondingly, in the geography game, when one player chooses a vertex, the player who did not choose that vertex chooses the next visited vertex. Finally, the winner of the tile-placing game is the last player to place a tile. Each vertex has three tiles which are always placed in sequence, so the last player to place a tile is the last player to place the first tile for a vertex, which corresponds to the last player to pick a vertex in the geography game. So the winner of the geography game is the winner of the tile-placing game, as desired. □

The same proof almost works in the case where the players draw from their own pools of tiles if we reduce from directed *vertex-partizan* vertex geography, because then we know which player places the first tile for each vertex. However, the other player needs to be able to choose the second tile for each vertex, and then the original player needs to be able to choose the remaining third tile, meaning we do not know which pools should have those two tiles. There is in fact an even simpler proof that avoids this problem:

Theorem 6.5. *The 2-player signed and unsigned edge-matching games are PSPACE-complete, whether players draw from their own pools of tiles or from a shared pool.*

Proof. We reduce from a version of edge geography. For signed edge matching, we reduce from directed edge geography. For unsigned edge matching, we reduce from undirected edge geography. For players drawing from their own tile pools, we reduce from edge-partizan edge geography. For players drawing from a shared pool, we reduce from impartial (nonpartizan) edge geography. All four of these versions of edge geography are PSPACE-complete by [15], [24] and Theorem 6.3.

In all cases, the reduction creates a single tile for each edge in the graph. For a directed edge (u, v), we make a signed tile ${}_{-u}\overset{U}{\underset{U}{\square}}{}_{+v}$. For an undirected edge $\{u, v\}$, we make an unsigned tile ${}_{u}\overset{U}{\underset{U}{\square}}{}_{v}$. Each U denotes a globally unique color, so these tiles can be rotated only by 180°. In the own-pool case, we put the tile in the pool of the player that can play the corresponding edge in edge-partizan geography. We set the left boundary edge color to $+s$ in the signed case and s in the unsigned case, where s is the given start vertex. We define the board size n to be the number of tiles (the number of edges in the input graph) so that there is no additional limit on the number of moves.

We claim that the resulting 2-player edge matching game faithfully simulates the edge geography game. By the left edge color, the first tile must have an edge colored s, and in the signed case, the edge must be colored $-s$; equivalently, the first edge played in geography must be incident to s, and in the directed case, it must be an edge outgoing from s. In a general move, the rightmost tile's right edge (exposed) color v represents the vertex v most recently visited by the path, and the current player must choose a tile representing an edge incident to or outgoing from that vertex, revealing the other endpoint of that edge. Because each tile can be played only once, each edge can be played only once (edge geography). The last player to play a tile/edge wins the game. □

Acknowledgments This work was initiated during open problem solving in the MIT class on Algorithmic Lower Bounds: Fun with Hardness Proofs (6.892) in Spring 2019. We thank the other participants of that class for related discussions and providing an inspiring atmosphere. We also thank the anonymous referees for helpful comments.

References

[1] Ariyibi, D., Gabor, J. and Williams, A.: Personal communication (2019).

[2] Bang-Jensen, J., Bessy, S., Jackson, B. and Kriesell, M.: Antistrong digraphs, *Journal of Combinatorial Theory, Series B*, Vol. 122, pp. 68–90 (online), DOI: 10.1016/j.jctb.2016.05.004 (2017).

[3] Berger, R.: The undecidability of the domino problem, *Memoirs of the American Mathematical Society*, Vol. 66 (1966).

[4] Berman, K. A.: Aneulerian digraphs and the determination of those Eulerian digraphs having an odd number of directed Eulerian paths, *Discrete Mathematics*, Vol. 22, No. 1, pp. 75–80 (online), DOI: 10.1016/0012-365X(78)90048-1 (1978).

[5] Bodlaender, H. L.: Complexity of Path-Forming Games, *Theor. Comput. Sci.*, Vol. 110, No. 1, pp. 215–245 (online), DOI: 10.1016/0304-3975(93)90357-Y (1993).

[6] Bondy, J. A. and Murty, U. S. R.: *Graph Theory with Applications*, North-Holland (1976).

[7] Bosboom, J., Demaine, E. D., Demaine, M. L., Hesterberg, A., Manurangsi, P. and Yodpinyanee, A.: Even $1 \times n$ edge-matching and jigsaw puzzles are really hard, *Journal of Information Processing*, Vol. 25, pp. 682–694 (2017).

[8] Brightwell, G. R. and Winkler, P.: Counting Eulerian Circuits is #P-Complete, *Proceedings of the 7th Workshop on Algorithm Engineering and Experiments and the 2nd Workshop on Analytic Algorithmics and Combinatorics*, Vancouver, Canada, pp. 259–262 (2005).

[9] Demaine, E. D.: Lecture 10: #P and ASP, *MIT class 6.892: Algorithmic Lower Bounds: Fun with Hardness Proofs* (2019). http://courses.csail.mit.edu/6.892/spring19/lectures/L10.html. Originally Lecture 15 of MIT class 6.890 in 2014.

[10] Demaine, E. D. and Demaine, M. L.: Jigsaw puzzles, edge matching, and polyomino packing: Connections and complexity, *Graphs and Combinatorics*, Vol. 23, No. 1, pp. 195–208 (2007).

[11] Ding, L., Fu, B. and Zhu, B.: Minimum Interval Cover and Its Application to Genome Sequencing, *Proceedings of the 5th International Conference on Combinatorial Optimization and Applications* (Wang, W., Zhu, X. and Du, D.-Z., eds.), Zhangjiajie, China, pp. 287–298 (2011).

[12] Filho, I. T. F. A.: Characterizing Boolean Satisfiability Variants, M.eng. thesis, Massachusetts Institute of Technology (2019).

[13] Fleischner, H.: Chapter VI: Various Types of Eulerian Trails, *Eulerian Graphs and Related Topics: Part 1, Volume 1*, Annals of Discrete Mathematics, Vol. 45, North-Holland (1990).

[14] Fleischner, H.: Chapter X: Algorithms for Eulerian Trails and Cycle Decompositions, Maze Search Algorithms, *Eulerian Graphs and Related Topics: Part 1, Volume 2*, Annals of Discrete Mathematics, Vol. 50, North-Holland (1991).

[15] Fraenkel, A. S., Scheinerman, E. R. and Ullman, D.: Undirected edge geography, *Theoretical Computer Science*, Vol. 112, No. 2, pp. 371–381 (1993).

[16] Fraenkel, A. S. and Simonson, S.: Geography, *Theoretical Computer Science*, Vol. 110, No. 1, pp. 197–214 (online), DOI: 10.1016/0304-3975(93)90356-X (1993).

[17] Garey, M. R. and Johnson, D. S.: *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W. H. Freeman & Co. (1979).

[18] Grünbaum, B.: Antidirected Hamiltonian paths in tournaments, *Journal of Combinatorial Theory, Series B*, Vol. 11, No. 3, pp. 249–257 (online), DOI: 10.1016/0095-8956(71)90035-9 (1971).

[19] III, H. B. H., Marathe, M. V., Radhakrishnan, V. and Stearns, R. E.: The Complexity of Planar Counting Problems, *SIAM Journal on Computing*, Vol. 27, No. 4, pp. 1142–1167 (online), DOI: 10.1137/S0097539793304601 (1998).

[20] Kotzig, A.: Moves Without Forbidden Transitions in a Graph, *Matematický časopis*, Vol. 18, No. 1, pp. 76–80 (1968).

[21] Lichtenstein, D. and Sipser, M.: Go is polynomial-space hard, *Journal of the ACM*, Vol. 27, No. 2, pp. 393–401 (1980).

[22] Miltzow, T.: Tron, a Combinatorial Game on Abstract Graphs, *Proceedings of the 6th International Conference on Fun with Algorithms* (Kranakis, E., Krizanc, D. and Luccio, F. L., eds.), Lecture Notes in Computer Science, Vol. 7288, Springer, pp. 293–304 (online), DOI: 10.1007/978-3-642-30347-0_29 (2012).

[23] Plesník, J.: The NP-completeness of the Hamiltonian cycle problem in planar diagraphs with degree bound two, *Information Processing Letters*, Vol. 8, No. 4, pp. 199–201 (online), DOI: 10.1016/0020-0190(79)90023-1 (1979).

[24] Schaefer, T. J.: On the Complexity of Some Two-Person Perfect-Information Games, *Journal of Computer and System Sciences*, Vol. 16, No. 2, pp. 185–225 (online), DOI: 10.1016/0022-0000(78)90045-4 (1978).

[25] Seta, T.: The Complexities of Puzzles, Cross Sum, and their Another Solution Problems (ASP), Senior thesis, University of Tokyo (2002).

[26] Thurston, E. L.: Puzzle, US Patent 487,798 (1892). https://patents.google.com/patent/US487798.

[27] Valiant, L. G.: The complexity of computing the permanent, *Theoretical Computer Science*, Vol. 8, No. 2, pp. 189–201 (online), DOI: 10.1016/0304-3975(79)90044-6 (1979).

[28] West, D. B.: *Introduction to Graph Theory*, Pearson Education, 2nd edition (2001).

[29] Wikipedia: Eternity II, https://en.wikipedia.org/wiki/Eternity_II_puzzle (2019).

[30] Yato, T. and Seta, T.: Complexity and Completeness of Finding Another Solution and Its Application to Puzzles, *IEICE Transactions on Fundamentals of Electronics, Communications, and Computer Sciences*, Vol. E86-A, No. 5, pp. 1052–1060 (2003). Also IPSJ SIG Notes 2002-AL-87-2, 2002.

[31] Žitnik, A.: Anti-directed Walks in 4-valent Graphs, Preprint series, volume 34, number 530, University of Ljubljana, Ljubljana, Slovenia (1996).

Jeffrey Bosboom received a B.S. degree from the University of California, Irvine, in 2011, and M.S. and Ph.D. degrees from Massachusetts Institute of Technology in 2014 and 2020, respectively. His research interests are compilers and programming languages for performance engineering, and the computational complexity of games and puzzles.

Charlotte Chen received her B.S. degree in Computer Science and Mathematics from Massachusetts Institute of Technology in 2019.

Lily Chung received her B.S. degree from Massachusetts Institute of Technology in 2019. Her research interests are cryptography, formally verified programming, and complexity theory.

Spencer Compton is pursuing a B.S. degree in Computer Science and Engineering from the Massachusetts Institute of Technology.

Michael Coulombe received his B.S. degree from the University of California at Davis in 2013 and his M.S. degree from the Massachusetts Institute of Technology in 2015. He is currently working towards a Ph.D. under Erik Demaine.

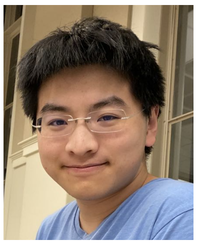
Erik D. Demaine received a B.Sc. degree from Dalhousie University in 1995, and M.Math. and Ph.D. degrees from the University of Waterloo in 1996 and 2001, respectively. Since 2001, he has been a professor in computer science at the Massachusetts Institute of Technology. His research interests range throughout algorithms, from data structures for improving web searches to the geometry of understanding how proteins fold to the computational difficulty of playing games. In 2003, he received a MacArthur Fellowship as a "computational geometer tackling and solving difficult problems related to folding and bending— moving readily between the theoretical and the playful, with a keen eye to revealing the former in the latter". He cowrote a book about the theory of folding, together with Joseph O'Rourke (*Geometric Folding Algorithms*, 2007), and a book about the computational complexity of games, together with Robert Hearn (*Games, Puzzles, and Computation*, 2009). With his father Martin, his interests span the connections between mathematics and art.

Martin L. Demaine is an artist and computer scientist. He started the first private hot glass studio in Canada and has been called the father of Canadian glass. Since 2005, he has been the Angelika and Barton Weller Artist-in-Residence at the Massachusetts Institute of Technology. Martin works together with his son Erik in paper, glass, and other material. Their artistic work includes curved origami sculptures in the permanent collections of the Museum of Modern Art in New York, and the Renwick Gallery in the Smithsonian. Their scientific work includes over 140 published joint papers, including several about combining mathematics and art.

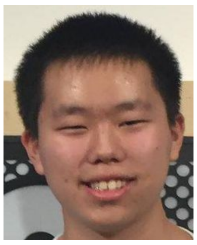
Ivan Tadeu Ferreira Antunes Filho received B.S. and M.Eng. degrees from Massachusetts Institute of Technology in 2017 and 2019, respectively. His research interests are Boolean Satisfiability Problems, the computational complexity of simple games, and more recently, resource-constrained project scheduling heuristics.

Dylan Hendrickson is a graduate student in computer science at MIT studying the computational complexity of motion planning problems under Erik Demaine. Dylan received a B.Sc. degree in mathematics and physics from MIT in 2019.

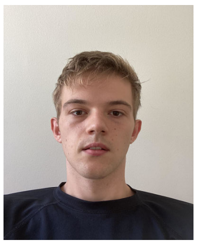
Adam Hesterberg received an A.B. degree summa cum laude from Princeton University in 2011 and a Ph.D. degree from Massachusetts Institute of Technology in 2018. He is now Assistant Director of Undergraduate Studies in Computer Science at Harvard University.



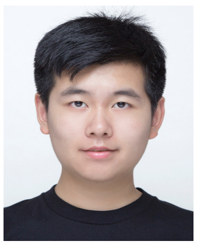
Calvin Hsu received a B.S. degree in Mathematics and Computer Science from the Massachusetts Institute of Technology. He is also currently pursuing a Ph.D. degree in Mathematics in Stanford University.



William Hu is a junior undergraduate at the Massachusetts Institute of Technology, Class of 2022. His interests include machine learning and algorithms. He has participated in a variety of programming competitions, including the 2018 International Olympiad in Informatics as a member of the USA team.



Oliver Korten was born in 1998. He recieved a B.S. degree in Mathematics and Computer Science from Tufts University in 2020. He is currently pursuing a Ph.D. degree in Computer Science at Columbia University.



Zhezheng Luo was born in 2000. He is pursuing a B.S. degree in Mathematics and Computer Science from the Massachusetts Institute of Technology.

Lillian Zhang received a B.S. degree in Mathematics with Computer Science from the Massachusetts Institute of Technology in 2020.