hexagon:2611.01244v1

Push and PushPush Puzzles Are PSPACE-Complete

Xuyang Peng*

Abstract

In the pushing-block puzzles Push-k and PushPush-k, a player walks on a grid and pushes a line of at most k boxes, by one cell in Push and until stopped in PushPush; $k = *$ means unbounded strength. Hoffmann proved Push-$*$ NP-hard in 2000 and asked whether it is PSPACE-complete; the question has remained open since. We prove that Push-$*$ and PushPush-$*$ are PSPACE-complete, as are the storage versions, where the boxes must end exactly on prescribed targets, for every fixed $k \geq 1$ and $k = *$, for both Push and PushPush, with and without fixed walls. Together with known results, this completes their complexity classification. All main theorems are formally verified in Lean 4.

At unbounded strength, no block of boxes is immovable, so standard box-wall constructions fail. For Push-$*$S, we instead use the exact target to freeze the boxes around a constraint-logic board. For PushPush, we simulate self-closing doors with sliding boxes; for storage goals, a cleanup chain resets every accepting run to one fixed configuration, covering all strengths at once. For Push-$*$ and PushPush-$*$, we replace the target with a wall-free gadget that requires a single configuration to pass a fixed sequence of row and column counts to reach the goal. For Push storage, one walled construction forces every solution to use single-box pushes at any strength, and $(k+1)$-wide box-wall gadgets give the wall-free case for fixed k.

* Shanghai Jiao Tong University. Email: controlvector@sjtu.edu.cn

Contents

1 Introduction **3**
1.1 Our results . . . 3
1.2 Related work . . . 4
2 Technical overview **5**
2.1 Push-∗S: trains of boxes and a protected background . . . 5
2.2 Push-∗: a certificate read from one configuration . . . 7
2.3 PushPush-kS and PushPush-∗: doors that store one box . . . 10
2.4 Push-kFS and Push-kS: single pushes at every strength . . . 12
3 Preliminaries **13**
4 Push-∗S **13**
4.1 The source problem . . . 13
4.2 The board with walls . . . 13
4.3 The board without walls . . . 17
5 Push-∗ **25**
5.1 Reading programs and readers . . . 25
5.2 The modified board and the certificate theorem . . . 32
5.3 Proof of the certificate theorem . . . 36
6 PushPush-kS and PushPush-∗ **45**
6.1 The work board: doors, locations, and arbitrary wiring . . . 45
6.2 Freezing the backbone without walls . . . 51
6.3 Exact storage: PushPush-kS for every $k \geq 1$ and $k = *$ . . . 52
6.4 Player goal: PushPush-∗ . . . 54
7 Push-kFS and Push-kS **56**
7.1 One walled construction for every strength: Push-kFS . . . 57
7.2 Removing walls at finite strength: Push-kS . . . 60
References **65**
A The source problem **67**
A.1 Bounded-bandwidth strip drawing and planarization . . . 67
A.2 Comb cut, two-page layout, and heavy-edge placement . . . 68
A.3 Wire subdivision, cursors, and ordered orientations . . . 70
B Details for Push-∗S **72**
B.1 Coordinates, scales, and separators . . . 72
B.2 Hole budgets, core freezing, and deep crossings . . . 74
B.3 Soundness: cut-off cases . . . 76
B.4 Completeness: reversing edges . . . 78
C Details for Push-∗ **78**
C.1 Readers: coordinates, rank bounds, recursion, and maximal slides . . . 79
C.2 Scales, fullness of $\mathcal{P}_\tau$, and completeness . . . 83
C.3 Separator certification and level-by-level freezing . . . 86
C.4 Occupancy strings and partner repair . . . 88
C.5 Decoding arithmetic and five-trap case analysis . . . 89
D Details for PushPush-kS and PushPush-∗ **91**
D.1 Coordinates, room layout, and walkability . . . 91
D.2 Door transitions and exhaustive wrong-push case analysis . . . 93
D.3 Seven-round quota accounting and verification . . . 95
E Details for Push-kFS **95**
E.1 Gadget coordinates, phase encoding, and the AND fan-out . . . 95
E.2 Single-box macros . . . 97
F Details for Push-kS **98**
F.1 Parameterized utilities, core rectangles, normal slots, and macro routes . . . 98
F.2 Fixed-core routing, offset adapters, and global walkability . . . 100

1 Introduction

Pushing-block puzzles are a classical family of motion-planning problems with movable obstacles. A player walks on a rectangular grid whose cells are empty or occupied by unit boxes. In one move, the player steps into an adjacent empty cell or pushes boxes in the direction of motion. The variants differ in four respects.

- *Movement.* In Push, a push moves the boxes by one cell; in PushPush, they slide until they are stopped by another box, a wall, or the boundary of the rectangle.
- *Strength.* In Push-k and PushPush-k, a push moves a contiguous line of at most k boxes; we write $*$ for unbounded strength.
- *Walls.* Without a suffix, the boundary of the rectangle is the only fixed obstacle; the suffix F permits fixed internal walls.
- *Goal.* Without a suffix, the player must reach a given cell. The suffix S replaces this by an exact storage goal: the final set of boxes must equal a prescribed target set of the same size, and the player may end anywhere.

For example, Sokoban is Push-1FS, which Culberson proved PSPACE-complete [6]. We call a cell that is neither a wall nor occupied by a box a *hole*; the cell of the player is a hole.

1.1 Our results

Two groups of cases have remained open. Hoffmann proved Push-$*$ NP-hard in 2000 with a reduction that also applies to PushPush-$*$, and asked whether the problem is in NP or PSPACE-complete [12]; the question has been listed as open repeatedly since [9, 3, 13]. Among the storage versions, only Push-1FS [6] and Push-1S [11] were known to be PSPACE-complete. We settle both.

▶ **Theorem 1.** *Push-$*$, PushPush-$*$, and PushPush-$*$F are PSPACE-complete.*

▶ **Theorem 2.** *For every fixed $k \geq 1$ and for $k = *$, Push-kS, Push-kFS, PushPush-kS, and PushPush-kFS are PSPACE-complete.*

Membership in PSPACE is standard (Section 3), so both theorems are hardness results; for $k = 1$, Push-1S and Push-1FS were known, and we include them for uniformity. Together with the known results, the theorems complete the classification in Table 1: for every fixed $k \geq 1$ and for unbounded strength, Push and PushPush are PSPACE-complete under both goals, with and without fixed walls.

Table 1. PSPACE-completeness of pushing-block puzzles. Citations give prior results; **New** marks results of this paper. Rows give the movement rule and suffixes, columns the pushing strength.

Problem family	1	Fixed $k \geq 2$	$*$
Push	[10, 13]	[3]	**New**
Push (F)	[2]	[8]	[5, 3]
Push (S)	[11]	**New**	**New**
Push (FS)	[6]	**New**	**New**
PushPush, PushPush (F)	[9]	[9]	**New**
PushPush (S), PushPush (FS)	**New**	**New**	**New**

This paper is mathematically self-contained. In addition, we have completely formalized the proofs of Theorems 1 and 2 in Lean 4; the formalization is available at `https://github.com/Galuisss/push-puzzles-lean`.

1.2 Related work

Walls and rigid backgrounds. Every F variant contains its wall-free counterpart and inherits its lower bounds. Bremner, O'Rourke, and Shermer showed in an unpublished manuscript that Push-$*$F is PSPACE-complete [5]; Ani et al. later gave a complete proof [3]. Demaine, Hearn, and Hoffmann proved Push-kF PSPACE-complete for every fixed $k \geq 2$ [8], and Push-1F followed in 2022 [2]. Without walls, a reduction needs boxes that act as walls. At fixed strength k, a $(k+1) \times (k+1)$ square of boxes can never move and serves as a fixed wall [9]; such walls are thick, so gadgets built from thin fixed walls do not carry over directly. With such backgrounds, the player-goal versions are resolved at every fixed strength: PushPush-k is PSPACE-complete for every $k \geq 1$ [9], Push-k for every $k \geq 2$ [3], and Push-1 was recently resolved by two independent proofs [10, 13].

Unbounded strength. Hoffmann's NP-hardness proof for Push-$*$ [12] is also included in [7]. The question whether Push-$*$ is PSPACE-complete was listed as open in 2004 [9] and 2024 [3], and was raised again after the resolution of Push-1 [13]. At unbounded strength a single push may move an arbitrarily long line of boxes, so no block of boxes, however thick, is rigid, and the player can shift or break any part of a gadget. Since Push-$*$F is PSPACE-complete, Hoffmann's question is about boards without fixed walls.

Storage goals. Culberson's proof for Sokoban [6] simulates a linear bounded automaton with fixed walls. Hearn and Demaine introduced nondeterministic constraint logic (NCL) and reduced it to Push-1S, using 2×2 blocks of boxes as walls [11]. No exact storage version of PushPush was known. The storage versions do not follow from the player-goal results, even with fixed walls: an exact storage goal fixes the final position of every box, while in the known reductions the boxes that carry the computation end in positions that depend on the run. For example, in the reduction for PushPush-k [9], the boxes that encode automaton states and input symbols end in channels that depend on the chosen word.[1] Nor does a reduction for one strength carry over to another, since a stronger player has more moves.

Checkable gadgets. Allowing gadgets to be misused and catching misuse in a subsequent forced check is an established method: Ani et al. [2, 3] introduced *checkable gadgets* for Push-1F, and the MIT Hardness Group et al. [13] extended this framework to reconfiguration via *checkable gizmos* for Push-1. Our contribution is therefore not end-state checking itself, but the problem-specific constructions needed where existing gadgets and rigid box walls fail.

Source problems. NCL is PSPACE-complete for planar graphs [11], and van der Zanden showed that it stays PSPACE-complete for planar graphs of bounded bandwidth [16], which our Push-$*$ constructions use to keep the nesting of gadgets bounded. For PushPush we reduce from reachability in networks of directed symmetric self-closing doors, which Ani et al. proved PSPACE-complete [1].

[1] Barish and Shibuya [4] note PSPACE-completeness of single-box PushPush with a covering goal, in which every target must be occupied and additional boxes may remain elsewhere. Their instances have more boxes than targets. On instances with equally many boxes and targets the two goals coincide, so Theorem 2 also gives hardness for covering goals; hardness for covering goals does not imply hardness for exact storage goals.

2 Technical overview

Following the checkable-gadget and checkable-gizmo approach [2, 3, 13], we catch unintended moves at the end rather than forbidding them locally. Our contributions are the constructions for each regime: multi-scale separator freezing for Push-∗S (Section 2.1), a wall-free count *reader* and certificate for Push-∗ (Section 2.2), role-swapping doors and a cleanup route for PushPush (Section 2.3), and a single-push move-graph condition with thick-wall gadgets for Push storage (Section 2.4).

2.1 Push-∗S: trains of boxes and a protected background

The source problem. A *constraint graph* [11] has oriented edges and two types of vertices, each with three edges. At an *OR* vertex at least one edge must point in, and at an *AND* vertex its *heavy* edge or both of its *light* edges must point in. An edge pointing into a vertex is a resource held by that vertex, and a *move* reverses one edge, handing the resource to the other endpoint, so that all constraints stay satisfied. NCL asks whether moves lead from a satisfying orientation σ to another one, τ; it is PSPACE-complete [11].

AND and OR gadgets. We first build the board with fixed walls. Each edge e becomes a long vertical line of boxes, its *train*, with an end at each endpoint of e. The train has two canonical positions one cell apart, and it *points into* the end at which it stops one cell short, which leaves room there. Figure 1 shows our gadget for an AND or OR vertex in representative states. The train of its middle edge ends above a gap of holes, the *core*, and the trains of its left and right edges pass on whether they point in through horizontal lines of boxes, the *bars*, and a single box, the *short bar*. The boundary rows and columns of the board form the empty *frame*, where the player starts, and empty paths from the frame, the *corridors*, reach the gadget from below. The core is joined to the corridors exactly when the edges that point in satisfy the constraint of the vertex. Otherwise the core is *cut off*, as in (d) and (g), and the gadget is built so that *a cut-off core can only be reopened by a push from inside it*. Our AND gadget lets its middle edge point in only after its left edge does, so we restrict NCL to such *ordered* orientations; this keeps it PSPACE-hard (Appendix A).

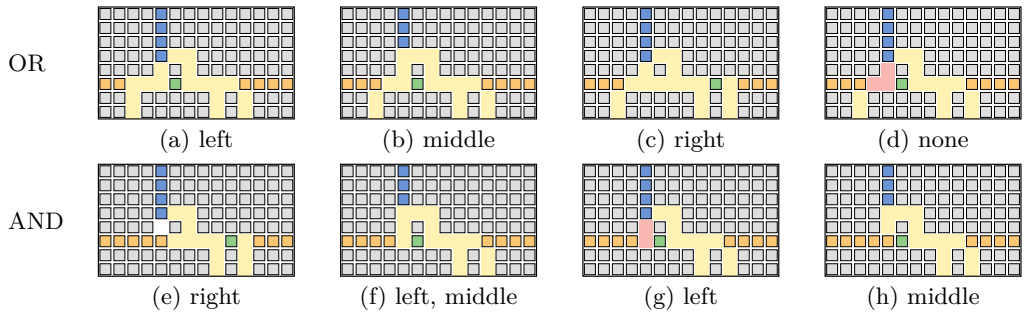


Figure 1. An OR (top) and an AND with its heavy edge on the right (bottom), labelled by the edges that point in. Trains are blue, bars orange, short bars green, walls gray, holes joined to the frame through the bottom corridors yellow, other holes white; in (d) and (g) the core (red) is cut off, and in (h), which is not ordered, it has closed.

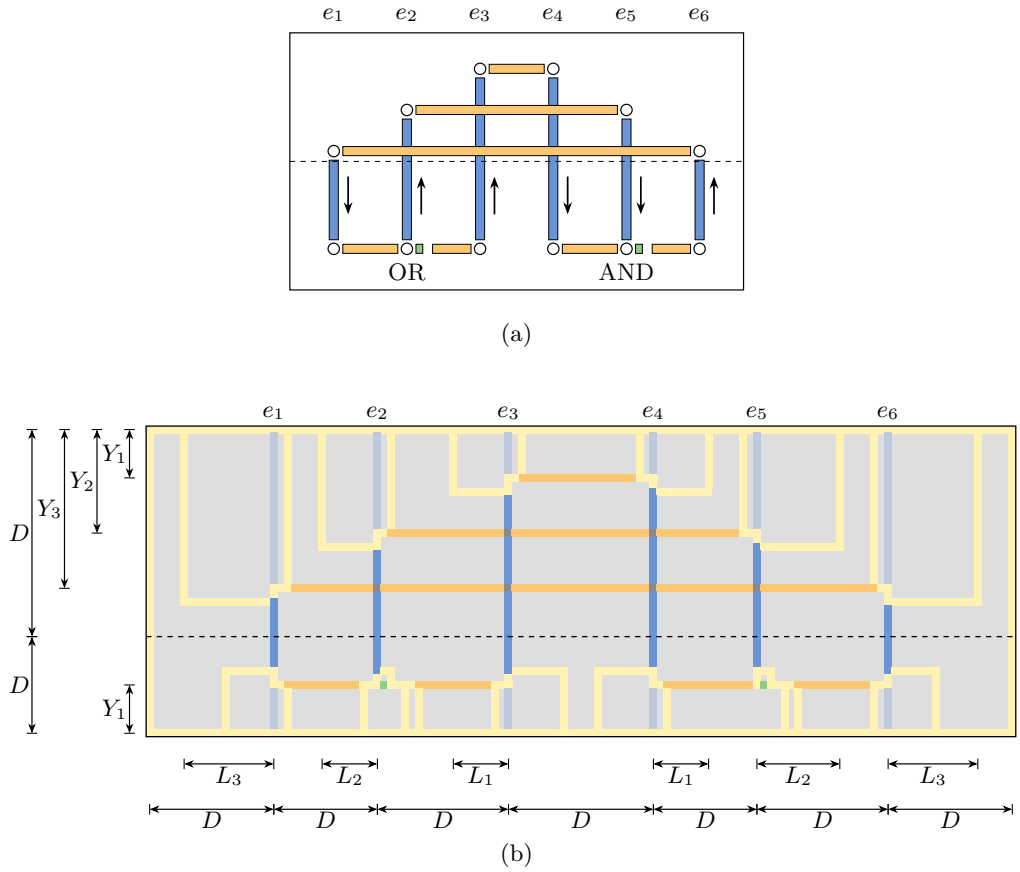


Figure 2. (a) A graph drawn as in Section 4.2, not to scale: three nested wires in the top half, an OR and an AND in the bottom half; the central row is dashed, and the arrows give an orientation. (b) The board of (a) in the canonical configuration of this orientation, compressed: long uniform stretches are drawn as two cells, so local structure is exact and long distances are not to scale; the scales are those of Section 4.3. Trains are blue, bars orange, short bars green, crossing cells dark, and holes yellow; gray cells are walls for Push-∗FS and boxes for Push-∗S, and light blue cells continue the column of a train to the frame.

Wires and layout. To route the edges, we subdivide them by *wires*, vertices with two edges that need at least one of them to point in. A bar joins the two trains of a wire, and where a train meets the bar there is a *pocket* of three holes, which is joined to a corridor while its train or the bar points into it. The bar points into one end only, so if neither train points in, a pocket is cut off, and again only a push from inside can reopen it. Figure 2 shows a whole board. The *central row* splits it into two halves, with wires above and wires, ANDs, and ORs below; each half's vertices nest like parentheses, with bars on their own *working row*.

This gives soundness with walls. The exact target B_τ is the canonical configuration of τ, so trains and bars keep their lengths along a successful trajectory and every push reverses at most one edge. Consider the first push after which some vertex u is violated. The constraints are monotone, so this push reversed an edge away from u by moving its train one cell towards u from the far end, where the player now stands; a pocket or the core at u is now cut off with the player outside, and it stays cut off, while B_τ has none. Hence every orientation along the trajectory is satisfying, which proves Push-∗FS PSPACE-complete (Theorem 5); completeness pushes each train from the open pocket or core at its new endpoint.

Walls become boxes. For Push-∗S we fill every wall cell with a box. The *tracks* are the columns of the trains (continued to the frame, except that a middle train stops at its working row) and the bar rows between their trains; the boxes off the tracks form the *background*. At unbounded strength the player could push a long segment of background along its row and later push it back, which the target cannot detect. The main lemma (Lemma 6) says that *along every trajectory that ends with B_τ, no push moves a box off the tracks or fills a hole off the tracks*; then the proof with walls applies word for word.

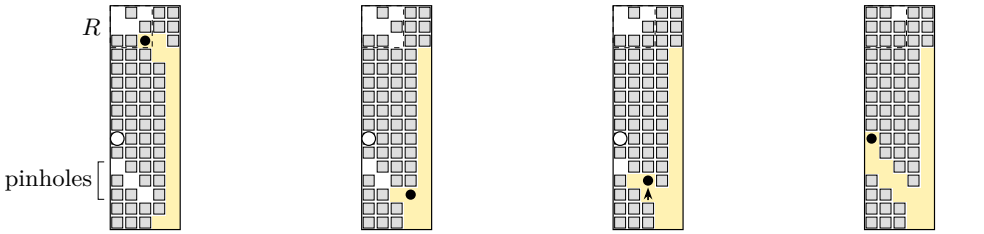
The tool is that the player cannot go far from the empty frame, since few holes lie elsewhere. If the cells of a set A stay empty and h holes lie outside A, then after leaving A at a cell s the player stays within distance h of s until it returns to A (Lemma 8), because a push moves a hole to the player's new cell, which lies between the player and the old place of the hole. So a hole farther than h from A that is a hole at the start and the end is never filled, and it keeps a segment of boxes through it from being pushed along its length; we add such single holes, the *separators*. Running the trajectory backwards, pulls keep the holes of B_τ on a set of pairwise non-adjacent rows and columns, the *streets*, so no box off the streets ever moves. On the streets, the corridors are the problem, since they bring holes deep into the board. A vertex of level j has its working row at depth Y_j and its train corridors at distance L_j from its trains, where the scales $Y_1 < L_1 < \cdots < Y_m < L_m < D$ grow by a factor $\Theta(n)$ for n trains. We proceed from the central row outwards: the separators of one level are blocking, so its corridors stay empty and join A, and h drops enough for the separators at the next, shorter scale (Figures 11–15 in Section 4.3). Planar NCL of bounded bandwidth [16] gives a constant nesting depth m, so the board has polynomial size.

2.2 Push-∗: a certificate read from one configuration

A player goal says nothing about the boxes. The proof of Section 2.1 used the exact target in three places: it started the backward street argument and kept the separators empty; it kept trains and bars at their lengths, so that tracks never exchange boxes; and it excluded cut-off pockets and cores at the end. Without it, the player could, for instance, cut off a core and then walk to the goal. We replace the target by a check that a player goal can enforce: *the hole set of a single configuration passes a fixed list of row and column counts.* This has two halves. A *reader* turns the player goal into such a check on any wall-free board, and a *certificate theorem* shows that the check suffices on our board. Neither forces a canonical final configuration, so we must extract from counts what the walled proof needs.

Reading programs. Let E be a set of holes of a board R. A *read* chooses a side of R and a quota $q_L \geq 0$ for every line L (row or column) perpendicular to it; on each such line it deletes the first q_L holes of E counted from that side, and it fails if L has fewer. A *clear* deletes all remaining holes on chosen lines. A *program* is a fixed sequence of reads and clears, and it *accepts* E if no read fails. It is *full* for E if its quotas add up to $|E|$. Pushes never change the number of holes, so fullness depends only on the initial configuration, and when a full program accepts, every clear deletes nothing: a clear then checks that earlier reads have emptied a line. So the order of a full program gives lower bounds (reads succeed) and upper bounds (clears delete nothing) at every stage. Acceptance is monotone: if a program accepts $F \subseteq E$, it accepts E. Hence deleting extra holes only makes later reads harder—if the player still passes, the prescribed deletions also pass—so the reader may allow it.

A reader for one read. Consider a read from the bottom with quotas q_x (Figure 3). We surround R by boxes, except for a single *exit cell* next to the lower right corner c of R, a



(a) at c (b) exit filled (c) one column read (d) at the goal

Figure 3. One read from the bottom with quotas $2, 1, 1$ on a 3×3 board R (dashed), on the actual reader; holes the player can reach are yellow. (a) The player is at the lower right corner c of R. (b) The exit cell east of c is filled, and the holes of R are out of reach. (c) One push north from its pinhole has filled the lowest hole of the right column. (d) All columns are read, and the player is at the goal (circle).

corridor down the east side, and in every column x a *pinhole* below R. The pinholes form a staircase: the pinhole of the next column to the west lies q_x cells north of that of column x, and the goal lies north of the westmost pinhole by the quota of its column. The canonical route leaves R through the exit, walks down the corridor, and fills the exit cell by two pushes north from below. It then visits the pinholes east to west: from column x's pinhole, q_x northward pushes shift the column, fill its lowest q_x holes in R, and reach the next pinhole.

Three counting arguments show that an arbitrary trajectory to the goal pays all quotas from one hole set. First, below R every cell holds a box except the pinholes, and a new hole there can only be made by a push that uses up a pinhole. So the strip west of a column can first be entered from the south only through one of its pinholes; these first entries happen from east to west, and since there are as many pinholes as columns, every pinhole is first entered by a walk from the east, unused. Second, until then the player leaves R only through the exit cell, and a counting argument along the exit column shows that the first pinhole cannot be reached before the exit cell is filled, after which nothing can empty it. Earlier excursions return through the same corner and can be cut out, which leaves a trajectory in R that ends at c with some hole set E. Third, once the exit is filled, approaching R requires pushing columns north, costing one hole of R per row; with more rows below R than holes in R, the player never returns and every quota is paid from E.

Many reads. For a read from another side, we rotate the board so that this side becomes the bottom and build the next reader around it. The previous reader leaves auxiliary rows that cross the lines of the next read; we place reserve holes on them far from R, and the next read takes a fixed number of holes from each, so that all leftover holes avoid the rows and columns of R. A clear is compiled into a read after preparing one reserve hole per row of R below every column of R: taking that many holes from the top of each chosen column always succeeds and deletes whatever holes of R remain on it, just as the clear does. Each step cubes the area, which gives the reader lemma (Lemma 12): *for a wall-free board R and a program $\mathcal{P}$ of fixed length, there is a polynomial-size Push-$*$ instance that contains R and is solvable if and only if R has a reachable configuration with the player at c whose hole set $\mathcal{P}$ accepts.* Our programs have $O(m)$ steps, which is a constant; this is the second place where bounded nesting depth is needed.

The certificate theorem. The board R is that of Section 2.1 with three changes, which we introduce where they are needed. From the canonical configuration B_τ we compute a

full program $\mathcal{P}_\tau$ with $O(m)$ steps that accepts the holes of B_τ, and we prove (Theorem 14): *if a trajectory from B_σ ends with a hole set E that $\mathcal{P}_\tau$ accepts, then τ is reachable from σ.* The final player position is arbitrary, and E need not be canonical; for example, a short bar may end one cell beyond its canonical position. So the proof derives from the counts exactly what the argument with walls needs: the background stays fixed, the tracks keep their boxes, the orientation can be decoded, and nothing is cut off.

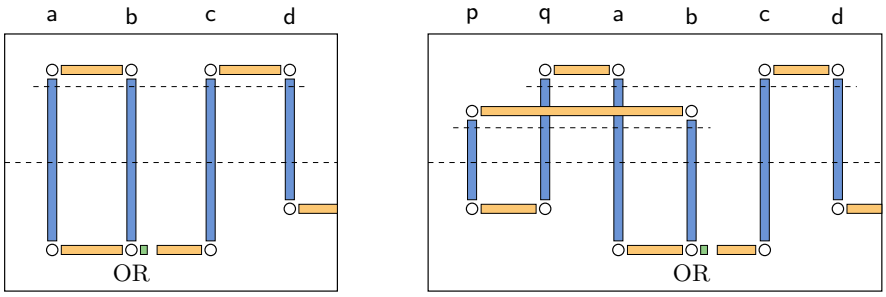
Fixing the background from counts. The first reads take every cell of the boundary lines, so the frame is empty at the end and hence throughout, and the number of other holes is fixed. Every quota takes holes on read lines and every clear deletes nothing, so every hole of E lies on a read line; the order of the reads places E in two unions of pairwise non-adjacent rows and columns, whose intersection is essentially the streets. The backward argument of Section 2.1 needs only this: if E lies in such a union, then so do the holes of every configuration of the trajectory, wherever the player ends. A read counts holes from an end of a line, so by itself it cannot certify that a particular cell, such as a separator on the central row, is still a hole. We therefore add *marker holes* a few cells from these separators, in both B_σ and B_τ. Markers lie beyond the reach of the player, so they can only be filled; saturated reads of the marker rows show that none ever was, so no push used a marker as its first hole, and turning the markers back into boxes leaves every push legal. With markers as cut points, the reads certify the separators level by level: when the separator row of a level is read, only the columns of that level can still pay, at most one hole each, and the quota equals their number. The stages of Section 2.1 then go through. Their budgets must come from the global number of holes minus holes already shown to stay empty, never from the boxes of individual tracks, which are not known yet.

Tracks that exchange boxes. This is the new difficulty. With an exact target, every train ends with its initial boxes; here, two kinds of exchange remain possible. A train whose lower end meets a long bar of an AND or OR vertex may absorb $J_i \geq 0$ boxes from it, and a middle train may give boxes to the working row of its core. All other exchanges would need the player to stand on or push the fixed background behind a pocket, and reads of the working rows show that every wire bar keeps its boxes. For a train i, let $\theta_i \in \{0, 1\}$ say whether τ points it into its upper end, and let $z_i \in \{0, 1\}$ be its number of holes left on the *probe row* of its level, two rows beyond the working row, when that row is read. The reads of the column of train i give $J_i = \theta_i - z_i$ for an ordinary train, and $z_i - \theta_i \geq 0$ for a middle train, with equality only if it gave no box away. Reading and clearing the probe row of a level gives $\sum_i z_i = \sum_i \theta_i$ over its trains, that is,

$$\sum_{\text{middle trains } i}(z_i - \theta_i) \;=\; \sum_{\text{ordinary trains } i} J_i.$$

Both sides are sums of non-negative terms, but a box absorbed by one train can be paid for by a box given away by another, so the equation alone does not make the terms vanish.

We remove this by changing the graph (Figure 4). Call the other train of the wire at the upper end of a middle train its *partner*. We subdivide edges by extra wires so that every partner is a middle train or has its lower end at a wire, and therefore cannot absorb boxes; and in the top half we give the wires that hold a middle train even levels and all others odd levels. Then on an even probe row every ordinary train has $J_i = 0$, so every middle term is zero; on an odd probe row there is no middle train, so every J_i is zero. Two local operations suffice: an *ear* where a wire joins two adjacent ends of one AND or OR vertex, and a *cut* at an AND or OR vertex whose middle train has an outer end of another such vertex as



(a) (b)

Figure 4. An ear of the partner repair, drawn as in Figure 2(a); dashed lines below the bars of the top half are probe rows. (a) The partner of the middle train b is a, whose lower end can absorb boxes from the bar of the OR, and both meet the probe row of level 1. (b) After inserting p and q with a wire between them in the bottom half, b has partner p on level 2, whose lower end is at a wire, and the probe row of level 1 meets no middle train.

partner. They do not change the answer, and the nesting depth stays at most $2m + 2$, still a constant. Hence every track keeps its boxes throughout.

Decoding and the first violation. Near each core, the reads certify cell by cell the working holes of B_τ; a marker below the central row keeps the other half from paying for them. These cells exclude every position of the short bar on the wrong side, so the bits of each vertex (whether its bars, short bar, and middle train point into the core) are those of τ, and both gaps of its working row are nonempty, so its runs of boxes never merged. Now the first-violation argument of Section 2.1 applies: a first violation produces one of five cut-off shapes (a pocket, the gap beyond the short bar, or one of three shapes of a core), none of them can be reopened from outside, and the decoded final configuration has none. Along an original edge subdivided into a chain, the wires forbid two consecutive edges pointing away from each other, so the chain carries a single cursor that reverses the original edge when it reaches the other end; this gives the moves of NCL.

The instance. The column reads near a core use pairwise non-adjacent (even) columns, so a one-box short bar could hide on an odd column undetected; moreover, the column that leads to the few holes beside each middle train must stay closed to the frame, so that the program can check those holes directly. We therefore widen the short bar to two boxes, which five pushes move across the core and which always covers an even column. The Push-$*$ instance is R with initial configuration B_σ and the reader of $\mathcal{P}_\tau$ attached at its corner. If τ is reachable, each train reversal takes at most eight pushes, after which the player walks to the corner and passes the reader; conversely, passing the reader invokes the certificate theorem.

2.3 PushPush-kS and PushPush-$*$: doors that store one box

A train in PushPush slides the length of the gap in front of it. Our construction stores each door state in a single box resting against a stopper.

Doors. A directed symmetric self-closing door has a state 0 or 1 and two directed passages; in state i only passage i can be traversed, and traversing it changes the state to $1 - i$. Reachability of a location in a network of such doors is PSPACE-complete [1]. Our board

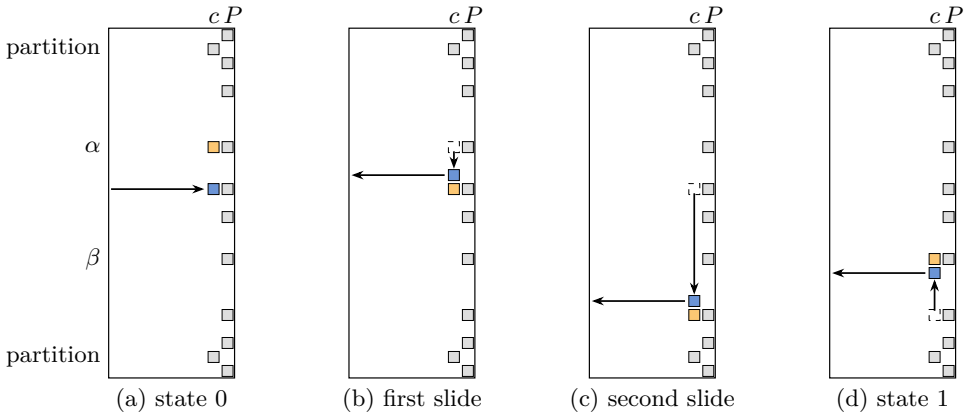


Figure 5. A door room traversing passage 0 from state 0, on the board of Section 6: the working column c, the stopper column P, and a few columns west; each long stretch of empty rows is drawn as one row. The stored box is orange and the signal blue; a dashed square marks the old cell of the box that slid. (a) The signal arrives at an input slot. (b)–(d) Each time a box arrives, the player slides the stored box onto it; the slid box stops next to it and leaves west as the new signal, and the arrived box is stored. After (d) the stored box is at β.

has a column P of fixed *stopper* boxes on its right, and every door and location is a room on the column c west of P, separated from the next room by a *partition*, a single box on c between two holes. A door stores its state in one box at slot α or β of its room (Figure 5), and one further box, the *signal*, carries the player's position through the network: it slides east into an input slot of a room and stops at the stopper in P. The signal and the stored box then lie on c, and the player slides one onto the other; whichever box stays, the other leaves westward along the row where it stops. Boxes are unlabelled, so the stored box may become the signal and vice versa. The six slots of a door are placed so that the correct choices traverse the passage and flip the stored state, while a wrong choice sends a box into a dead row from which no successful trajectory continues. Wires between rooms use ordinary boxes as stoppers at their turns; the player may push these the wrong way, but such pushes lead to dead ends as well (Section 6).

Protecting the stoppers. The doors work only while P and the partitions stay in place. Suppose no box ever touches the board's boundary. Then every slide stops at a box, so no box ever moves east of P; column P can only lose boxes, and vertical slides along P merge runs of boxes that never split again. Hence if P ends with its initial boxes, it never changes; a similar argument keeps every partition, and a storage target gives these conditions.

One target for every run. When the signal reaches the goal location, each door is in a state that depends on the run, so no exact target can be fixed in advance. We add a *cleanup chain* that starts at the goal location and visits the doors one by one. At door i, the signal enters a new slot E_i south of α and β, and the player pushes the old stored box south onto it. The stored box stops one cell north of E_i and becomes the signal for door $i+1$, and the incoming box stays at E_i; neither step depends on the state of door i. The target consists of the boxes that never move (the stoppers, the partitions, the caps that end the rooms, and the boxes at wire turns), the cells E_i, and the final signal cell; the board has quadratic area and a linear number of boxes. A successful trajectory must enter the cleanup chain, and its

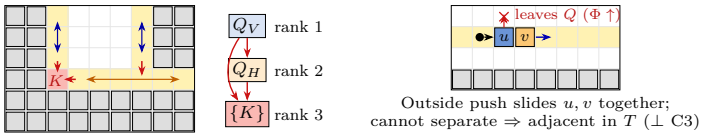
prefix up to that moment decodes into a path to the goal. Completeness uses single pushes and soundness allows unbounded strength, so one instance covers every $k \geq 1$ and $k = *$.

A player goal. Without the cleanup chain, a reader for maximal slides checks the conditions: a northward slide from a pinhole fills a complete run of holes, so the reads become *run reads*, and a full program of seven rounds certifies the frame, the stoppers, the partitions, the caps, and arrival at the goal location. This proves PushPush-$*$ PSPACE-complete.

2.4 Push-kFS and Push-kS: single pushes at every strength

A Push-1FS reduction may fail at higher strength because a stronger player has more moves; we give a static condition forcing every solution to use only single pushes.

Strongly connected components of box moves. Let D_W be the directed graph on non-wall cells with an arc $u \to u + d$ whenever $u - d$ and $u + d$ are non-wall cells of the board (Figure 6). Every pushed box moves along an arc of D_W, including inner boxes of a pushed line. Suppose that (C1) each strongly connected component Q of D_W contains as many initial boxes as targets, (C2) every box-bearing component lies in one row or column, and (C3) no two targets in one component are adjacent. Then every Push-$*$FS solution uses only single pushes: in a topological order of the components, the rank sum Φ rises strictly whenever a box leaves its component, so by (C1) none does; adjacent boxes in a straight component then never separate, against (C3), ruling out multi-box pushes.



(a) Move graph D_W and topological ranks (b) Glued pair in a straight SCC

Figure 6. The SCC transfer argument (Lemma 23): (a) straight tracks form collinear SCCs in D_W, and cross-track arcs raise Φ; (b) adjacent boxes u, v in a straight SCC Q never separate, against (C3).

Push-kFS. We realize Hearn and Demaine's Sokoban reduction [11] with walls replacing immovable blocks, targets for all boxes, and all logic gates in one orientation so that (C1)–(C3) hold. Completeness uses single pushes; conversely, any solution at any strength uses only single pushes, and the local soundness argument below, taken at width 2, decodes it into an NCL path, covering all $k \geq 1$ and $k = *$ (Section 7).

Push-kS for fixed k. Let $w = k + 1$. We replace walls W of thickness w by boxes B_W with targets: if every box of B_W lies in a $w \times w$ square of boxes of B_W, none ever moves, since the first push touching B_W would push at least $w > k$ boxes [9]. Because the width-2 gadgets of [11] do not directly generalize, we redesign every gadget for width w (lines, corners, phase shifters, flips, AND, OR) and prove soundness directly: if two consecutive boxes on a track enclose a walled empty interval with the player outside, reopening it needs a push from inside, so any bad state first entered from outside is permanent and excluded by the target. With B_W frozen, (C1)–(C3) hold for the remaining boxes, proving Push-kS PSPACE-complete for every fixed k; this needs $w > k$ and fails at $k = *$, which is why Push-$*$S requires Section 2.1.

3 Preliminaries

We make the rules of the introduction precise. A board is a rectangle R of cells with a set W of walls, where $W = \emptyset$ unless the problem has the suffix F; whenever we use coordinates, x increases rightward (east) and y increases downward (south). A state (p, B) consists of the player cell p and the *configuration* $B \subseteq R \setminus W$, the set of cells holding boxes. The other cells of $R \setminus W$, including p, are *holes*. If the player moves in direction d and $p + d$ holds a box, it pushes the maximal line of ℓ consecutive boxes starting at $p + d$, never a part of it. The push requires $\ell \leq k$ at strength k, with no bound at strength $*$, and a hole just beyond the line. In Push the line moves by one cell, and in PushPush it slides until the cell in front of it is not a hole; under both rules the player moves to $p + d$. A *trajectory* is a sequence of moves from the initial state. An instance gives the board, the initial state (p_0, B_0), and the goal cell by cell. A storage goal is a target set T with $|T| = |B_0|$, and it is met when the configuration equals T.

Each problem asks for reachability in a graph on at most $|R| \cdot 2^{|R|}$ states, in which an edge can be checked in polynomial space. Hence all problems in this paper lie in NPSPACE = PSPACE.

4 Push-∗S

We reduce from nondeterministic constraint logic (NCL). Section 4.2 first builds the board with walls, which proves Push-∗FS PSPACE-complete. Each edge is a long vertical line of boxes, a *train*, and at each vertex a few holes are joined to the empty frame of the board only while the constraint of the vertex holds; once cut off, they stay cut off. Section 4.3 then replaces every wall by a box. These boxes could now be pushed, and most of the work is to show that the exact target keeps those off the tracks in place.

4.1 The source problem

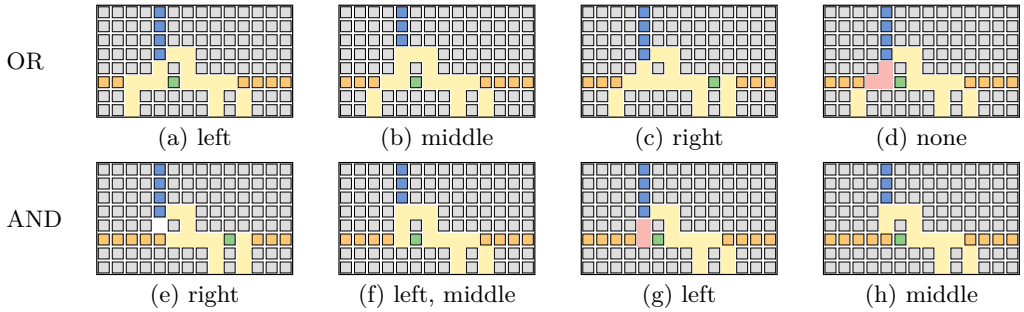
A *constraint graph* [11] has oriented edges and two types of vertices, each with three edges. At an *OR* vertex, at least one of its edges must point in. At an *AND* vertex, its *heavy* edge or both of its two *light* edges must point in. A move reverses one edge so that all constraints stay satisfied. *NCL* asks whether moves lead from a satisfying orientation σ to another one, τ, and it is PSPACE-complete [11].

4.2 The board with walls

Trains. The boundary rows and columns of the board form the empty *frame*, and the player starts at a corner p_0 of it. A box pushed onto the frame could never leave it, and the target leaves the frame empty, so the frame stays empty. Each edge e is a vertical line of boxes, its *train*, with an end at each endpoint of e. The train has two canonical positions, one cell apart, and it *points into* the end at which it stops one cell short; so e points into v when its train is shifted away from v, which leaves room at v. *Corridors* are empty paths from the frame, and all other cells are walls, except for the boxes and holes described below.

AND and OR vertices. Figure 7 shows, in representative states, our gadget for an AND or OR vertex v on three edges, from left to right e_i, e_{i+1}, and e_{i+2}. The train of e_{i+1} ends above a gap of holes, the *core*, and the trains of e_i and e_{i+2} pass on whether they point in through a horizontal line of boxes, a *bar*, and a single box, the *short bar*. The core is joined to

the corridors exactly when the edges that point in satisfy the constraint of v; otherwise it is cut off, and only a push from inside the core can reopen it. Our AND gadget lets e_{i+1} point in only after e_i does, so we call an orientation *ordered* if every AND vertex on e_i, e_{i+1}, e_{i+2} has e_{i+1} pointing in only if e_i points in.



(a) left (b) middle (c) right (d) none

(e) right (f) left, middle (g) left (h) middle

Figure 7. An OR (top) and an AND with its heavy edge on the right (bottom), labelled by the edges that point in (a copy of Figure 1). Trains are blue, bars orange, short bars green, walls gray, holes joined to the frame through the bottom corridors yellow, and other holes white; in (d) and (g) the core (red) is cut off, and in (h), which is not ordered, the left bar meets the short bar and the core has closed.

Wires. To route the edges, we subdivide them by *wires*, vertices with two edges that need at least one of them to point in. A wire has a bar between its two trains, with a *pocket* of three holes where a train meets the bar; the extra hole at the end that a train or bar points into leads to a corridor, so a pocket is joined to the frame if its train or its bar points into it (Figure 8). The bar points into one end only, so if neither train points in, one pocket is cut off, and only a push from inside it can reopen it. The bars of an AND or OR vertex have the same pockets at the trains of e_i and e_{i+2}, so a bar that points into the core while its edge points out cuts off its pocket.



(a) train and bar (b) train (c) bar (d) neither

Figure 8. One end of a wire in the top half, exact crops, with the train below, the bar on the right, and the pocket outlined. The pocket is joined to the frame through the train corridor on the left when its train points in, and through the bar corridor at the top when its bar points in; in (d) it is cut off (red).

Layout. The *central row* splits the board into two halves, and every train crosses it (Figure 9). The upper end of a train belongs to a wire in the top half, and its lower end to a wire in the bottom half or to an AND or OR vertex on three consecutive edges, with the heavy edge of an AND last. Vertices in the same half are nested like parentheses, and each has a *working row* in its half for its bar, closer to the central row than the working rows of the vertices nested inside it, so a bar crosses only trains of vertices nested inside it; both can be pushed through a crossing cell.

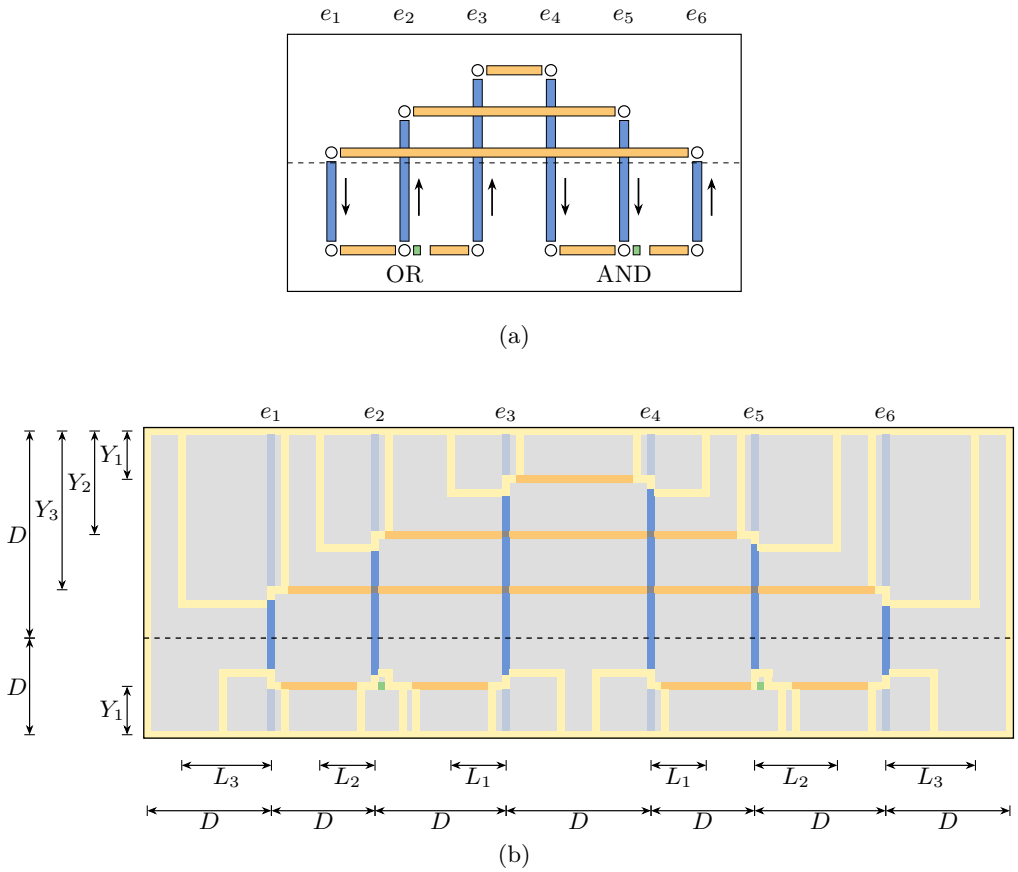


Figure 9. (a) A graph drawn as in Section 4.2, not to scale: three nested wires in the top half, an OR and an AND in the bottom half; the central row is dashed, and the arrows give an orientation. (b) The board of (a) in the canonical configuration of this orientation, compressed: long uniform stretches are drawn as two cells, so local structure is exact and long distances are not to scale; the scales are those of Section 4.3. Trains are blue, bars orange, short bars green, crossing cells dark, and holes yellow; gray cells are walls in Section 4.2 and boxes in Section 4.3, and light blue cells continue the column of a train to the frame (a copy of Figure 2).

Let n be the number of trains; all distances are large compared with n, and Section 4.3 fixes them. Starting from planar NCL of bounded bandwidth [16, Theorem 6], Appendix A shows that NCL stays PSPACE-hard on such graphs with ordered σ and τ, and that restricting moves to ordered orientations does not change the answer.

Reduction. For an ordered satisfying orientation ρ, the canonical configuration B_ρ places every train as ρ says, every bar of an AND or OR vertex so that it points into the core exactly when its edge points in, the short bar in the cell right of the middle train's column when e_{i+2} points out and four cells further right when it points in, and the bar of a wire so that it points into its left end exactly when the left train does not. No pocket or core of B_ρ is cut off, and all B_ρ have equal size. We map an instance (G, σ, τ) to the board with initial configuration B_σ and target set B_τ.

▶ **Lemma 3** (completeness). *If τ is reachable from σ through ordered orientations, then there is a trajectory from (p_0, B_σ) to (p_0, B_τ).*

The player reverses an edge e by one push on its train from the open pocket (or open core, for a middle train) at its new endpoint. Before this push, it moves the bars at the old endpoint so that the pocket of e there stays open, and after it, it moves the bars at the new endpoint into their canonical positions; at an AND or OR vertex, these are the bar at the train of e and, if e is the right edge, the short bar. Appendix B.4 gives the moves.

Tracks. Before reading an orientation off an arbitrary trajectory, we verify that every train and bar stays one run of boxes of its original length and never merges with another run. The *track* of a train is its column up to the walls beyond its ends (for a middle train, down to the cell above its working row), and the track of a bar is its row between its trains. At an AND or OR vertex, the two bars and the short bar share the working row between the trains of e_i and e_{i+2}.

With walls, every push takes place on the tracks. A push through a crossing cell moves boxes along one of its two tracks and keeps the box counts of both. A box can leave its track in only two ways: a box of a bar can enter the corner of a pocket, or the lowest box of a middle train can drop onto its working row, which belongs to the track of the bars. Reversing either transfer would require the player to stand on a wall. Moreover, every such transfer moves a box strictly forward in the order: wire bars, middle trains, working rows of AND and OR vertices, and other trains. Weighting each track's box count by its position in this order gives a non-decreasing potential that agrees on B_σ and B_τ; hence no box ever leaves its track on a trajectory ending with B_τ.

Along any track, a push translates a whole maximal run of consecutive boxes and empties the cell at its trailing end. Runs therefore never split, and once two runs meet they can never separate again. Because B_τ has as many runs on each track as B_σ, no two runs ever meet. For arbitrary (possibly non-canonical) positions, a train *points into* an endpoint when its gap at that end reaches the train corridor (or, for a middle train at its bottom end, leaves at least two holes above the working row). Consequently, trains and bars keep their lengths, the left bar of an AND or OR vertex never meets its short bar, so every core keeps a hole (unlike Figure 7(h)), every train points into a unique endpoint at every moment, and each push reverses at most one edge.

▶ **Lemma 4** (soundness). *If a trajectory from (p_0, B_σ) ends with B_τ, then τ is reachable from σ.*

Proof. Let π be such a trajectory. By the paragraph above, the orientations during π run from σ to τ, and consecutive ones differ in at most one edge, so it suffices to show that all of them are satisfying. Suppose not, and consider the first push after which a vertex u is violated. The constraints are monotone, so this push reversed an edge e away from u: it moved the train of e one cell towards u, from the end of e at its other endpoint, where the player now stands. If u is a wire, neither of its trains points in now, and its bar points into one end only, so one of its pockets is cut off (Figure 8(d)). If u is an AND or OR vertex, either one of its bars points into the core while the train at its pocket does not point in, which cuts off that pocket, or the core is cut off (Figure 7(d),(g)); Appendix B.3 lists the cases. A cut-off pocket or core can be reopened only by a push from inside it: every run of boxes on a track that borders it, a train, a bar, or any other run, moves away from it only when pushed from its side, so pushes from outside only move such runs further in. The player stands at the far end of e, outside this region, so the region stays cut off to the end of π, but B_τ has no cut-off pocket or core, a contradiction. ◀

Since τ is reachable from σ exactly when it is reachable through ordered orientations, Lemmas 3 and 4 and Section 3 give the following.

▶ **Theorem 5.** *Push-∗FS is PSPACE-complete.*

4.3 The board without walls

Without walls, we fill every wall cell of Section 4.2 with a box. The track of a train now continues through these boxes to the frame (for a middle train, down to the cell above its working row), since the player can push the boxes on the column of a train from the frame and later push them back. The new boxes off the tracks form the *background*. A push of unbounded strength could now shift a long segment of the background along its row and later shift it back, which the exact target cannot detect, and meanwhile the trains would no longer be held in their columns. We use the instance of Section 4.2 on the new board, and we choose its distances and add single holes so that this cannot happen.

▶ **Lemma 6** (background). *Along every trajectory from (p_0, B_σ) that ends with B_τ, no push moves a box off the tracks or fills a hole off the tracks.*

▶ **Theorem 7.** *Push-∗S is PSPACE-complete.*

The trajectory of Lemma 3 pushes only trains and bars, so it still works. Conversely, by Lemma 6, background boxes never move and every push takes place on the tracks, as with walls, so the paragraph on tracks and the proof of Lemma 4 apply word for word, with background boxes in place of walls; the only addition is that the boxes continuing a train's column to the frame stay within two cells of their initial position, by the same cut-off argument (Appendix B.3), so whether a train points into an end still means whether its pocket is joined to a corridor.

The rest of this section proves Lemma 6. Fix a trajectory π from (p_0, B_σ) that ends with B_τ.

Holes far from the player. The background is made of long segments of boxes, and the player can push a whole column of it from the frame into a corridor and later push it back. Without walls, the only obstacles we can add are holes that the player never reaches. The frame stays empty, so the number b of other holes never changes, and with so few holes the player cannot get far from the empty cells it comes from.

▶ **Lemma 8** (hole radius). *Let a set A stay empty during π, with $p_0 \in A$ if $A \neq \emptyset$, and let h holes lie outside A. After the player leaves A from a cell s, it stays within Manhattan distance h of s until it returns to A; if $A = \emptyset$, the player stays within Manhattan distance $h - 1$ of its initial cell p_0 throughout π. Hence the player never enters a cell farther than h from A (or farther than $h - 1$ from p_0 when $A = \emptyset$), and such a cell can only be filled: an occupied cell stays occupied, and a hole there at the start and at the end stays a hole, into which no box is ever pushed; we call such a hole* blocking.

Proof. Mark s (or p_0 when $A = \emptyset$), and mark a hole outside A when the player first stands on it or a push first moves it (Figure 10). We show that once i holes are marked, the player and all marked holes lie within distance $i - 1$ of s (resp. p_0). A newly marked hole lies at the player's new cell, next to its previous cell. A push moves the hole beyond the pushed boxes to the cell that the player enters, which lies on the straight segment between the player and the old place of the hole, and a Manhattan ball contains the straight segment between two of its cells. At most $h + 1$ holes get marked when $A \neq \emptyset$ (and at most h when $A = \emptyset$, since p_0 is already one of the h holes), which gives the bound. ◀

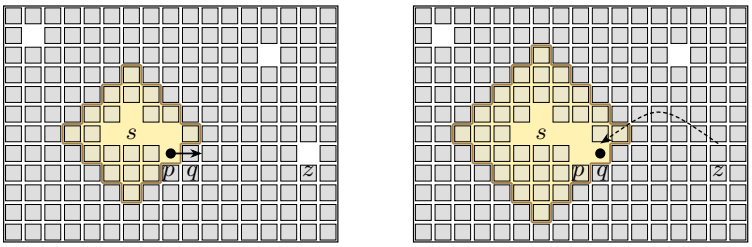


Figure 10. One push in the proof of Lemma 8. (a) The player at p, with all marked holes in the shaded ball around s, pushes the boxes from q up to the first hole z, which is not marked yet. (b) The hole of z has moved to q, the new cell of the player next to p; it is marked now, and the ball with radius one larger still contains the player and every marked hole. Had z been marked, q would lie on the segment from p to z, inside the old ball.

A blocking hole cuts a segment of boxes like a wall. If a straight segment of boxes contains a blocking hole and its side neighbours never move, then its boxes never move: the player could only push them from beyond the ends of the segment, along it, and such a push would fill the hole. The single holes that we add for this purpose are the *separators*; Appendix B.1 lists them all. We start with A the frame, so that $h = b$, and make blocking the separators that are far from the frame or, along their street, from the ends of their segment. Then, one level at a time, we show that the corridors stay empty and add them to A; each time h drops, and the separators at the next, shorter distance become blocking. In the end, every background box on a street lies in a segment that never moves.

Separators on the central row. We place a separator on the central row in every corridor column, and also in each AND or OR vertex's private column (the column two cells past its middle train on the short-bar side, holding the *private branch*, the three holes beside the lower end of the middle train in Figure 7). Each of these separators lies at distance more than $D/2$ from the frame, where D is the depth of each half. Choosing $D > 4b$ makes all of them blocking by Lemma 8, so no push along these columns ever crosses the central row.

Holes stay on streets. The exact target already keeps most of the background motionless. The *streets* of a half consist of its boundary row, the left and right boundary columns, all train columns, its working rows, the rows and columns along which its corridors run, and the private AND/OR columns that received a central-row separator above. Each non-train street column counts only within its own half; continuing it into the opposite half would open a path from the far frame into a corridor, so those opposite-half cells stay off the streets. We place all street rows and columns on even coordinates so that no two street rows are adjacent and no two street columns—even from opposite halves—are adjacent. Every hole of B_τ lies on a street of its half or at a central-row separator.

Now run π backwards from B_τ. A push in reverse is a *pull*: the player steps from a cell x back to an adjacent hole and pulls boxes from beyond x towards itself. When both holes lie on streets, non-adjacency forces them onto a common street, so the pull takes its boxes from that street. Because every non-train street column ends at its blocking separator on the central row, no pull crosses into the opposite half, and every hole stays on a street. By backward induction from B_τ, no box off the streets ever moves (Figure 11).

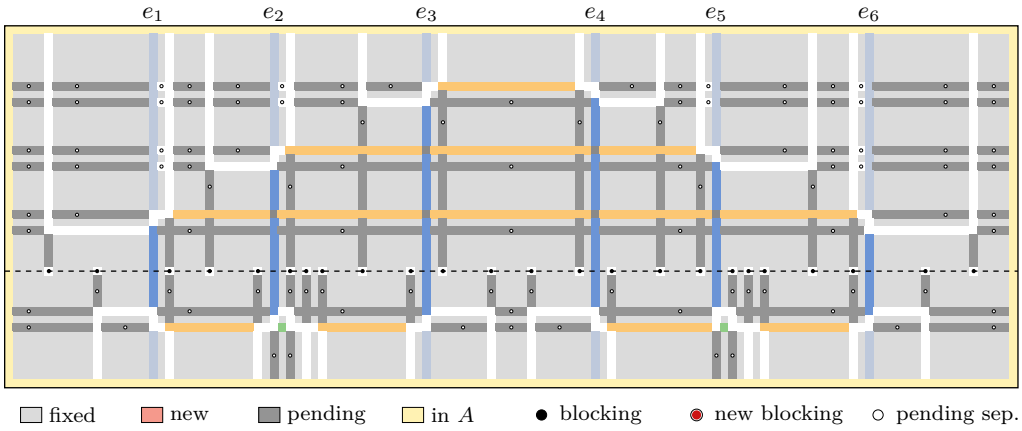
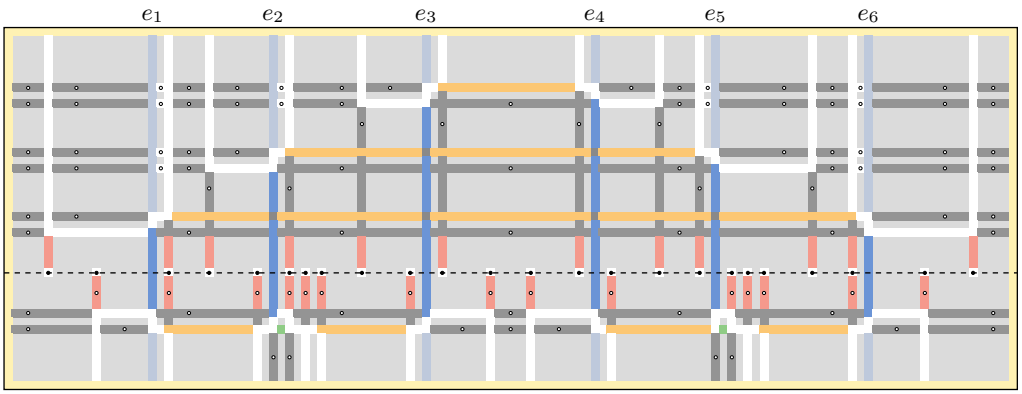
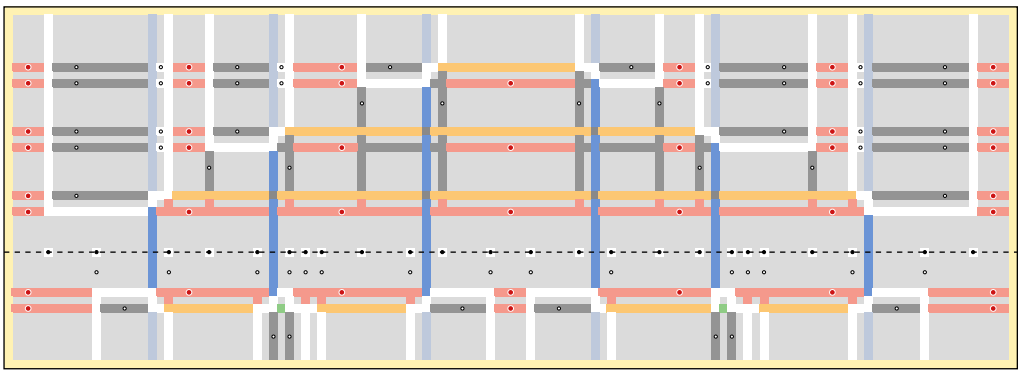


Figure 11. The board of Figure 9(b) after the street argument, with all streets and separators; the legend below applies to all whole-board freezing figures in this section.

Long segments. Between the central row and the deepest street row of its half, a street column crosses no street rows, so its side neighbours on that segment are off the streets and never move. Each separator on the central row therefore fixes the boxes of its column from the central row to the first hole or street row towards the frame: for a column of a level $j < m$ that is the deepest level of its half, this is its inter-level separator introduced below, and otherwise the deepest street row of the half (Figure 12(a)). Street rows need one more step. Between consecutive street columns of the same half that are more than L_m apart (between the streets of consecutive trains, or between a boundary column and the nearest train), each street row crosses a gap of length more than $D/2$ with no crossing street columns, which gets a separator at its middle unless a bar spans it. A street row near the frame is close to the frame, so Lemma 8 does not make these separators blocking directly. But the side neighbours of the gap are off the streets and never move, so the player can enter the gap only from the two cells beyond its ends, and inside it moves only along it. The proof of Lemma 8 then works with distance along the gap: after entering from one of these cells, the player stays within distance h of it. Since $D/4 > b$, the separators are blocking, and the boxes in the gaps never move (Figure 12(b)).



(a) Central-row separators fix the deep column segments



(b) Long street-row separators fix the wide gaps

Figure 12. Freezing the long segments (same palette as Figure 11). (a) The separators on the central row fix the street-column segments up to the first hole or street row towards the frame. (b) The separators at the midpoints of the wide horizontal gaps become blocking and fix the long street rows.

Corridors and scales. What remains are the streets near the trains, between a train and its corridors. There the corridors are the problem: they lead the player deep into the board, and their holes make b large, too large to show that the separators near a train are blocking. We therefore show that the corridors stay empty and add them to A, one level at a time, from the central row towards the frame. Each time, h drops enough for the separators at the next, shorter distance to become blocking, so the distances have to shrink by a large factor from level to level.

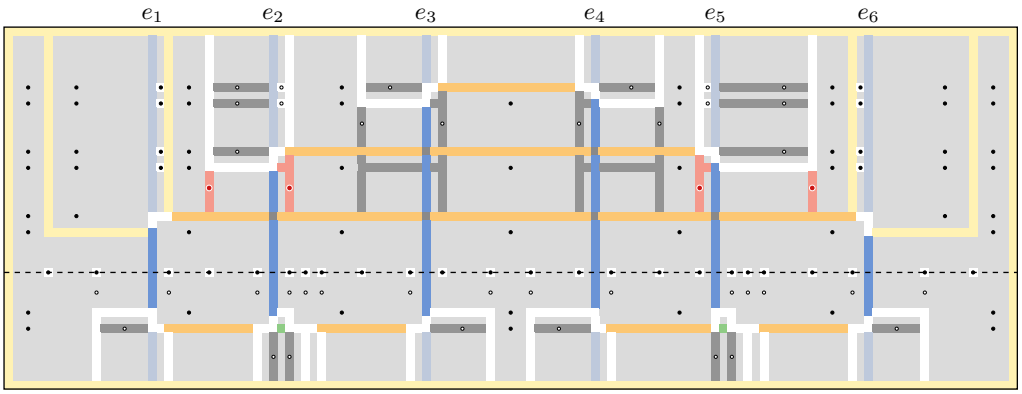
A vertex u lies *inside* a vertex v in the same half if all edges of u lie between the outermost edges of v. The *level* of v is one more than the largest level of a vertex inside it, or 1 if there is none, and the *nesting depth* m is the largest level; Appendix A bounds it by a constant m_0. Let $Y_1 < L_1 < Y_2 < \cdots < Y_m < L_m < D$ be scales in which consecutive ones differ by a factor $S = c(n+1)$ for a large enough constant c. The *depth* of a cell is its distance from the boundary row of its half. A vertex of level j has its working row at depth Y_j. The corridor of each of its trains runs along the column at distance L_j from the train, on the side away from the bar, to just beyond the working row, and then across to the train; the corridor of its bar runs along the column two cells from the train, on the side of the bar, to just before the working row (Figure 9(b)). Consecutive trains are D apart, so the board has height $2D$ and width $\Theta(nD)$, which is polynomial. Then $b = O(nL_m)$, so $D > 4b$, and we choose c so that in each stage below, h is less than half the next scale. Coordinates are in Appendix B.1.

Stage of level j, for $j = m, \ldots, 1$. It starts with $h = O(nL_j)$; for $j = m$, A is the frame and $h = b$. It has three steps; Figures 13–15 show them on the board of Figure 9, whose nesting depth is $m = 3$.

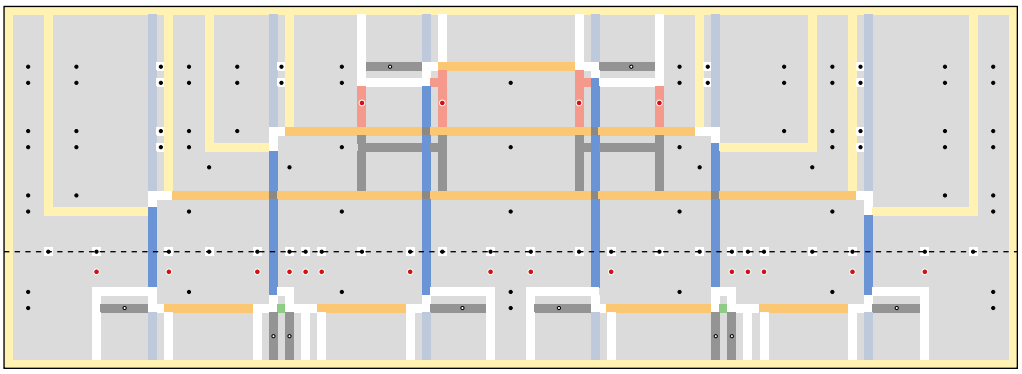
Closing the corridors. Between depths $Y_j + 2$ and Y_{j+1} (or between $Y_m + 2$ and the central row for $j = m$), each level-j corridor column crosses no street rows and gets a separator halfway between Y_j and Y_{j+1} (for $j = m$, the central-row separator); at level 1, so do the private column and the *public columns* of each AND or OR vertex, which carry the corridors from the bottom of the frame to its working row. These separators are blocking by the along-gap argument above, so the column segment that closes each level-j corridor on the side of the central row never moves. On each level-j bar-corridor column, the crossing box at depth $Y_j + 2$ is now backed both vertically by this fixed segment and horizontally by the wide gap fixed earlier, so it cannot move either, and neither can the box between it and the train or the box between it and the working row. For $j = m = 3$ this already holds from the long-segments step; Figure 13 shows levels 2 and 1.

Train corridors. The train corridors of level j now stay empty (Figure 14(a)). Consider first the horizontal leg of such a corridor. Its top and bottom neighbours are off the streets, so a box could enter it only from one of its two ends. From the corner end, pushing a box in would require standing on or pushing the background on the far side of the vertical leg, which was already fixed in the long-segments step. From the train end, the player would have to stand on or push the cell just beyond the train; that cell has off-street vertical neighbours and is backed on its far side by the box on the bar-corridor column just fixed in the closing step, so it never moves either. Now consider the vertical leg: a box can enter it from a crossing street row, but it can never leave. Pushing towards the frame only moves boxes along the leg, which can never exit onto the empty frame; pushing towards the central row hits the closed background; and pushing sideways requires standing on or pushing the fixed background on the far side. Since B_τ leaves these corridors empty, no box ever enters them. Adding all level-j train corridors to A reduces the remaining hole count to $h = O(nY_j)$.

Next to the trains. On each street row up to the working row, the background segment between a level-j train and its train corridor has length $L_j - 1$ and carries a midpoint



(a) Level 2: closing the corridors



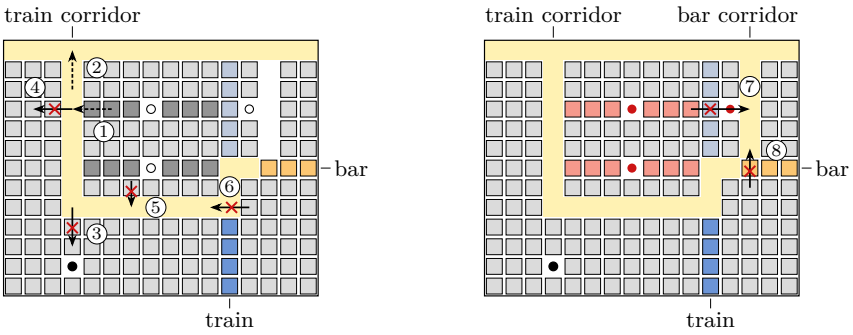
(b) Level 1: closing the corridors (including the bottom half)

Figure 13. Closing the corridors of levels 2 and 1 on the side of the central row (level 3 is already closed by the central-row separators in Figure 12(a)).

separator. With $h = O(nY_j)$, this separator is now blocking and fixes the entire segment (Figure 14(b)). On each strictly shallower street row, we also leave the single cell between the train and its bar corridor as a hole (Appendix B.1). No box can then enter a level-j bar corridor:

- along a shallower street row, that intervening cell is empty, so pushing a box in from the train side requires standing on or pushing the background just fixed, and pushing from the far side requires standing on or pushing the wide-gap background fixed earlier;
- from the working row, pushing a box of the bar into its corridor would require standing on the cell of the corridor column just beyond the working row, whose box was fixed in the closing step.

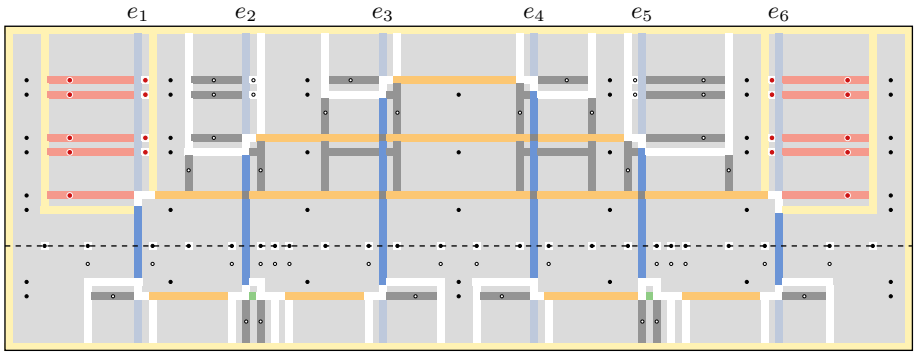
Hence every level-j bar corridor stays empty as well. For $j > 1$, adding them to A drops the hole count to $h = O(nL_{j-1})$. Figure 15(a)–(c) shows this step for levels 3, 2, and 1; the corridors of each level turn yellow as they join A.



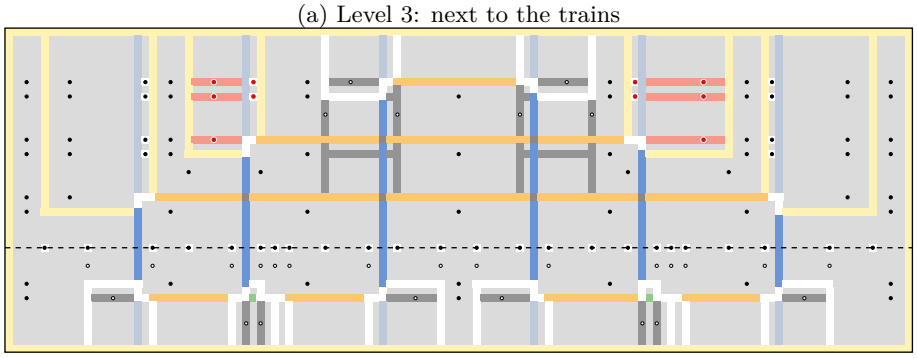
(a) Train corridor joins A (b) Bar corridor joins A

Figure 14. Local freezing at the top end of a level-j train, not to scale (same palette as Figure 11). (a) Train corridor: a box may enter the vertical part from a crossing street row (1), but upward pushes only move boxes along the leg and can never exit onto the empty frame (2), while pushing down (3) or sideways (4) is blocked; no box enters the horizontal part from above (5) or from the train (6). (b) Next to the train: once the train corridor joins A, the midpoint separators on the shallow street rows become blocking and fix the background beside the train (pink), blocking pushes into the bar corridor from the train side (7) or from below the bar (8).

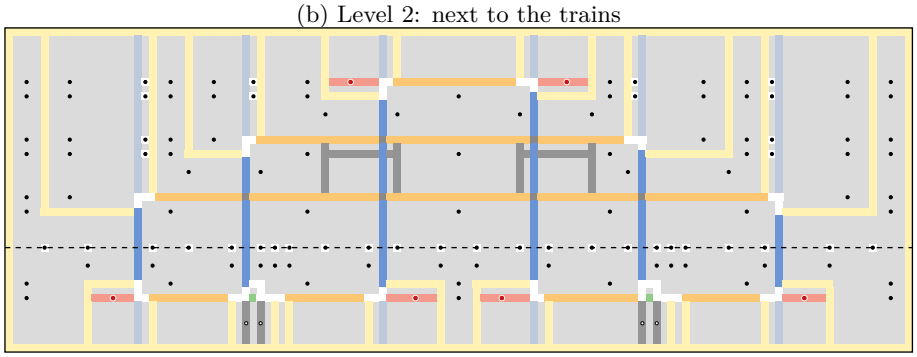
Cores. At level 1, the public columns of the AND and OR vertices remain outside A, with $O(nY_1)$ holes, far more than $O(n)$. No budget is needed for them: around each core, the boxes just beyond the working row on the public columns and on the street row two cells beyond it form segments that each end next to a box already fixed, so none of them can be the first to move (Appendix B.2). Hence no box enters a public column, and the public columns join A at the end of level 1, which leaves $O(n)$ holes outside A. Below each AND or OR vertex, the middle train's column and the private column each hold a *half-depth separator*, at depth $Y_1/2$; by the along-gap argument these are now blocking and fix both columns between the frame and the working row. The same first-move arguments keep the private branch empty, and every deep crossing on an inner train or bar and every cell strictly between a shallower column's inter-level separator and the central row lies farther than h from A and stays occupied throughout π (so bars and trains may shift across deep crossings without emptying them, and the remaining deep background segments are pinned between their blocking separators so their boxes never move; Figure 15(d) and Appendix B.2). Hence no background box ever moves and no hole off the tracks is ever filled, which proves Lemma 6.



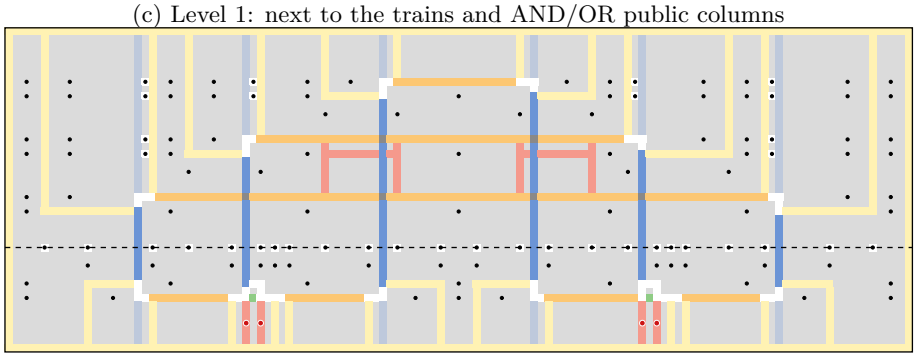
(a) Level 3: next to the trains



(b) Level 2: next to the trains



(c) Level 1: next to the trains and AND/OR public columns



(d) Cores: half-depth separators fix the remaining columns

Figure 15. Completing the freeze from level 3 inwards to the cores. (a)–(c) Next to the trains across levels 3, 2, and 1: each level's separators fix the adjacent street rows (pink) and shield its bar corridors, which join A in turn; at the end of level 1, the AND/OR public columns join A as well. (d) The cores: the half-depth separators below each AND and OR vertex become blocking and fix their columns, and the remaining deep background segments are pinned; every background box is now fixed.

5 Push-$*$

In Push-$*$, the goal asks only for the player to reach a given cell; it says nothing about the final positions of the boxes. The reduction for Push-$*$S in Section 4 relied on the exact storage target B_τ in three places:

1. *Background:* the backward street induction started from B_τ, and B_τ guaranteed that every separator was still a hole at the end.
2. *Tracks:* matching the initial and final configurations kept every train and bar at its original length and prevented runs of boxes from merging.
3. *Final state:* B_τ ruled out cut-off pockets and cores at the end of the trajectory.

Without an exact target, a player could push background boxes out of place, move boxes from one track to another, or cut off a core and walk straight to the goal.

We replace the exact target by a condition that a player goal can enforce: *the hole set of a single configuration passes a fixed list of row and column counts.* The proof splits into two independent layers:

- *A black-box reader compiler (Section 5.1, independent of NCL):* we isolate the exact input–output contract of wrapping one read and rotating the board in a single composition interface (Lemma 11), and deduce both the Push-$*$ reader (Lemma 12) and the PushPush-$*$ reader (Lemma 13). Once Section 5.1 is complete, pinholes, corridors, and rotations disappear from the rest of the proof.
- *A counting certificate on the NCL board (Sections 5.2–5.3):* we modify the board of Section 4, define a program $\mathcal{P}_\tau$ from the canonical configuration B_τ, and prove the *certificate theorem* (Theorem 14)—any trajectory ending with a hole set accepted by $\mathcal{P}_\tau$ yields an NCL path from σ to τ—in three phases: freezing the background (Steps 1–3), sealing the tracks (Step 4), and decoding the gadgets (Steps 5–6).

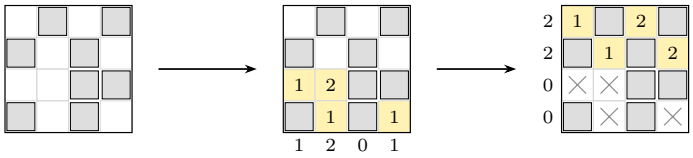
Routine inductions, closed-form scale arithmetic, and case analyses are collected in Appendix C.

5.1 Reading programs and readers

Reading programs. Let R be a wall-free board and let $E \subseteq R$ be a set of holes. A *line* of R is a row or a column.

- A *read* chooses one side of R (bottom, left, top, or right) and a quota $q_L \geq 0$ for each line L perpendicular to that side. On each such line L, it deletes the first q_L holes of E counted from the chosen side; if L contains fewer than q_L holes, the read *fails*.
- A *clear* chooses a set of lines and deletes all remaining holes of E on them. A clear never fails.
- A *program* is a fixed sequence of reads and clears applied in order to the remaining holes; it *accepts* E if no read fails.
- The program is *full* for E if the sum of all its quotas equals $|E|$.

Every push replaces one hole (the first hole beyond the pushed boxes) by one new hole (the player's new cell), so the total number of holes never changes during play, and fullness depends only on the initial state. When a full program accepts E, its positive quotas already delete $|E|$ distinct holes, so *every clear must delete zero holes* and no hole remains at the end (Figure 16). Thus a clear in a full program certifies that earlier reads have already emptied the cleared lines. The order of operations therefore imposes both lower bounds (reads succeed) and upper bounds (clears delete nothing) at every intermediate stage, not only at the end.



(a) hole set E ($|E| = 8$) (b) read from bottom (c) read from left

Figure 16. A full reading program on a 4×4 hole set E. (a) Initial hole set E ($|E| = 8$). (b) Read from the bottom with column quotas $1, 2, 0, 1$; numbered yellow holes are deleted in order from the bottom. (c) Read from the left with row quotas $2, 2, 0, 0$; crossed cells were deleted in (b). All 8 holes are deleted by positive quotas, so any subsequent clear deletes nothing.

A physical reader will allow a player to push further than required, deleting extra holes from R. This never helps a player pass a program that E itself would fail:

▶ **Lemma 9** (inclusion monotonicity). *If $F \subseteq E$ and a program $\mathcal{P}$ accepts F, then $\mathcal{P}$ accepts E.*

Indeed, for one read of quota q on a line L, if F has at least q holes on L then so does E, and every hole of F surviving the read has at least q holes of F (hence of E) ahead of it, so it survives on E as well; clears preserve inclusion directly, and induction over the steps of $\mathcal{P}$ gives the lemma.

A reader for one read. Suppose first that $\mathcal{P}$ consists of a single read from the bottom with column quotas $q_x \geq 0$ on a $w_0 \times h_0$ wall-free board R with n_0 initial holes (if $Q = \sum_x q_x > n_0$, output a trivial unsolvable instance; assume $Q \leq n_0$). We want a Push-$*$ instance that is solvable if and only if R has a reachable *corner state* (c, B)—with the player at the lower right corner c of R—whose hole set $E = R \setminus B$ has at least q_x holes in every column x.

The main obstacle is enforcing a *single snapshot*. If the player could read some columns of R, return to R to rearrange boxes, and then read the remaining columns, the quotas would be met by different configurations at different times. We prevent this in two ways: the player can leave R only through a single *exit cell* C and must fill C before reaching any reading entry, and a solid band of boxes taller than n_0 separates R from the reading area so that pushing north after C is filled can never climb back into R.

We extend R (columns $-w_0, \ldots, -1$, bottom row $y = 0$, corner $c = (-1, 0)$) by two columns on the east (columns 0 and 1) and by a number L of rows on the south (we write L for this added height throughout the reader construction, distinct from a generic line above), keeping R unchanged and filling every added cell with a box except for the following holes (Figure 17; exact coordinates in Appendix C.1):

- In column 0, we leave only three holes: the *exit cell* C directly east of c, the cell just below C, and the bottom cell of column 0. Thus C is the only hole outside R adjacent to R.
- In column 1, an open vertical corridor connects the row below C down to the bottom row.
- Below a solid band of boxes in rows $1, \ldots, y^* - 1$, where $y^* = L - 2 - Q$ and the band height $y^* - 1$ exceeds $\max(n_0, w_0)$, each column x of R receives a single *pinhole* at row s_x. The pinholes form a staircase rising from east to west: the eastmost pinhole lies two rows above the bottom, each step west rises by q_x rows, and the player goal sits q_{-w_0} rows north of the westmost pinhole (at row y^*).

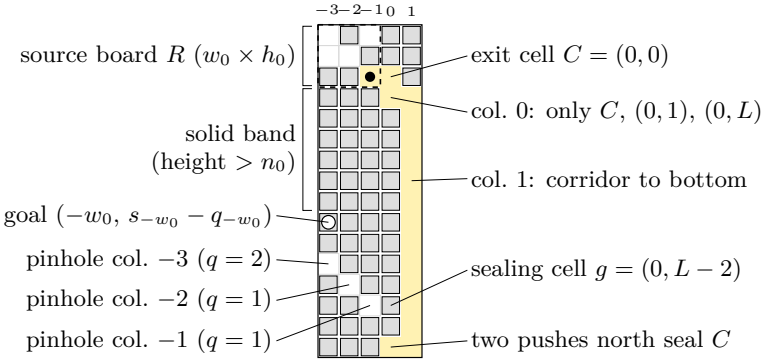


Figure 17. The one-read reader on a 3×3 source board R (dashed, $w_0 = h_0 = 3$, $n_0 = 5$) with bottom quotas $2, 1, 1$ from west to east; holes reachable from the player (dot) at c are yellow, and the circle is the goal.

Canonical route. Starting from c with a hole set E that passes the quotas, the player steps through C and the cell below it into the east corridor, walks down to the bottom row, and steps west into column 0. Two pushes north along column 0 fill the two upper holes (ending with C) and place the player at the *sealing cell* $g = (0, L - 2)$, directly east of the eastmost pinhole. Next, visiting the columns of R from east to west, the player steps west into each pinhole and pushes north q_x times. Each push shifts the solid column of boxes north by one cell, fills the lowest remaining hole of column x in R, and leaves a hole under the player. After q_x pushes, the player stands directly east of the next pinhole; after the westmost column, the player stands at the goal (Figure 18).

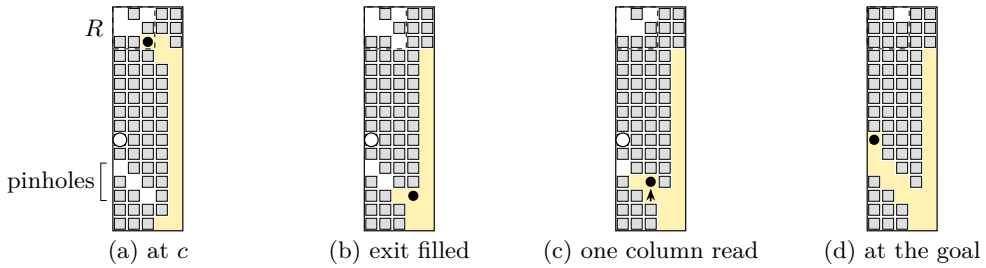
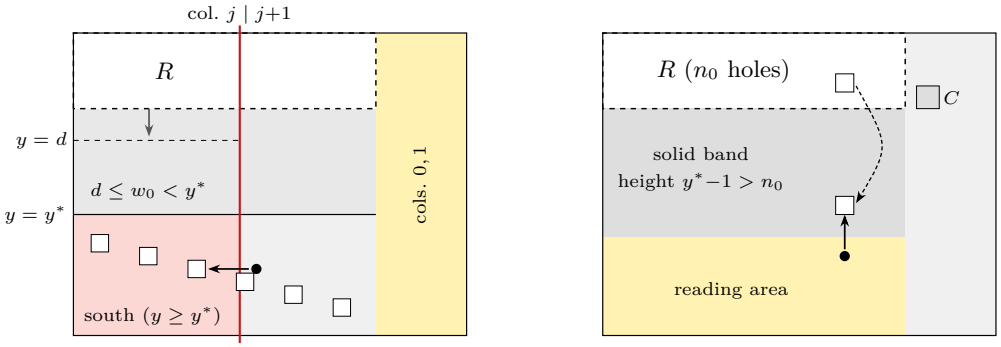


Figure 18. Canonical route of the one-read reader of Figure 17 (a copy of Figure 3). (a) At c. (b) After filling C and reaching g, the holes of R are out of reach (white). (c) After reading the rightmost column (quota 1). (d) At the goal, with $1, 1, 2$ lowest holes deleted from columns $-1, -2, -3$.

▶ **Lemma 10** (one read). *The one-read reader has size $(w_0 + 2) \times (h_0 + L)$ and is solvable if and only if R has a corner state (c, B) whose hole set $E = R \setminus B$ has at least q_x holes in every column x. Moreover, along every trajectory to the goal, there is a corner state (c, B) of R such that at the goal the remaining holes in R are obtained from $E = R \setminus B$ by deleting the lowest $t_x \geq q_x$ holes in each column x.*

Proof. The canonical route gives the "if" direction. Conversely, fix any trajectory to the goal, and recall that every pinhole and the goal lie in $y \geq y^*$, where $y^* = L - 2 - Q > \max(w_0, n_0)$. Three counting arguments show that all quotas are paid from a single snapshot of R:

(a) Strip injection. For each $j \in \{-1, \ldots, -w_0\}$, consider the strip $\{x \le j\}$ and its *south region* $\{x \le j,\ y \ge y^*\}$ (Figure 19(a)). Before the player enters this south region, let d be the deepest non-pinhole row in $\{x \le j\}$ (initially $d = 0$). Any new hole below R in $\{x \le j\}$ must be created by pushing a column south into an original pinhole of $\{x \le j\}$; each push increases d by at most 1 and consumes one of the at most w_0 pinholes in $\{x \le j\}$, so all non-pinhole holes stay in $y \le d \le w_0 < y^*$. Thus the player cannot reach the south region from the north; first entry must come from the east, using an original pinhole in some column $i(j) \le j$. Because the player crosses one column per move, strips are first entered from east to west; a push destroys pinhole $i(j)$ while a walk enters $i(j) = j$ (outside all smaller strips), so i is an injection on $\{-w_0, \ldots, -1\}$ with $i(j) \le j$. Hence $i(j) = j$ for every j: every strip is first entered by walking from the east into its own pinhole (j, s_j), untouched until that visit.



(a) strip injection (b) barrier after sealing C

Figure 19. Two counting arguments in Lemma 10. (a) Strip injection: southward pushes cannot pass $y = d \le w_0 < y^*$, so first entry into the south of strip $x \le j$ comes from the east and uses an unused pinhole. (b) Barrier after sealing C: each new row entered north of $y = y^*$ consumes one source hole from R, and the solid band has height $y^* - 1 > n_0$.

(b) Exit C is filled before reaching g. Until the first visit to $g = (0, L-2)$, all w_0 pinholes are untouched by (a), the northeast quadrants above and right of C remain full, and the player leaves R only from c through C, which can never reopen once filled. Moreover, no horizontal push can create a hole in column 0, so column 0 undergoes only vertical pushes along itself and horizontal deletions. Measuring height $z = L - y$ upward from the bottom of column 0, colour the bottom hole ($z = 0$) red and the top two holes ($z = L-1, L$) blue (Figure 20), giving each newly created hole along column 0 the colour of the pushing player's hole. A simple rank induction (Appendix C.1) keeps all red holes in $z \le 2$ and all blue holes in $z \ge L - 2$, separated by boxes since $L \ge 7$, with every cell gained by red costing one blue hole. Reaching g ($z = 2$) therefore forces all blue holes—including C—to be gone. Deleting all earlier excursions outside R (each returning through the still-empty corner c) leaves a trajectory inside R ending at c with the hole set E present at the last departure from R (Appendix C.1).

(c) No return to R after sealing C. When the player reaches g, the holes in R are E and the band $\{x < 0,\ 1 \le y < y^*\}$ is solid (Figure 19(b)). Entering any row $y \in \{1, \ldots, y^* - 1\}$ in $\{x < 0\}$ for the first time requires a northward push from row $y + 1$, which moves one hole of R down to row y; that hole can never pay for a higher unvisited row. Since R has only $n_0 < y^* - 1$ holes, the player never climbs back to R, and R changes only via northward pushes deleting the lowest remaining hole of their column. Finally, between first stepping onto pinhole (j, s_j) and first entering $(j-1, s_{j-1})$ (or the goal), no horizontal push can create

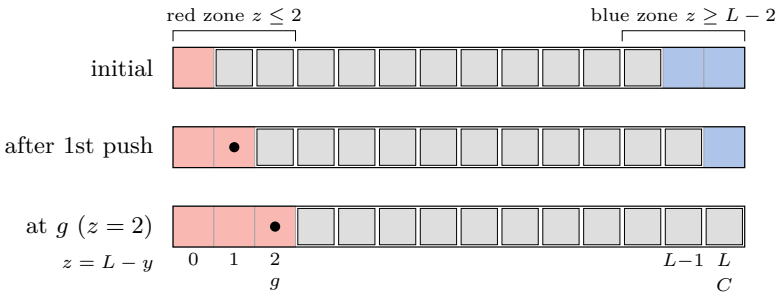


Figure 20. The colouring invariant on column 0 ($z = L - y$). Red holes stay in $z \le 2$ and blue holes in $z \ge L - 2$; reaching g ($z = 2$) forces all blue holes, including C, to vanish.

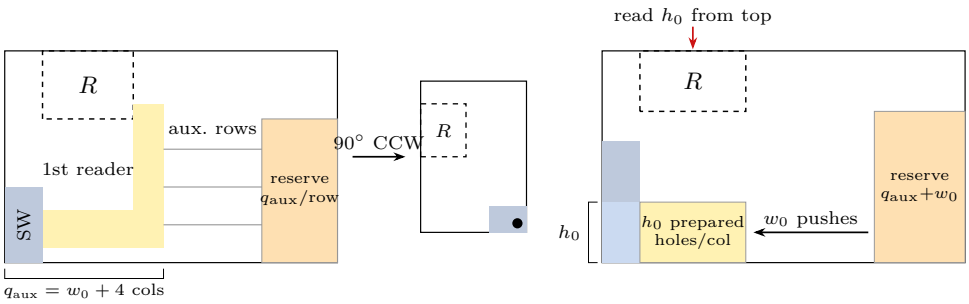
a hole in column j (by (a), no hole to the west can be used without consuming an unused pinhole), so climbing q_j rows from s_j to $s_j - q_j$ consumes at least q_j holes from column j of E. ◀

Composing reads and compiling clears. Our certificate program $\mathcal{P}_\tau$ alternates reads from all four sides and inserts clears between them. Composing a second read after the first raises two problems: the one-read gadget expects the player at the lower right corner of a rectangle, and the first reader leaves *auxiliary rows* south of R whose leftover holes sit in the same columns as R and could falsely pay for later column reads.

We solve both by wrapping the one-read board R_1 inside a larger rectangle R^+ (Figure 21(a); exact dimensions and coordinates in Appendix C.1). Specifically, we can arrange coordinates so that R^+ extends R_1 in three directions with all added cells holding boxes except for:

- two added columns on the west, containing only a single *southwest entry* directly west of the old goal, a vertical *push-down shaft* south of R_1 of height one more than the box stack above it, and an open *west corridor* down to the new southwest corner; and
- a far-east reserve placed strictly beyond the total hole count of R^+ (so the player can never stand on it by Lemma 8 with $A = \emptyset$), holding $q_{\text{aux}} = w_0 + 4$ pairwise non-adjacent *reserve holes* on every auxiliary row below R.

Because the southwest entry is the only hole west of R_1 down to its bottom row and the strip directly south of R_1 is solid, the player can first enter the two west columns or the rows south of R_1 only by stepping west from the old goal. (Earlier, pushes east from column 1 may spend reserve holes on the rows of R_1, but the player never stands on the reserve.) In the canonical route, the player steps into the southwest entry, pushes the box stack south into the shaft (one maximal slide and walk in PushPush-$*$), and walks down the west corridor to the new southwest corner. Rotating R^+ by 90° counter-clockwise produces a rotated extended board $\widetilde{R}$ whose new lower right corner c^+ is this southwest corner; the rotation turns the left side of R into the new bottom and turns every auxiliary row into an auxiliary column. The next read from the left therefore becomes a bottom read on $\widetilde{R}$, taking R's row quotas on R's lines and quota $q_{\text{aux}} = w_0 + 4$ (the combined width of R_1 and the two west columns) on every auxiliary row. The canonical route pays q_{aux} exactly using its at most q_{aux} leftover holes topped up by isolated reserve holes (with zero overshoot even under maximal slides in PushPush-$*$). On any passing trajectory, reading q_{aux} from the left sweeps all leftover holes out of R's columns on every auxiliary row, so *every surviving auxiliary hole lies strictly east of R's columns and south of R's rows* and never interferes with later reads on R.



(a) rotate and clear auxiliary rows

(b) preparing h_0 rows for a clear

Figure 21. Composing reads and compiling a clear. (a) Adding the two-column southwest extension and isolated far-east reserve holes ($q_{\text{aux}} = w_0 + 4$ per auxiliary row), then rotating 90° counter-clockwise; reading q_{aux} from the left on each auxiliary row sweeps all leftover holes east of R's columns. (b) To compile a clear of chosen columns, extend south by h_0 rows with $q_{\text{aux}} + w_0$ reserve holes per row, push east w_0 times on each extra row to leave w_0 holes under R's columns, and read h_0 from the top on each chosen column.

A *clear* of a set I of columns cannot simply be replaced by a top-to-bottom read of quota h_0: on a valid configuration, a column in I may have fewer than h_0 holes left (in a full program, it has zero), so a read of quota h_0 would reject valid inputs. Instead, we extend R^+ south by h_0 extra rows, each with an entry from the west corridor and $q_{\text{aux}} + w_0$ isolated far-east reserve holes (Figure 21(b); coordinates in Appendix C.1). Along the west corridor, the canonical route visits each extra row and pushes east w_0 times, leaving one prepared hole directly under every column of R while keeping q_{aux} reserve holes untouched. On the rotated board $\widetilde{R}$, five cyclic reads then compile the clear: (1) the left read on R's rows (quota 0 on auxiliary rows), (2) a top read of quota h_0 on every column in I (deleting every hole left inside the h_0-row board R on those columns, with the h_0 prepared holes below R guaranteeing payment even on an empty column), (3)–(4) two zero-quota turns back to the left side, and (5) a left read of quota q_{aux} sweeping all auxiliary rows, which leaves the orientation aligned with the next top read.

Run reads and the composition interface. Section 6 uses the same wrapper $\widetilde{R}$ under maximal slides. In PushPush-$*$, a northward slide from a pinhole fills an entire maximal run of consecutive holes in its column. Define a *run read* on a line L with quota q_L to delete the shortest prefix of complete maximal runs of holes on L, counted from the chosen side, that contains at least q_L holes (failing if L has fewer). In a full program of run reads, each run read deletes exactly q_L holes, so every cut must fall between two runs.

The next lemma collects every property that a single wrapped-and-rotated layer $\widetilde{R}$ exports to the recursive reader constructions for both Push-$*$ and PushPush-$*$.

▶ **Lemma 11** (reader composition interface). *Let R be a $w_0 \times h_0$ wall-free board of area A, corner c, and n_0 initial holes, with bottom column quotas $q_x \geq 0$ ($\sum_x q_x \leq n_0$). Let $\widetilde{R}$ be its 90° counter-clockwise rotated extended board (in plain mode or clear-preparation mode), with new lower right corner c^+ and auxiliary-sweep quota $q_{\text{aux}} = w_0 + 4$. Then $\widetilde{R}$ has area $O(A^3)$ and satisfies:*

1. Source extraction and prefix deletion: *Along any Push-$*$ trajectory on $\widetilde{R}$ reaching a corner state (c^+, B^+) with hole set $E^+ = \widetilde{R} \setminus B^+$, there is a Push-$*$ trajectory inside R reaching a corner state (c, B) with hole set $E = R \setminus B$ such that:* (i) *on every column x of R,*

$E^+ \cap R$ *is obtained from* E *by deleting a bottom prefix of* $t_x \geq q_x$ *holes of* E*; and* (ii) *on every row of* R*,* E^+ *has no hole outside* R*. Under PushPush-*, the same holds with a PushPush-* trajectory inside* R *and each deleted column prefix consisting of complete maximal runs of total length* $t_x \geq q_x$*.*

2. Auxiliary isolation after sweep: *On* E^+*, any read from the old left taking at least* q_{aux} *holes on every auxiliary row (merged into the next read in plain mode, or as step* (5) *in clear-preparation mode) empties* $[-w_0 - 2, 1]$ *on every auxiliary row. Every surviving auxiliary hole then lies strictly south of* R*'s rows and strictly east of* R*'s columns, hence off every reading line of* R *in all subsequent reads and clears. (In clear-preparation mode, step* (1) *touches only* R*'s rows, which by* (1) *have no holes outside* R*, and step* (2)*'s top read of* h_0 *on columns* I *empties* R *on* I *before step* (5) *sweeps the auxiliary rows.)*
3. Canonical forward pass: *Conversely, from any reachable corner state* (c, B) *of* R *whose hole set* E *has at least* q_x *holes in every column* x*, the canonical Push-* route on* $\widetilde{R}$ *reaches a corner state* (c^+, B^+) *where* $E^+ \cap R$ *deletes the exact bottom* q_x *holes per column, every auxiliary row holds at most* q_{aux} *holes in* $[-w_0 - 2, 1]$ *plus* q_{aux} *isolated far-east reserve holes (paying* q_{aux} *exactly), and in clear-preparation mode each of the* h_0 *extra rows additionally holds one prepared hole under every column of* R*. Under PushPush-* (plain mode), if* E *passes* $\{q_x\}$ *with exact run cuts, the canonical slide-and-walk route reaches the same corner state with exact cuts on both* R *and the subsequent auxiliary sweep.*
4. From extra deletions to exact acceptance:
   - Ordinary Push-$*$ reads and clears: *writing* $\text{read}_{\text{bot},q}(E)$ *for the holes of* E *left by the bottom read with quotas* q_x*,* $t_x \geq q_x$ *implies* $E^+ \cap R \subseteq \text{read}_{\text{bot},q}(E)$*, so parts* (1) *and* (2) *together with inclusion monotonicity (Lemma 9) ensure that if the reduced program on* $\widetilde{R}$ *accepts* E^+*, then the original program accepts* E*.*
   - Full PushPush-$*$ run-read programs: *unwrapping the recursion across all layers via* (1) *and* (2) *(with the slide expansion described after this lemma applied to the one-read base case of Lemma 10) extracts a PushPush-* corner state* E *of the original board* R *on which every step* s *and line* L *of* R *deletes a prefix of complete maximal runs of length* $t_L^{(s)} \geq q_L^{(s)}$ *from pairwise disjoint subsets of* E*. When the program is full* ($\sum_{s,L} q_L^{(s)} = |E|$)*, we have* $|E| = \sum_{s,L} q_L^{(s)} \leq \sum_{s,L} t_L^{(s)} \leq |E|$*, forcing* $t_L^{(s)} = q_L^{(s)}$ *term by term and hence exact run-read acceptance of* E*.*

Because the far-east reserve lies beyond the total hole count (unreachable by Lemma 8) and east of every source strip, and the southwest entry is the sole hole west of R_1 at height $y \leq L$, the strip, rank, and solid-band counts of Lemma 10 extend to arrival at c^+, while the full northwest and northeast quadrants and sealed exit C keep R's rows box-filled outside R. Under PushPush-$*$, expanding each slide of displacement d into d unit pushes followed by $d - 1$ steps back preserves these exact bounds while forcing each post-sealing slide into R to delete a complete maximal run; part (4) then follows directly from parts (1) and (2) via Lemma 9 and fullness. Appendix C.1 verifies parts (1)–(4) and the $O(A^3)$ area bound in full.

▶ **Lemma 12** (Push-$*$ reader)**.** *For every fixed number of steps, given a wall-free board* R *with an initial state and a program* $\mathcal{P}$*, one can construct in polynomial time a Push-* instance without walls and with a player goal, which contains* R *and its initial state, and which is solvable if and only if* R *has a corner state accepted by* $\mathcal{P}$*.*

We first normalize $\mathcal{P}$ into cyclic four-side order: merge consecutive clears of the same orientation, then pad the start and gaps with at most three zero-quota reads so that reads cycle bottom–left–top–right, every column clear follows a left read, every row clear follows a

top read, and no clear appears in the first two steps (multiplying the length by at most 4). We then recurse on (r, e)—the remaining reads and clears—down to the base cases $e = 0, r \leq 1$ (trivial for $r = 0$; Lemma 10 for $r = 1$; details in Appendix C.1). Each plain step wraps R into $\widetilde{R}$ and maps $(r, e) \mapsto (r - 1, e)$; each clear step wraps R in clear-preparation mode and replaces two reads and one clear by five cyclic reads, mapping $(r, e) \mapsto (r + 3, e - 1)$. Thus $r + 4e$ decreases strictly at every step, and $O(A^3)$ area per step gives polynomial size for fixed program length; completeness follows by induction from Lemma 11(3) and soundness from Lemma 11(1),(2),(4).

▶ **Lemma 13** (PushPush-$*$ reader)**.** *For every fixed number of steps, given a wall-free board R with an initial state and a program $\mathcal{P}$ of run reads that is full for the initial state of R, one can construct in polynomial time a PushPush-$*$ instance without walls and with a player goal, which contains R and its initial state, and which is solvable if and only if R has a corner state accepted by $\mathcal{P}$ under PushPush-$*$.*

Normalize $\mathcal{P}$ into cyclic four-side order (with $e = 0$, no clears) and apply the same plain-step recursion down to the one-read base case $r = 1$ (polynomial size by Lemma 11). For completeness, fullness makes every cut of $\mathcal{P}$ on R exact, and induction via Lemma 11(3) lifts exact-cut acceptance across every wrapped layer and auxiliary sweep down to $r = 1$ (Appendix C.1). For soundness, Lemma 11(4) immediately extracts a PushPush-$*$ corner state E of R accepted by $\mathcal{P}$ with exact term-by-term payments.

5.2 The modified board and the certificate theorem

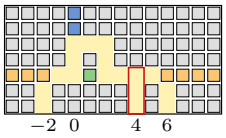
We now adapt the wall-free board of Section 4.3 so that a reading program $\mathcal{P}_\tau$ of $O(m)$ steps replaces the exact target B_τ. Three modifications are needed:

1. *Two-box short bar* (introduced right below): ensures that every short-bar shift covers an even column so sparse even-column reads decode on which side of the core it lies, and makes column +4 beside each AND/OR core private for verifying the private branch.
2. *Central marker holes* (introduced before Table 2 and analysed in Step 1 of Section 5.3): act as fences near the central row so that column reads from one side can certify interior separators without borrowing holes from the opposite half.
3. *Partner repair and parity levels* (introduced in Step 4 of Section 5.3): separate middle trains from ordinary trains that can absorb boxes onto different probe rows, so that box exchanges cannot cancel in a row sum.

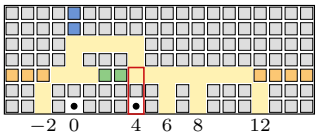
First modification: the two-box short bar. Recall from Sections 4.2 and 4.3 that an AND or OR vertex lies in the bottom half at level 1 on three consecutive trains e_i, e_{i+1}, e_{i+2} (using local x-offsets relative to the middle train e_{i+1} at offset 0), with its working row at $y_1 = 2D - Y_1$ (depth Y_1 from the bottom frame $y = 2D$). To keep the backward street support sparse (Step 2), all street and sampling columns must be pairwise non-adjacent, so our column reads inspect only *even* offsets around the core. In the one-box gadget of Section 4, however, a one-box short bar can sit on an odd column (such as offset 3) undetected by even-column reads, and column +4 is a public column rather than a private branch column (Figure 22(a)).

We therefore widen the core to a *two-box short bar* on offsets $[p + 1, p + 2]$, which always covers one even cell and one odd cell (Figure 22(b),(c); exact bar lengths and coordinates in Appendix C.2). Writing l, r for the outer-end hole counts at the left and right bars and v for the number of holes directly above y_1 on column 0 (region V), the two working-row gaps are

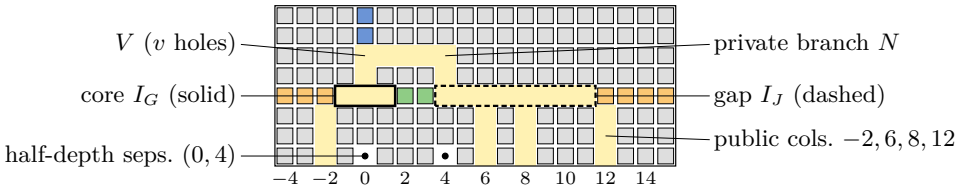
$$I_G = [\, l - \kappa + 1,\ p \,], \qquad I_J = [\, p + 3,\ 13 - r \,],$$



(a) Section 4 one-box core (col. 4 public)



(b) two-box core (col. 4 private)



(c) two-box OR core ($l = 2$, $p = 1$, $r = 2$, $v = 2$)

Figure 22. Why we switch to a two-box short bar, on an OR core (local coordinates relative to the middle train at column 0, bottom frame below; blue, orange, and green boxes are trains, bars, and short bars as in Section 4). (a) In the one-box gadget of Section 4, the single green box can hide on an odd column, and column 4 (red outline) is public. (b) In the two-box gadget, the short bar always covers an even column, and column 4 is private, leading only to the private branch and a half-depth separator (black dot). (c) Notation on the two-box OR core in state $(l, p, r, v) = (2, 1, 2, 2)$.

with $\kappa = 4$ for OR and $\kappa = 2$ for AND, and total working-row hole count $\kappa + 11$ independent of the state. Unlike Section 4 (where AND omitted column -2), here both OR and AND include all four *public columns* at even offsets $-2, 6, 8, 12$, running empty from the bottom frame up to just below y_1 (in AND, the cell $(-2, y_1)$ is always covered by the left bar); a five-cell *private branch* N joins $(4, y_1)$ to V whenever $v \geq 2$; and columns 0 and 4 hold *half-depth separators* halfway between y_1 and the bottom frame. Column 4 is now private: below y_1 it holds boxes except for its single half-depth separator.

Given incoming edge bits $\iota_l, \iota_m, \iota_r \in \{0, 1\}$, we define four internal bits

$$\gamma_H = [p \geq 6], \qquad \gamma_L = [l \leq 1], \qquad \gamma_M = [v \geq 2], \qquad \gamma_R = [r \leq 1],$$

recording whether the short bar, left bar, and middle train point into I_G, and whether the right bar makes room for the short bar (the *relay* bit). In the canonical configuration (exact formulas in Appendix C.2), $\gamma_H = \gamma_R = \iota_r$, $\gamma_L = \iota_l$, and $\gamma_M = \iota_m$; Figure 23 shows representative states of the two-box OR and AND gadgets. Wire endpoints keep the exact three-hole pocket and corridors of Section 4, and all canonical configurations share the same total hole count $K = |R \setminus B_\sigma| = |R \setminus B_\tau|$.

Scales and central markers. Starting from the planarized AND/OR instance (H, σ_H, τ_H) and its two-page wire-subdivided layout G_0 of Appendix A, we first apply the partner repair of Lemma 18 (defined in Step 4 of Section 5.3, where the problem it solves appears; it further subdivides wires while keeping nesting depth $m = O(1)$ and increasing the train count linearly) to obtain the repaired two-page graph G with n trains and maximum working level m. By Appendix A.3, applying the canonical cursor encoding of (σ_H, τ_H) on G yields ordered orientations σ, τ of G such that τ_H is reachable from σ_H in H if and only if τ is reachable from σ in G (and equivalently through ordered moves in G). Throughout the rest of this section, B_σ and B_τ denote the canonical configurations of these orientations σ, τ on the repaired graph G. Just as in Section 4.3, we can arrange geometric scales $Y_1 < L_1 < \cdots < Y_m < L_m < D$ growing by a factor $\Theta(n)$ so that the board R spans

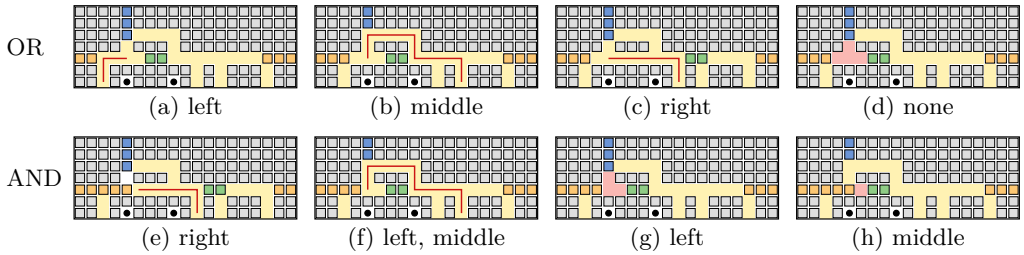


Figure 23. Representative states of the two-box OR (top) and AND (bottom), labelled by the edges pointing in; yellow holes connect to the public columns (red paths show a route to the core), and red holes are cut off. In (d), (g), and (h) the core is cut off.

$[0, (n+1)D] \times [0, 2D]$, central row $M = D$, and trains e_i at $x_i = iD$, with every street line, inter-level row, and marker row on an even coordinate and with every freezing stage's remaining hole budget strictly below half the next scale (exact formulas and inequalities in Appendix C.2).

The reading program groups all inspected lines into three geometric families (with every column in $X \cup X_{\text{gap}} \cup X_{\text{samp}}$ on an even coordinate, hence pairwise non-adjacent):

- *Streets:* X and Y are the unions of the street columns and street rows of both halves (including the frame), and $U \subseteq X$ denotes the non-train, non-boundary street columns.
- *Off-street separator and sampling columns:* X_{gap} contains the even-rounded wide-gap and midpoint separator columns, and X_{samp} contains the two *sampling columns* at offsets $+2, +10$ of each AND/OR vertex.
- *Separator and marker rows:* Z consists of the inter-level separator rows, the bottom half-depth row $2D - Y_1/2$, and the central row M. To prevent column reads from borrowing holes across M, our *second modification* adds *central marker holes* in both B_σ and B_τ on four marker rows near M (Figure 25(a)): rows $M \pm 2$ at every $x \in U$ (flanking the central separators (x, M)), row $M - 4$ at every $x \in X_{\text{gap}}$, and row $M + 4$ at every $x \in X_{\text{samp}}$.

Let b be the initial number of holes strictly inside the frame (including these markers); by our scales, every interior hole lies more than b cells from the left and right boundaries, and all four marker rows lie more than b cells from the top and bottom boundaries.

The reading program $\mathcal{P}_\tau$. From $R \setminus B_\tau$, we compute the seven-stage reading program $\mathcal{P}_\tau$ summarized in Table 2, where every pre-read takes the exact hole count of $R \setminus B_\tau$ before its cutoff and the *probe row* of a top level j is row $Y_j + 2$, two rows beyond its working row (its role appears in Step 4). Geometrically, the seven stages scan the board in three acts: Stages 1–2 peel the outer frame, open corridors, and train ends while clearing blank rows; Stages 3–5 lock the four marker fences and read all working rows, probe rows, and sampling columns; and Stages 6–7 sweep the surviving separators from shallowest to deepest. Because $\mathcal{P}_\tau$ runs statically on the fixed final hole set E, the proof in Section 5.3 may extract invariants from later stages (such as Stage 6–7's separator checks in Step 2) before inspecting earlier stages (such as Stage 4's probe rows in Step 4) without circularity. Appendix C.2 gives the exact generator pseudocode (maintaining the remaining target hole set $S \subseteq R \setminus B_\tau$ across all seven stages) and verifies column by column that $\mathcal{P}_\tau$ accepts $R \setminus B_\tau$, is full (quotas sum to K), and uses $4m + 26 = O(m)$ steps.

▶ **Theorem 14** (certificate). *If a trajectory π on R from (p_0, B_σ) ends in a state (p_{end}, B) whose hole set $E = R \setminus B$ is accepted by $\mathcal{P}_\tau$, then τ is reachable from σ.*

Table 2. The seven stages of the reading program $\mathcal{P}_\tau$ computed from B_τ ($O(m)$ steps in total).

Stage	Operations (reads and clears)	Role in Section 5.3
1	Read full boundary columns and remaining boundary rows; clear the b inner columns on each side; pre-read corridor and public columns from their frame up to their first inter-level separator (or $M \pm 2$), leaving private columns $+4$ untouched.	Empty frame (Step 1); corridors stay empty (Step 3).
2	Clear rows outside $Y \cup Z$, marker rows, and working rows ± 1; pre-read each train from the top to Y_j+1, and ordinary trains by 2 from the bottom; clear extra working-neighbour rows except row $y_1 - 1$ (which holds the private-branch cell $(4, y_1 - 1)$).	Locates each train's upper gap on its probe row (Step 4).
3	Read 4 holes from the bottom on each private column $+4$; read rows $M \pm 2$ to $\lvert U\rvert$; pre-read X_{gap} from the top above $M - 4$, then read row $M - 4$ to $\lvert X_{\mathrm{gap}}\rvert$.	Central markers (Step 1); private branch and cell $+4$ (Step 5).
4	Read to full count and clear every top probe row $Y_j + 2$; finish and clear ordinary trains; on middle trains, read remaining target holes north of y_1, clear rows outside $Y \cup Z \cup \{M + 4\}$, and read 1 more hole from the top if $(0, y_1)$ is open in B_τ.	Probe-row equations and track conservation (Step 4); core cell 0 (Step 5).
5	Finish and clear X_{gap} from the bottom; pre-read X_{samp} from the bottom below $M + 4$, then read row $M + 4$ to $\lvert X_{\mathrm{samp}}\rvert$; read all remaining street rows in Y (working rows and bottom corridor rows) to full count; clear columns outside U and middle trains.	Wire bar inventories (Step 4); sampling cells $+2, +10$ (Step 5).
6	Bottom verification: read 1 from the bottom on U columns without a bottom inter-level separator; read the half-depth row and clear middle trains; from shallowest to deepest bottom level, read each inter-level separator row and then read 1 from the bottom on that group's columns.	Central, bottom inter-level, and half-depth separators (Step 2).
7	Top verification: clear U columns without a top inter-level separator; from shallowest to deepest top level, read each inter-level separator row and clear that group's columns.	Top inter-level separators (Step 2).

Note three features of Theorem 14: the final player cell $p_{\text{end}} \in E$ is arbitrary (it need not lie on the frame or at c); E is *not* forced to equal $R \setminus B_\tau$ (for example, a short bar at $p = 7$ instead of 6 can still pass $\mathcal{P}_\tau$); and intermediate states along π are unrestricted.

Completeness and the main theorem. Given the repaired ordered instance (G, σ, τ) constructed above, we build R with initial state (p_0, B_σ), compute $\mathcal{P}_\tau$ from B_τ, and attach the Push-$*$ reader of Lemma 12 for $\mathcal{P}_\tau$ at the lower right corner c of R. Since $m = O(1)$, $\mathcal{P}_\tau$ has constant length and the construction runs in polynomial time.

▶ **Lemma 15** (completeness). *If τ is reachable from σ through ordered orientations in G, then there is a trajectory on R from (p_0, B_σ) to (p_0, B_τ), and hence the Push-$*$ instance is solvable.*

Each ordered move in G reverses a single train while keeping every AND vertex ordered (Appendix A.3, Figure 45). Simulating one train reversal shifts its bars and train in at most eight pushes from the frame, using the five-push macro of Figure 24 for the two-box short bar; central markers avoid all canonical rays. Once (p_0, B_τ) is reached, the player walks along the empty frame to c and passes the reader (Appendix C.2).

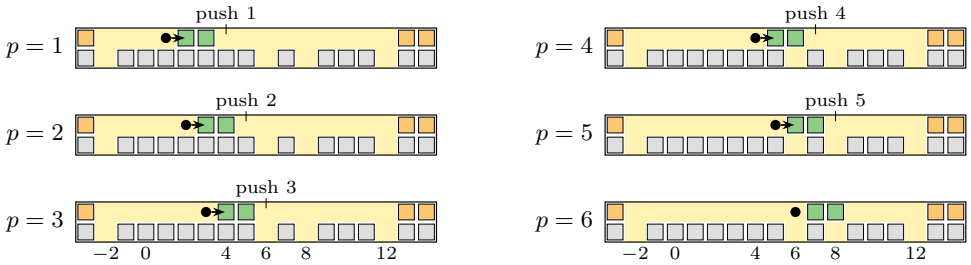


Figure 24. The five-push macro moving the two-box short bar from $p = 1$ to $p = 6$ on an OR working row ($l = 1, r = 1, v = 1$). Standing at (p, y_1), each push right shifts the two green boxes by one cell and fills the next hole in I_J; at $p = 6$, the core I_G reaches public column 6 and the player returns to the frame.

▶ **Theorem 16.** *Push-$*$ is PSPACE-complete.*

Proof. Membership in PSPACE is given in Section 3. By Appendix A.3, τ_H is reachable from σ_H in H if and only if τ is reachable from σ in G, and equivalently through ordered orientations in G. Lemma 15 therefore gives completeness. Conversely, if the Push-$*$ instance is solvable, Lemma 12 extracts a trajectory on R from (p_0, B_σ) to a corner state (c, B) whose hole set $R \setminus B$ is accepted by $\mathcal{P}_\tau$, and Theorem 14 yields an NCL path from σ to τ in G. ◀

5.3 Proof of the certificate theorem

Fix a trajectory π on R from (p_0, B_σ) to a state (p_{end}, B) whose hole set $E = R \setminus B$ is accepted by $\mathcal{P}_\tau$. Since $\mathcal{P}_\tau$ is full for E, every clear in $\mathcal{P}_\tau$ deletes zero holes and no hole remains after Stage 7.

Throughout this proof we keep three notions distinct:

1. *Physical trajectory invariants (throughout π):* a cell stays *permanently empty along* π (such as cells in A or never-filled markers and separators), or a box is *fixed throughout* π.
2. *Terminal snapshot properties (in E):* a cell belongs to the single final hole set $E = R \setminus B$ at the end of π; being a hole of E alone does not say the cell was empty earlier in π.

3. *Static reading state (in $\mathcal{P}_\tau$):* as $\mathcal{P}_\tau$ runs on the fixed snapshot E, it maintains a shrinking working set of *remaining* (unconsumed) holes of E. Reads and clears in $\mathcal{P}_\tau$ never modify the physical board or π; fullness implies that every clear deletes zero remaining holes, certifying that every hole of E on the cleared lines was already consumed earlier (or absent from E).

We prove Theorem 14 in six consecutive steps on the same π and E, summarized in Table 3. To avoid circularity, Steps 1–3 never assume that tracks keep their initial box counts (that is proved in Step 4); every hole-radius budget in Steps 1–3 counts all holes outside the set A already proved permanently empty along π.

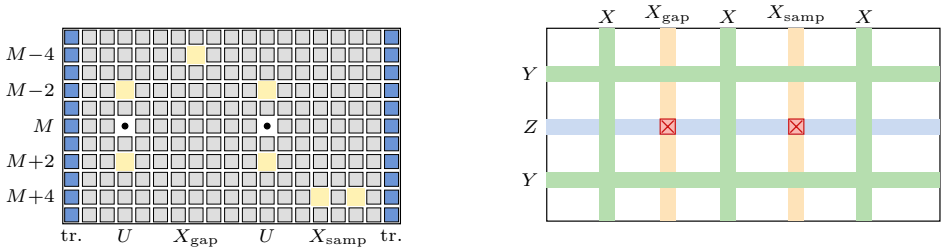
Table 3. Interface of the six steps proving Theorem 14 (separating physical invariants throughout π from terminal properties in E and static reading stages of Table 2).

Step	Inputs & prior invariants	$\mathcal{P}_\tau$ stages	New outputs (in E vs. throughout π)
1	Fullness on E; scale bounds ($> b$); Lemma 8.	Stage 1; Stages 3, 5 (marker rows)	*Throughout π:* frame and all central markers stay permanently empty; projecting markers to boxes gives a legal marker-free trajectory π' (same pushes, terminal holes E', interior hole count $< b_m$, where b_m bounds the marker-free non-frame holes of B_τ in Appendix C.2).
2	Step 1 (empty frame, marker-free π' and E'); fullness (clears delete 0); backward support (Lemma 17, stated in Step 2).	Stages 1–7 (clear/read order); Stages 1, 3, 6, 7	*Throughout π' (hence π):* holes of π' stay on streets $F(X,Y)$, boxes off $F(X,Y)$ never move, central separators (x, M) stay empty, and $M \pm 1$ split the halves. *In E:* all inter-level, half-depth, and private-column +4 cells are holes.
3	Step 1 (budget $< b_m$); Step 2 (street support, certified separators in E); Lemma 8 (global budgets outside A).	Stage 1 (corridor/public pre-reads); Stages 3, 5 (X_{gap})	*Throughout π:* all separators are blocking; every corridor, public column, and private branch N joins A (permanently empty); every background box off tracks is fixed by identity; deep crossings stay occupied.
4	Step 3 (fixed background, empty corridors and N, occupied deep crossings); partner repair (Lemma 18).	Stage 2; Stage 4 (probes, trains); Stage 5 (wire rows)	*Throughout π:* no track exchanges boxes (trains, wire bars, AND/OR working rows keep initial box counts). *In E:* ordinary-train gaps match θ_i; middle trains satisfy $g_i = 2 + \theta_i$, $h_i = 2 - \theta_i$ ($\gamma_M = \gamma_M^*$).
5	Step 2 (private +4 and separators in E); Step 3 (empty corridors, fixed background); Step 4 (track counts, γ_M).	Stage 1 (public/bar pre-reads); Stages 3, 4, 5 ($+4, 0, +2, +10$)	*In E:* working cells $-2, 0, 2, 4, 6, 8, 10, 12$ decode all internal bits $(\gamma_H, \gamma_L, \gamma_M, \gamma_R)$ and wire bars to B_τ; both I_G, I_J are nonempty. *Throughout π:* runs on every track never merged.
6	Step 3 (fixed background, empty corridors); Steps 4–5 (closed non-merging tracks throughout π; valid final bits and train orientations in E).	none (uses Step 5 output)	*Throughout π:* every intermediate orientation satisfies all wire, OR, and AND constraints of G (five cut-off shapes), yielding a valid NCL path $\sigma \to \tau$ in G.

Step 1: The frame and central marker holes. Stage 1 reads every cell of the boundary columns and remaining boundary rows, so the frame lies in E. Since a box pushed onto the frame can never leave and the frame starts empty, the frame stays empty throughout π. The

number b of interior holes is therefore constant, and by Lemma 8 (with A the frame) the player never enters any cell farther than b from the frame.

Distance alone does not make the central marker holes of Section 5.2 harmless: although the player cannot stand on them, a marker could still serve as the first hole of a long push from afar. We rule this out via $\mathcal{P}_\tau$. All four marker rows ($M \pm 2, M-4, M+4$) lie farther than b from the top and bottom frame, and Stage 1 clears the b inner columns on each side (deleting zero). Every remaining cell on those rows is unreachable by the player, so no new hole can appear there and existing markers can only be filled. Stages 3 and 5 read rows $M-2, M+2, M-4, M+4$ to their full initial counts $|U|, |U|, |X_{\text{gap}}|, |X_{\text{samp}}|$ (while earlier pre-reads stop before them), proving that every marker survives in E and was *never filled* during π. Replacing every marker hole by a box therefore produces a legal marker-free trajectory π' with the exact same sequence of pushes and player cells (and with every marker box motionless), lowering the interior hole count below the marker-free bound b_m (Appendix C.2) for all hole-radius bounds in Steps 2–3 while $\mathcal{P}_\tau$ continues to evaluate on the original terminal hole set E of π. Write $E' = E \setminus \{\text{markers}\}$ for the terminal hole set of π'; by the above, E' has zero holes on all four marker rows.



(a) central band markers (b) intersection of the two supports

Figure 25. Steps 1 and 2. (a) Central band around row M: marker holes (yellow) at $M \pm 2, M-4, M+4$ fence the original central separators (x, M) (black dots) and separate top and bottom column reads. (b) Intersection of the two sparse supports $F(X, Y \cup Z)$ (blue and green) and $F(X \cup X_{\text{gap}} \cup X_{\text{samp}}, Y)$ (orange and green): every cell outside the streets $F(X, Y)$ (green) is an isolated intersection (red cross) that can never become a hole.

Step 2: Two sparse supports and the separators. For columns X' and rows Y', let $F(X', Y')$ be the union of all cells on columns in X' and rows in Y'; call it *sparse* if no two columns in X' are adjacent and no two rows in Y' are adjacent.

▶ **Lemma 17** (backward support). *If $F = F(X', Y')$ is sparse and $E' \subseteq F$, then throughout π' every hole lies in F and no box outside F ever moves.*

Just as in Section 4.3, running π' backwards from $p_{\text{end}} \in E' \subseteq F$ (for any final player cell p_{end}, which cannot be a marker by Step 1), each pull steps between two adjacent holes of F, which by sparsity share a line of F, so the entire pull stays on that line.

Write $U = U_{\text{top}} \cup U_{\text{bot}}$ for the top- and bottom-half columns of U, let X_{top} (respectively X_{bot}) be the street columns of the top (respectively bottom) half—the boundary columns, all train columns, and U_{top} (respectively U_{bot})—and write $Y = Y_{\text{top}} \cup Y_{\text{bot}}$ and $Z_{\text{top}} = Z \cap \{y < M\}$, $Z_{\text{bot}} = Z \cap \{y > M\}$. Note that U_{top} may intersect U_{bot} or X_{samp} (for instance, when a middle train's top-half partner lies to its right, $x_c + 2 \in U_{\text{top}} \cap X_{\text{samp}}$), so X_{samp} is off-street in the bottom half rather than globally.

Because $\mathcal{P}_\tau$ is full and E' has zero holes on the four marker rows, every hole of E' is consumed by some positive read outside the marker rows, placing E' in $F(X \cup X_{\text{gap}} \cup X_{\text{samp}},\, Y \cup Z)$. That single union is too coarse, since it allows holes on $X_{\text{gap}} \cup X_{\text{samp}}$ off Y and on Z off X. The *order* of clears and reads splits it into two sparse supports: when columns in X_{gap} or X_{samp} are read (Stages 3 and 5), all non-marker rows outside $Y \cup Z$ in the active half have already been cleared, so $E' \subseteq F(X,\, Y \cup Z)$; when rows in Z are read (Stages 6 and 7), all columns outside X have already been cleared (Stage 5), so $E' \subseteq F(X \cup X_{\text{gap}} \cup X_{\text{samp}},\, Y)$. Applying Lemma 17 to each keeps every hole of π' in their intersection (Figure 25(b)):

$$F(X,\, Y \cup Z) \,\cap\, F(X \cup X_{\text{gap}} \cup X_{\text{samp}},\, Y) \;=\; F(X,Y) \,\cup\, \big(((X_{\text{gap}} \cup X_{\text{samp}}) \setminus X) \times (Z \setminus Y)\big).$$

Every extra cell in $((X_{\text{gap}} \cup X_{\text{samp}}) \setminus X) \times (Z \setminus Y)$ starts as a box and has no orthogonal neighbour inside the intersection, so the player can never step into it to create a hole. Hence every hole of π' stays on the streets $F(X,Y)$, and *no box off the streets ever moves* (in π' and therefore in π).

This certifies all separators:

- *Original central separators:* For $x \in U$, row $M \notin Y$ prevents horizontal pushes, and the never-filled markers at $(x, M \pm 2)$ block vertical pushes while row $M \pm 1$ is unreachable. Thus (x, M) stays empty throughout π, and its unique consumption in $\mathcal{P}_\tau$ is the single bottom read on column x in Stage 6, certifying that column x had no remaining unconsumed holes of E below M at Stage 6.
- *Inter-level and half-depth separators:* In Stage 6 (bottom half, processed from shallowest to deepest level), shallower U columns have already spent their central separator, while deeper corridor columns have already pre-read past the current inter-level row in Stage 1. Only the U columns of the current level can contribute—at most one hole each, matching the quota—so every bottom inter-level separator survives in E. Stage 7 symmetrically certifies all top inter-level separators, and Stage 6's half-depth row read together with Stage 3's four bottom reads on each private column $+4$ certify all half-depth separators (Appendix C.3).

In the central band, the boxes on each $x \in U$ at $(x, M \pm 1)$ are fixed by (x, M) and their off-street side neighbours, so rows $M \pm 1$ are solid outside the boundary and train columns (which belong to both X_{top} and X_{bot}) and split the two halves. We now repeat the two-support argument inside each half separately. In the bottom half $\{y > M\}$, every $x \in U_{\text{top}} \setminus U_{\text{bot}}$ has no bottom inter-level separator, so Stage 6's initial single bottom read consumes (x, M) and certifies that column x has no remaining holes of E' below M at Stage 6. Hence $E' \cap \{y > M\}$ lies in both $F(X_{\text{bot}},\, Y_{\text{bot}} \cup Z_{\text{bot}})$ (columns in $X_{\text{gap}} \cup X_{\text{samp}}$ are read in Stage 5 only after Stage 4 clears all non-marker rows outside $Y \cup Z$) and $F(X_{\text{bot}} \cup X_{\text{gap}} \cup X_{\text{samp}},\, Y_{\text{bot}})$ (no column outside X_{bot} contributes to Stage 6's Z_{bot} row reads). Lemma 17 assumes $E' \subseteq F$ on the whole board, but its proof only uses that a pull stays on the line shared by its two adjacent holes. Since no pull crosses row $M + 1$ outside X_{bot}, a pull that adds a hole in $\{y > M\}$ either has both adjacent holes there or runs along a column of X_{bot}, so the same backward induction, applied to the holes in $\{y > M\}$ only, confines them throughout π' to the intersection $F(X_{\text{bot}}, Y_{\text{bot}}) \cup \big(((X_{\text{gap}} \cup X_{\text{samp}}) \setminus X_{\text{bot}}) \times (Z_{\text{bot}} \setminus Y_{\text{bot}})\big)$ of the two supports. The extra cells of this intersection, including $(x_c + 2, z)$ for $z \in Z_{\text{bot}}$, are isolated and can never become holes. Thus every hole of π' in $\{y > M\}$ stays on $F(X_{\text{bot}}, Y_{\text{bot}})$, and symmetrically every hole of π' in $\{y < M\}$ stays on $F(X_{\text{top}}, Y_{\text{top}})$.

Step 3: Freezing the background level by level. With the streets and separators certified, the freezing induction of Section 4.3 goes through using global budgets from $\mathcal{P}_\tau$ (exact budgets and the first-move rule are verified in Appendix C.3):

- *Wide gaps:* Each wide-gap separator column in X_{gap} lies farther than b from the nearest street column, so its full column reads in Stages 3 and 5 prove all its separators survive, making them blocking and fixing the wide street-row segments.
- *Level j, for $j = m, \ldots, 1$:* Entering level j, each corridor column of level j is already closed on the central side (by the central band for $j = m$, or its blocking level-j inter-level separator made blocking at the end of level $j + 1$ for $j < m$). Stage 1's pre-read on each level-j train corridor column, combined with its surviving inter-level separator, forces the entire train corridor to be empty in E; as in Figure 14 of Section 4.3, no box can leave it, so every level-j train corridor and its horizontal branch stay empty throughout π and join A. Subtracting them drops the global remaining hole count to at most K_j, strictly below the distance (roughly $L_j/2$) to the midpoint separators between level-j trains and their corridors. On each such midpoint column $x_{\text{mid}} \in X_{\text{gap}}$, Step 2 restricts holes of E' in this half to street rows, every street-row cell at depth $d > Y_j + 2$ is occupied in E' (either fixed in the wide-gap step, or lying on a deeper level-j' bar with $j' > j$ at Manhattan distance $> K_j$ from A, hence occupied throughout π by Lemma 8), and depth $Y_j + 2$ is already in A. Thus the X_{gap} column read (Stage 3 in the top half, fenced by marker $M - 4$, and Stage 5 in the bottom half) forces every level-j midpoint separator ($d \le Y_j$) to survive in E'; being farther than K_j from A, those separators are blocking and fix the background next to the trains. As in Section 4.3, the bar corridors and the single holes between each level-j train and its bar corridor on shallower street rows then stay empty and join A. The remaining hole count then drops enough to make the level-$(j-1)$ inter-level separators blocking for the next stage.
- *Level 1, cores, and deep crossings:* Adding the public columns $-2, 6, 8, 12$ to A drops the remaining holes below our non-corridor bound q_0, making the half-depth separators at depth $Y_1/2 > q_0$ in columns 0 and 4 blocking and fixing the background around each core and private branch N. Finally, every deep crossing where a bar crosses another train lies farther than q_0 from any reachable hole, so it holds a box throughout π (though pushes may pass through it and swap box identities).

Thus every background box off the tracks is fixed by identity, and every corridor and separator stays empty throughout π.

Step 4: Tracks never exchange boxes. This step resolves the new difficulty absent in Section 4: without an exact target, tracks could trade boxes at their junctions. The argument works as a conservation ledger: every inter-track transfer is one-way, column reads express each train's net leak in terms of its probe-row bit z_i, and probe-row reads balance all leaks—until partner repair separates the two sides of the balance equation onto odd and even levels.

First, every possible exchange is *one-way.* Pushing a box from a train into a bar at a pocket requires standing on, or pushing, the fixed background behind the pocket corner; pushing a box from an AND/OR working row up into its middle train requires the fixed background below column 0; and permanently occupied deep crossings do not change track box counts. Hence:

> *No wire bar or AND/OR bar ever gains a box from a train, and no middle train ever gains a box from its AND/OR working row.*

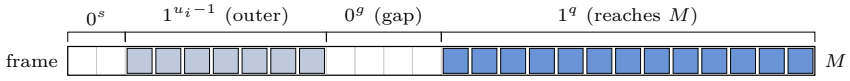
Wire bars keep their boxes. On any top-half working row carrying ℓ wire bars, each bar starts with 3 holes between its trains and cannot gain boxes, so its final hole count satisfies $h_e \ge 3$. Stage 5 reads that working row to full count (together with Stage 1's one-hole

bar-corridor pre-read per bar), yielding $\sum_e h_e \le 3\ell$, so every $h_e = 3$; since box counts cannot increase, no wire bar ever loses a box. The same holds on bottom-half wires (levels ≥ 2). Hence every ordinary train whose lower end meets a wire is sealed at both ends and keeps its 5 holes.

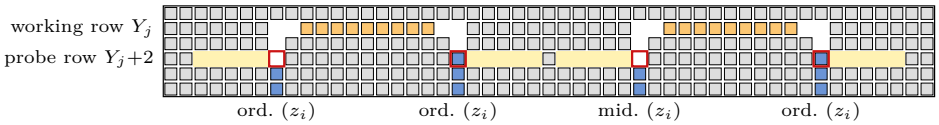
Ordinary trains. An ordinary train i whose top working row lies at depth $u_i = Y_j$ and whose lower end meets an AND/OR outer bar may absorb $J_i \ge 0$ boxes from that bar, leaving $5 - J_i$ holes in E. Let $\theta_i \in \{0, 1\}$ indicate whether τ points train i up, and let $z_i \in \{0, 1\}$ be the number of *remaining (unconsumed)* holes of column x_i on its top-half probe row $Y_j + 2$ entering Stage 4. Stage 2 pre-reads 2 holes from each end of column x_i, and Stage 4 consumes z_i on the probe row and then reads $1 - \theta_i$ before clearing the column (deleting zero). Summing all $5 - J_i$ holes gives

$$5 - J_i = 2 + 2 + z_i + (1 - \theta_i), \qquad \text{so} \qquad J_i = \theta_i - z_i. \tag{1}$$

Because the midpoint (x_i, M) stays occupied, the upper half of column x_i keeps the occupancy pattern $0^s 1^{u_i - 1} 0^g 1^q$ from the top frame to M (Figure 26(a); Appendix C.4). Writing g_i and g_{bot} for its upper and lower gap sizes, Stage 2 clears the row just inside the top frame and row $Y_j + 3$, forcing $s = 0$ and confining the upper gap to $\{Y_j, Y_j + 1, Y_j + 2\}$ ($g_i \le 3$), and symmetrically forces $g_{\text{bot}} \le 3$. Once $J_i = 0$ is established below, $g_i + g_{\text{bot}} = 5$ with $g_i, g_{\text{bot}} \le 3$ forces $g_i, g_{\text{bot}} \ge 2$, so Stage 2's pre-reads consume 2 holes at each end without crossing M and z_i coincides with the physical hole at $(x_i, Y_j + 2)$.



(a) upper half of a train column from the frame to M



(b) probe row $Y_j + 2$ in the top half (red squares are the probed train cells)

Figure 26. Step 4: locating probe-row holes. (a) Occupancy of the upper half of a train column from the frame to midpoint M; clearing row 1 forces $s = 0$, placing the upper gap 0^g starting at $u_i = Y_j$. (b) A top-half probe row $Y_j + 2$ crossed by ordinary trains and a middle train; yellow cells are horizontal train-corridor branches (already in A), and red squares are the probed train cells (contributing z_i).

Middle trains. A middle train i starts with 4 holes strictly above its bottom working row y_1, has a sealed top bar, and cannot gain boxes from below. If it loses $\delta_i \ge 0$ boxes into the core, its $4 + \delta_i$ holes split between its upper gap (g_i holes) and its lower gap directly above y_1 ($h_i = v$ holes; Appendix C.4). Stage 2's row clears force $g_i \le 3$ and $h_i \le 2$, so $g_i \ge 2$, $h_i \ge 1$, and its probe-row hole count is $z_i = g_i - 2 \in \{0, 1\}$. In Stage 4, after reading the probe row, $\mathcal{P}_\tau$ reads $2 - \theta_i$ more holes from the top of column x_i and clears row $y_1 - 1$; fullness forces $h_i \le 2 - \theta_i$, and since $g_i + h_i = 4 + \delta_i$,

$$z_i - \theta_i = (g_i - 2) - \theta_i = \delta_i + (2 - \theta_i - h_i) \;\ge\; \delta_i \;\ge\; 0. \tag{2}$$

The cancellation problem on a probe row. On a top-half probe row $Y_j + 2$, let I_j^{ord} and I_j^{mid} be the ordinary trains and middle trains of top level j (Figure 26(b)). After subtracting

the horizontal train-corridor branches and the bar-side holes of deeper levels $> j$ (all already proved empty in A), reading and clearing row $Y_j + 2$ gives $\sum_{i \in I_j^{\mathrm{ord}} \cup I_j^{\mathrm{mid}}} z_i = \sum_{i \in I_j^{\mathrm{ord}} \cup I_j^{\mathrm{mid}}} \theta_i$. Substituting (1) yields

$$\sum_{i \in I_j^{\mathrm{mid}}} (z_i - \theta_i) \;=\; \sum_{i \in I_j^{\mathrm{ord}}} (\theta_i - z_i) \;=\; \sum_{i \in I_j^{\mathrm{ord}}} J_i. \tag{3}$$

Every term on both sides is non-negative, but they sit on *opposite* sides: a middle train dumping a box into its core ($z_i - \theta_i = 1$) can cancel an ordinary train absorbing a box from an AND/OR bar ($J_{i'} = 1$) on the same probe row.

Third modification: partner repair and parity levels. To prevent cancellation, we ensure that no probe row contains both a middle train and an ordinary train whose lower end meets an AND/OR vertex. Call the other train in the top-half wire of a middle train its *partner*. We subdivide edges with extra wires so that:

(P1) Every partner of a middle train is either another middle train or an ordinary train whose lower end meets a wire (hence already known to have $J_i = 0$).

(P2) In the top half, wires containing a middle train receive even working levels and all other wires receive odd levels; in the bottom half, AND/OR vertices receive level 1 and wires receive levels ≥ 2.

Two local operations achieve (P1) and (P2) (Figure 27): an *ear* replaces a top wire pairing adjacent ports a, b of one AND/OR vertex by q–a and p–b with a bottom wire p–q (and symmetrically for b, c); a *cut* at the boundary of an AND/OR vertex whose middle train pairs with an outer port of another AND/OR vertex simultaneously slices all t active upper wires crossing that boundary, inserting $\mathsf{p}_t, \ldots, \mathsf{p}_1, \mathsf{q}_1, \ldots, \mathsf{q}_t$ with nested bottom wires p_j–q_j.

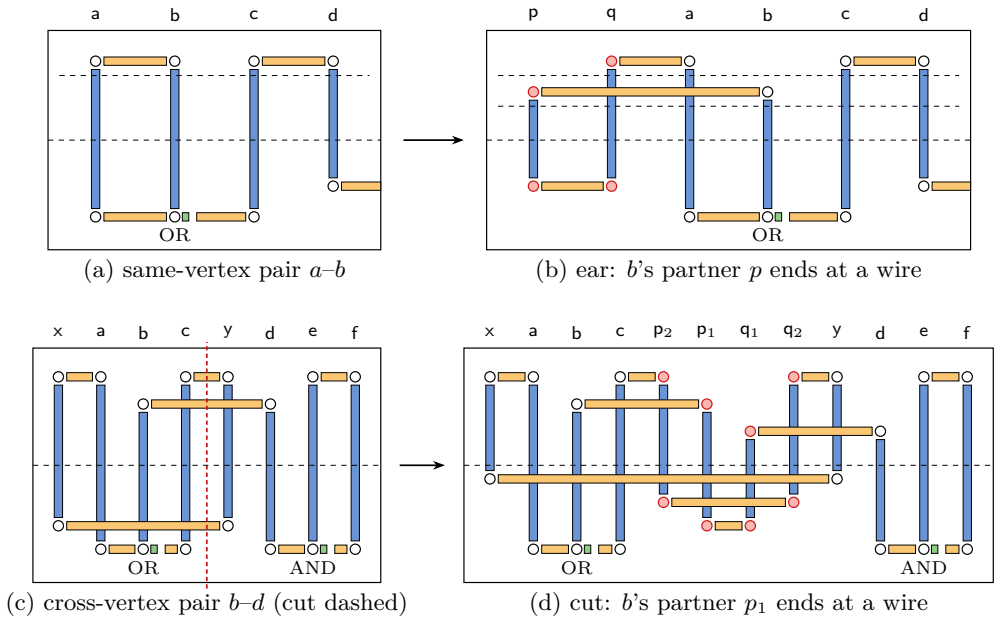
▶ **Lemma 18** (partner repair). *Starting from a valid two-page layout with n_{init} trains, n_3 AND/OR vertices, and top and bottom nesting depths m_T, m_B, partner repair produces an equivalent wire-subdivided layout satisfying* (P1) *and* (P2) *with*

$$m \;\leq\; \max(2m_T + 2,\; m_T + m_B + 2), \qquad n \;\leq\; n_{\mathrm{init}} + 2n_3(m_T + 2).$$

Ears raise top depth by at most 1 without increasing bottom depth; each cut slices at most $m_T + 1$ nested upper arcs without increasing top depth and adds one disjoint nested wire chain in the bottom half; placing every AND/OR vertex on level 1 and giving each wire the smallest level allowed by (P2) (even for a top wire with a middle train, odd for other top wires, at least 2 for bottom wires) that exceeds the levels of everything nested inside it preserves strict nesting; by induction, a wire of nesting depth d gets level at most $2d$ in the top half and at most $d + 1$ in the bottom half, which gives both bounds (Appendix C.4).

Now return to (3) on the repaired board. On an *even* top level j, every ordinary train $i \in I_j^{\mathrm{ord}}$ is a partner of a middle train, so by (P1) its lower end meets a wire and $J_i = 0$; hence $\sum_{i \in I_j^{\mathrm{mid}}} (z_i - \theta_i) = 0$, forcing $z_i = \theta_i$, $\delta_i = 0$, and $h_i = 2 - \theta_i$. On an *odd* top level j, $I_j^{\mathrm{mid}} = \emptyset$, so $\sum_{i \in I_j^{\mathrm{ord}}} J_i = 0$, forcing $J_i = 0$ for every ordinary train. Because every transfer was one-way and initial and final counts match, *no track ever exchanges a box with another track throughout π*. Every AND/OR working row keeps its $\kappa + 11$ holes, every ordinary train ends with gap sizes 2 and 3 matching θ_i, and every middle train ends with $g_i = 2 + \theta_i$ and $h_i = 2 - \theta_i$.

Step 5: Decoding the final bits and non-merging. Fix an AND or OR vertex. By Step 4, its working row holds its three runs of boxes (left bar, two-box short bar, right bar) in order,



(a) same-vertex pair a–b (b) ear: b's partner p ends at a wire

(c) cross-vertex pair b–d (cut dashed) (d) cut: b's partner p_1 ends at a wire

Figure 27. Partner repair on the two-page layout (drawn as in Figure 2(a); new ports are red). Top (ear): (a) same-OR pair a–b shares probe row 1; (b) inserting p, q puts b–p on even level 2, with p ending at a bottom wire. Bottom (cut): (c) middle train b pairs with outer port d of the AND; (d) cutting all active upper arcs across the dashed line inserts $\mathsf{p}_2, \mathsf{p}_1, \mathsf{q}_1, \mathsf{q}_2$ with nested bottom wires, so b's new partner p_1 ends at a wire.

though we have not yet ruled out two runs touching. Allowing touching runs temporarily, write its final state in E by $l, r \geq 0$ and short-bar position p ($l - \kappa \leq p \leq 11 - r$), with $I_G = [l - \kappa + 1, p]$ and $I_J = [p + 3, 13 - r]$.

The column reads of $\mathcal{P}_\tau$ certify cell by cell every hole of B_τ on the working row at local offsets $-2, 0, 2, 4, 6, 8, 10, 12$: offsets $-2, 6, 8, 12$ come from Stage 1's public-column pre-reads, 2 and 10 from Stage 5's bottom reads on sampling columns X_{samp} (fenced by marker $M + 4$), 4 from Stage 3's four bottom reads on the private column, and 0 from Stage 4's extra top read on the middle train after its 4 upper/lower holes are accounted for. Likewise, bar-corridor pre-reads force $l \geq 2$ or $r \geq 2$ whenever B_τ has 2 holes at an outer end (Appendix C.5).

These certified holes force the four internal bits $(\gamma_H, \gamma_L, \gamma_M, \gamma_R)$ to match their target values $(\gamma_H^*, \gamma_L^*, \gamma_M^*, \gamma_R^*)$ in B_τ: $\gamma_M = [h_i \geq 2]$ is already exact from Step 4; cell 12 can lie only in I_J and $x_L = 2 - \kappa$ (-2 for OR, 0 for AND) can lie only in I_G, giving $\gamma_R = [r \leq 1]$ and $\gamma_L = [l \leq 1]$; and since the short bar occupies $[p + 1, p + 2]$, each certified working-row hole at x rules out $p \in \{x - 2, x - 1\}$. Figure 28 shows that the certified holes rule out every $p \leq 5$ when $\gamma_H^* = 1$ and every $p \geq 6$ when $\gamma_H^* = 0$, giving $\gamma_H = [p \geq 6] = \gamma_H^*$ even though non-canonical shifts such as $p = 7$ can survive (full interval check in Appendix C.5).

Furthermore, in every valid target state both I_G and I_J contain at least one certified working-row hole (e.g., $2 \in I_G$ and $12 \in I_J$ when $\gamma_H^* = 1$, and $8 \in I_J$ together with x_L or 0 in I_G when $\gamma_H^* = 0$; Appendix C.5), so both gaps are nonempty in E. Because two runs on a closed track can never separate once they meet, the three runs on every AND/OR working row never merged during π, and neither did the runs on any train or wire bar. Bar-corridor pre-reads likewise decode every wire bar to its direction in B_τ.

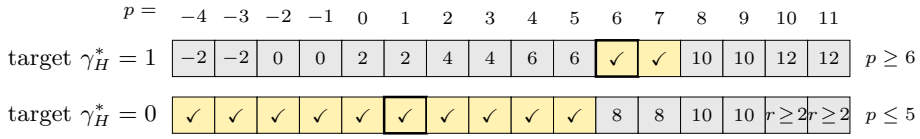
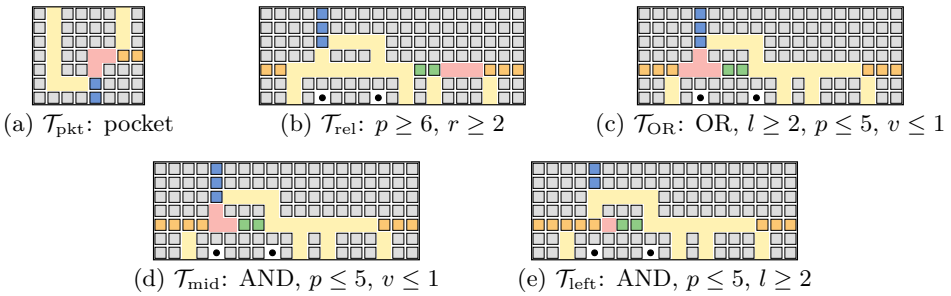


Figure 28. Decoding the short-bar bit $\gamma_H = [p \geq 6]$ in Step 5. Each gray cell shows why that shift p is excluded: a certified working-row hole (for $\gamma_H^* = 1$, the shown exclusions $-2, 0$ on $[-4, -1]$ apply when $\gamma_L^* = 1$; when $\gamma_L^* = 0$, the bound $p \geq l - \kappa$ with $l \geq 2$ excludes $p \leq -3$, and all of $[-4, -1]$ in an AND, while in an OR the certified hole 0 still excludes $p \in \{-2, -1\}$), or $r \geq 2$ from the right bar-corridor pre-read when $\gamma_H^* = \gamma_R^* = 0$ (forcing $p \leq 11 - r \leq 9$ and excluding $p \in \{10, 11\}$). Yellow cells (✓) mark shifts on the surviving side of the threshold, with the canonical shift ($p = 6$ or $p = 1$) framed in bold (when $\gamma_H^* = 0$, some shifts on $p \leq 5$ may also be excluded depending on l). Non-canonical shifts such as $p = 7$ can survive, yet $\gamma_H = [p \geq 6]$ is still exact.

Step 6: First violation and decoding back to NCL. By Steps 4 and 5, every track is closed and its runs never merge, so at every step of π each train points into one endpoint and each push reverses at most one train. Suppose for contradiction that some vertex violates its constraint along π, and consider the first push after which a vertex u is violated. Since all constraints are monotone, this push reversed some edge e away from u: it moved the train of e one cell towards u from the far end of e, leaving the player *outside* u.



(a) $\mathcal{T}_{\text{pkt}}$: pocket (b) $\mathcal{T}_{\text{rel}}$: $p \geq 6$, $r \geq 2$ (c) $\mathcal{T}_{\text{OR}}$: OR, $l \geq 2$, $p \leq 5$, $v \leq 1$

(d) $\mathcal{T}_{\text{mid}}$: AND, $p \leq 5$, $v \leq 1$ (e) $\mathcal{T}_{\text{left}}$: AND, $p \leq 5$, $l \geq 2$

Figure 29. The five cut-off shapes in Step 6 (yellow holes connect to the frame, red holes are cut off). In the two-box gadget, I_J being nonempty allows $p + r \leq 10$, so the relay trap $\mathcal{T}_{\text{rel}}$ in (b) (where $I_J \subseteq [9, 11]$ is cut off) must be listed alongside the pocket $\mathcal{T}_{\text{pkt}}$ and the three core shapes $\mathcal{T}_{\text{OR}}, \mathcal{T}_{\text{mid}}, \mathcal{T}_{\text{left}}$.

Figure 29 shows the five cut-off shapes in the two-box geometry (each paired with "the player is outside the red region"): the ordinary pocket $\mathcal{T}_{\text{pkt}}$, the *relay gap* $\mathcal{T}_{\text{rel}}$ ($p \geq 6$ and $r \geq 2$, cutting off $I_J \subseteq [9, 11]$—needed here because $I_J \neq \emptyset$ only gives $p + r \leq 10$, whereas in Section 4's one-box gadget it forced $p + r \leq 5$), the OR core $\mathcal{T}_{\text{OR}}$ ($l \geq 2, p \leq 5, v \leq 1$), and the two AND cores $\mathcal{T}_{\text{mid}}$ ($p \leq 5, v \leq 1$) and $\mathcal{T}_{\text{left}}$ ($p \leq 5, l \geq 2$). In all five shapes, restoring the violated condition requires pushing from inside the cut-off region, while pushes from outside only shrink its gaps; every first violation at a wire, OR, or AND hits one of these five shapes (case analysis in Appendix C.5). Hence the cut-off condition persists to the end of π, contradicting the valid final bits decoded in Step 5. Therefore every orientation along π satisfies all wire, OR, and AND constraints of G. Since π starts at σ, reverses at most one train per push, and ends with every ordinary and middle train matching its target orientation θ_i in τ (Step 4), the sequence of orientations along π yields a valid NCL path from σ to τ in G (which decodes to a valid path from σ_H to τ_H in H by Appendix A.3), proving Theorem 14.

6 PushPush-kS and PushPush-$*$

Maximal slides change the geometry of pushing completely. In Push (Sections 4 and 5), a long train of boxes could be nudged by a single cell, encoding one bit in a one-cell offset while distant holes froze the background. In PushPush, a train advances exactly one cell when the gap ahead is one cell long. A longer gap forces a longer slide, so the player cannot choose the one-cell displacement used in the Push construction.

We therefore reverse the encoding. Instead of moving trains, *every unit of state is a single box resting against a fixed stopper*, and *one extra moving box—the signal—carries the player's location through a network of doors*. Whenever the signal enters a door room on the same column as the door's stored box, sliding one box onto the other lets them swap roles: one stays parked as the new stored state, and the other shoots out westward as the new signal.

The source problem. We reduce from reachability in a network of *directed symmetric self-closing doors*, which Ani et al. [1] (Corollary 2.4) proved PSPACE-complete. Each door has a state 0 or 1 and two directed passages (s_0, t_0) and (s_1, t_1); in state i, only passage i may be traversed (from s_i to t_i), and traversing it flips the state to $1 - i$. Contracting each connected component of free-walking space between doors into a *location*, an instance specifies a set of locations, a list of d doors $(s_0, t_0, s_1, t_1, \text{bit})$ with initial states, a start location, and a target location, and asks whether a sequence of legal door traversals reaches the target location from the start. No planar embedding is needed: our wiring routes arbitrary pairs of rows in two dimensions.

Organization of the section. Both PushPush-kS (for all fixed $k \geq 1$ and $k = *$) and PushPush-$*$ build on the same wall-free *work board*:

1. Section 6.1 designs the work board—doors, location rooms, and arbitrary six-pin wires—under a *backbone invariant* stating that the stopper column and room dividers never move.
2. Section 6.2 proves that four end-of-trajectory conditions enforce the backbone invariant throughout the entire play, with no fixed walls.
3. Section 6.3 settles exact storage (PushPush-kS for every $k \geq 1$ and $k = *$). The target supplies the four backbone conditions for free, but the doors end in run-dependent states; we append a door-by-door *cleanup chain* that merges every arrival state into a single target configuration.
4. Section 6.4 settles the player goal (PushPush-$*$). Dropping the cleanup chain, we check the four backbone conditions and arrival at the target room via a seven-round full run-read program compiled by Lemma 13, whose quotas do not depend on the final door states.

Coordinate formulas, exhaustive wrong-push case checks, and quota accounting are collected in Appendix D.

6.1 The work board: doors, locations, and arbitrary wiring

Our work board R is a wall-free $w_0 \times h_0$ rectangle divided into two regions (Figure 30). Near the east side, the column with x-coordinate P holds the *stopper rack*, a set of ordinary boxes, and the $\mu = w_0 - 2 - P > 0$ interior columns east of it are empty. Immediately west of column P is the *working column* $c = P - 1$, along which all door rooms and location rooms are stacked vertically. Consecutive rooms are separated by a *partition*, a single box at (c, g) flanked by rack stoppers at $(P, g-1)$ and $(P, g+1)$ with (P, g) empty; we write Y_{part} for the set of partition rows and Y_P for the set of rows holding a box in column P. Some location

rooms also place an *end-cap* box at (c, y) two rows inside a partition ($y = g \pm 2$); write Y_{cap} for the set of these cap rows. The wide area west of column c holds only sparse *wire pins* (ordinary boxes) routing signals from room outputs back to room inputs.

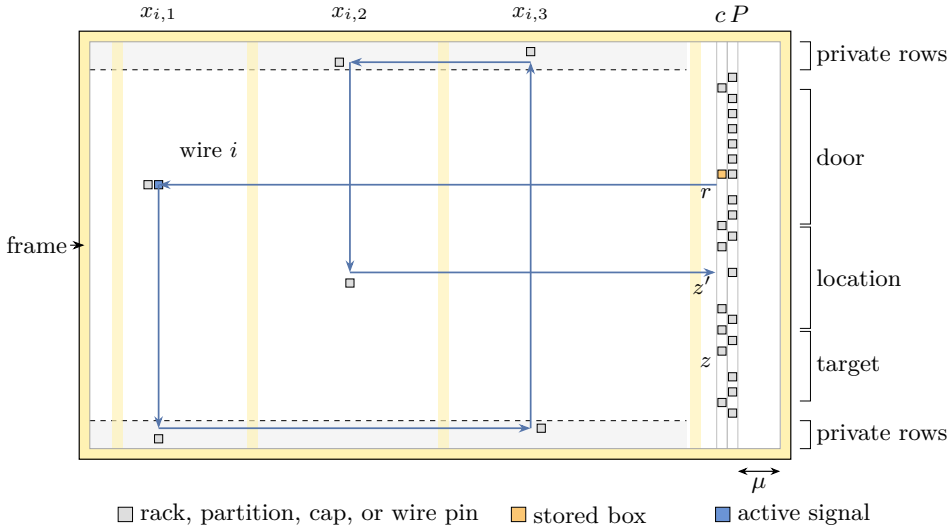


Figure 30. Architecture of the PushPush work board R. Rooms stack vertically along the working column c, backed by the stopper rack in column P and separated by partitions. West of c, each wire routes one output row to one input row using six ordinary pins placed on private columns and private top/bottom rows, leaving empty vertical lanes (pale yellow) connected to the empty top frame row. The wire shown leaves a door from row $r = g + 61$ and enters a location room at row z'; long stretches of empty rows and columns are drawn shortened.

For this subsection, assume the following *backbone invariant* holds throughout the trajectory:

> *No box ever touches the outer frame of R; every cell of column P keeps its initial occupancy (boxes on Y_P, holes elsewhere); every partition (c, g) ($g \in Y_{\text{part}}$) and cap (c, y) ($y \in Y_{\text{cap}}$) stays occupied; and the forbidden cells $(c, g \pm 1)$ adjacent to every partition stay empty.*

Section 6.2 will guarantee this invariant from four conditions on the final configuration alone.

Doors: why six slots and role-swapping work. Each door occupies a room of height 90 along column c, between north partition g and south partition $g + 90$, with no end-caps and a single stored box initially at slot α ($g + 30$, state 0) or slot β ($g + 60$, state 1). Why can a door not use only these two slots? When the signal slides east into an input slot e (stopped by a rack box at (P, e)), it shares column c with the stored box at some slot $m \neq e$. Under the backbone invariant, neither box can slide east (blocked by column P) or west (the player cannot stand on the frozen rack boxes in column P or push them west), and sliding either box into a partition hits a forbidden cell $(c, g \pm 1)$. Thus the only legal next push slides one of the two boxes vertically onto the other:

- sliding the old stored box from m onto e leaves the box at e parked as the new stored box, while the slid box stops adjacent to e (at row $e \pm 1$, on the side toward m) and can now slide west as the new signal;

- sliding the incoming signal from e onto m keeps the box at m stored, while the slid box stops adjacent to m and leaves west.

Because boxes are unlabelled, the first option lets the incoming signal and the stored box *swap roles*, shifting the stored box from m to e. With only the slots α and β, the second option breaks the door: a signal entering β for passage 0 may be slid onto the stored box at α, which leaves the state unchanged and emits a box on row $\alpha + 1$. Passage 1 emits on the same row (its signal enters α, and the stored box slides from β onto it), so this wrong choice would lead from s_0 to t_1. To traverse passage 0 from state 0, we therefore first *test* that the stored box is at α (not β), which shifts the stored box into a check slot; we then need a staging hop to move the stored box from that check slot to β without colliding with passage 1.

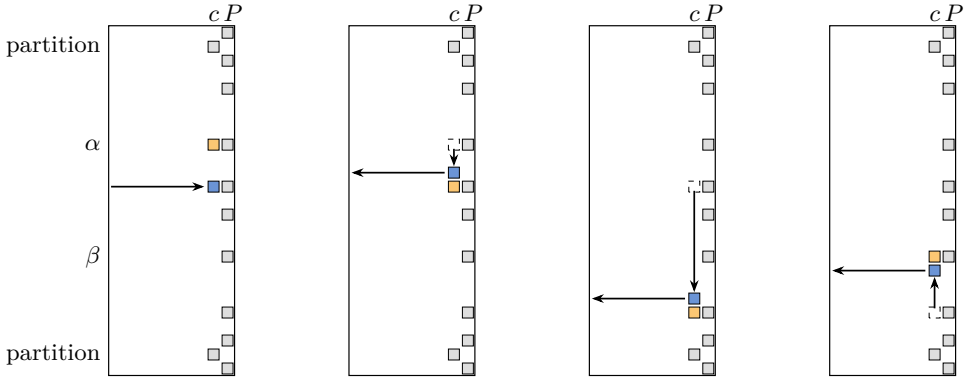


Figure 31. Traversing passage 0 from state 0 on the actual work board (a copy of Figure 5; long stretches of empty rows are drawn as one row). Stored box orange, signal blue, old cell of the slid box dashed. (a) Signal arrives at check α (row 40). (b)–(d) Three slides shuttle the stored box $30 \to 40 \to 80 \to 60$ (β) while emitting the signal on rows $39, 79, 61$.

Six slots in each door room—each with a rack stopper in column P, spaced by at least 10 rows so their ± 1 neighbours never overlap—implement this three-step "test, stage, commit" shuttle symmetrically for both passages (Figures 31 and 32, coordinates relative to g):

Slot	Offset	Valid output row	Destination of wire
Low staging	10	11 (+1)	Stable α (30)
Stable α (state 0)	30	29 (−1)	Passage-1 destination t_1
Check α (entry s_0)	40	39 (−1)	High staging (80)
Check β (entry s_1)	50	51 (+1)	Low staging (10)
Stable β (state 1)	60	61 (+1)	Passage-0 destination t_0
High staging	80	79 (−1)	Stable β (60)

Every slot has only *one* valid output side; its opposite neighbouring row is a *dead row* with no outgoing wire. Let us trace passage 0 when the signal enters check α (40):

- If the door is in state 0 (box at 30), sliding 30 south onto 40 emits the signal on row 39 (wired to high staging 80) and stores a box at 40. At 80, sliding 40 south onto 80 emits on row 79 (wired to $\beta = 60$) and stores at 80. At 60, sliding 80 north onto 60 emits on row 61 (wired to t_0) and leaves the stored box at $\beta = 60$, flipping the state. At steps 1 and 2, sliding the incoming box onto the old stored box lands on dead rows 31 and 41; at step 3, sliding 60 south onto 80 lands on row 79, which simply routes back to 60 and retries the final step.

- If instead the door is in the wrong state 1 (box at 60), sliding 60 north onto 40 lands on row 41 and sliding 40 south onto 60 lands on row 59—both dead rows.

Passage 1 (entering check β at 50 from state 1) is the exact vertical mirror image, shuttling $60 \to 50 \to 10 \to 30$ via rows $51, 11, 29$ (Figure 32; full transition check in Appendix D.2). Note that while no signal is in a door room, its single stored box is completely immobile under the backbone invariant.

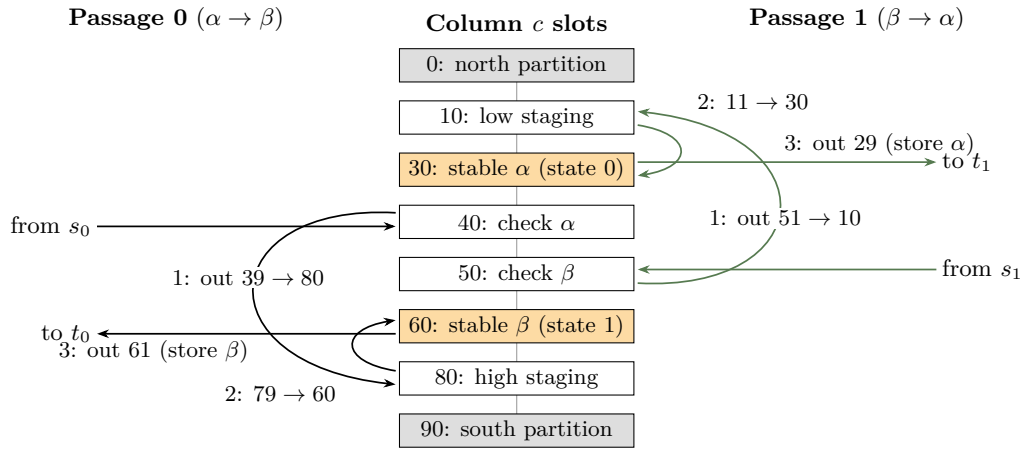


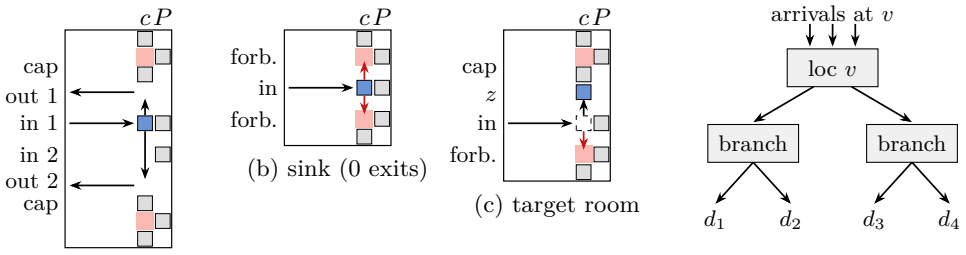
Figure 32. The three-step "test, stage, commit" shuttle of a six-slot door room. Passage 0 (left, black) verifies state 0, stages at 80, and exits to t_0 leaving state 1; passage 1 (right, green) is the exact mirror image. Entering in the wrong state forces both possible slides onto dead rows.

Locations, branching, and the target room. A location room holds no stored box; it merges any number ℓ of incoming wires on private, well-spaced input rows, each backed by a rack stopper in P (Figure 33; exact coordinates in Appendix D.1). When a signal arrives at an input row, its only legal moves are vertical slides along column c. Placing a fixed end-cap two rows inside the north partition catches a northward slide one row further in, from which the signal can leave west; a south end-cap symmetrically offers a second exit. Omitting a cap turns that direction into a forbidden collision with $(c, g \pm 1)$:

- 0 caps build a *sink* room for a dead-end location (Figure 33(b));
- 1 or 2 caps build a relay or a 2-way nondeterministic branch (Figure 33(a));
- a location with more than two outgoing door passages expands into a binary tree of 1-input, 2-output junction rooms (Figure 33(d));
- the start location receives one extra input row holding the initial signal box;
- the target location (which without loss of generality has no outgoing passages) has only a north cap, and its adjacent exit row z has no outgoing wire—it is the *target parking row* (Figure 33(c)).

The total number of rooms, input rows, and wires is $O(N)$, where N is the encoding length of the source door network (Appendix D.1).

Arbitrary wiring with movable pins. Because merging and branching happen inside rooms, every valid output row r sends at most one wire and every input row z' receives at most one wire. To connect (c, r) to (c, z') without fixed walls or planarity, we assign wire i three private columns $x_{i,1} < x_{i,2} < x_{i,3}$ west of c and two private rows $y_{i,1}, y_{i,2}$ in dedicated horizontal



(a) 2-way location (b) sink (0 exits) (c) target room (d) binary choice tree

Figure 33. Location rooms and branching. Forbidden partition neighbours $(c, g \pm 1)$ are shaded red. (a) A 2-way location merges incoming wires and lets the signal choose either cap's exit. (b) With no caps, both vertical slides hit forbidden cells. (c) The target room has only a north cap, parking the signal at row z. (d) Arbitrary out-degree is realized by a binary tree of junction rooms.

bands above or below all rooms (Figure 34(a); exact coordinates in Appendix D.1). We arrange these lines with wide spacing so that all ± 1 column neighbours are disjoint across wires and all private rows are separated from each other, from every room row, and from the frame.

The signal follows a six-turn polyline

$$(c, r) \to (x_{i,1}, r) \to (x_{i,1}, y_{i,1}) \to (x_{i,3}, y_{i,1}) \to (x_{i,3}, y_{i,2}) \to (x_{i,2}, y_{i,2}) \to (x_{i,2}, z') \to (c, z'),$$

stopped at each turn by one ordinary box—the six *wire pins* $p_1, \ldots, p_6$—and finally stopped at (c, z') by the frozen rack box (P, z'). Writing $\varepsilon_1, \varepsilon_2, \varepsilon_3 \in \{-1, +1\}$ for the signs of the three vertical legs ($+1$ is south), we place $y_{i,1}$ and $y_{i,2}$ in the north or south band so that:

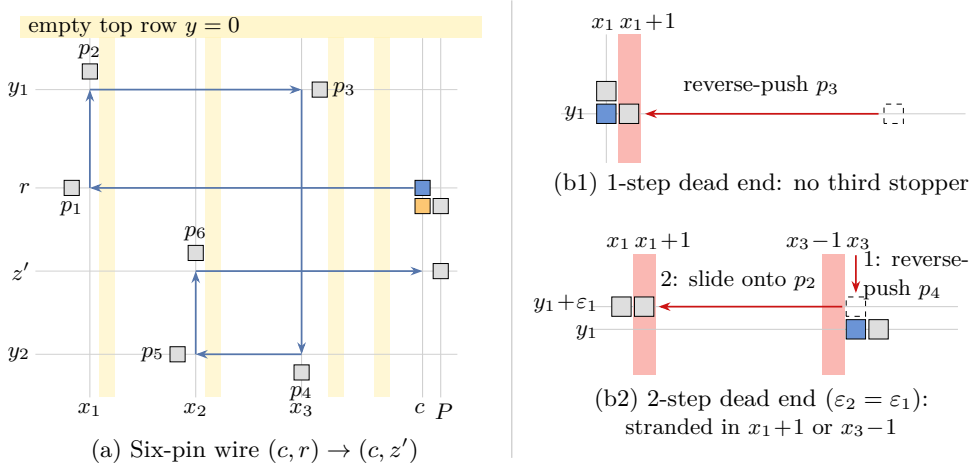
- ε_1 points *away* from the room box that (c, r) rested against (so row $r + \varepsilon_1$ is empty, never colliding with a door slot);
- ε_3 points toward the *dead* output side of a door slot z' (and south for a junction input), so p_6 lies on a dead row, never colliding with any valid output row, slot, partition, or cap.

Because every wire has its own columns and its own private rows, the $6n$ pins of n wires have *pairwise distinct x-coordinates and pairwise distinct y-coordinates.* Since p_1 lies on an output row, p_6 on a dead row, and $p_2, \ldots, p_5$ in the private bands, no pin lies on a row in $Y_P \cup Y_{\text{part}} \cup Y_{\text{cap}}$ or on the target parking row (Appendix D.1). Wires may cross freely: because there is only one active signal, a crossing contains no box and never deflects a slide.

Why wrong pushes cannot cheat. Unlike the frozen rack boxes in column P, the wire pins are ordinary movable boxes. At any turn along a wire, the signal rests against the previous pin and aligns with the next pin; nothing stops the player from walking around to the *far pin* and reverse-pushing it toward the signal. Why does this never break soundness?

Because no two wire pins share a row or column and every room without the signal is immobile, at a turn only two boxes can move without sliding a box into the frame or moving the backbone: the signal or its next pin. If the player reverse-pushes that pin onto the signal, the two boxes collide on an otherwise empty row or column, forming a single adjacent pair with no third stopper left along their line (Figure 34(b1)). Even at unbounded strength, pushing an adjacent pair together with no third stopper hits the boundary (violating the backbone invariant), and no player can stand between two adjacent boxes to separate them. Only two pins (p_4 when $\varepsilon_2 = \varepsilon_1$, and p_6 when $\varepsilon_3 = \varepsilon_2$) can align with one earlier pin (p_2 or p_4) after a reverse-push; even then, at most one further horizontal slide is possible, stranding that pair in an empty column ($x_{i,1} + 1$, $x_{i,3} - 1$, or $x_{i,2} + 1$) while the signal stays pinned

with no third stopper (Figure 34(b2)). A box that a door emits onto a dead row $z' + \varepsilon_3$ fares the same way: the only other box on that row is the final pin p_6 of the wire feeding slot z' (if there is such a wire), and one more slide (the box onto p_6, or p_6 reverse-pushed east to $(c-1, z'+\varepsilon_3)$) leaves an adjacent pair with no third stopper on that row.



(a) Six-pin wire $(c, r) \to (c, z')$

(b1) 1-step dead end: no third stopper

(b2) 2-step dead end ($\varepsilon_2 = \varepsilon_1$): stranded in x_1+1 or x_3-1

Figure 34. Arbitrary six-pin wiring and wrong-push dead ends. (a) A wire from (c, r) to (c, z') uses six pins on private columns x_1, x_2, x_3 and private rows y_1, y_2; shaded vertical lanes connect every two-column band to the empty top row $y = 0$. (b1) Reverse-pushing p_3 onto the signal leaves an adjacent pair in empty column $x_1 + 1$ with no third stopper. (b2) Even in the two-step case ($\varepsilon_2 = \varepsilon_1$), reverse-pushing p_4 onto row $y_1 + \varepsilon_1$ allows at most one slide with p_2, stranding both in empty column $x_1 + 1$ or $x_3 - 1$ while the signal remains pinned.

Walkability. Along the canonical trajectory, every box lies in a pairwise separated two-column band (one for each wire column and one for the columns c and P), separated by completely empty columns that all reach the empty top row $y = 0$ (Figure 34(a)). Every empty cell lies either in an empty column or horizontally adjacent to one, so the player can walk from any empty cell to any other pushing cell in five straight segments via row $y = 0$, and at each turn the previous pin leaves the next pushing cell free (Appendix D.1).

Summarizing this step-by-step induction (complete six-case wrong-push check in Appendix D.2):

▶ **Lemma 19** (work-board decoding). *Let R be the work board, and let π be a PushPush-$*$ trajectory on R that satisfies the backbone invariant.*

1. *Before any wrong push or any arrival at the target parking cell (c, z), the trajectory simulates a valid walk in the source door network (up to harmless retries inside doors and traversal of binary choice trees): all wire pins remain in place, each door whose room does not contain the signal holds one box at α or β recording its state, and no box occupies (c, z).*
2. *After the first wrong push prior to reaching (c, z)—emitting onto a dead row, sliding into a capless end, or reverse-pushing any wire pin—no continuation obeying the backbone invariant can ever place a box at (c, z).*

In particular, if π places a box at (c, z), then the target location is reachable from the start location in the source door network.

6.2 Freezing the backbone without walls

Why do the rack in column P, the partitions Y_{part}, and the end-caps Y_{cap} stay fixed when every box on the board is movable? Unlike in Push-∗S, we need no distant hole-radius argument: maximal slides themselves make every displacement of P, Y_{part}, or Y_{cap} irreversible, so four simple checks on the *final* configuration freeze the entire backbone from start to finish (Figure 35).

▶ **Lemma 20** (backbone invariance). *Let R be the work board, and let π be a PushPush-∗ trajectory on R ending in a configuration B such that:*
(i) the outer frame of R is empty in B;
(ii) $(P, y) \in B$ for every rack row $y \in Y_P$;
(iii) $(c, g) \in B$ and $(c, g \pm 1) \notin B$ for every partition $g \in Y_{\text{part}}$; and
(iv) $(c, y) \in B$ for every end-cap row $y \in Y_{\text{cap}}$.
Then π satisfies the backbone invariant throughout: no box ever touches the frame; every cell of column P keeps its initial occupancy; every partition (c, g) ($g \in Y_{\text{part}}$) and cap (c, y) ($y \in Y_{\text{cap}}$) stays occupied; and every forbidden cell $(c, g \pm 1)$ ($g \in Y_{\text{part}}$) stays empty.

Proof. We establish the four claims in order (Figure 35).

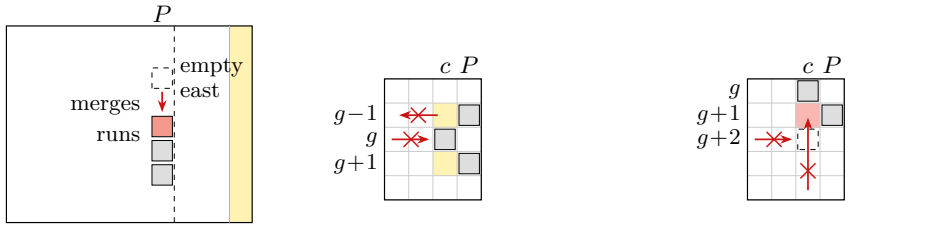
1. No boundary hits and empty east margin. Once a box lands on a boundary row or column of R, no push can move it inward: the player cannot stand outside R, and pushing along the boundary keeps the box on the boundary. Since the frame is empty in B by (i), no box ever touches the frame along π. Consequently, every slide stops against another box rather than the boundary, so every moved box lands strictly between its starting cell and a stationary box. The axis-aligned bounding box of all boxes therefore can only shrink; since all initial boxes lie in $x \le P$, no box ever enters $x > P$.

2. Column P is frozen. Column P cannot receive a box from the east (nothing lies in $x > P$) and cannot receive one from the west (stopping an eastward slide on column P would require a stopper at $x = P + 1$). Hence the number of boxes in column P is non-increasing along π. Initially column P holds boxes on Y_P, and (ii) requires boxes on all of Y_P at the end; thus column P keeps the exact same box count throughout, and every move that displaces a box on column P must be a vertical slide along column P. Any vertical maximal slide along column P stops against another box on P, strictly decreasing the number of maximal contiguous box runs in column P (Figure 35(a)). Because a push moves an entire maximal run, purely vertical slides can never split a run, so the run count on P is non-increasing. Since the initial and final sets of boxes on P are both Y_P, they have the same run count; therefore no box on column P ever moves, and every cell of P keeps its initial occupancy.

3. Partitions and forbidden neighbours. Fix $g \in Y_{\text{part}}$. By construction, $(P, g \pm 1)$ hold boxes and (P, g) is empty, and by Step 2 this holds throughout π. Call the pattern on column c in which (c, g) holds a box and $(c, g \pm 1)$ are empty an *isolated partition* (Figure 35(b)). We claim that no single push can create an isolated partition from any other occupancy of $\{(c, g-1), (c, g), (c, g+1)\}$. Indeed, no horizontal push can place a box into (c, g) (nothing lies in $x > P$, and (P, g) is empty so an eastward slide cannot stop at (c, g)), and no horizontal push can remove a box from $(c, g \pm 1)$ (eastward motion is blocked by the frozen box $(P, g \pm 1)$, while westward motion requires standing on or pushing $(P, g \pm 1)$). Consider a vertical push along column c after which (c, g) is occupied and $(c, g \pm 1)$ are empty. If a box of the moving run lands on (c, g), the cell ahead of it in the direction of motion holds the next box of the run or, for the front box, the box that stopped the slide; either way $(c, g-1)$ or $(c, g+1)$ is occupied afterwards, a contradiction. Otherwise the box at (c, g) did not move, so neither did its maximal run, which contains every box on $(c, g \pm 1)$; the three cells only gain boxes, so

the pattern was already an isolated partition. By (iii), every $g \in Y_{\text{part}}$ is an isolated partition at the end of π; backward induction over the pushes of π shows that (c, g) holds a box and $(c, g \pm 1)$ stay empty throughout π.

4. End-caps are frozen. Fix a cap row $y = g + 2 \in Y_{\text{cap}}$ (the case $y = g - 2$ is symmetric). Since $y \notin Y_P$, (P, y) is permanently empty by Step 2, so no horizontal slide can place a box into (c, y) (Figure 35(c)). If (c, y) were ever empty during π, refilling it by a southward slide would have to cross the permanently occupied partition (c, g) or start from the permanently empty cell $(c, g + 1)$, while refilling it by a northward slide would have to stop against or land on $(c, g + 1)$—all contradicting Step 3. Since $(c, y) \in B$ by (iv), (c, y) stays occupied throughout π. ◀



(a) Column P cannot gain boxes or split runs

(b) Partition (c, g) cannot be re-isolated

(c) Vacated cap $(c, g+2)$ cannot be refilled

Figure 35. The three mechanisms of Lemma 20. (a) Without boundary hits, $x > P$ stays empty; column P cannot gain boxes, and any vertical slide along P irreversibly merges two runs. (b) With $(P, g \pm 1)$ frozen and (P, g) empty, neither horizontal nor vertical slides can recreate an isolated partition at (c, g). (c) With $(P, g + 2)$ empty, (c, g) frozen, and $(c, g + 1)$ permanently empty (red), a vacated cap $(c, g + 2)$ can never be refilled.

Notice that Lemma 20 makes no assumption about *how* conditions (i)–(iv) are enforced at the end of π: exact storage enforces them directly, and the player-goal version enforces them via a run-read program.

6.3 Exact storage: PushPush-kS for every $k \geq 1$ and $k = *$

With an exact storage goal, conditions (i)–(iv) of Lemma 20 come for free as soon as the target set T leaves the frame and forbidden cells $(c, g \pm 1)$ empty and includes every rack box, partition, and end-cap. So why are we not already done?

The obstacle is that *an exact target must fix the final cell of every box before the puzzle begins.* When the signal reaches the target room, each door $i \in \{1, \ldots, d\}$ holds its stored box at α_i or β_i depending on the parity of traversals along the chosen path. No single target configuration matches every accepting run of the source door network.

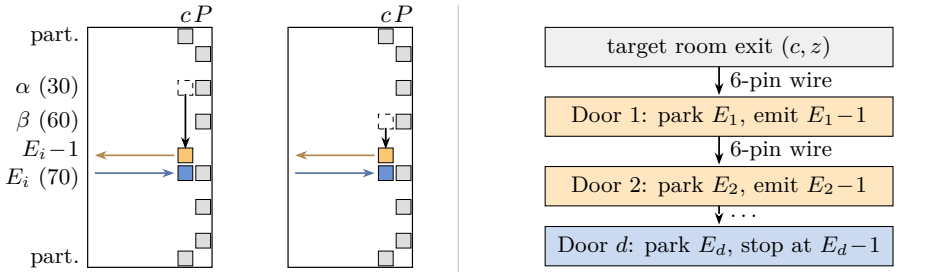
The door-by-door cleanup chain. We collapse all 2^d possible arrival states into one fixed configuration by appending a *cleanup chain* right after the target room (Figure 36). In each door room $i \in \{1, \ldots, d\}$, we add one new input slot E_i (backed by a rack stopper in P) with unique valid output row $E_i - 1$ directly to its north, placed strictly *south* of both stable slots α_i and β_i and spaced far enough from all six door slots and their ± 1 neighbours that (c, E_i) stays empty and never obstructs vertical slides during normal door traversals (exact offset in Appendix D.1). We then add d ordinary six-pin wires (re-allocating private columns and

rows across all wires together):

$$(c, z) \;\to\; (c, E_1), \qquad (c, E_i - 1) \;\to\; (c, E_{i+1}) \quad (1 \le i < d).$$

When the signal reaches (c, z) in the target room, every door i has its stored box at α_i or β_i. The signal leaves (c, z) along the first cleanup wire and arrives at (c, E_1). Because both α_1 and β_1 lie north of E_1, the player pushes the old stored box of door 1 *south*: whether it started at α_1 or β_1, it slides onto the box at E_1, stops at $(c, E_1 - 1)$, and (when $d > 1$) leaves west along the next cleanup wire to E_2, leaving the incoming box parked at (c, E_1) (Figure 36(a)). Repeating for $i = 1, \ldots, d$ parks one box at (c, E_i) in every door, and on door d the old stored box simply stops at $(c, E_d - 1)$ without leaving west (Figure 36(b)).

The resulting *storage board* is again a work board in the sense of Section 6.1: each E_i is one more rack row, and the cleanup wires are ordinary wires whose pins keep pairwise distinct coordinates and avoid every row in $Y_P \cup Y_{\text{part}} \cup Y_{\text{cap}}$; the only pin on the target parking row is p_1 of the first cleanup wire (Appendix D.1). Hence Lemma 20 applies to it, and Lemma 19 holds on it with "(c, z)" replaced by "(c, z) or a cleanup slot (c, E_i)" throughout: before the signal first reaches (c, z), no slide stops at the empty cells (c, E_i), and the wrong-push dead ends never reach them (Appendix D.2).



(a1) from state 0 (a2) from state 1 (b) Cleanup chain (2^d states → 1 target)

Figure 36. Collapsing run-dependent door states into one target. (a1)–(a2) In door i, slot E_i lies south of both α_i and β_i: whether the old stored box (orange) starts at α_i or β_i, one southward slide onto the arriving signal (blue) parks the arriving box at E_i and emits the old box west from $E_i - 1$. (b) Chaining the d doors from the target exit (c, z) collapses all 2^d arrival states to a single exact target.

We define the exact storage target T to be the union of:

- all rack boxes $\{(P, y) : y \in Y_P\}$, partitions $\{(c, g) : g \in Y_{\text{part}}\}$, caps $\{(c, y) : y \in Y_{\text{cap}}\}$, and wire pins;
- every door's cleanup slot (c, E_i) for $i = 1, \ldots, d$; and
- the final signal cell $(c, E_d - 1)$ (or (c, z) if $d = 0$).

These cells are pairwise distinct, $|T| = |B_0|$, and T depends only on the source instance, never on the path taken.

▶ **Theorem 21.** *For every fixed $k \ge 1$ and for $k = *$, PushPush-kS and PushPush-kFS are PSPACE-complete.*

Proof. Membership in PSPACE is proved in Section 3. Because an instance without walls is also a valid walled instance ($W = \emptyset$), it suffices to prove hardness for PushPush-kS ($k \ge 1$ and $k = *$). Given a door network of encoding length N, construct the storage work board R with target T; it has $O(N)$ rooms, wires, and boxes, width and height $O(N)$, and area $O(N^2)$, so it is built in polynomial time (Appendix D.1).

Completeness (using only single-box pushes). If the source door network has a path from start to target, follow it through the rooms and wires to park the signal at (c, z), then run the cleanup chain through doors $1, \ldots, d$, ending at $(c, E_d - 1)$ (or stopping at (c, z) when $d = 0$). Every pushing support cell is reachable via the empty vertical lanes and top row $y = 0$ (Appendix D.1). Every push along this canonical trajectory starts from a single box against empty space along its direction of motion (the second box it meets only stops the slide), so the trajectory is legal under PushPush-1S and therefore under PushPush-kS for every $k \geq 1$ and for $k = *$.

Soundness (at unbounded strength). Because increasing strength only adds legal moves, any solution under PushPush-kS ($k \geq 1$) is also a solution under PushPush-$*$S. Fix any PushPush-$*$S trajectory π reaching T. By definition of T, conditions (i)–(iv) of Lemma 20 hold at the end of π, so π obeys the backbone invariant throughout. When $d > 0$, T requires a box at (c, E_1); when $d = 0$, T requires a box at (c, z). In either case, Lemma 19 on the storage board yields a path from the start location to the target location in the source door network (we need not even analyse moves after the first arrival at (c, z), since the prefix up to that moment already witnesses reachability). ◀

6.4 Player goal: PushPush-$*$

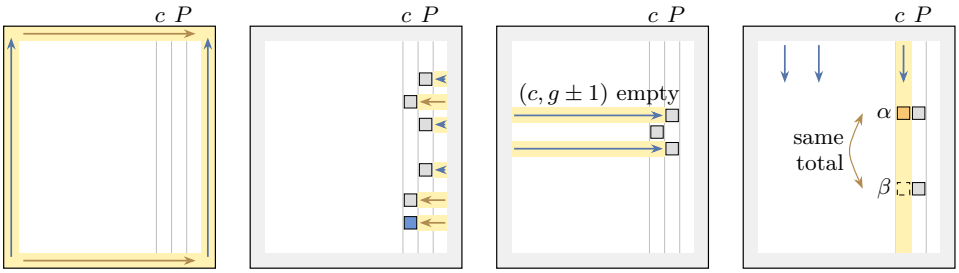
We now drop the storage goal and ask only for the player to reach a single goal cell. Without a storage target, we no longer need doors to end in fixed states, so we use the work board R of Section 6.1, without the cleanup slots E_i and cleanup wires. The challenge is the opposite one: how do we enforce conditions (i)–(iv) of Lemma 20 and arrival at (c, z) using only a player goal?

We invoke the PushPush-$*$ reader of Section 5.1 (Lemma 13), which compiles any fixed-length *full* program of run reads on R into a polynomial-size wall-free PushPush-$*$ instance. Recall that a run read deletes the shortest prefix of complete maximal hole runs on each line reaching its quota, and in a *full* program (whose quotas sum to the exact total number of holes $|R| - |B_0|$), every run read must pay its quota *exactly*: no cut may fall inside a run of holes, and no line may overshoot its quota.

The seven-round program. Let B_{can} be any canonical parked configuration on R: all rack boxes, partitions, caps, and wire pins sit in their initial cells, each door holds its stored box at α_i or β_i (for definiteness, take the initial door states), and the signal rests at the target parking cell (c, z). We define a seven-round program $\mathcal{P}_7$ of run reads cycling through the sides *bottom, left, top, right, bottom, left, top* (unlisted lines have quota 0; Figure 37):

1. *Round 1 (bottom):* quota h_0 on the left and right boundary columns 0 and $w_0 - 1$.
2. *Round 2 (left):* quota $w_0 - 2$ on the top and bottom boundary rows 0 and $h_0 - 1$.
3. *Round 3 (top):* all quotas 0 (an empty turn to rotate to the right side).
4. *Round 4 (right):* on every rack row $y \in Y_P$, quota $\mu = w_0 - 2 - P$; on every partition, cap, or target row $y \in Y_{\text{part}} \cup Y_{\text{cap}} \cup \{z\}$, quota $\mu + 1$.
5. *Round 5 (bottom):* all quotas 0 (an empty turn to rotate to the left side).
6. *Round 6 (left):* on every partition-neighbour row $g \pm 1$ ($g \in Y_{\text{part}}$), quota $P - 1$.
7. *Round 7 (top):* on every column x, quota equal to the number of holes of $R \setminus B_{\text{can}}$ still remaining in column x after Rounds 1–6.

Why the program is state-independent and certifies the backbone. Two facts make this seven-round program work (verified line by line in Appendix D.3):



(a) Rounds 1–2 empty frame

(b) Round 4 (right) μ on Y_P; $\mu{+}1$ on $Y_{\text{part}}, Y_{\text{cap}}, z$

(c) Round 6 (left) $P-1$ on $g \pm 1$

(d) Round 7 (top) state-independent

Figure 37. The seven-round full run-read program $\mathcal{P}_7$ on the work board R (Rounds 3 and 5 are empty turns). (a) Rounds 1–2 strip the four boundary lines, forcing an empty frame. (b) Round 4 reads μ from the right on Y_P-rows (forcing boxes on P) and $\mu + 1$ on $Y_{\text{part}} \cup Y_{\text{cap}} \cup \{z\}$ (forcing boxes on c). (c) Round 6 sweeps $P - 1$ from the left on $g \pm 1$ (forcing $(c, g \pm 1)$ empty). (d) Round 7 finishes every column; because Rounds 4 and 6 never touch (c, α_i) or (c, β_i), column c's remaining hole count is independent of the final door states.

- *State independence and fullness.* Changing a door's final state moves its stored box from (c, α_i) to (c, β_i) within the *same* column c. Both α_i and β_i lie in Y_P (disjoint from $Y_{\text{part}} \cup Y_{\text{cap}} \cup \{z\}$ and from $\{g \pm 1 : g \in Y_{\text{part}}\}$), so Round 4 reads only strictly east of column P on rows α_i, β_i and Round 6 skips them completely. Thus Rounds 1–6 delete the exact same set of cells on every canonical parked configuration, and the remaining hole count of column c in Round 7 is identical regardless of which doors end in state 0 or 1. Moreover, every positive quota in Rounds 4 and 6 stops directly against a box in B_{can} (while Rounds 1 and 2 strip full boundary lines to the board edges), so every canonical parked configuration passes all seven rounds with exact run cuts, and the quotas sum to $|R| - |B_0|$ (the program is full).
- *Exact certification of conditions (i)–(iv) and (c, z) along a trajectory.* Suppose a PushPush-$*$ trajectory π on R ends in a corner state $((w_0 - 1, h_0 - 1), B)$ whose hole set $R \setminus B$ is accepted by $\mathcal{P}_7$. We interleave the rounds with Steps 1 and 2 of Lemma 20:
  1. Rounds 1 and 2 exhaust the full capacities of all four boundary lines, forcing the frame empty in B (condition (i)). By Step 1 of Lemma 20 (which uses only condition (i)), no box ever touches the frame or enters $x > P$ along π, so all μ interior cells strictly east of P (columns $P + 1, \ldots, w_0 - 2$) are empty in B on every row.
  2. In Round 4, after Round 1 has stripped column $w_0 - 1$, for every $y \in Y_P$ the μ empty cells east of P form a rightmost hole run of length at least μ; if (P, y) were also empty, this run would have length at least $\mu + 1$, overshooting quota μ and violating fullness. Hence $(P, y) \in B$ for all $y \in Y_P$ (condition (ii)). By Step 2 of Lemma 20 (which uses only conditions (i) and (ii)), column P keeps its initial occupancy throughout π; in particular, $(P, y) \notin B$ for every $y \in Y_{\text{part}} \cup Y_{\text{cap}} \cup \{z\}$.
  3. Returning to Round 4 for $y \in Y_{\text{part}} \cup Y_{\text{cap}} \cup \{z\}$, all $\mu + 1$ interior cells east of c (columns $P, \ldots, w_0 - 2$) are empty in B; if (c, y) were also empty, the rightmost hole run on row y would have length at least $\mu + 2$, overshooting quota $\mu + 1$ and violating fullness. Hence $(c, y) \in B$ for every $y \in Y_{\text{part}} \cup Y_{\text{cap}} \cup \{z\}$, giving partition occupancy, condition (iv), and a parked signal at (c, z).
  4. Finally, in Round 6, for every $g \in Y_{\text{part}}$, row $g \pm 1$ lies in Y_P, so Round 4 already deleted columns $P + 1, \ldots, w_0 - 2$ and $(P, g \pm 1) \in B$, leaving zero holes at or east of P on row

$g \pm 1$. After Round 1 has stripped column 0, only the $P-1$ cells $\{1,\dots,c\} \times \{g \pm 1\}$ remain available; reading $P-1$ holes from the left forces every one of them—in particular $(c, g \pm 1)$—to be empty, completing condition (iii).

▶ **Theorem 22.** *PushPush-$*$ and PushPush-$*$F are PSPACE-complete.*

Proof. Membership in PSPACE is in Section 3, and PushPush-$*$F follows from PushPush-$*$ with $W = \emptyset$. Given a door network of encoding length N, build the work board R of Section 6.1, of area $O(N^2)$, compute the seven-round full run-read program $\mathcal{P}_7$ from a canonical parked configuration, and apply Lemma 13 to obtain a wall-free PushPush-$*$ instance with a player goal. Because the number of rounds is a fixed constant (7), Lemma 13 runs in polynomial time and produces a polynomial-size board.

If the door network has a path from start to target, simulate it on R to park the signal at (c, z) with every door's stored box at α_i or β_i, then walk via the empty lanes and boundary frame to the lower right corner $(w_0 - 1, h_0 - 1)$ of R (Appendix D.1). As shown above, this corner state is accepted by $\mathcal{P}_7$ regardless of the final door states, so Lemma 13 yields a solution to the PushPush-$*$ instance.

Conversely, if the PushPush-$*$ instance is solvable, Lemma 13 extracts a PushPush-$*$ trajectory π on R ending at a corner state whose configuration B is accepted by $\mathcal{P}_7$. By the interleaved certification argument above, B satisfies conditions (i)–(iv) of Lemma 20 and contains a box at (c, z), so Lemma 20 guarantees the backbone invariant throughout π and Lemma 19 decodes a valid path from the start location to the target location in the source door network. ◀

7 Push-kFS and Push-kS

Sections 4–6 settle unbounded strength and every PushPush storage variant. What remains for Theorem 2 are the finite-strength storage variants of Push: Push-kFS and Push-kS for every fixed $k \geq 1$. In fact, our walled construction goes further—one fixed instance works simultaneously for every finite $k \geq 1$ and for $k = *$.

At first sight, a reduction already proved for Push-1FS (Sokoban) might seem to carry over to higher strength, because every single-box move remains legal when the player can push up to k boxes. The obstacle is soundness: increasing strength strictly enlarges the player's moves, so a stronger player can shove a whole line of boxes through a corridor that was designed to jam under single pushes. Even knowing that both Push-1FS and Push-$*$FS are PSPACE-complete does not by itself settle intermediate k, because a stronger player might solve instances that had no Push-1FS solution.

We break this obstacle in two stages.

1. In Section 7.1, we isolate a static condition on a walled board and its storage target under which *every successful trajectory, even at unbounded strength, pushes only single boxes.* Realizing Hearn and Demaine's Sokoban reduction [11] so that this condition holds yields a single k-independent reduction proving Push-kFS PSPACE-complete for every $k \geq 1$ and for $k = *$.
2. In Section 7.2, we remove the fixed walls at every finite strength k by replacing them with $w \times w$ frozen blocks of boxes ($w = k + 1$). Thickening the walls stretches the tunnels inside every gadget, breaking the width-2 soundness of [11]; we redesign every gadget for width w and prove soundness directly from one enclosed-cavity trap together with the same single-push lemma.

Both reductions start from configuration-to-configuration reachability on a 3-connected simple planar AND/OR constraint graph G with satisfying orientations σ, τ, which Hearn

and Demaine proved PSPACE-complete [11] (Corollary 6 and Theorems 9 and 12). Every vertex has degree 3; each AND vertex has two light edges (weight 1) and one heavy edge (weight 2), each OR vertex has three heavy edges, and every vertex requires total incoming weight at least 2. Coordinate tables, single-box macros, and global routing details are collected in Appendices E and F.

7.1 One walled construction for every strength: Push-kFS

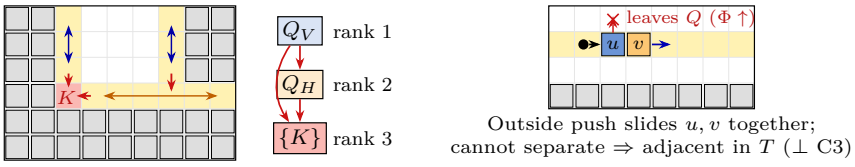
Why can a storage target strip an unbounded-strength player of multi-box pushes? Watch what happens whenever a box moves from u to $u + d$ in direction d: the cell $u + d$ in front must be a non-wall cell, and the cell $u - d$ behind must also be a non-wall cell—either it holds the player, or (for an inner box of a multi-box push) it holds the preceding box of the pushed line. Define the *static move graph* D_W on the non-wall cells $R \setminus W$ by placing a directed arc $u \to u + d$ whenever $u - d, u, u + d \in R \setminus W$, for each unit cardinal direction d. Ignoring other boxes and player reachability, *every pushed box moves along an arc of* D_W, including every inner box of a multi-box push.

Let $Q_1, Q_2, \ldots$ be the strongly connected components (SCCs) of D_W. Consider three static properties of an instance with initial box set B_0 and target set T:

(C1) For every component Q_i, $|B_0 \cap Q_i| = |T \cap Q_i|$.
(C2) Every component Q_i with $B_0 \cap Q_i \neq \emptyset$ lies in a single row or a single column.
(C3) No component Q_i contains two orthogonally adjacent targets in T.

A *single push* is a push of one box.

▶ **Lemma 23** (SCC transfer). *If a walled instance satisfies* (C1)–(C3)*, then along every successful Push-∗FS trajectory:*
(i) no box ever leaves its initial SCC of D_W*;*
(ii) no two boxes in the same SCC are ever adjacent; and
(iii) every push is a single push.
Consequently, the instance has the same solvability under Push-kFS for every $k \geq 1$ *and under Push-∗FS.*



(a) Move graph D_W and topological ranks (b) Glued pair in a straight SCC

Figure 38. The SCC transfer lemma (Lemma 23; a copy of Figure 6). (a) In the static move graph D_W, straight tracks form collinear bidirectional SCCs (blue vertical, orange horizontal); cross-track arcs (red) go strictly down the condensation DAG, so any box leaving its SCC raises the potential Φ. (b) Once boxes remain inside a straight SCC Q, two adjacent boxes $u, v \in Q$ can never separate: pushing either out of Q raises Φ, pushing one away along Q requires standing on the other, and pushing from outside slides both together, forcing adjacent targets in $T \cap Q$ against (C3).

Proof. Order the SCCs of D_W topologically by assigning an integer rank $\mathrm{rk}(Q)$ that increases strictly along every arc between distinct components, and define the potential $\Phi(B) = \sum_{u \in B} \mathrm{rk}(Q(u))$, where $Q(u)$ is the SCC of u (Figure 38(a)). When a push moves boxes along arcs of D_W, no box decreases its rank, and any box crossing into a new SCC increases

Φ strictly. By (C1), $\Phi(B_0) = \sum_i |B_0 \cap Q_i| \operatorname{rk}(Q_i) = \sum_i |T \cap Q_i| \operatorname{rk}(Q_i) = \Phi(T)$, so no box ever changes SCC; this proves (i).

For (ii), suppose two boxes occupy adjacent cells $u, u + d$ in the same component Q. By (i), Q contains initial boxes, so by (C2) it lies in one row or column, and both boxes must remain in Q until the end (Figure 38(b)). Any push perpendicular to Q moves a box out of Q, contradicting (i). Along Q, pushing just the leading box forward requires standing on the trailing box, and pushing from behind moves both boxes together as a contiguous line. Thus the two boxes remain adjacent all the way to T, giving two adjacent targets in $T \cap Q$ and contradicting (C3).

For (iii), suppose a push moves at least two boxes, and let u and $u + d$ be the cells of the two boxes closest to the player. The push moves the first box from u to $u + d$, so by (i) both cells lie in the same SCC Q. Before the push, u and $u + d$ already held adjacent boxes in Q, contradicting (ii).

Finally, every Push-1FS solution is a Push-kFS solution and every Push-kFS solution is a Push-$*$FS solution, while by (iii) every Push-$*$FS solution uses only single pushes and is therefore a Push-1FS solution; so solvability is the same at all strengths. ◀

Notice what Lemma 23 does *not* claim: wrong multi-box pushes may still be legal on the board, and the reachable state graphs at different strengths differ—it is only the *successful* trajectories that collapse to single pushes. Notice also a useful bonus of (i)–(ii) that we will need in Section 7.2: on any straight SCC Q, boxes cannot pass one another or become adjacent, so they preserve their linear order along Q at distance at least 2 and match the targets in $T \cap Q$ in that exact order, even though boxes are unlabelled.

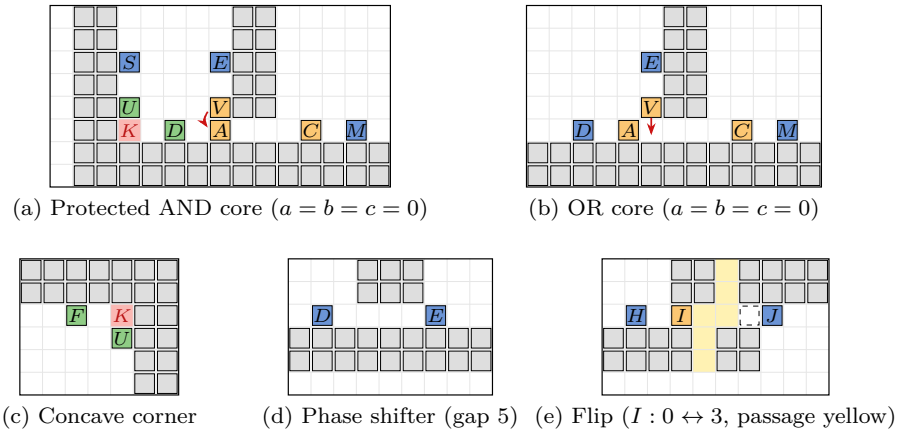
Adapting Hearn and Demaine's Sokoban gadgets to (C1)–(C3). Can we build a walled board from Hearn and Demaine's Sokoban reduction [11] (Section 4.3, Figure 15) that obeys (C1)–(C3)? In [11], every gadget is a narrow track flanked by 2×2 blocks of immovable boxes; we first replace those immovable blocks by real fixed walls of thickness 2 (Figure 39; exact coordinates in Appendix E.1). A straight track runs alongside a 2-thick wall strip with boxes spaced by 2; shifting every box along the track by one cell flips the edge's orientation between phase 0 and phase 1. Three routing utilities adapt tracks in the plane:

- a *concave corner* turns a track by 90° around an empty dead corner cell K, which has incoming arcs from both arms and no outgoing arc;
- a *phase shifter* places a 3×2 roof over the track to create an odd gap of 5 between boxes D and E, repairing parity mismatches along an edge; and
- a *flip* swaps which side of the track holds the wall using two 2×2 wall islands, sliding its middle box I by three cells ($0 \leftrightarrow 3$) while leaving a 1-cell-wide crossing passage through the walls open for the player to cross the track in either normal phase.

The AND and OR cores join a vertical input track (V, E) perpendicularly to a horizontal track carrying input A (with neighbour D) on the left and output C (with neighbour M) on the right. As required in [11], every AND core attaches a protective concave corner directly to its left input D so that D cannot back up without entering the dead cell K; this bends the A input upward only four columns west of V.

Three design issues arise when we try to enforce (C1)–(C3) globally:

1. *Avoiding cross-track cycles for (C2).* Inside every straight track, phase shifter, and flip, each track cell borders a wall on at least one side, so no box has an arc leaving its line. A concave corner only sends boxes into the sink cell K, which has no outgoing arc. Thus the *only* cross-track arcs between live tracks enter the horizontal track from the vertical V-track inside AND and OR. If we rotated one logic gate by 90° relative to another, a



(a) Protected AND core ($a = b = c = 0$) (b) OR core ($a = b = c = 0$)

(c) Concave corner (d) Phase shifter (gap 5) (e) Flip ($I : 0 \leftrightarrow 3$, passage yellow)

Figure 39. The width-2 walled gadgets of Section 7.1 in normal phase 0 (all coordinates in Appendix E.1). Fixed walls are gray, the gate boxes A, V, C and the flip middle box I orange, corner boxes green, ordinary line boxes blue, dead corners K pale red, and the flip crossing passage pale yellow. The only non-dead cross-track arcs run from the vertical V-track into the horizontal A/C-track (red arrows); keeping every gate in this single orientation makes all cross-track arcs run strictly vertical to horizontal.

vertical-to-horizontal arc in one gate and a horizontal-to-vertical arc in another could link perpendicular tracks into a single non-collinear SCC, breaking (C2). We therefore place *every AND and OR core in the exact same orientation* (only translating them, never rotating or reflecting), while allowing the routing utilities to rotate and reflect freely. Every live cross-track arc now points strictly from a global vertical track to a global horizontal track, so no cycle can mix directions and every box-bearing SCC is collinear.

2. *Separating AND's narrow inputs and embedding in the plane.* Because AND's protective corner places the upward A-track only four columns west of V, we cannot assume arbitrary free space around them. We instead attach a fixed finite fan-out template to every AND core (Appendix E.1, Figure 48(a)): a vertical flip whose detour clears V, followed by a concave corner turning A westward. Every AND and OR core now exposes three well-separated straight tails facing west, north, and east. Starting from an orthogonal grid drawing of the planar cubic graph G [15] (rank-compressed so coordinates and area are polynomial, then scaled by a constant spacing), we place each core in its vertex box, permute the symmetric ports (the two light ports of AND, or all three ports of OR) to match the drawing's cyclic order without rotating the gate, and wire the three tails inside the vertex box using constant-size orthogonal templates. Since the player can cross the wall strip of an edge only through the passage of a flip, we also place an extra pair of flips on every edge: each flip opens a crossing, and two flips leave the wall side unchanged. As the planar dual is connected, the player can then travel between all faces.
3. *Exact targets for every box to satisfy (C1) and (C3).* Rather than leaving auxiliary boxes unconstrained, every NCL orientation ρ uniquely determines a full normal box configuration B_ρ (every edge box, flip box I, corner box, and gate box): each port shifts by one cell outward on inflow, a phase shifter on each straight segment whose endpoint reference-slot distance is odd repairs parity so every straight splice has spacing 2, and every box moves between its phase-0 and phase-1 slots inside its own bidirectional track SCC (Appendix E.1). Pairing initial and target boxes of the same track slot immediately gives (C1), and every intra-track gap in B_τ is at least 2 (with K always

empty), giving (C3).

Putting $B_0 = B_\sigma$ and $T = B_\tau$ with the player in an outer empty margin gives a polynomial-size walled board $\mathcal{R}(G, \sigma, \tau)$ completely independent of k.

▶ **Theorem 24.** *There is a single polynomial-time reduction $\mathcal{R}$ from planar AND/OR NCL such that for every instance $\mathcal{N} = (G, \sigma, \tau)$,*

$$\begin{aligned} \mathcal{N} \text{ is solvable in NCL} &\iff \mathcal{R}(\mathcal{N}) \text{ is solvable in Push-}k\text{FS} \quad (\text{for every } k \geq 1) \\ &\iff \mathcal{R}(\mathcal{N}) \text{ is solvable in Push-}*\text{ FS}. \end{aligned}$$

In particular, Push-kFS is PSPACE-complete for every fixed $k \geq 1$ and for $k = *$.

Proof. Membership in PSPACE is proved in Section 3.

Completeness (single pushes). Suppose an NCL move reverses an edge e from $v \to u$ to $u \to v$, so u loses inflow and v gains inflow. We update the boxes of e by single-box macros in three steps: execute the local macro at u (which keeps all other edges in place), sweep the utilities and straight-line boxes along e from u toward v (moving the leading box ahead of the trailing box on each straight segment, and executing each flip atomically so all crossing passages stay open between macros), and finish with the local macro at v (full macros and pushing-cell checks in Appendix E.2). Every auxiliary box lands in its exact slot of the new normal configuration, so by induction on the NCL path, $\mathcal{R}(\mathcal{N})$ has a Push-1FS trajectory from B_σ to B_τ.

Soundness (at unbounded strength). Suppose $\mathcal{R}(\mathcal{N})$ has a Push-$*$FS trajectory from B_σ to B_τ. Because $\mathcal{R}(\mathcal{N})$ satisfies (C1)–(C3) (checked above and in Appendix E.1), Lemma 23 implies that this trajectory uses only single pushes. We decode it as on the wall-free board of Section 7.2. For $w = 2$, the line, corner, phase shifter, flip, and OR core of Section 7.2 coincide cell for cell with ours, and our protected AND core is the AND core of Section 7.2 with $w = 2$ and protective-chain length $L = 0$ (so $F_1 = D$); only the AND fan-out differs, and it is built from the same $w = 2$ lines, flip, and corner. Lemma 25 covers $\mathcal{R}(\mathcal{N})$, and the decoding that follows it uses only Lemmas 23 and 25 and the outer port boxes (located on $\mathcal{R}(\mathcal{N})$ in Appendix E.1), so it yields an NCL path from σ to τ.

By Lemma 23, solvability under Push-1FS, Push-kFS (every $k \geq 1$), and Push-$*$FS all coincide on $\mathcal{R}(\mathcal{N})$. ◀

Push-$*$FS was already established as a stepping stone in Section 4; Theorem 24 settles all finite strengths $k \geq 1$ simultaneously on a single strength-independent board.

7.2 Removing walls at finite strength: Push-kS

We now remove the fixed walls for a fixed finite strength $k \geq 1$ and set $w = k + 1 \geq 2$. Why can we not simply turn the thickness-2 walls of Section 7.1 into boxes with targets? Because a 2×2 block freezes boxes only when $k = 1$; at strength $k \geq 2$, the player can push an entire row of two boxes right through the wall. We therefore design a walled board with thick walls W, and in the wall-free instance we place a box on every cell of W; call this set of boxes B_W. We use the following *freezing rule* [9]: if *every cell of W belongs to a $w \times w$ square contained in W*, then no box of B_W ever moves at strength k, because the very first push moving a box of B_W would start from outside such a square (which is still completely filled) and therefore push an entire row or column of at least $w = k + 1$ boxes, exceeding the strength bound k. So the boxes of B_W act exactly as the walls W, and we may apply Lemma 23 on the walled board to the remaining *working boxes* alone.

Making every wall a union of $\geq w \times w$ rectangles, however, creates two real difficulties:

1. *Unit boxes in thick tunnels.* We cannot scale the whole board by w, because the working boxes remain 1×1. Thickening the roofs of the phase shifter, flip, AND, and OR to width w stretches their internal tunnels to length $\Theta(w)$, and widens the flip islands so that AND's narrow 4-column fan-out would collide with the V-track.
2. *Non-canonical intermediate states.* Inside a tunnel of length w, a working box can stop at intermediate cells that do not exist when $w = 2$. So we prove soundness directly for arbitrary Push-kS trajectories on our redesigned gadgets; for $w = 2$ the same proof also gives the soundness of Section 7.1.

One mechanism for direct soundness: the enclosed cavity. Both the redesign and its soundness proof are driven by a single observation, which relies on the working-box invariants already guaranteed by Lemma 23 (every working box stays in its straight SCC, and no two boxes in the same SCC are ever adjacent):

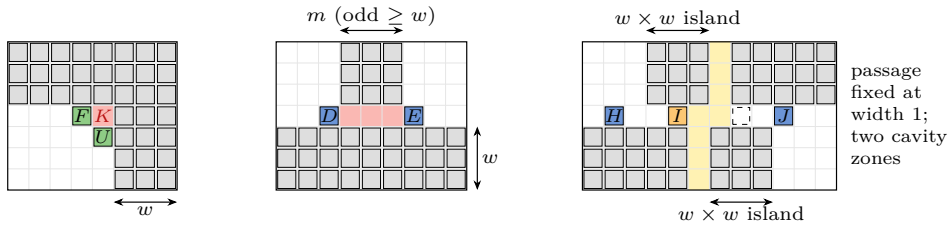
> *Enclosed-cavity trap.* Suppose two consecutive working boxes X, Y on a straight track enclose a non-empty interval of empty cells, the player is outside the interval, and every cell of the interval is flanked on both sides of the track by a frozen wall or by a working box Z whose SCC is perpendicular to the track and does not contain that cell, so that Z cannot move into the interval by Lemma 23(i). Then along any successful trajectory, the player can never enter the interval, and neither X, Y, nor any such Z moves away from it: each of these pushes requires standing inside the interval, pushing X or Y toward the other only shrinks the interval while keeping the player outside, and X and Y can never become adjacent (Lemma 23(ii)) to crush the interval away.

Whenever a target configuration forbids such a closed pair, the state is unrecoverable—provided we check that the *first* move forming the bad state is a push from outside, leaving the player outside the newly sealed cavity.

Parameterized gadgets of width w. We can arrange every gadget of Section 7.1 using unions of $\geq w \times w$ wall rectangles so that all tracks, ports, and enclosed cavities behave uniformly for every $w = k + 1$ (Figures 40 and 41; exact coordinates in Appendix F.1):

- *Straight line and thick concave corner* (Figure 40(a)): widen the wall strips to thickness w while keeping the unit track and dead corner K unchanged.
- *Thick phase shifter* (Figure 40(b)): let m be the smallest odd integer with $m \geq \max(3, w)$, and place an $m \times w$ roof over the track above a w-thick bottom wall, creating an odd gap $m + 2$ between D and E.
- *Thick flip* (Figure 40(c)): the crossing passage runs through the lower wall in column $x = 1$ and through the upper wall in column $x = 2$, crossing the track at $(1, 0)$ and $(2, 0)$; we keep these two columns adjacent and grow both wall islands away from them to $w \times w$ squares. So the middle box I still slides only three cells between its two phases ($x = 0$ and $x = 3$), leaving the crossing passage open in both normal phases and creating two walled cavity zones flanking the crossing.

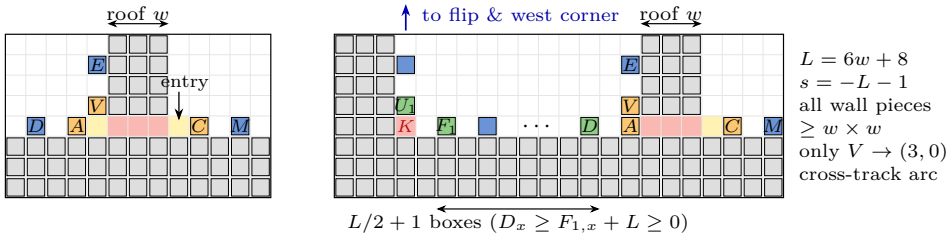
- *Thick OR core* (Figure 41(a)): thicken both the bottom wall and the right wall/roof to width w; this lengthens the right entry tunnel to length w while keeping the A/V junction identical to Section 7.1.



(a) Corner (K sealed) (b) Phase shifter (cavity) (c) Thick flip ($q = 0$; $H \Rightarrow I \Rightarrow J$)

Figure 40. Parameterized width-w routing utilities for Push-kS, illustrated at $w = 3$ ($k = 2$), where every wall cell is a frozen ordinary box (gray) belonging to a $w \times w$ square. (a) Thick concave corner in the bad state sealing dead cell K (red). (b) Thick phase shifter ($m = 3$) in the bad state sealing the cavity under the $m \times w$ roof (red). (c) Thick flip in normal phase $q = 0$: the 1-cell-wide crossing passage (yellow, length $2w + 2$) stays open while the two islands grow outward to $w \times w$.

- *Thick AND core with distant protective corner* (Figure 41(b)): thicken the bottom wall and right roof to width w. To protect D from moving left without crowding V when the flip on A has width $\Theta(w)$, we move the protective concave corner L cells to the left, so that its dead cell is $K = (-L - 1, 0)$ for an even $L \geq 0$, and fill the straight horizontal track from the corner box F_1 to D with $L/2 + 1$ boxes at spacing 2 (F_1, then ordinary line boxes, then D) and *no utilities in between*. Why does moving the corner far away cost nothing? Because K is a dead cell ($F_{1,x} \geq -L$), and by Lemma 23 the $L/2+1$ boxes from F_1 to D keep their linear order at distance at least 2, giving the same lower bounds $D_x \geq 0$ and $A_x \geq 2$ as when the corner sits adjacent to D (the case $L = 0$, $F_1 = D$, which at $w = 2$ is the protected AND core of Section 7.1). For Push-kS we take $L = 6w + 8$ (Appendix F.1); then the vertical A-branch sits more than $5w$ columns west of V, leaving room for a thick flip and an upper corner turning A westward (Figure 48(b)).



(a) Thick OR core ($w = 3$) (b) Thick AND core ($w = 3$, chain compressed)

Figure 41. Parameterized width-w OR and AND cores ($w = 3$, $a = b = c = 0$). (a) Thick OR core: the w-wide roof creates a tunnel over $[5, w + 4] \times \{0\}$ (red) with right entry at $(w + 5, 0)$ when $c = 0$. (b) Thick AND core: moving the protective corner to $K = (-L - 1, 0)$ ($L = 6w + 8$) leaves $> 5w$ columns clear between the A and V vertical branches while the $L/2+1$ boxes from F_1 to D preserve the bound $D_x \geq 0$ by spacing ≥ 2. The drawing shortens this chain, omitting the boxes at $\cdots$.

Every wall piece in these cores and utilities is a union of rectangles of size at least $w \times w$, so by the freezing rule every box of B_W is frozen at strength k. Keeping all AND and OR cores in this single orientation, assembling the planar orthogonal drawing [15] with constant-size fixed-core routing templates (without scaling the unit-grid cores; Appendix F.2), adding a pair of extra flips on every edge, fixing parity with at most one thick phase shifter on each straight segment, and surrounding the bounding box by an outer empty margin of

width $r = 100(w + 3)$ holding the initial player p_0 produces a polynomial-size wall-free board $\mathcal{R}_k(G, \sigma, \tau)$ whose working boxes satisfy (C1)–(C3) (Appendix F.1). By Lemma 23, in any successful Push-kS trajectory every working box stays in its straight SCC, preserves its linear order at distance ≥ 2, matches its exact named target at the end, and moves only by single pushes.

Threshold contracts and two-tier cavity exclusion. Rather than assuming boxes remain near canonical slots, we state a simple *threshold contract* for every gadget along any successful trajectory. On an oriented edge segment with unit tangent t, say a box X with phase-0 reference slot $\bar{X}$ is *high* if $\langle X - \bar{X}, t\rangle \geq 1$.

▶ **Lemma 25** (Gadget threshold contracts). *For any planar AND/OR NCL instance $\mathcal{N}$, along any successful Push-kS trajectory on $\mathcal{R}_k(\mathcal{N})$, and for $w = 2$ along any successful Push-$*$FS trajectory on the walled board $\mathcal{R}(\mathcal{N})$ of Section 7.1:*

1. *Every straight line, thick concave corner, thick phase shifter, and thick flip propagates the high threshold forward from its entry box to its exit box (and reversed along a path, propagates the contrapositive).*
2. *In every thick AND core, for every even $L \geq 0$, defining $a^* = [A_x \leq 2]$, $b^* = [V_y \leq -2]$, and $c^* = [C_x \leq w+4]$ (in the local core coordinates of Appendix F.1, where $A = (3-a, 0)$, $V = (3, -1-b)$, and $C = (w+5-c, 0)$ do not depend on L), we have $c^* \Rightarrow a^* \wedge b^*$.*
3. *In every thick OR core, defining $a^* = [A_x \leq 2]$, $b^* = [V_y \leq -2]$, and $c^* = [C_x \leq w+5]$, we have $c^* \Rightarrow a^* \vee b^*$.*

Proof. All claims use the working-box invariants of Lemma 23 (order preservation, spacing ≥ 2, exact named targets, and no box ever entering a dead cell K or crossing tracks).

1. Routing utilities. On a straight line with reference spacing 2, box spacing ≥ 2 makes the offset $\langle X_j - \bar{X}_j, t\rangle$ non-decreasing in j. In a concave corner, if F is high and U is not high, neither box can enter K, so F and U sit directly adjacent to K (Figure 40(a)); the first move forming this state pushes F or U inward from outside K, sealing K forever because pushing either box back requires standing at K, whereas no normal target seals K. In a phase shifter, if D is high and E is not high (Figure 40(b)), the open interval between D and E lies completely under the $m \times w$ roof; the first move entering this state pushes D or E inward from outside, forming an enclosed cavity that no normal target contains. A thick flip applies the enclosed-cavity trap twice in cascade:

$$H \text{ high} \implies I \text{ past the left island} \implies J \text{ high}.$$

Violating the first implication seals a non-empty cavity between the bottom wall and the left upper island; violating the second seals a cavity between the right lower island and the upper wall (Figure 40(c)). In each case the first move forming the bad state leaves the player outside, and neither normal phase lies in either bad set. When a utility is traversed in reverse, its normal phase-1 slot becomes the new phase-0 slot with reversed tangent, so the new high predicate is the negation of the old one and the contrapositive gives the reversed contract.

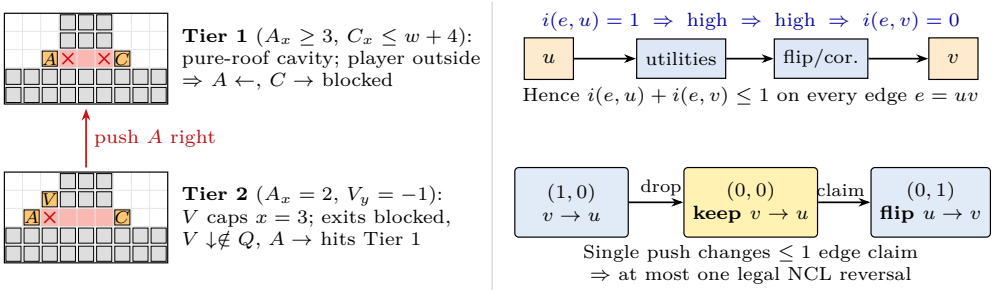
2. Thick AND core (two-tier cavity exclusion; Figure 42(a)). Recall that the protective chain gives $A_x \geq 2$, and the vertical SCC of V gives $V_y \leq -1$. We exclude two bad sets in order:

- *Tier 1:* $A_x \geq 3$ and $C_x \leq w + 4$. Here the open interval between A and C lies completely under the w-wide AND roof, walled above and below. The first push entering Tier 1 pushes A right or C left from outside, sealing an enclosed cavity; no legal normal target of AND lies in Tier 1, so Tier 1 never occurs.

- *Tier 2:* $A_x = 2$, $V_y = -1$, and $C_x \le w + 4$. Here V caps the single gap column left of the roof from above, while the remaining columns between A and C are capped by the roof; together with A, C, and the bottom wall, they enclose the open interval between A and C (the trap with $Z = V$). Because Tier 1 never occurs, the first move into Tier 2 cannot come from pushing A left from $x = 3$; it must push V down or C left, leaving the player outside the cavity. From inside Tier 2, pushing A left, V up, or C right all require standing inside the cavity; pushing V down leaves V's SCC; and pushing A right drops immediately into Tier 1 (already excluded). Since no legal AND target lies in Tier 2, Tier 2 never occurs either.

Therefore c^* ($C_x \le w + 4$) forces $A_x = 2$ (so $a^* = 1$) and $V_y \le -2$ (so $b^* = 1$).

3. Thick OR core. The exact same two-tier argument applies with the roof shifted one cell right: Tier 1 ($A_x \ge 4$, $C_x \le w + 5$) is a pure-roof enclosed cavity, and once Tier 1 is excluded, Tier 2 ($A_x = 3$, $V_y = -1$, $C_x \le w + 5$) can first be entered only by pushing A right from 2 to 3, V down, or C left (all from outside), and any escape either requires standing inside the cavity, leaves V's SCC, or pushes A right into Tier 1. Hence c^* forces $A_x \le 2$ or $V_y \le -2$, i.e., $a^* \vee b^* = 1$. ◀



(a) Two-tier enclosed-cavity exclusion (AND)

(b) Edge threshold contract & completion

Figure 42. Direct soundness for Push-kS. (a) Two-tier cavity exclusion on thick AND: Tier 1 is a pure-roof enclosed cavity; once Tier 1 is ruled out, Tier 2 (capped by V at $x = 3$) can only be entered from outside, and pushing A right drops straight into Tier 1, making their union unrecoverable. (b) Threshold claims compose along each edge $e = uv$ to enforce $i(e,u) + i(e,v) \le 1$, permitting asynchronous $(0,0)$ intermediate states; keeping unclaimed edges fixed and flipping only on new claims extracts a legal NCL trajectory.

Decoding asynchronous trajectories via partial-orientation completion. During an arbitrary successful Push-kS trajectory, boxes along an edge need not move in lockstep: a box near one endpoint can pull back long before the far endpoint advances. To decode such asynchronous trajectories, we read each vertex's inflow claims at its three *outer port boxes* (where each edge meets the vertex box): for an edge e incident to u with outward unit vector d_u and non-inflow slot $X_{\mathrm{out},u}$, set the *inflow claim*

$$i(e,u) \;=\; \big[\langle X_{e,u} - X_{\mathrm{out},u},\, d_u\rangle \ge 1\big] \;\in\; \{0,1\}.$$

At any state of a successful trajectory, two properties hold simultaneously:

- *Every vertex is satisfied by its inflow claims alone.* For AND, if $i(e_C, u) = 0$, propagating inward along the C-branch by the contrapositive of Part 1 of Lemma 25 gives $M_x \le w + 6$; spacing ≥ 2 gives $C_x \le w + 4$, so Part 2 of Lemma 25 gives $A_x = 2$ and $V_y \le -2$; spacing ≥ 2 then forces $D_x = 0$, $F_{1,x} = -L$ (via the protective chain), $U_{1,y} \le -2$ (via

the protective corner), and $E_y \le -4$. Propagating these near-core thresholds outward along the A- and V-branches via Part 1 of Lemma 25 yields $i(e_A, u) = i(e_V, u) = 1$, so $i(e_C, u) = 0 \Rightarrow i(e_A, u) = i(e_V, u) = 1$. For OR, if $i(e_C, u) = 0$, the same inward propagation gives $M_x \le w + 7$ and $C_x \le w + 5$, so Part 3 gives $A_x \le 2$ or $V_y \le -2$; spacing ≥ 2 turns this into $D_x \le 0$ or $E_y \le -4$, and propagating outward along the A- or V-branch gives $i(e_A, u) \vee i(e_V, u) = 1$ (branch details in Appendix F.2).

- *No edge claims inflow at both ends.* Orienting edge $e = uv$ from u to v, the claim $i(e, u) = 1$ is the high predicate at the start of the edge, and $i(e, v) = 1$ is the negation of the high predicate at the end of the edge. Composing Part 1 of Lemma 25 along all utilities of e gives $i(e, u) = 1 \Rightarrow i(e, v) = 0$, so $i(e, u) + i(e, v) \le 1$ (Figure 42(b)).

An edge may temporarily have $i(e, u) = i(e, v) = 0$, forming a *partial orientation* of G. We complete it to a valid NCL trajectory greedily: maintain a full orientation ρ agreeing with every active claim (orienting any unclaimed edge arbitrarily still satisfies every vertex, because adding inflow cannot break a lower bound). Whenever a single push occurs along the Push-kS trajectory, it moves at most one outer port box, changing the claim of at most one edge e: if a claim disappears, keep ρ unchanged; if a new claim $i(e, u) = 1$ appears and ρ points away from u, flip e toward u. Both before and after that single flip, ρ extends a valid partial orientation and is therefore a satisfying NCL orientation. Since every edge has exactly one active claim in B_σ and in B_τ (by exact order-preserving target matching), the resulting sequence of single-edge reversals runs from σ to τ.

▶ **Theorem 26.** *For every fixed $k \ge 1$, Push-kS is PSPACE-complete.*

Proof. Membership in PSPACE is in Section 3. Fix $k \ge 1$ and $w = k + 1$. Given a simple, 3-connected planar AND/OR NCL instance (G, σ, τ), build the wall-free board $\mathcal{R}_k(G, \sigma, \tau)$ with initial configuration $B_W \cup B_\sigma$, target $B_W \cup B_\tau$, and initial player p_0 in the outer empty margin; since w and all core/routing constants are fixed, the board has polynomial size and is built in polynomial time (Appendix F.2).

Completeness. Every single-edge NCL reversal from $v \to u$ to $u \to v$ is simulated by single-box pushes (legal for all $k \ge 1$) that never touch B_W: run the lose-inflow macro at u, propagate along e utility by utility to v, and run the gain-inflow macro at v (Appendix F.1). Thickening the AND/OR roofs only lengthens the empty entry tunnel from $(w + 4, 0)$ (AND) or $(w + 5, 0)$ (OR) without blocking any pushing cell. As in Section 7.1, the extra pair of flips on every edge lets the player cross every edge between macros, and during a flip macro the player stays on one side of that flip, so its blocked passage ($I \in \{1, 2\}$) is never needed (Appendix F.2). Every auxiliary box returns to its exact slot in $B_{\rho'}$, so an NCL path from σ to τ yields a Push-kS solution.

Soundness. In any Push-kS trajectory from $B_W \cup B_\sigma$ to $B_W \cup B_\tau$, every cell of W lies in a $w \times w$ square in W, so by the freezing rule no box of B_W ever moves. Turning B_W back into the walls W gives a walled board satisfying (C1)–(C3), so Lemma 23 guarantees single pushes, straight-SCC containment, order preservation with spacing ≥ 2, and exact named target matching for all working boxes. Lemma 25 and the partial-orientation completion above then extract a valid NCL trajectory from σ to τ. ◀

The freezing rule requires $w = k + 1 > k$ and therefore fails at unbounded strength $k = *$, which is why Push-$*$S needed the wall-free train construction of Section 4. Together, Theorems 7, 16, 21, 22, 24, and 26 prove Theorems 1 and 2 and complete Table 1.

References

1 Hayashi Ani, Jeffrey Bosboom, Erik D. Demaine, Jenny Diomidova, Della Hendrickson, and Jayson Lynch. Walking through doors is hard, even without staircases: Proving PSPACE-hardness via planar assemblies of door gadgets. In *10th International Conference on Fun with Algorithms (FUN 2021)*, volume 157 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 3:1–3:23. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.FUN.2021.3.

2 Hayashi Ani, Lily Chung, Erik D. Demaine, Jenny Diomidova, Della Hendrickson, and Jayson Lynch. Pushing blocks via checkable gadgets: PSPACE-completeness of Push-1F and Block/Box Dude. In *11th International Conference on Fun with Algorithms (FUN 2022)*, volume 226 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 3:1–3:30. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.FUN.2022.3.

3 Hayashi Ani, Lily Chung, Erik D. Demaine, Jenny Diomidova, Della Hendrickson, and Jayson Lynch. Pushing blocks via checkable gadgets: PSPACE-completeness of Push-1F and Block/Box Dude, 2024. Full version of the FUN 2022 paper. arXiv:2412.20079, doi:10.48550/arXiv.2412.20079.

4 Robert D. Barish and Tetsuo Shibuya. Sokoban-style PushPush games with unique solutions. *Journal of Information Processing*, 33:1101–1109, 2025. doi:10.2197/ipsjjip.33.1101.

5 David Bremner, Joseph O'Rourke, and Thomas Shermer. Motion planning amidst movable square blocks is PSPACE-complete. Draft, June, 1994.

6 Joseph C. Culberson. Sokoban is PSPACE-complete. In *Proceedings of the 1st International Conference on Fun with Algorithms (FUN 1998)*, pages 65–76. Carleton Scientific, 1998. URL: https://www.sokoban.dk/wp-content/uploads/2016/02/Sokoban-is-PSPACE-complete.pdf.

7 Erik D. Demaine, Martin L. Demaine, Michael Hoffmann, and Joseph O'Rourke. Pushing blocks is hard. *Computational Geometry: Theory and Applications*, 26(1):21–36, 2003. doi:10.1016/S0925-7721(02)00170-0.

8 Erik D. Demaine, Robert A. Hearn, and Michael Hoffmann. Push-2-F is PSPACE-complete. In *Proceedings of the 14th Canadian Conference on Computational Geometry (CCCG 2002)*, pages 31–35, 2002. URL: https://erikdemaine.org/papers/Push2F_CCCG2002/.

9 Erik D. Demaine, Michael Hoffmann, and Markus Holzer. PushPush-k is PSPACE-complete. In *Proceedings of the 3rd International Conference on Fun with Algorithms (FUN 2004)*, pages 159–170, 2004. URL: https://erikdemaine.org/papers/PushPush1_PSPACE_FUN2004/.

10 Zachary DeStefano and Bufang Liang. Push-1 is PSPACE-complete, and the automated verification of motion planning gadgets, 2025. arXiv:2508.17602, doi:10.48550/arXiv.2508.17602.

11 Robert A. Hearn and Erik D. Demaine. PSPACE-completeness of sliding-block puzzles and other problems through the nondeterministic constraint logic model of computation. *Theoretical Computer Science*, 343(1–2):72–96, 2005. doi:10.1016/j.tcs.2005.05.008.

12 Michael Hoffmann. Motion planning amidst movable square blocks: Push-$*$ is NP-hard. In *Proceedings of the 12th Canadian Conference on Computational Geometry (CCCG 2000)*, pages 205–210, 2000. URL: https://people.inf.ethz.ch/hoffmann/pub/h-psnh-00.pdf.

13 MIT Hardness Group, Josh Brunner, Lily Chung, Erik D. Demaine, Jenny Diomidova, Della Hendrickson, and Jayson Lynch. Pushing blocks without fixed walls via checkable gizmos: Push-1 is PSPACE-complete, 2026. arXiv:2508.19759, doi:10.48550/arXiv.2508.19759.

14 James B. Saxe. Dynamic-programming algorithms for recognizing small-bandwidth graphs in polynomial time. *SIAM Journal on Algebraic and Discrete Methods*, 1(4):363–369, 1980. doi:10.1137/0601042.

15 Roberto Tamassia. On embedding a graph in the grid with the minimum number of bends. *SIAM Journal on Computing*, 16(3):421–444, 1987. doi:10.1137/0216030.

16 Tom C. van der Zanden. Parameterized complexity of graph constraint logic. In *10th International Symposium on Parameterized and Exact Computation (IPEC 2015)*, volume 43

of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 282–293. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2015. doi:10.4230/LIPIcs.IPEC.2015.282.

A The source problem

This appendix reduces standard NCL to the two-page, bounded-nesting, ordered form used in Sections 4 and 5.

▶ **Proposition 27.** *There is a constant m_0 such that NCL reduces in polynomial time to instances (G, σ, τ) of NCL with wires satisfying:*

1. *G is drawn across a horizontal central row with edges $e_1, \ldots, e_n$ crossing it in order: every top vertex is a wire, and every bottom vertex is either a wire or an AND/OR vertex on three consecutive edges e_i, e_{i+1}, e_{i+2}, with the heavy edge of every AND vertex rightmost (e_{i+2});*
2. *vertices in the same half do not cross, and the nesting depth in each half is at most m_0;*
3. *σ and τ are ordered, and τ is reachable from σ through ordered orientations if and only if τ is reachable from σ through arbitrary satisfying orientations.*

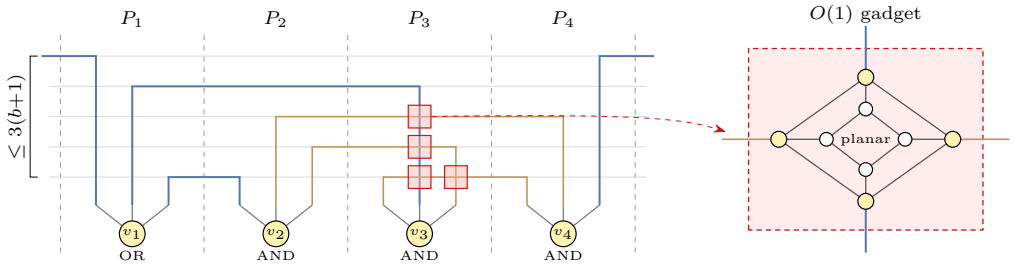
The reduction proceeds in three steps: drawing a bounded-bandwidth constraint graph in a strip of constant height and planarizing its crossings (Appendix A.1), cutting the drawing along a comb tree into two nested pages (Appendix A.2), and encoding the initial and target orientations by edge cursors so that ordered moves suffice (Appendix A.3).

A.1 Bounded-bandwidth strip drawing and planarization

Edge weights and vertex constraints. In the standard formulation of NCL [11, 16], every vertex has degree 3, every edge has weight 1 (light) or 2 (heavy), and every vertex requires total incoming weight at least 2. Two vertex types suffice: an *OR* vertex has three weight-2 edges (satisfied when at least one points in), and an *AND* vertex has one weight-2 edge and two weight-1 edges (satisfied when the heavy edge points in, or both light edges point in). This agrees with Section 4.1, with the additional property that every edge joins either two weight-2 ports (OR or AND-heavy) or two weight-1 ports (AND-light).

Constant-height strip drawing. We start from configuration-to-configuration reachability for AND/OR NCL on graphs of bandwidth at most a fixed constant ω, which is PSPACE-complete [16, Theorem 6]. Given such a constraint graph H_0 with initial and target orientations σ_0, τ_0, we obtain a linear vertex ordering $v_1, \ldots, v_N$ with $|i - j| \leq \omega$ for every edge $v_i v_j$ directly from the reduction of [16], or in polynomial time $O(N^{\omega+1})$ by Saxe's dynamic-programming algorithm [14].

Place $v_1, \ldots, v_N$ from left to right along the bottom of a horizontal strip, giving each vertex v_i its own constant-width *panel* P_i containing three distinct vertical columns for its three incident half-edges (a self-loop uses two distinct columns of P_i). Above the vertices, reserve $3(\omega + 1)$ horizontal tracks indexed by $(i \bmod (\omega + 1), s)$ for $s \in \{1, 2, 3\}$. Route each edge $v_i v_j$ $(i \leq j)$ upward along its column in P_i to track $(i \bmod (\omega + 1), s)$ (where s is its half-edge slot at v_i), horizontally across to its column in P_j, and vertically down to v_j (Figure 43(a)). Two edges assigned to the same track start at least $\omega + 1$ panels apart, and every edge spans at most ω panels, so horizontal segments on the same track never overlap. Crossings occur only transversely between vertical segments and horizontal tracks, with at most $3 \cdot 3(\omega + 1) = O(\omega)$ crossings per panel.



(a) strip drawing (bandwidth b) (b) crossover replacement

Figure 43. (a) A bounded-bandwidth constraint graph drawn in a strip of constant height: each vertex v_i sits in a panel P_i with three vertical half-edge columns, and edges route along at most $3(\omega + 1)$ horizontal tracks (blue for heavy weight-2 edges, orange for light weight-1 edges). (b) Each crossing sits in a small disjoint box (red) and is replaced by Hearn and Demaine's constant-size planar crossover gadget [11, Theorem 9].

Planarization. Although the source graphs of [16, Theorem 6] are planar, planarity together with bounded bandwidth does not automatically yield a *planar* drawing inside a strip of constant height (as noted in the proof of [16, Theorem 11]). We therefore enclose each crossing in a small disjoint box inside its panel and replace the crossing by Hearn and Demaine's constant-size planar crossover gadget [11, Theorem 9] (Figure 43(b)), drawn on a fixed grid template. To map σ_0 and τ_0 to the planarized graph H, orient all segments of each original edge consistently and choose any valid internal orientation of each constant-size crossover gadget matching its four boundary ports; by [11, Theorem 9], the resulting orientations σ_H, τ_H preserve reachability. The planar strip drawing H has two geometric properties, both independent of the input size:

- each panel P_i contains at most a constant number C of vertices, edge pieces, and bends;
- the entire polyline of every edge of H lies in the union of at most $s = \omega + 1$ consecutive panels.

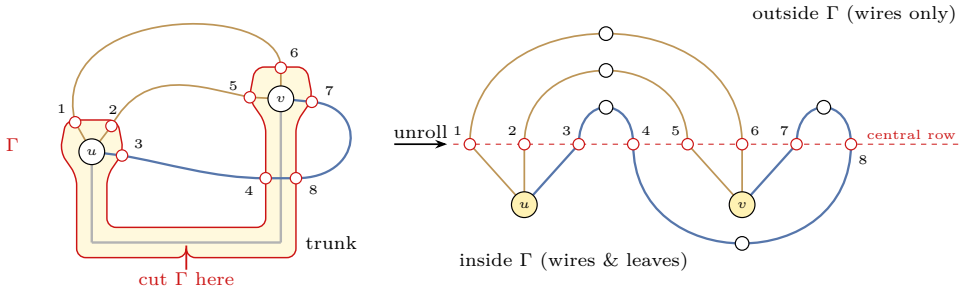
A.2 Comb cut, two-page layout, and heavy-edge placement

The comb tree and its contour Γ. Draw a horizontal *trunk* below the strip, completely disjoint from H. Within each panel P_i, perturb vertex x-coordinates slightly if needed so all vertices of H have distinct x-coordinates and no vertical line through a vertex hits another vertex or bend, then drop a straight vertical *twig* from each vertex of H down to the trunk. The twigs are pairwise disjoint and may cross edges of H transversely; since P_i has at most C geometric features, the twigs in P_i cross edges of H at most $O(C^2)$ times.

Take a sufficiently thin closed regular neighbourhood of this comb tree, and let Γ be its boundary, a simple closed curve (Figure 44(a)). Near each vertex v of H (a leaf of the comb), Γ cuts the three edges incident to v once each, in three *consecutive* crossings along Γ. At each transverse intersection between a twig and an edge of H, Γ cuts that edge twice, once on each side of the twig.

Unrolling into two pages. Cut Γ open along the empty underside of the trunk and flatten it into a horizontal line, the *central row* (Figure 44(b)). Each crossing of H with Γ becomes an edge $e_1, \ldots, e_n$ (a train) crossing the central row in order:

- *Top half (outside Γ):* every connected component of H outside Γ is a sub-arc of an edge of H joining two crossings on Γ; contracting it gives a degree-2 *wire* (requiring at least one



(a) comb tree and contour Γ (b) unrolled two-page layout

Figure 44. From a planar strip drawing to a two-page layout. (a) Two vertices u, v joined by two light edges (orange) and one heavy edge (blue) that crosses the twig of v. The comb tree (gray) is surrounded by its thin neighbourhood boundary Γ (red), producing eight crossings $1, \ldots, 8$. (b) Cutting Γ on the underside of the trunk and flattening it into the central row yields non-crossing wires in the top half (outside Γ) and non-crossing wires plus consecutive three-port vertices in the bottom half (inside Γ).

incoming edge). Because H is planar, these sub-arcs do not cross outside Γ, so top-half wires nest like parentheses.

- *Bottom half (inside Γ):* inside the thin neighbourhood of the comb, every component of H is either a short sub-arc crossing a twig (giving a bottom-half wire) or a three-edge star at a leaf (giving an AND or OR vertex on three consecutive crossings e_i, e_{i+1}, e_{i+2}). Planarity again prevents crossings inside Γ, so bottom-half vertices nest like parentheses.

Each original edge of H leaves both of its endpoint leaves across Γ and may cross twigs along the way, so it is subdivided into a chain of $d \geq 2$ trains alternating between top and bottom wires.

Putting the heavy edge rightmost. At an AND vertex on consecutive ports (a, b, c), we want the heavy edge to occupy the rightmost slot c. One way is to bend the tip of the twig inside a tiny disc around the vertex so it approaches between the heavy edge and its clockwise neighbour along Γ. Alternatively, a purely discrete cyclic shift on the two-page layout achieves the same effect without modifying the drawing: if the heavy edge sits at a, replace (a, b, c) by five consecutive ports (a, b, c, m, f), add a top wire m–f and a bottom wire a–f, and attach the AND star to (b, c, m) while keeping a's original top wire attached to a (Figure 46(a)); symmetrically, if the heavy edge sits at b, replace (a, b, c) by (f, m, a, b, c) with top wire f–m, bottom wire f–c, and AND star on (m, a, b) (making b rightmost). Each shift subdivides one original edge by two extra wires, keeps both halves non-crossing, adds at most 2 ports per AND vertex, and increases nesting depth by at most 1. Call the resulting wire-subdivided graph G.

Constant nesting depth m_0. Because Γ is cut along the underside of the trunk, all crossings coming from a single panel P_i—including the at most 2 extra ports per AND vertex added by the discrete cyclic shifts, whose shift wires stay within the same panel—appear in a contiguous interval of at most B ports along the central row, where B is a constant depending only on C (and thus only on ω). Every AND or OR vertex occupies 3 consecutive ports. Every wire joins two consecutive Γ-crossings along some edge of H; since the entire polyline of that edge lies in at most $s = \omega + 1$ consecutive panels, the two endpoints of any wire are separated by at most sB ports along the central row. In any strictly nested chain of vertices

in the same half, each enclosing wire strictly contains the previous interval and therefore widens the port span by at least 2. Hence the nesting depth in either half is bounded by the constant $m_0 = sB$ (independent of N).

A.3 Wire subdivision, cursors, and ordered orientations

It remains to construct ordered orientations σ, τ of G from σ_H, τ_H so that

$$\begin{aligned} \tau_H \text{ is reachable from } \sigma_H \text{ in } H &\iff \tau \text{ is reachable from } \sigma \text{ via ordered moves in } G \\ &\iff \tau \text{ is reachable from } \sigma \text{ in } G. \end{aligned}$$

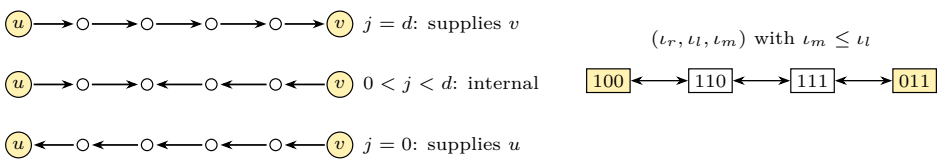
Note first that a self-loop in H contributes its full edge weight to its vertex in either orientation, so flipping a self-loop never affects any constraint. We may therefore first adjust τ_H on every self-loop to match σ_H (and restore it at the end of a path in H), and keep self-loops fixed when lifting moves.

Cursors and canonical ordered encoding. Fix an arbitrary ordering (u, v) of the endpoints of each original edge e of H; in G, e is subdivided into a chain of $d \geq 2$ trains separated by $d - 1$ wires. Number the trains from u to v, writing 1 if a train points toward v and 0 if it points toward u. Every wire requires at least one incoming edge, forbidding the adjacent pair 01; thus every satisfying orientation of the chain has the *cursor form* $1^j 0^{d-j}$ for a unique $j \in \{0, 1, \ldots, d\}$ (Figure 45(a)): $j = d$ supplies v, $j = 0$ supplies u, and $0 < j < d$ is *withheld*, supplying neither endpoint. Reversing one train shifts j by ± 1 while keeping all internal wires satisfied.

Given any satisfying orientation ρ_H of H, assign *demands* at each original vertex:

- every OR vertex demands all its incoming edges in ρ_H;
- every AND vertex demands *only its heavy edge* when its heavy edge points in (local incoming bits $(\iota_r, \iota_l, \iota_m) = 100$, where ι_r is the heavy edge e_{i+2} and ι_l, ι_m are the left and middle light edges e_i, e_{i+1}), and demands *both light edges* otherwise (011).

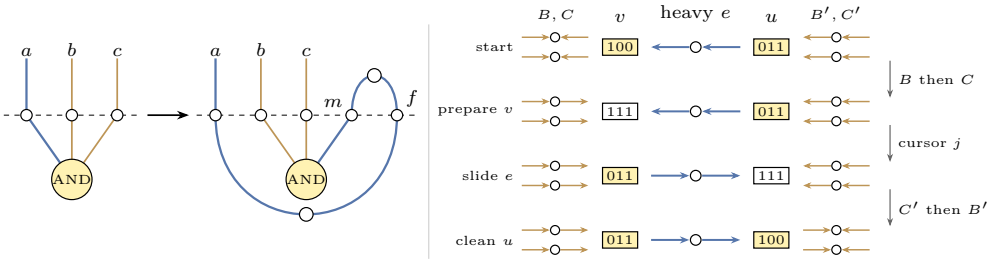
Because a vertex demands only edges pointing into it in ρ_H, at most one endpoint demands any original edge e. Define the canonical orientation ρ of G by setting e's cursor to $j = d$ if v demands e, to $j = 0$ if u demands e, and to the withheld position $j = 1$ if neither endpoint demands e (for a self-loop at an OR vertex, set $j = d$ so one end supplies the OR; a light self-loop at an AND vertex uses both light ports and only supplies weight 1, so the heavy edge must point in and the self-loop stays at $j = 1$). Every vertex of G is satisfied in ρ, and every AND vertex has local incoming bits 100 or 011, both of which satisfy the ordering condition $\iota_m \leq \iota_l$. Set σ and τ to the canonical encodings of σ_H and τ_H.



(a) cursor $1^j 0^{d-j}$ on a subdivided edge

(b) ordered AND transitions

Figure 45. Subdivided edges and ordered AND transitions. (a) Along a chain of d trains between original vertices u and v, wire constraints forbid consecutive edges pointing away from each other, so every valid orientation has the cursor form $1^j 0^{d-j}$. (b) Ordered AND transitions used in the completeness lift.



(a) shifting heavy edge a to the right (b) ordered lift of a heavy-edge reversal ($v \to u$)

Figure 46. (a) Discrete cyclic shift moving a heavy edge a (blue) to the rightmost slot m of an AND vertex (b, c, m) using two extra wires (m–f above, a–f below). (b) Lifting a heavy-edge reversal from AND vertex v to AND vertex u through ordered states: prepare v (B then C: $100 \to 110 \to 111$), slide e's cursor from v to u ($v : 111 \to 011$, $u : 011 \to 111$), and clean up u (C' then B': $111 \to 110 \to 100$).

Completeness (ordered moves in G). Suppose an NCL move in H reverses an edge e from v to u.

- *If e is a light (weight-1) edge:* both endpoints are AND vertices. Because the orientation is valid both before and after the flip, u lacks e before the flip and v lacks e after the flip, so both u and v must have their heavy edge pointing in throughout. Hence both endpoints have canonical demand 100 before and after the flip, e's cursor stays at $j = 1$, and the canonical encoding of G does not change (zero moves).
- *If e is a heavy (weight-2) edge:* perform three stages (Figure 46(b)):
  1. *Prepare v:* if v is an AND vertex losing its heavy edge e, both of its light edges—left light edge B (e_i) and middle light edge C (e_{i+1})—must point into v in H. Currently neither is demanded at either endpoint, so both sit at $j = 1$. Slide B's cursor from 1 to v, then slide C's cursor from 1 to v; v's local state moves $100 \to 110 \to 111$.
  2. *Slide e:* move e's cursor step by step from v to u. If v is an AND vertex it moves $111 \to 011$, and if v is an OR vertex another incoming edge still supplies it; u moves $011 \to 111$ (if AND) or simply gains an incoming edge (if OR).
  3. *Clean up u:* if u is an AND vertex that just gained its heavy edge e, retract its middle light edge C''s cursor from u back to 1, then retract its left light edge B''s cursor from u back to 1; u's local state moves $111 \to 110 \to 100$.

  A light edge cannot be both prepared at v and cleaned up at u, since that would require it to point into both v and u simultaneously in H. Every intermediate AND state lies in $\{100, 110, 111, 011\}$ and is therefore ordered.

Concatenating these lifted moves yields an ordered NCL path from σ to τ in G.

Soundness (arbitrary moves in G). Conversely, suppose τ is reachable from σ in G through any sequence of satisfying orientations (not necessarily ordered). Maintain a recorded orientation $\tilde{\rho}$ of H, initialized to σ_H. Along the sequence in G, every subdivided chain keeps the cursor form $1^j 0^{d-j}$: for each non-loop edge e between u and v, keep $\tilde{\rho}(e)$ unchanged while $0 < j < d$, and flip $\tilde{\rho}(e)$ to point into v (respectively u) only when j reaches d (respectively 0); keep self-loops at their initial orientation. At every step, whenever a chain supplies an endpoint w in G, the recorded edge $\tilde{\rho}(e)$ already points into w (and every self-loop always contributes its full weight). Since vertex constraints are monotone and every vertex of G is satisfied, every update of $\tilde{\rho}$ is a valid NCL move in H.

When the sequence reaches τ in G, every edge demanded by τ_H has its cursor at the demanding endpoint and therefore already agrees with τ_H in $\tilde{\rho}$. Any remaining edge where $\tilde{\rho}$ differs from τ_H is undemanded in τ_H (or a self-loop); flipping these edges one by one to match τ_H preserves all demands of τ_H and hence keeps every vertex satisfied. Thus τ_H is reachable from σ_H in H.

Combining the two directions gives all three equivalences and proves Proposition 27.

Further wire subdivisions. The cursor encoding and the three-way equivalence above depend only on the property that every original edge of H is subdivided into a chain of $d \geq 2$ trains separated by $d-1$ wires. Consequently, if we further subdivide wires of the two-page layout—in particular, by applying the partner repair of Lemma 18 in Section 5, which replaces each repaired upper wire by an alternating chain of upper and lower wires—to obtain a refined two-page graph G', applying the exact same canonical cursor encoding to (σ_H, τ_H) on G' produces ordered orientations σ', τ' of G' such that τ_H is reachable from σ_H in H if and only if τ' is reachable from σ' in G' (and if and only if τ' is reachable from σ' via ordered moves in G').

B Details for Push-$*$S

This appendix supplies the exact coordinates, scales, separator list, core freezing and hole budgets, track and cut-off case analysis, and canonical moves omitted from Section 4.

B.1 Coordinates, scales, and separators

Let n be the number of trains (edges of the drawn graph G), $n_3 \leq n/3$ the number of AND and OR vertices, and $m \leq m_0$ the nesting depth (Appendix A). Set

$$\bar{n} = n+1, \qquad q_0 = 512\bar{n}(m+1), \qquad S = 64\bar{n}, \qquad Y_1 = 8q_0,$$

$$Y_j = Y_1 S^{2j-2}, \qquad L_j = Y_1 S^{2j-1} \quad (1 \leq j \leq m), \qquad L_0 = 0, \qquad D = Y_1 S^{2m}.$$

The board is the rectangle $[0, \bar{n}D] \times [0, 2D]$ with central row $y = D$, and train e_i $(1 \leq i \leq n)$ on column $x_i = iD$; the player starts at $p_0 = (0, 0)$. Every scale Y_j, L_j, D is divisible by 4096. For a half $s \in \{\text{top}, \text{bot}\}$ and depth $d \in [0, D-1]$, write the physical row as $\text{phys}_{\text{top}}(d) = d$ and $\text{phys}_{\text{bot}}(d) = 2D - d$, and let $\eta_{\text{top}} = +1$, $\eta_{\text{bot}} = -1$ be the inward vertical sign. Since $D = 4096(m+1)64^{2m}\bar{n}^{2m+1}$, the board area $(\bar{n}D+1)(2D+1) = O\big((m+1)^2 64^{4m}\bar{n}^{4m+3}\big)$ is polynomial in n for constant m, and every cell is classified as box or hole in polynomial time.

Streets per half. Let $\mathcal{L}_s \subseteq \{1, \ldots, m\}$ be the set of levels of vertices in half s (every AND or OR vertex lies in the bottom half on three consecutive trains, so it has level 1). The *street rows* of half s are depths $\{0\} \cup \{Y_j, Y_j + 2 : j \in \mathcal{L}_s\}$. Each wire on trains $i < i'$ at level j in half s has two *ends* $(x_i, \varepsilon = +1)$ and $(x_{i'}, \varepsilon = -1)$ at depth Y_j, where $\varepsilon \in \{+1, -1\}$ points towards its bar; each AND or OR vertex on consecutive trains $c-1, c, c+1$ has two outer ends $(x_{c-1}, +1)$ and $(x_{c+1}, -1)$ at level 1 in the bottom half. For each such end (x_i, ε) at level j in half s, define its train-corridor column $f = x_i - \varepsilon L_j$ and its bar-corridor column $g = x_i + 2\varepsilon$. For each AND or OR vertex with middle train x_c, define its *public columns* $x_c + 4, x_c + 6$ (and $x_c - 2$ for an OR vertex; an AND vertex omits $x_c - 2$), and its *private column* $x_c + 2$. The *street columns* of half s consist of the boundary columns $0, \bar{n}D$, all train columns $x_1, \ldots, x_n$, and the auxiliary column set U_s containing every f, g of ends in half s together with (for $s = \text{bot}$) the public and private columns of every AND and OR vertex.

Every street row and street column has an even coordinate, so no two street rows of the same half are adjacent and no two street columns of the same half are adjacent.

Fixed corridors and private branches. The board starts full of boxes (or walls in Section 4.2) except for the following holes:

- *Frame:* rows $0, 2D$ and columns $0, \bar{n}D$.
- *Train and bar corridors:* at every wire or outer AND/OR end (x_i, ε) at level j in half s, the vertical leg on column $f = x_i - \varepsilon L_j$ at depths $1, \ldots, Y_j + 2$, the horizontal leg at depth $Y_j + 2$ strictly between f and x_i, and the bar corridor on column $g = x_i + 2\varepsilon$ at depths $1, \ldots, Y_j - 1$.
- *Public columns and private branches:* at every bottom AND or OR vertex with middle train x_c (working row $y_1 = 2D - Y_1$), each public column ($x_c - 2, x_c + 4, x_c + 6$ for OR; $x_c + 4, x_c + 6$ for AND) is empty at depths $1, \ldots, Y_1 - 1$, and the *private branch* consists of the three holes

  $$N = \{(x_c + 2, y_1 - 1),\ (x_c + 1, y_1 - 2),\ (x_c + 2, y_1 - 2)\}$$

  (depths $Y_1 + 1$ and $Y_1 + 2$).

Canonical working holes of B_ρ. Given an ordered satisfying orientation ρ, the working holes of B_ρ are as follows:

- *Wire and outer AND/OR ends:* at each end (x_i, ε) on working row $y = \mathrm{phys}_s(Y_j)$, include the three-cell *pocket* $\{(x_i, y), (x_i, y + \eta_s), (x_i + \varepsilon, y)\}$. Add $(x_i, y + 2\eta_s)$ if e_i points into this end, and add $(x_i + 2\varepsilon, y)$ if the bar points into this end (for a wire, its bar points into its left end iff the left edge does not point in; for an AND or OR vertex, the left or right bar points into its outer pocket iff e_{c-1} or e_{c+1} does not point into the vertex).
- *AND and OR cores:* at a vertex with middle train x_c and working row $y_1 = 2D - Y_1$, set $\kappa = 4$ for OR and $\kappa = 2$ for AND. Let $l = 1$ if e_{c-1} points in and $l = 2$ otherwise; $v = 2$ if e_c points in and $v = 1$ otherwise; $r = 1$ and $p = 4$ if e_{c+1} points in, and $r = 2$ and $p = 0$ otherwise. Include the middle-train holes $(x_c, y_1 - 1), \ldots, (x_c, y_1 - v)$, the core gap $I_G = \{(x_c + \delta, y_1) : \delta \in [l - \kappa + 1, p]\}$, and the relay gap $I_J = \{(x_c + \delta, y_1) : \delta \in [p + 2, 7 - r]\}$.

On the working row of an AND or OR vertex (strictly between x_{c-1} and x_{c+1}, length $2D - 1$), the remaining occupied cells form three runs: the left bar of length $D - \kappa$ on $[x_{c-1} + l + 1,\ x_c + l - \kappa]$, the single-box short bar at $x_c + p + 1$, and the right bar of length $D - 8$ on $[x_c + 8 - r,\ x_{c+1} - r - 1]$, leaving $(2D - 1) - (D - \kappa) - 1 - (D - 8) = \kappa + 6$ holes. Since ρ is ordered and satisfying, I_G and I_J are nonempty in B_ρ: for OR, $l - \kappa + 1 \le -1 \le p$; for AND, $p = 0$ forces e_{c-1} and e_c to point in, so $l = 1$ and $I_G = \{x_c\}$. Each ordinary train has length $y_B - y_T - 4$ (where y_T, y_B are its two working rows), each middle train has length $y_1 - y_T - 4$, and each wire bar between x_a and x_b has length $x_b - x_a - 4$, independent of ρ. Hence all B_ρ have equal size.

Separators (wall-free board of Section 4.3). When walls become boxes, we add five families of single-cell holes:

1. *Central-row separators:* (x, D) for every $x \in U_{\mathrm{top}} \cup U_{\mathrm{bot}}$.
2. *Wide-gap horizontal separators:* in each half s, on every street row at depth $d \in \{Y_j, Y_j + 2 : j \in \mathcal{L}_s\}$ and for every pair of consecutive street columns $a < b$ of half s with $b - a > L_m$, place a hole at the even-rounded midpoint $(2\lfloor (a + b)/4 \rfloor, \mathrm{phys}_s(d))$, unless that midpoint lies on a bar track.

3. *Inter-level vertical separators:* in each half s, for every column $x \in U_s$ belonging to a level-j vertex with $j < m$ (including the public and private columns of AND and OR vertices, which have level 1), place a hole at $(x, \mathrm{phys}_s((Y_j + Y_{j+1})/2))$.
4. *Local horizontal separators next to trains:* for each wire or outer AND/OR end (x_i, ε) at level j in half s with $f = x_i - \varepsilon L_j$, place a hole at $((f + x_i)/2, \mathrm{phys}_s(d))$ on every street row of half s at depth $0 < d \le Y_j$, and place a hole at $(x_i + \varepsilon, \mathrm{phys}_s(d))$ on every street row of half s at depth $0 < d < Y_j$.
5. *Half-depth core separators:* for each AND or OR vertex with middle train x_c, place holes at $(x_c, 2D - Y_1/2)$ and $(x_c + 2, 2D - Y_1/2)$.

Every hole of B_τ lies on a street of its half or on the central row in a column of $U_{\mathrm{top}} \cup U_{\mathrm{bot}}$, and no separator is adjacent to a hole off its own street, so the backward pull induction of Section 4.3 applies directly.

B.2 Hole budgets, core freezing, and deep crossings

Global hole budgets and scale inequalities. For $1 \le j \le m$, define the global hole budgets

$$b_j = q_0 + 8\bar{n}(Y_j + L_j + 10), \qquad K_j = q_0 + 4n(Y_j + 2) + 2nL_{j-1}.$$

Because $q_0 \ge 2048$, $S \ge 128$, and $Y_j \ge 8q_0 \ge 16384$, we have $q_0/Y_j \le 1/8$, $16/Y_j \le 1/8$, and $L_{j-1}/Y_j \le 1/S$, yielding four bounds:

1. $Y_1/2 = 4q_0 > q_0$.
2. $K_j/Y_j \le 1/8 + 5n + 2n/S < 16\bar{n} = L_j/(4Y_j)$, so $K_j < L_j/4 < L_j/2$; also $L_j/4 = 16\bar{n}Y_j > L_{j-1} + 12$, giving $L_j/2 - L_{j-1} - 12 > K_j$.
3. $b_j/Y_j \le 1/8 + 8\bar{n}(S + 1 + 1/8) = 512\bar{n}^2 + 9\bar{n} + 1/8 < S^2/4$ and $(Y_j + 4b_j + 16)/Y_j < 4096\bar{n}^2 = S^2 = Y_{j+1}/Y_j$, so $(Y_{j+1} - Y_j)/2 - 2 > 2b_j + 6 > b_j$.
4. $4(b_m + L_m + 10)/Y_m < 4096\bar{n}^2 = D/Y_m$, so $D > 4(b_m + L_m + 10)$, and in particular $D - L_m > b_m$ and $(D - 2L_m)/2 - 1 > b_m$.

Outside the frame, corridors, horizontal legs, and public columns, the board contains at most $4m(4n + 1)$ wide-gap separators, $8nm$ local separators, $5n$ inter-level separators, $5n$ central-row separators, $5n_3$ half-depth separators and private-branch holes, and $10n + 10n_3$ working holes, totalling strictly fewer than q_0. Initially, all $2n$ train corridors and horizontal legs, $2n$ bar corridors, and at most $3n_3 \le n$ public columns contribute at most $5n(Y_m + 2) + 2nL_m$ holes, so the total interior hole count is $b < q_0 + 5n(Y_m + 2) + 2nL_m < b_m$. Thus every central-row separator lies at distance at least $D - L_m > b_m > b$ from the frame, and every wide-gap separator lies at distance at least $(D - 2L_m)/2 - 1 > b_m > b$ from the ends of its gap.

Stage budgets and bar-corridor shielding. At the start of stage j ($j = m, \dots, 1$), all corridors of levels $> j$ are already in A, so the holes outside A number fewer than $q_0 + 5n(Y_j + 2) + 2nL_j < b_j$. For $j < m$, the segment of each level-j column $x \in U_s$ strictly between depth $Y_j + 2$ and depth Y_{j+1} (or the central row D if j is the deepest level of half s) crosses no street rows (so its side neighbours are off the streets and fixed); its inter-level separator lies at distance at least $(Y_{j+1} - Y_j)/2 - 2 > b_j$ from both endpoints, so it is blocking by the along-segment version of Lemma 8 and fixes that segment (for $j = m$, the central-row separator already fixed the segment strictly between $Y_m + 2$ and D). On each level-j bar-corridor column $g = x_i + 2\varepsilon$, the crossing box $(g, \mathrm{phys}_s(Y_j + 2))$ is then backed vertically by the fixed box at depth $Y_j + 3$ and horizontally by the wide-gap box $(x_i + 3\varepsilon, \mathrm{phys}_s(Y_j + 2))$ fixed in the long-segments step, while $(g, \mathrm{phys}_s(Y_j + 1))$ has off-street

horizontal neighbours and $(x_i + \varepsilon, \mathrm{phys}_s(Y_j + 2))$ has off-street vertical neighbours; thus none of these three boxes can be the first to move, which closes the train end of the level-j horizontal corridor leg. After the level-j train corridors and horizontal legs join A, the remaining corridors outside A consist of at most $3n$ bar and public corridors of depth at most $Y_j - 1$ and at most $2n$ shallower train corridors and legs of depth at most $Y_{j-1} + 2$ and length at most L_{j-1} (zero when $j = 1$). Since $2(Y_{j-1} + 2) \le Y_j + 2$ for $j > 1$, the remaining holes outside A number fewer than K_j. Each level-j midpoint separator $((f + x_i)/2, \mathrm{phys}_s(d))$ $(d \le Y_j)$ lies at distance $L_j/2 > K_j$ from f and x_i, so it is blocking and fixes the entire segment between f and x_i on row $\mathrm{phys}_s(d)$.

On any shallower street row $d < Y_j$ crossing a level-j train x_i and its bar corridor $g = x_i + 2\varepsilon$, the cell $(x_i - \varepsilon, \mathrm{phys}_s(d))$ was just fixed, and the cell $(x_i + 3\varepsilon, \mathrm{phys}_s(d))$ on the far side of the bar corridor lies in a wide gap already fixed in the long-segments step (no shallower bar spans across $(x_i + 2\varepsilon, \mathrm{phys}_s(d))$ because vertices nested inside level j lie strictly between x_i and its partner). Moreover, the single cell $(x_i + \varepsilon, \mathrm{phys}_s(d))$ is a hole whose vertical neighbours are off the streets and fixed. Therefore no horizontal push can move a box into $(x_i + \varepsilon, \mathrm{phys}_s(d))$ or into the bar corridor $(x_i + 2\varepsilon, \mathrm{phys}_s(d))$ (pushing from the train side requires standing on or pushing $(x_i - \varepsilon, \mathrm{phys}_s(d))$, and pushing from the far side requires standing on or pushing $(x_i + 3\varepsilon, \mathrm{phys}_s(d))$), and no vertical push can enter the bar corridor from its working row, since that would require standing on $(g, \mathrm{phys}_s(Y_j + 1))$, whose box was fixed in the closing step above. Thus every level-j bar corridor and every adjacent hole $(x_i + \varepsilon, \mathrm{phys}_s(d))$ stays empty; adding the bar corridors to A reduces the holes outside A below b_{j-1} for $j > 1$ (and below q_0 once the level-1 public columns also join A).

Core freezing. Fix an AND or OR vertex in the bottom half with middle train x_c and working row $y_1 = 2D - Y_1$ (depth Y_1). We use local column offsets $\delta = x - x_c$ and depth $d = 2D - y$. In the long-segments step, the wide gaps on street row $d = Y_1 + 2$ fixed $(+7, Y_1 + 2)$, together with $(-3, Y_1 + 2)$ for an OR vertex and $(-1, Y_1 + 2)$ for an AND vertex (for AND, column -2 is off the streets, so the wide gap reaches column 0). In the level-1 closing step, the inter-level (or central-row) separators fixed every public column $(+4, +6,$ and -2 for OR) and the private column $+2$ from depth $Y_1 + 3$ down to its separator. Now consider the set F_{core} of initial boxes consisting of:

- on row $d = Y_1 + 2$: offsets $[+3, +6]$, and for OR also $[-2, -1]$;
- on each public column $(+4, +6,$ and -2 for OR): depths $Y_1 + 1$ and $Y_1 + 2$.

Every maximal horizontal segment of F_{core} has an endpoint adjacent to a fixed box $((+7, Y_1 + 2), (-3, Y_1 + 2),$ or an off-street box at depth $Y_1 + 1)$, and every maximal vertical segment of F_{core} has an endpoint adjacent to a fixed box (at depth $Y_1 + 3$, or an off-street box at columns $-1, +3, +5$). Consider the hypothetical first push in π that moves a box of F_{core}: the player cannot start inside the still-full set F_{core}, pushing along a maximal segment towards its fixed neighbour would push that fixed box, and pushing away from it would require standing on or pushing that fixed box. Hence no box of F_{core} ever moves. In particular, every public column is backed at depth $Y_1 + 1$ by a fixed box of F_{core} and has no crossing street rows at depths $1, \ldots, Y_1 - 1$, so no box can enter it; all public columns stay empty and join A at the end of level 1.

At this point A contains the frame, every train and bar corridor, and every public column, leaving fewer than q_0 holes outside A. On columns 0 (middle train) and $+2$ (private column), at depths $1, \ldots, Y_1 - 1$, there are no crossing street rows (the shallowest street row is Y_1), so all side neighbours are off the streets and fixed. Each column segment starts full of boxes except for its single half-depth separator at depth $Y_1/2 = 4q_0 > q_0$. By

Lemma 8 along each segment, both half-depth separators are blocking, fixing every box on columns 0 and +2 at depths $1, \dots, Y_1 - 1$. Finally, no box can enter the private branch $N = \{(+2, Y_1 + 1), (+1, Y_1 + 2), (+2, Y_1 + 2)\}$: cell $(+2, Y_1 + 1)$ has off-street fixed horizontal neighbours $(+1, Y_1 + 1)$ and $(+3, Y_1 + 1)$; column +2 is bounded by the just-fixed $(+2, Y_1 - 1)$ (blocking pushes from depth Y_1 into N) and fixed $(+2, Y_1 + 3)$; cell $(+1, Y_1 + 2)$ has off-street fixed vertical neighbours $(+1, Y_1 + 1)$ and $(+1, Y_1 + 3)$; and row $Y_1 + 2$ is bounded on the right by fixed $(+3, Y_1 + 2) \in F_{\text{core}}$ and on the left by fixed $(-1, Y_1 + 2)$ (blocking pushes from column 0 into N). Thus N stays empty as well.

Deep crossings and remaining deep segments. With fewer than q_0 holes remaining outside A, consider any cell that lies either (i) at a deep crossing $(x, \text{phys}_s(d))$ where a deeper street row $d \in \{Y_{j'}, Y_{j'} + 2\}$ $(j' > j)$ crosses a level-j train x_i, or where a level-j' bar row $Y_{j'}$ crosses an inner column $x \in U_s$ of level $j < j'$, or (ii) on a level-j $(j < m)$ column $x \in U_s$ strictly between its inter-level separator at depth $(Y_j + Y_{j+1})/2$ and the central row D. Every such cell lies at Manhattan distance at least $\min((Y_{j+1} - Y_j)/2 - 2,\ D - 2L_m,\ D - Y_m - 2) > q_0$ from the frame, from every corridor, and from every public column (a level-j' horizontal corridor leg runs away from its bar and hence stays at distance at least $D - L_j > q_0$ from every inner column x). Since every such cell starts occupied, Lemma 8 implies that the player never enters it, so it stays occupied throughout π. In particular:

- Every deep crossing on a bar (meeting an inner train or an inner column $x \in U_s$) and every deeper street-row cell $(x_i, \text{phys}_s(d))$ $(d \in \{Y_{j'}, Y_{j'} + 2\},\ j' > j)$ on a level-j train stays occupied throughout π, while still allowing the bar or train to shift boxes across it (so its cell occupancy is preserved, not box identity).
- On each level-j $(j < m)$ column $x \in U_s$, because every cell strictly between its blocking inter-level separator and blocking central-row separator stays occupied, no vertical push along x can move any box in that interval (any such push would fill one of the two blocking separators). Hence every off-row background box on x in that interval is fixed by identity; and on every deeper non-bar street row (such as depth $Y_{j'} + 2$ with $j' > j$, or depth $Y_{j'}$ outside level-j' bars), the crossing boxes at x and the background boxes between a level-j train and its level-j columns are backed horizontally by the fixed wide gaps on one side and the permanently occupied train cell $(x_i, \text{phys}_s(d))$ on the other (and vertically by fixed boxes or off-street neighbours), so by the first-move argument they are fixed by identity as well.

Every background box off the tracks is now fixed by identity and no hole off the tracks is ever filled, completing the proof of Lemma 6.

B.3 Soundness: cut-off cases

Parameters along a trajectory. By the *Tracks* paragraph of Section 4.2 (and Lemma 6 without walls), every run of boxes on a track keeps its exact length throughout π and consecutive runs never merge. This gives the following parameter identities at every configuration along π:

- *Wire bar:* the hole counts $h_l, h_r \geq 0$ at its two ends satisfy $h_l + h_r = 3$.
- *Ordinary train x_i* (working rows y_T, y_B): let $a_T =$ (top cell of train) $- y_T$ and $a_B = y_B -$ (bottom cell of train). Train length $y_B - y_T - 4$ gives $a_T + a_B = 5$, while non-merging with the outer extensions (shifted inward by $t_T, t_B \geq 0$) gives $a_T \geq t_T + 1 \geq 1$ and $a_B \geq t_B + 1 \geq 1$. Along π, the shifts always satisfy $t_T \leq 2$ and $t_B \leq 2$: since $a_T \leq 4$ (from $a_T + a_B = 5, a_B \geq 1$), we have $t_T \leq 3$, and if t_T ever reached 3 for the first time, the push shifting the top extension from $t_T = 2$ to $t_T = 3$ would leave the player above the

extension (at $(x_i, 3)$) outside the top gap, which would then be the single cell $(x_i, y_T + 3)$ (as $a_T = 4$). Because y_T is even, $y_T + 3$ is odd and lies off all street rows (its horizontal neighbours are walls or fixed background), while the extension at $y_T + 2$ and the train at $y_T + 4$ bound it vertically; no later push from outside can move the extension back up, so $t_T = 3$ would persist to the end of π, contradicting $t_T = 0$ in B_τ. Symmetrically $t_B \le 2$. Since $t_T \le 2$, the top gap $[y_T + t_T,\, y_T + a_T - 1]$ reaches the train corridor at depth $y_T + 2$ iff $a_T \ge 3$ (equivalently $a_B \le 2$), and symmetrically the bottom gap reaches its train corridor iff $a_B \ge 3$; thus the train points into its top end iff $a_T \ge 3$ and into its bottom end iff $a_B \ge 3$.

- *Middle train x_c* (top working row y_T, bottom working row $y_1 = 2D - Y_1$): with $a_T =$ (top cell of train) $- y_T$ and $v \ge 0$ holes directly above y_1, train length $y_1 - y_T - 4$ gives $a_T + v = 4$ (with $a_T \ge t_T + 1 \ge 1$). Since $a_T \le 4$, the exact same enclosure argument gives $t_T \le 2$ throughout π, so the middle train points up iff $a_T \ge 3$ ($v \le 1$) and into the bottom core iff $v \ge 2$ ($a_T \le 2$).
- *AND or OR working row:* with $l, r \ge 0$ holes at the left and right outer ends and the single-box short bar at $x_c + p + 1$, non-emptiness of $I_G = [l - \kappa + 1, p]$ and $I_J = [p + 2, 7 - r]$ gives $p \ge l - \kappa + 1$ and $p + r \le 5$.

Cut-off invariants and exhaustive case analysis. Consider the four cut-off conditions below, each paired with the requirement that the player stands *outside* the listed cut-off region:

$\mathcal{T}_{\text{pkt}}$ *Ordinary pocket* ($a \le 2$ and $h \le 1$ at a wire or outer AND/OR end): the train gap $[t, a - 1] \subseteq [0, 1]$ misses the train corridor at depth offset $+2$, and the bar gap (at most 1 cell) misses the bar corridor at horizontal offset $+2$. The cut-off region is the union of the train gap and bar gap (when $t = 1$, the extension box at the corner splits this union into at most two components, neither connected to the frame or any public column, and the player stands outside both).

$\mathcal{T}_{\text{OR}}$ *OR core* ($l \ge 2$, $p \le 3$, $v \le 1$): the nonempty core $I_G = [l - 3, p] \subseteq [-1, 3]$ misses all three public columns $-2, 4, 6$; $I_G \ne \emptyset$ implies $p \ge -1$, so $I_J = [p + 2, 7 - r]$ starts at ≥ 1 and misses column 0; and $v \le 1$ disconnects the middle-train gap V from the private branch N. The cut-off region is $I_G \cup V$ (plus N if $2 \in I_G$; if $2 \notin I_G$, then N is separated from $I_G \cup V$).

$\mathcal{T}_{\text{mid}}$ *AND core without middle* ($p \le 3$, $v \le 1$): since $\kappa = 2$, $I_G = [l - 1, p] \subseteq [-1, 3]$ misses both public columns $4, 6$, and the same region as in $\mathcal{T}_{\text{OR}}$ is cut off.

$\mathcal{T}_{\text{left}}$ *AND core without left* ($p \le 3$, $l \ge 2$): $I_G = [l - 1, p] \subseteq [1, 3]$, so the left bar occupies column 0, blocking V from reaching I_G even if $v \ge 2$; moreover $I_G \ne \emptyset$ gives $p \ge 1$, so $I_J = [p + 2, 7 - r]$ starts at ≥ 3 and misses private column 2. If $2 \in I_G$, the cut-off region is $I_G \cup N$ (plus V if $v \ge 2$); if $2 \notin I_G$, it is I_G alone.

In every one of $\mathcal{T}_{\text{pkt}}, \mathcal{T}_{\text{OR}}, \mathcal{T}_{\text{mid}}, \mathcal{T}_{\text{left}}$, the cut-off region has no path of holes to the frame or to any public column, and breaking any defining inequality requires pushing a train, bar, short bar, or extension *away* from the cut-off region, which is only possible from inside it. Any push from outside only moves those runs further in, preserving the condition and keeping the player outside. Because B_τ is ordered and satisfying, it avoids all four conditions ($\mathcal{T}_{\text{pkt}}$ fails at every pocket; in an OR, either $p = 4$, or $p = 0$ with $l = 1$ or $v = 2$; in an AND, either $p = 4$, or $p = 0$ with $l = 1$ and $v = 2$).

At the first vertex u violated during π, the reversing push moved the train of an edge e towards u from its far endpoint, leaving the player outside u. Every case hits one of the four conditions:

- *Wire:* neither train points in $(a_l, a_r \le 2)$. Since $h_l + h_r = 3$, at least one end has $h \le 1$, hitting $\mathcal{T}_{\text{pkt}}$.
- *OR:* all three edges point away ($a_l, a_r \le 2$ and $v \le 1$). If $l \le 1$ or $r \le 1$, that outer pocket hits $\mathcal{T}_{\text{pkt}}$; otherwise $l, r \ge 2$, and $p + r \le 5$ gives $p \le 3$, hitting $\mathcal{T}_{\text{OR}}$.
- *AND:* the heavy edge points away ($a_r \le 2$) and at least one light edge points away ($a_l \le 2$ or $v \le 1$). If $r \le 1$, the right pocket hits $\mathcal{T}_{\text{pkt}}$; otherwise $r \ge 2$, so $p + r \le 5$ gives $p \le 3$. If $v \le 1$, this hits $\mathcal{T}_{\text{mid}}$; otherwise $a_l \le 2$, and then $l \le 1$ hits $\mathcal{T}_{\text{pkt}}$ on the left while $l \ge 2$ hits $\mathcal{T}_{\text{left}}$.

This contradiction completes the proof of Lemma 4.

B.4 Completeness: reversing edges

To prove Lemma 3, we simulate one ordered step $\rho \to \rho'$ reversing an edge e from its old endpoint u to its new endpoint w, moving from (p_0, B_ρ) to $(p_0, B_{\rho'})$ by pushing only trains, bars, and short bars (so the same trajectory works with and without walls).

Each basic move starts and ends on the empty frame at p_0: shifting a train, wire bar, or outer AND/OR bar takes 1 push from an open pocket or core (returning via the same or newly opened corridor), while shifting the single-box short bar between $p = 0$ and $p = 4$ takes 4 consecutive pushes (from $(x_c, y_1) \in I_G$ right to $p = 4$ when $r = 1$, returning via $+4$; or from $(x_c + 6, y_1) \in I_J$ left to $p = 0$ via $+6$, returning via $+4$). To reverse e from u to w, first retract any internal bar at u, next push e's train 1 cell away from w towards u, and finally advance any internal bar at w:

Role of e	Before pushing e (at losing u)	After pushing e (at gaining w)
Wire, left	flip bar to point left	flip bar to point right
Wire, right	none	none
AND/OR, e_{c-1}	retract left ($l : 1 \to 2$)	advance left ($l : 2 \to 1$)
AND/OR, e_c	none	none
AND/OR, e_{c+1}	retract short ($4 \to 0$), right ($1 \to 2$)	advance right ($2 \to 1$), short ($0 \to 4$)

Every move enters an open pocket or core. At a wire, satisfaction of ρ and ρ' keeps the opposite edge pointing in whenever the bar flips, and keeps w's pocket open via its bar before e's train is pushed. At an AND or OR vertex:

- outer pockets are opened by their train corridors when retracting r or l, and by their bar corridors before pushing e's train; advancing r and retracting p enter I_J directly via public columns $+4$ and $+6$;
- advancing l (at w), pushing the middle train e_c up from $(x_c, y_1 - 1)$ (at w), and advancing p (at w) all enter I_G (with $(x_c, y_1) \in I_G$ whenever $p = 0$ or when pushing e_c or advancing p): whenever $p = 4$, public column $+4$ opens I_G; when $p = 0$, an OR with $l = 1$ uses public column -2, and otherwise $v = 2$ gives the route $+4 \to I_J \to N \to V \to (x_c, y_1) \in I_G$. Satisfaction and orderedness of ρ and ρ' guarantee that an AND vertex never needs I_G with $l = 2, p = 0$: before gaining e_{c-1}, the AND is satisfied without it, so its heavy edge e_{c+1} points in and $p = 4$; acquiring e_c forces e_{c-1} to point in, so $l = 1$, and e_{c+1} already points in, so $p = 4$; and advancing p happens only with both light edges pointing in, so $l = 1$ and $v = 2$.

Each edge reversal uses at most 4 basic moves (at most one 4-push short-bar shift), hence at most 7 pushes, and leaves $(p_0, B_{\rho'})$.

C Details for Push-$*$

This appendix supplies the detailed proofs omitted from Section 5.

C.1 Readers: coordinates, rank bounds, recursion, and maximal slides

Exact coordinates of the one-read and extended boards (Section 5.1). For a $w_0 \times h_0$ source board R with n_0 initial holes and bottom quotas q_x summing to $Q \le n_0$, place R on $[-w_0, -1] \times [1 - h_0, 0]$ with $c = (-1, 0)$, and set $L = \max(7,\, Q + \max(n_0, w_0) + 4)$ and $y^* = L - 2 - Q \in [2, L-2]$. The one-read board $R_1 = [-w_0, 1] \times [1 - h_0, L]$ fills all cells outside R with boxes except:

- column 0: $C = (0,0)$, $(0,1)$, and $(0,L)$ (with sealing cell $g = (0, L-2)$);
- column 1: $(1,1), \ldots, (1,L)$; and
- pinholes at (x, s_x) for $x \in \{-w_0, \ldots, -1\}$, where $s_{-1} = L - 2$ and $s_{x-1} = s_x - q_x$.

The player goal is $g_{\text{old}} = (-w_0, s_{-w_0} - q_{-w_0}) = (-w_0, y^*)$ (which holds a box initially unless $q_{-w_0} = 0$, when it coincides with the westmost pinhole). For composition, set $y_{\text{mid}} = 2L - y^* + 1$ and $q_{\text{aux}} = w_0 + 4$; take $e_{\text{rows}} = e_{\text{wid}} = 0$ for a plain read step, or $e_{\text{rows}} = h_0$ and $e_{\text{wid}} = w_0$ for a clear-preparation step, and set $y_{\max} = y_{\text{mid}} + e_{\text{rows}}$. Let $x_{\text{res}} = n_0 + w_0 + L + 2 + 2y_{\max} + (q_{\text{aux}} + e_{\text{wid}})y_{\max} + 4$, which exceeds the total hole count N_{total} of the extended board R^+ by at least 2. Extend R_1 to $[-w_0 - 2,\, x_{\text{res}} + 2(q_{\text{aux}} + e_{\text{wid}} - 1)] \times [1 - h_0, y_{\max}]$ by filling all new cells with boxes except:

- the southwest entry $(-w_0 - 1, y^*)$, push-down shaft $(-w_0 - 1, y)$ $(L + 1 \le y \le y_{\text{mid}})$, and west corridor $(-w_0 - 2, y)$ $(L + 1 \le y \le y_{\max})$;
- when $e_{\text{rows}} = h_0$, clear-preparation entries $(-w_0 - 1, y)$ $(y_{\text{mid}} + 1 \le y \le y_{\max})$; and
- isolated far-east reserve holes at $(x_{\text{res}} + 2j, y)$ on every auxiliary row $y \in [1, y_{\max}]$, for $0 \le j < q_{\text{aux}}$ (and $0 \le j < q_{\text{aux}} + w_0$ when $y > y_{\text{mid}}$).

Rotating R^+ by 90° counter-clockwise ($(x, y) \mapsto (y, -x)$) maps the new southwest corner $(-w_0 - 2, y_{\max})$ to the new lower right corner $(y_{\max}, w_0 + 2)$ and maps every row of R ($y \in [1 - h_0, 0]$) and auxiliary row ($y \in [1, y_{\max}]$) to a vertical column for the next bottom read.

Column-0 colouring and excursion deletion (Lemma 10). In part (b) of the proof of Lemma 10, the two northeast quadrants $\{x \ge 0,\ y < 0\}$ and $\{x > 0,\ y \le 0\}$ start full of boxes; their complements are intervals along every row and column, and every push moves its first hole to the player's new cell, which lies between two existing holes. Hence both quadrants stay full forever, and the player leaves R only from $c = (-1, 0)$ through $C = (0, 0)$. Once C holds a box, it cannot become a hole again because no hole or player position lies to its north or east. Before the first visit to $g = (0, L-2)$, strip injection (part (a)) keeps all w_0 pinholes untouched and every other cell of $\{x < 0,\ y > 0\}$ full, so no horizontal push can create a hole in column 0: pushing west from column 1 into $y > 0$ would consume a pinhole, pushing east requires a player in $\{x < 0,\ y > 0\}$, and C cannot reopen.

Write $z = L - y$ along column 0, with the initial three holes at $z = 0$ (red) and $z = L-1, L$ (blue). Let D_R and D_B be the numbers of red and blue holes deleted horizontally from column 0. We prove by induction on moves before the first visit to g that:

$$\text{the } i\text{-th red hole from } z = 0 \text{ has } z \le i - 1 + D_R,$$
$$\text{the } j\text{-th blue hole from } z = L \text{ has } z \ge L - j + 1 - D_B.$$

Initially $D_R = D_B = 0$, the sole red hole is at $z = 0$, and the two blue holes are at $z = L, L-1$, so both bounds hold; since total holes plus deletions equal 3, red holes stay in $z \le 2$ and blue holes in $z \ge L - 2$ (disjoint since $L \ge 7$). At each step:

- A vertical push between two holes of the same colour moves the pushed hole to a cell adjacent to the pushing hole, whose rank differs by 1, keeping both bounds.

- A vertical push from a hole of one colour into a hole of the opposite colour removes the innermost rank of the opposite colour (leaving its remaining ranks unchanged) and creates a new hole of the pushing colour at distance 1 past its previous maximum rank, preserving both bounds.
- A horizontal deletion of a red (or blue) hole decreases the rank of every subsequent hole of that colour by 1 while increasing D_R (or D_B) by 1, leaving $i-1+D_R$ (or $L-j+1-D_B$) unchanged.

When the player first stands on g ($z=2$), g must be a red hole of some rank i with $i+D_R \geq 3$. Since $i + D_R \leq (\#\text{ red holes}) + D_R = 3 - (\#\text{ blue holes}) - D_B$, both the number of blue holes and D_B must be 0; in particular, C ($z=L$) is filled.

Before this visit to g, whenever the player leaves R from c through C, c remains empty while the player is outside R (no push from outside R can reach c without consuming a pinhole, and the only cell adjacent to R is C). Any return into R therefore steps directly from C back into c, leaving every hole inside R unchanged since departure. Deleting every excursion outside R produces a valid trajectory inside R ending at c with the exact hole set E present at the last departure from R.

Proof of Lemma 11 (one-layer composition interface). We verify parts (1)–(4) and the $O(A^3)$ area bound for the rotated extended board $\widetilde{R}$ built from a $w_0 \times h_0$ board R of area $A = w_0 h_0$ (coordinates above).

Part (1): Source extraction and prefix deletion (Push-$*$ and PushPush-$*$). Fix first a Push-$*$ trajectory on R^+ reaching the new southwest corner $(-w_0-2, y_{\max})$ (which becomes c^+ on $\widetilde{R}$). Before the first entry into $\{x < -w_0\} \cup \{y > L\}$, this region can only lose holes: a push creates a hole where the player steps. A first entry from the north would need a hole below L in the player's column. Such holes lie only in the two west columns or in the far-east reserve, and the player cannot reach $x \geq x_{\text{res}}$ by Lemma 8 (since $x_{\text{res}} > N_{\text{total}}$). The only hole west of $-w_0$ at height $y \leq L$ is the southwest entry $(-w_0-1, y^*)$, which must therefore survive until the player steps into it from g_{old}; consuming it earlier would close the only entrance.

Until that step the player may visit $x > 1$ and spend reserve holes from afar. Apply the strip count in part (a) of Lemma 10 to the source columns $-w_0 \leq x \leq j$, for $j = -1, \ldots, -w_0$. The reserve lies east of every strip, and the southwest entry must still be preserved. Before a strip's first southern visit, advancing the southernmost of its non-pinhole holes by one row costs an original pinhole in that strip. There are at most $w_0 < y^*$ such pinholes, so first entry into its southern part comes from the east and uses an original pinhole. The same injection on the w_0 strips forces entry by walking into (j, s_j) for every j. In particular, until the first visit to $g = (0, L-2)$ all pinholes are intact. A horizontal push into column 0 on a positive row would need either a western pinhole or a player already south of R in a western column. On row 0, the two initially full northeast quadrants of part (b) keep C from reopening. Column 0 thus undergoes only vertical pushes and horizontal deletions of holes, and the red–blue rank bound above forces C to be filled by the visit to g. Removing the excursions through C as in part (b) gives a trajectory inside R reaching its corner c with hole set E at the last departure.

At g the entire western band $\{x < 0,\ 1 \leq y < y^*\}$ is solid, including the two added west columns. To reach a new northernmost row in this band from the east would require a western hole on that row; entry from the south spends a hole in the same column farther north, hence in R (since all intervening unvisited rows are solid), and that hole now lies on a visited row and cannot pay for another advance. Since $y^* - 1 > n_0$, the western player cannot

return to the data rows $y \le 0$, even after entering the southwest extension. The initially full northwest quadrant $\{x < -w_0,\ y < y^*\}$ stays full (its complement meets each row and column in an interval), while the northeast quadrants and the sealed C exclude holes east of R on the data rows. Thus on every row of R, E^+ has no hole outside R, proving (1)(ii). Moreover, the data rows can lose holes only to northward pushes, lowest hole first in each source column. Between first visits to successive pinholes, advancing q_x rows in column x costs at least q_x such holes by part (c) of Lemma 10 (the southwest entry is still intact during the last column's read), and later excursions can only lengthen these deleted prefixes. At the new corner c^+, $E^+ \cap R$ is therefore obtained from E by deleting a bottom prefix of $t_x \ge q_x$ holes in each column x, proving (1)(i) for Push-$*$.

Under PushPush-$*$, a slide of displacement d has the same effect on the full state as d unit Push-$*$ pushes followed by $d-1$ steps back. Applying the strip, rank, and band arguments above to this expansion yields the same last departure from R through C, the same solid data-row exterior (1)(ii), and allows only bottom-up deletions in R thereafter through arrival at c^+. Before that departure, any slide extending past the old boundary of R would consume C or an unvisited pinhole, sealing the exit or violating strip injection; deleting the outside excursions therefore leaves a valid PushPush-$*$ trajectory inside R with the exact same stopping cells, ending at corner state (c, B) with hole set E. After sealing, each physical slide that reaches R consumes a complete maximal hole run of its column and creates its new holes south of R, so $E^+ \cap R$ deletes a bottom prefix of complete maximal runs of total length $t_x \ge q_x$ in each column x.

Part (2): Auxiliary isolation after sweep. On R^+, every column of R_1 and the two added west columns lie in $[-w_0-2, 1]$ (width $q_{\text{aux}} = w_0 + 4$). Any read from the old left taking at least q_{aux} holes on an auxiliary row $y \ge 1$ therefore deletes every hole in $[-w_0-2, 1]$ on row y; every surviving auxiliary hole lies in $\{x > 1,\ y \ge 1\}$, strictly south of R's rows and strictly east of R's columns, hence off every reading line of R in all subsequent reads and clears. In plain mode, this sweep is merged into the next read on $\widetilde{R}$ (whose quotas on R's rows see no holes outside R by (1)(ii)). In clear-preparation mode, step (1) reads only R's rows (again disjoint from auxiliary rows and box-filled outside R by (1)(ii)), step (2) reads quota h_0 from the old top on each selected column $x \in I$ (which exhausts every surviving hole of R in column x, since R has height h_0 and all auxiliary holes lie south of R), steps (3)–(4) have quota 0, and step (5) runs the q_{aux} sweep before any further read inspects R.

Part (3): Canonical forward pass (Push-$*$ and PushPush-$*$). Under Push-$*$, starting from any reachable corner state (c, B) of R with at least q_x holes in every column x, the canonical route of Section 5.1 reaches g_{old} after deleting the exact bottom q_x holes per column of R, steps west into $(-w_0-1, y^*)$, pushes the column stack south into the shaft $[L+1, y_{\text{mid}}]$, and walks down the west corridor to c^+. In clear-preparation mode, before reaching c^+ it visits each of the h_0 extra rows $y \in [y_{\text{mid}}+1, y_{\max}]$ and pushes east w_0 times, leaving one prepared hole directly under every column of R while leaving q_{aux} isolated far-east reserve holes untouched. On every auxiliary row, $[-w_0-2, 1]$ holds at most q_{aux} leftover holes and the far-east reserve holds q_{aux} isolated holes, so the subsequent left read of quota q_{aux} succeeds exactly (and in clear-preparation mode, the h_0 prepared holes below each column of R ensure that step (2)'s top read of h_0 succeeds even if a column in I is empty in R).

Under PushPush-$*$ (plain mode), suppose (c, B) is reachable in R under PushPush-$*$ and its hole set E passes $\{q_x\}$ with exact run cuts. First, the source trajectory on R embeds unchanged into R^+: the corner c stays empty throughout the source trajectory (a box pushed onto c could never leave it), so the only added hole adjacent to R—namely $C = (0, 0)$—could lengthen a source slide only if that slide already occupied c in R alone; on every other row the

east extension starts with a box, and boxes separate the southern pinholes from R, keeping all stopping cells identical. From c, the player walks to $(0, L)$; one northward slide fills $(0, 1)$ and C, vacates $(0, L-1)$ and $g = (0, L-2)$, and leaves the player at $(0, L-1)$, from which one step north reaches g. At each pinhole (j, s_j): if $q_j = 0$, turn west immediately; otherwise, for each successive source hole run of length d, starting from the player's current cell (j, y) (initially (j, s_j)), one northward slide consumes the entire run and vacates d consecutive holes starting at $(j, y-1)$, after which the player walks north through the remaining $d-1$ vacated holes to reach $(j, y-d)$ (still south of R because $y^* - 1 > n_0$). Because the cut at q_j falls between runs, the player reaches $(j, s_j - q_j)$ after deleting the exact q_j holes prescribed on column j, ending at $g_{\text{old}} = (-w_0, y^*)$. Stepping west into $(-w_0 - 1, y^*)$, the box stack in column $-w_0 - 1$ faces the shaft hole run $[L+1, y_{\text{mid}}]$ of length $L - y^* + 1$; one southward slide vacates that many holes starting at $(-w_0 - 1, y^* + 1)$, and the player walks south to $(-w_0 - 1, L+1)$, steps west into the corridor, and walks to c^+. On each auxiliary row of this corner state, if $t \le q_{\text{aux}}$ leftover holes lie in $[-w_0 - 2, 1]$, the next left run read consumes those t holes in complete runs and then $q_{\text{aux}} - t$ isolated far-east reserve holes (each of run length 1, separated by boxes), paying q_{aux} with an exact cut and leaving all remaining auxiliary holes strictly south of R's rows and east of R's columns.

Part (4): Lifting extra deletions to acceptance. For ordinary Push-$*$ reads and clears, suppose the reduced program $\mathcal{P}'$ on $\widetilde{R}$ accepts E^+. By parts (1) and (2), restricting the execution of $\mathcal{P}'$ on E^+ to the sub-rectangle R executes the remaining suffix of $\mathcal{P}$ on $E^+ \cap R$ (in clear-preparation mode, step (1) executes the left read on R's rows, step (2) clears columns I in R, steps (3)–(4) delete zero holes, and step (5) sweeps all auxiliary holes off R's lines before any later read). Thus the remaining suffix of $\mathcal{P}$ accepts $E^+ \cap R$. Since $t_x \ge q_x$ implies $E^+ \cap R \subseteq \text{read}_{\text{bot},q}(E)$, inclusion monotonicity (Lemma 9) implies that the suffix also accepts $\text{read}_{\text{bot},q}(E)$, so $\mathcal{P}$ accepts E.

For PushPush-$*$ run-read programs (no clears), we induct on the number of reads $r \ge 1$ to show that any solution to the reader extracts a PushPush-$*$ corner state E of the original board R admitting a *relaxed run execution*—where each step s and line L of R deletes a prefix of complete maximal runs of total length $t_L^{(s)} \ge q_L^{(s)}$ from the currently remaining holes of E. For the base case $r = 1$ (Lemma 10 under PushPush-$*$), the slide expansion in part (1) applied to R_1 extracts a PushPush-$*$ corner state E of R whose bottom columns lose prefixes of complete runs of lengths $t_x \ge q_x$. For $r \ge 2$, the inductive hypothesis on $\widetilde{R}$ yields a PushPush-$*$ trajectory to (c^+, B^+) whose hole set E^+ admits a relaxed run execution of the reduced $(r-1)$-step program $\mathcal{P}'$. By part (1), this extracts a PushPush-$*$ corner state E of R with bottom prefixes of complete runs of lengths $t_x^{(1)} \ge q_x^{(1)}$ deleted, and no holes outside R on R's rows. In the relaxed run execution of $\mathcal{P}'$ on E^+, step 1 (which reads R's rows and sweeps every auxiliary row with quota q_{aux}) deletes complete run prefixes of lengths $t_y^{(2)} \ge q_y^{(2)}$ on R's rows and empties $[-w_0 - 2, 1]$ on every auxiliary row by part (2). Every remaining hole on every row and column of R now lies strictly inside R, so the remaining steps of $\mathcal{P}'$'s relaxed run execution restrict to complete run prefixes of lengths $t_L^{(s)} \ge q_L^{(s)}$ on R, completing the induction. Finally, these prefixes delete pairwise disjoint subsets of the same extracted hole set E, so when $\mathcal{P}$ is full ($\sum_{s,L} q_L^{(s)} = |E|$) we get $|E| = \sum_{s,L} q_L^{(s)} \le \sum_{s,L} t_L^{(s)} \le |E|$, forcing $t_L^{(s)} = q_L^{(s)}$ for every (s, L) and exact run-read acceptance of E.

Area bound. For $A = w_0 h_0$, we have $L = O(A)$, $y_{\max} \le 2L + h_0 = O(A)$, $N_{\text{total}} = O(w_0\, y_{\max}) = O(A^2)$, and $x_{\text{res}} = O(A^2)$, so $\widetilde{R}$ has width $O(A)$, height $O(A^2)$, and area $O(A^3)$.

Direction normalization and recursion (Lemmas 12 and 13). First normalize an arbitrary four-direction program $\mathcal{P}$ without changing acceptance (or fullness) by merging any consecutive clears of the same orientation into one clear and then inserting zero-quota reads: insert at most three zero-quota reads before the first step (starting with a bottom read so that no clear appears in steps 1 or 2) and at most three zero-quota reads between consecutive steps so that all reads cycle strictly in the order *bottom, left, top, right, bottom, ...*, every clear of columns sits immediately after a *left* read (before a *top* read), and every clear of rows sits immediately after a *top* read (before a *right* read). This multiplies the length of $\mathcal{P}$ by at most 4 (still constant for fixed length). Under a 90° counter-clockwise rotation $(x, y) \mapsto (y, -x)$ after peeling off one read, a clear of rows following a top read becomes a clear of columns following a left read, so both clear orientations are handled by the same rule.

Let r be the number of remaining reads and e the number of remaining clears:

- *Base cases ($e = 0$, $r \leq 1$).* If $r = 0$, the instance is R with player goal c. If $r = 1$ (a single bottom read), apply Lemma 10 directly (and under PushPush-$*$, the canonical one-read slide-and-walk route from part (3) of Lemma 11 gives completeness for any exact-cut corner state).
- *Plain read step ($e = 0$ with $r \geq 2$, or step 3 is not a clear).* Build the rotated extended board $\widetilde{R}$ in plain mode ($e_{\text{rows}} = e_{\text{wid}} = 0$) for step 1, merge quota $q_{\text{aux}} = w_0 + 4$ on every auxiliary row into step 2 (now the first bottom read on $\widetilde{R}$), and recurse on the remaining $r - 1$ reads and e clears.
- *Clear compilation step (step 3 is a clear of column set I, following bottom read 1 and left read 2).* Build $\widetilde{R}$ in clear-preparation mode ($e_{\text{rows}} = h_0$, $e_{\text{wid}} = w_0$) for step 1, and replace the first three steps by five cyclic reads on $\widetilde{R}$: (1) bottom read on $\widetilde{R}$ (= old step 2, left read on R's rows; quota 0 on auxiliary rows), (2) left read on $\widetilde{R}$ (= old top read of quota h_0 on columns in I), (3)–(4) two zero-quota turns (top and right on $\widetilde{R}$), and (5) bottom read on $\widetilde{R}$ of quota $q_{\text{aux}} = w_0 + 4$ on all auxiliary rows (= old left read sweeping $[-w_0 - 2, 1]$). After (5), the next cyclic side is left on $\widetilde{R}$ (= old top on R), already aligned with step 4 of $\mathcal{P}$ (if step 3 was the last step of $\mathcal{P}$, stop after (2)). This removes 2 reads and 1 clear and adds at most 5 reads, mapping $(r, e) \mapsto (r + 3, e - 1)$.

Hence the potential $r + 4e$ decreases by at least 1 at every recursive step, terminating at the base cases after a constant number of steps (each cubing the area), so the final instance has polynomial area and is built in polynomial time. For Lemma 12, completeness follows by induction from Lemma 11(3) and soundness from Lemma 11(1),(2),(4). For Lemma 13 ($e = 0$, full program $\mathcal{P}$), completeness follows by induction on r for programs accepted with exact run cuts: fullness gives exact cuts on R, Lemma 11(3) preserves exact cuts across both R and the auxiliary sweep (and later zero-quota reads on auxiliary lines ignore surviving far-east reserve holes), reaching the $r = 1$ base case; soundness follows directly from Lemma 11(4).

C.2 Scales, fullness of $\mathcal{P}_\tau$, and completeness

Two-box core geometry and closed-form scales. On the working row $y_1 = 2D - Y_1$ of a bottom AND or OR vertex (middle train at local offset 0, outer trains at $-D, D$), the left bar has length $D - \kappa$ ($\kappa = 4$ for OR, $\kappa = 2$ for AND), the two-box short bar occupies $[p + 1, p + 2]$, and the right bar has length $D - 14$, leaving $\kappa + 11$ working-row holes split into $I_G = [l - \kappa + 1, p]$ and $I_J = [p + 3, 13 - r]$. The four public columns at $-2, 6, 8, 12$ are empty at depths $1, \ldots, Y_1 - 1$ (unlike Section 4, both OR and AND include column -2; in AND, $(-2, y_1)$ is always covered by the left bar since $l - 2 \geq -2$); the five-cell private branch is $N = \{(4, y_1 - 1), (1, y_1 - 2), (2, y_1 - 2), (3, y_1 - 2), (4, y_1 - 2)\}$; and columns $0, 4$

hold half-depth separators at row $2D - Y_1/2$. Given incoming edge bits $\iota_l, \iota_m, \iota_r \in \{0, 1\}$, the canonical state sets $l = 2 - \iota_l$, $r = 2 - \iota_r$, $p = 1 + 5\iota_r$, and $v = 1 + \iota_m$. Every ordinary train has 5 holes across its ends, every wire bar has 3 holes, and every middle train has 4 holes strictly above y_1, so all canonical configurations have the same total hole count K.

Let n and m be the number of trains and maximum working level after partner repair (Lemma 18), and take the exact same scales $\bar{n}, q_0, S, Y_j, L_j, D$, board $[0, \bar{n}D] \times [0, 2D]$, central row $M = D$, and budgets b_j, K_j as in Appendices B.1 and B.2; inequalities (1)–(4) and the polynomial area bound of Appendix B.2 hold verbatim. With the two-box short bar ($7n_3$ half-depth and five-cell private-branch holes and $12n+15n_3$ working holes), the non-corridor holes of the marker-free board still total fewer than q_0, so the marker-free interior hole count stays below b_m. The added central markers number $2|U| + |X_{\text{gap}}| + |X_{\text{samp}}| \le 20n + 2 + 2n_3 < q_0$, so the full interior hole count b satisfies $b < b_m + q_0 < 2b_m$. By inequality (4) of Appendix B.2, $D > 4(b_m + L_m + 10)$, so the nearest interior hole to the left or right boundary lies at distance at least $(D - L_m)/2 - 1 > b$, every even-rounded wide-gap X_{gap} column lies at distance at least $D/2 - 3L_m/2 - 2 > b$ from the nearest street column, and the central marker rows lie at distance at least $D - 4 > b$ from the top and bottom boundaries.

Generator pseudocode, fullness, and acceptance of $R \setminus B_\tau$. To specify every side, line set, and quota of $\mathcal{P}_\tau$ (Table 2) unambiguously, Figure 47 gives the generator pseudocode. It maintains a working set S of remaining target holes, initialized to $S \leftarrow R \setminus B_\tau$, and emits operations using four primitives that simultaneously update S:

- READ(side, $L \mapsto q_L$) appends a read from side $\in \{\text{top}, \text{bot}, \text{left}, \text{right}\}$ with quota q_L on each listed line L (and 0 on all other perpendicular lines), and deletes from S the first q_L holes of $S \cap L$ counted from side.
- READALL(side, $\mathcal{L}$) invokes READ(side, $L \mapsto |S \cap L|$ for $L \in \mathcal{L}$).
- READPREFIX(side, $\mathcal{L}$, ϕ_L) sets $q_L = |\{c \in S \cap L : \phi_L(c)\}|$ for each $L \in \mathcal{L}$ (where ϕ_L selects a prefix from side) and invokes READ(side, $L \mapsto q_L$).
- CLEAR($\mathcal{L}$) appends a clear of the lines in $\mathcal{L}$ and sets $S \leftarrow S \setminus \bigcup_{L \in \mathcal{L}} L$.

Here $X_{\text{ord}}, X_{\text{mid}}$ are the ordinary and middle train columns, with $j_T(x)$ the top working level of train x and $\theta_i \in \{0, 1\}$ recording whether τ points train i into its upper end; $U = U_{\text{top}} \cup U_{\text{bot}}$ with $U_{\text{bot}} = U_{\text{bot}}^{\text{open}} \cup U_{\text{bot}}^{\text{priv}}$ ($U_{\text{bot}}^{\text{priv}}$ being the private columns $x_c + 4$); $U_s^{(j)} \subseteq U_s$ ($1 \le j < m$) are the half-s level-j columns with inter-level separator row $z_{s,j} = \text{phys}_s((Y_j + Y_{j+1})/2)$ and $U_s^{(\emptyset)} = U \setminus \bigcup_{j=1}^{m-1} U_s^{(j)}$; $r_{\text{top}}(x) = z_{\text{top},j}$ if $x \in U_{\text{top}}^{(j)}$ ($j < m$) and $M-2$ otherwise (symmetrically $r_{\text{bot}}(x) = z_{\text{bot},j}$ or $M + 2$); Y_{work} is the set of working rows, $Y_{\text{probe}} = \{Y_j + 2 : j \in \mathcal{L}_{\text{top}}\}$, $Y_{\text{mark}} = \{M - 4, M - 2, M + 2, M + 4\}$, and $Y_{\text{adj}} = \{y \pm 1 : y \in Y_{\text{work}}\}$.

We verify the evolution of S column by column across Figure 47:

- *Frame and inner side bands:* Steps 1–2 consume every boundary hole, and Step 3 clears the empty b-column inner side bands (deleting zero).
- *Ordinary trains:* Steps 7–8 delete 2 holes from each end, leaving 1 hole (at top probe row $Y_j + 2$ if $\theta_i = 1$, or at the bottom end if $\theta_i = 0$). Steps 14 and 16 consume θ_i on the probe row and $1 - \theta_i$ on the column, leaving 0 before Step 17 clears the column.
- *Middle trains:* Step 7 deletes 2 holes from the top; Steps 14, 18, and 20 delete θ_i on the probe row, then $2 - \theta_i$ holes directly above y_1, then $(0, y_1)$ if open in B_τ. Only the half-depth separator at $(0, 2D - Y_1/2)$ remains, which Step 28 consumes before Step 29 clears the column.
- *Private columns +4:* Step 10 reads 4 holes from the bottom: the half-depth separator $(4, 2D - Y_1/2)$, the working-row hole $(4, y_1)$ (which lies in I_J when $p = 1$ and in I_G when

Generator for $\mathcal{P}_\tau$ (input: R, B_τ; state: remaining target hole set $S \leftarrow R \setminus B_\tau$)

Stage 1 *(frame, side bands, corridor/public pre-reads;* 5 *steps)*
1. ReadAll(top, $\{x = 0,\ x = \bar{n}D\}$); 2. ReadAll(left, $\{y = 0,\ y = 2D\}$);
3. Clear($\{x \in [1, b] \cup [\bar{n}D - b, \bar{n}D - 1]\}$);
4. ReadPrefix(top, U_{top}, $y < r_{\mathrm{top}}(x)$); 5. ReadPrefix(bot, $U^{\mathrm{open}}_{\mathrm{bot}}$, $y > r_{\mathrm{bot}}(x)$).

Stage 2 *(row clears and train pre-reads;* 4 *steps)*
6. Clear($\{y \notin Y \cup Z \cup Y_{\mathrm{mark}} \cup Y_{\mathrm{adj}}\}$);
7. ReadPrefix(top, $X_{\mathrm{ord}} \cup X_{\mathrm{mid}}$, $y \le Y_{j_T(x)} + 1$) *(quota* 2 *per train)*;
8. Read(bot, $x \mapsto 2$ for $x \in X_{\mathrm{ord}}$); 9. Clear($Y_{\mathrm{adj}} \setminus \{y_1 - 1\}$).

Stage 3 *(private columns, markers* $M \pm 2$*, top* X_{gap} *and marker* $M - 4$*;* 4 *steps)*
10. Read(bot, $x \mapsto 4$ for $x \in U^{\mathrm{priv}}_{\mathrm{bot}}$); 11. ReadAll(left, $\{M - 2,\ M + 2\}$) *(quota* $|U|$ *each)*;
12. ReadPrefix(top, X_{gap}, $y < M - 4$); 13. ReadAll(left, $\{M - 4\}$) *(quota* $|X_{\mathrm{gap}}|$*)*.

Stage 4 *(top probe rows, finishing trains, core cell* 0*;* 7 *steps)*
14. ReadAll(left, Y_{probe}); 15. Clear(Y_{probe});
16. ReadAll(top, X_{ord}) *(quota* $1 - \theta_i$*)*; 17. Clear(X_{ord});
18. ReadPrefix(top, X_{mid}, $y < y_1$) *(quota* $2 - \theta_i$*)*;
19. Clear($\{y \notin Y \cup Z \cup \{M + 4\}\}$); 20. ReadPrefix(top, X_{mid}, $y \le y_1$) *(quota* $\mathbf{1}[(x, y_1) \in S]$*)*.

Stage 5 *(bottom* X_{gap}*, sampling columns* X_{samp}*, remaining street rows;* 6 *steps)*
21. ReadAll(bot, X_{gap}); 22. Clear(X_{gap});
23. ReadPrefix(bot, X_{samp}, $y > M + 4$); 24. ReadAll(left, $\{M + 4\}$) *(quota* $|X_{\mathrm{samp}}|$*)*;
25. ReadAll(left, $Y \setminus (\{0, 2D\} \cup Y_{\mathrm{probe}})$); 26. Clear($\{x \notin U \cup X_{\mathrm{mid}}\}$).

Stage 6 *(bottom separator verification;* $2m + 1$ *steps)*
27. Read(bot, $x \mapsto 1$ for $x \in U^{(\emptyset)}_{\mathrm{bot}}$); 28. ReadAll(left, $\{H - Y_1/2\}$) *(quota* $|X_{\mathrm{mid}}|$*)*; 29. Clear(X_{mid});
for $j = 1, \ldots, m - 1$ **do** ReadAll(left, $\{z_{\mathrm{bot},j}\}$) *(quota* $|U^{(j)}_{\mathrm{bot}}|$*)*; Read(bot, $x \mapsto 1$ for $x \in U^{(j)}_{\mathrm{bot}}$).

Stage 7 *(top separator verification;* $2m - 1$ *steps)*
Clear($U^{(\emptyset)}_{\mathrm{top}}$);
for $j = 1, \ldots, m - 1$ **do** ReadAll(left, $\{z_{\mathrm{top},j}\}$) *(quota* $|U^{(j)}_{\mathrm{top}}|$*)*; Clear($U^{(j)}_{\mathrm{top}}$).

Figure 47. Explicit generator for the seven-stage reading program $\mathcal{P}_\tau$, maintaining the remaining hole set S of $R \setminus B_\tau$ ($4m + 26$ steps total: $3m + 17$ reads and $m + 9$ clears).

$p = 6$), and the two branch cells $(4, y_1 - 1), (4, y_1 - 2)$. Only the inter-level separator (if any), markers $(4, M \pm 2)$, and central separator $(4, M)$ remain for Steps 11 and Stages 6–7.

- *Markers and $X_{\text{gap}}, X_{\text{samp}}$:* Step 11 reads $M \pm 2$ ($|U|$ each), Step 12 reads X_{gap} above $M - 4$, and Step 13 reads $M - 4$ ($|X_{\text{gap}}|$). Steps 21–24 finish and clear X_{gap} from the bottom, read X_{samp} strictly below $M + 4$ (consuming $(10, y_1)$ if open on $x_c + 10$, and $(2, y_1 - 2) \in N$ together with $(2, y_1)$ if open on $x_c + 2$), and read $M + 4$ ($|X_{\text{samp}}|$). Every remaining hole on Y—wire bars (two per bar after Stage 1's bar-corridor pre-reads), remaining AND/OR core cells, bottom horizontal train-corridor legs, the remaining private-branch cells $(1, y_1 - 2), (3, y_1 - 2)$, and deeper bar-side holes $(x_i + \varepsilon, \text{phys}_s(d))$—is consumed by Step 25, so Step 26 clears all columns outside $U \cup X_{\text{mid}}$ deleting zero.
- *Remaining U columns:* Entering Stage 6, each $x \in U$ holds only its central separator (x, M) and at most one inter-level separator per half. Stage 6 consumes every bottom inter-level separator (in row order) and every (x, M) (one bottom read per column in U); Stage 7 consumes every top inter-level separator and clears each group.

By construction, every read consumes exactly its quota from S, every clear acts on lines disjoint from S (deleting zero), and $S = \emptyset$ after Stage 7. Hence $\mathcal{P}_\tau$ accepts $R \setminus B_\tau$, its quotas sum to $|R \setminus B_\tau| = K$ (fullness), and the total number of steps is $5+4+4+7+6+(2m+1)+(2m-1) = 4m + 26 = O(m)$.

Completeness (Lemma 15). Given an ordered NCL path from σ to τ in the repaired graph G (obtained from a path $\sigma_H \to \tau_H$ in H by Appendix A.3), we simulate each single-train reversal $\rho \to \rho'$ from (p_0, B_ρ) to $(p_0, B_{\rho'})$ by the same three-step schedule as in Appendix B.4: retract the internal bar or short bar at the losing endpoint, push the train one cell, and advance the bar or short bar at the gaining endpoint. Each reversal uses at most four basic moves, at most one being the five-push short-bar macro of Figure 24 (moving $[p + 1, p + 2]$ between $p = 1$ and $p = 6$), hence at most eight pushes. Wire pockets are entered through the train corridor branch (depth $Y_j + 2$) and bar corridor, as in Section 4.2. At an AND/OR core, $\gamma_H = 1$ ($p = 6$) joins I_G to public column 6 and I_J to 12, while $\gamma_H = 0$ ($p = 1$) joins I_J to public columns $6, 8$ and, through the working-row cell $(4, y_1) \in I_J$ below N, joins N to the frame; an ordered move lets an AND vertex acquire $\gamma_M = 1$ only when $\gamma_L = 1$ (and retract γ_L only when $\gamma_M = 0$, so $\gamma_H = 1$), ensuring access to V and I_G at every step. All central marker holes lie at rows $M \pm 2, M \pm 4$ on columns outside the canonical push rays (and the even-rounded wide-gap X_{gap} separators stay inside their wide background gaps), so every canonical walk and first-hole push remains legal on the marked board. Once (p_0, B_τ) is reached, the player walks along the empty frame to c and executes the canonical pass of Lemma 12 for $\mathcal{P}_\tau$.

C.3 Separator certification and level-by-level freezing

Inter-level and half-depth separators (Step 2). In Stage 6, bottom inter-level separator rows are read from shallowest to deepest. Consider level j:

- Columns in U without a bottom inter-level separator had their single bottom read at the start of Stage 6, and columns of shallower levels $1, \ldots, j - 1$ had theirs right after their own separator row was read; each of those reads consumed the intact central separator (x, M) (leaving no remaining unconsumed holes of E on that column below M), so none of those columns can contribute to the level-j row.
- Every corridor column of a deeper level $> j$ was pre-read from the bottom in Stage 1 by at least $Y_{j+1} - 1$ holes. Since the level-j inter-level row lies at depth strictly less than $Y_{j+1} - 1$ from the bottom frame, by rank from the bottom no unconsumed hole of E

remains on those columns at or closer to the bottom frame than the level-j inter-level row.

- Private columns +4 belong only to level 1.

Thus only the U columns of level j can pay on this row, at most one hole each, and the quota equals their count; hence every bottom inter-level separator survives in E. Stage 7 applies the same argument to the top half (clearing each level's columns immediately after reading its row).

For the bottom half-depth row $2D - Y_1/2$ in Stage 6, every bottom corridor and public column has already pre-read past $Y_1/2$ in Stage 1 or spent its central separator; every private column +4 already paid 4 holes from the bottom in Stage 3 (and Stage 2's zero-deleting row clears confine any holes of E below y_1 on column +4 to the single half-depth cell), so column +4 has no unconsumed hole left at $2D - Y_1/2$ either. Only the middle trains remain able to pay on row $2D - Y_1/2$, one hole each, matching the quota and certifying every middle train's half-depth separator in E. Finally, on each private column +4, its inter-level separator above y_1 survives to Stage 6 (after partner repair (P2), any board with an AND or OR vertex has $m \geq 2$; even if $m = 1$, marker $M + 2$ survives until Stage 3 Step 11, still after Step 10, and row $M + 4 \in Y_{\text{mark}}$ is hole-free on column +4 by Step 1), so Stage 3's four bottom reads must come strictly below that separator; Stage 2's zero-deleting row clears leave only four cells in that interval where E can hold a hole (the half-depth separator, $(4, y_1)$, and $(4, y_1 - 1), (4, y_1 - 2)$), certifying all four cells as holes in E.

First-move rule and global freezing budgets (Step 3). Suppose a set S_{box} starts full of boxes, and every maximal horizontal or vertical segment of S_{box} has at least one endpoint adjacent to a known blocking cell (either a permanently empty blocking hole or an already fixed box). Consider the first push along π that moves a box of S_{box}. It cannot start with the player inside S_{box} (since S_{box} is still full). Along its line of action, it must act along a maximal segment of S_{box}; pushing toward the blocking endpoint would either push that fixed box or fill that blocking hole, while pushing away from it would require the player to stand on the blocking cell (or include it in the pushed line), both impossible. Hence no box of S_{box} ever moves. (When applied to a one-cell-wide street segment, its side neighbours off the streets are already fixed by Step 2.)

In the level-j stage ($j = m, \ldots, 1$), after the level-j train corridors and horizontal branches join A, the remaining uncertified corridors consist of at most $3n$ bar/public corridors of depth $\leq Y_j + 2$, at most $2n$ shallower train corridors of depth $\leq Y_{j-1} + 2$, and at most $2n$ shallower horizontal branches of length $\leq L_{j-1}$ (zero when $j = 1$). Since $2(Y_{j-1} + 2) \leq Y_j + 2$ for $j > 1$, the total number of holes outside A is at most $K_j = q_0 + 4n(Y_j + 2) + 2nL_{j-1}$. For each level-$j$ midpoint column $x_{\text{mid}} = (f + x_i)/2 \in X_{\text{gap}}$, every street-row cell at depth $d > Y_j + 2$ is already occupied in E' (either fixed in the wide-gap step, or lying on a deeper level-j' bar with $j' > j$ at Manhattan distance $> K_j$ from A, hence occupied throughout π by Lemma 8), and depth $Y_j + 2$ is already in A. Hence Stage 3 Step 12 (top) / Stage 5 Step 21 (bottom) forces every level-j midpoint separator ($0 < d \leq Y_j$) to survive in E'; by inequality (2) of Appendix B.2, each lies at distance at least $L_j/2 - L_{j-1} - 12 > K_j$ from any street column and is therefore blocking by Lemma 8 (along its street row). After level-j bar corridors join A, the holes outside A drop below $q_0 + 5n(Y_{j-1} + 2) + 2nL_{j-1} < b_{j-1}$, and by inequality (3) of Appendix B.2 the level-$(j-1)$ inter-level separators lie at distance at least $(Y_j - Y_{j-1})/2 - 2 > 2b_{j-1} + 6 > b_{j-1}$ from the ends of their segments, making them blocking for level $j - 1$. At level 1, the core-freezing argument of Appendix B.2 applies verbatim with the two-box set F_{core} consisting of offsets $[-2, -1]$ and $[5, 12]$ on row $d = Y_1 + 2$ (column

$+10 \in X_{\text{samp}}$ is off-street in the bottom half) and depths $Y_1 + 1, Y_1 + 2$ on all four public columns $-2, 6, 8, 12$ (for both OR and AND); this adds all public columns to A. Fewer than q_0 holes then remain outside A, and $Y_1/2 = 4q_0 > q_0$ freezes columns 0 and 4 below y_1 and then the five-cell private branch N; every deep crossing (on bars and on inner trains at deeper street rows) and every remaining deep background segment between an inter-level separator and the central row lies farther than q_0 from A ($\min((Y_{j+1} - Y_j)/2 - 2,\ D - 2L_m,\ D - Y_m - 2) > q_0$) and therefore stays occupied throughout π, pinning the remaining deep background boxes by identity as in Appendix B.2.

C.4 Occupancy strings and partner repair

Half-train occupancy strings (Step 4). For any train i, its midpoint (x_i, M) lies at distance $D > b$ from the frame and starts occupied, so (x_i, M) holds a box throughout π. Consider the upper half of column x_i from depth 1 down to M. We show by induction on pushes that its occupancy string (0 for hole, 1 for box) has the form

$$0^s\, 1^{u_i - 1}\, 0^g\, 1^q \qquad (s, g \geq 0,\ q \geq 1),$$

where the block $1^{u_i - 1}$ is tagged as the outer box run (even when $g = 0$ momentarily joins it with 1^q). Initially $s = 0$, $g \in \{2, 3\}$, and $q \geq 1$. Horizontal pushes across deep crossings preserve occupancy, while all other side neighbours are either fixed background or permanently empty corridors, so no horizontal push alters the string. A vertical push along column x_i translates a maximal contiguous run of boxes by 1 cell: if $g > 0$, pushing the outer run down increases s by 1 and decreases g by 1 (and pushing it up does the reverse), while pushing the inner run across M changes g and q without altering the outer run; if $g = 0$, pushing the combined run translates the tagged block $1^{u_i - 1}$ rigidly. Because the s holes above the outer run are interior holes, $s \leq b_m$, and $b_m + u_i - 1 < M$, so the tagged outer run never crosses M, keeping $q \geq 1$.

In Stage 2, $\mathcal{P}_\tau$ clears depth 1 (just inside the top frame) before reading any train column; fullness forces $s = 0$ in E. Thus the upper gap 0^g begins at depth $u_i = Y_j$. Stage 2 also clears row $Y_j + 3$, forcing $g \leq 3$ and confining 0^g to $\{Y_j, Y_j + 1, Y_j + 2\}$; because all other un-cleared rows are disjoint from these three cells, the remaining probe-row hole count z_i entering Stage 4 ($g_i - 2$ for a middle train) comes strictly from $(x_i, Y_j + 2)$. Symmetrically, for the lower half of an ordinary train i (working row y_B), the same induction from M down to $2D - 1$ with Stage 2 clearing rows $2D - 1$ and $y_B - 3$ gives $s_{\text{bot}} = 0$ and $g_{\text{bot}} \leq 3$ in E (confining its lower gap to $\{y_B, y_B - 1, y_B - 2\}$). For a middle train i, the lower half of column x_i from M down to $y_1 - 1$ (one cell above the bottom working row) starts as $1^d 0^{h_i}$ ($d \geq 1$, $h_i \in \{1, 2\}$) and maintains this form by the same induction (since it cannot gain boxes from y_1, boxes can only slide down into y_1, keeping all holes directly above y_1); Stage 2 clears row $y_1 - 3$, forcing $h_i \leq 2$.

Proof of Lemma 18 (partner repair). Each ear or cut replaces an upper wire by an alternating path of upper and lower wires, preserving every AND/OR vertex and every original incidence without creating pure-wire cycles; every original edge of H remains subdivided into a chain of $d \geq 2$ trains separated by wires, so the cursor encoding of Appendix A.3 applies directly to the repaired layout and preserves reachability. After all ears and cuts, every middle train either kept an already safe partner or meets a newly inserted port whose lower end belongs to a newly created bottom wire, establishing (P1).

Ears. An ear replaces a top leaf wire by a two-level nested pair; any other upper wire either encloses both or is disjoint from them, and ears of different AND/OR vertices are

disjoint, so the top nesting depth increases by at most 1, to $m'_T \le m_T + 1$. In the bottom half, the new wire p–q sits directly beside the contiguous AND/OR block as a sibling leaf sharing the exact same strict ancestors, so the bottom nesting depth does not increase ($m'_B \le m_B$). At most n_3 ears add at most $2n_3$ ports.

Cuts. Each middle train has one partner, so each AND/OR vertex triggers at most one cut; merging duplicate cuts leaves at most n_3 cuts, none splitting an AND/OR block. At any cut, the t active upper wires crossing it form a nested chain, so $t \le m'_T$. Inserting $\mathsf{p}_t, \ldots, \mathsf{p}_1, \mathsf{q}_1, \ldots, \mathsf{q}_t$ closes the active arcs inside-out on the left and reopens them outside-in on the right, so no new gap has more than t active upper arcs; thus the top nesting depth stays at most m'_T. In the bottom half, the new wires p_j–q_j ($j = 1, \ldots, t$) form a local nested chain of length $t \le m'_T$, disjoint from every other cut block and every AND/OR block, so any bottom nesting chain consists of original ancestors plus at most one cut chain, giving bottom nesting depth at most $m'_B + m'_T \le m_B + m_T + 1$. Cuts add at most $2n_3 m'_T \le 2n_3(m_T + 1)$ ports, so the total port count is $n \le n_{\text{init}} + 2n_3 + 2n_3(m_T + 1) = n_{\text{init}} + 2n_3(m_T + 2)$.

Parity levels. For each vertex v, let $d(v)$ be its nesting depth within its half ($d(v) = 1$ for leaves). Assign levels from the inside out: bottom AND/OR vertices (which are leaves) get level 1; every other vertex v gets the smallest level that exceeds the levels of all vertices nested inside it and has the parity or range required by (P2), namely even for a top-half wire containing a middle train, odd for any other top-half wire, and at least 2 for a bottom wire. Strict nesting holds by construction. By induction on $d(v)$, every top-half wire gets level at most $2d(v)$ (its children have levels at most $2d(v) - 2$, and one of $2d(v) - 1, 2d(v)$ has the required parity), and every bottom wire gets level at most $d(v) + 1$ (its children have levels at most $d(v)$). For example, in the ear of Figure 27(b) the wire q–a gets level 1 and the enclosing wire b–p, which holds the middle train b, gets level 2. This establishes (P2), with maximum working level

$$m \;\le\; \max(2m'_T,\; m'_B + m'_T + 1) \;\le\; \max(2m_T + 2,\; m_T + m_B + 2).$$

C.5 Decoding arithmetic and five-trap case analysis

Cell-by-cell certification and interval decoding (Step 5). Fix an AND or OR vertex with local working-row offsets $-2, 0, 2, 4, 6, 8, 10, 12$. Every hole of B_τ among these eight cells is forced to be a hole in E:

- Offsets $-2, 6, 8, 12$ (public columns): Stage 1 pre-reads each public column x from the bottom frame up to just before $r_{\text{bot}}(x)$ (its level-1 inter-level separator, or marker $M + 2$), and the only target holes in that range are the corridor below y_1 and possibly (x, y_1). In E the corridor is permanently empty (Step 3), and the cells strictly between y_1 and $r_{\text{bot}}(x)$ hold boxes: depths $Y_1 + 1, Y_1 + 2$ belong to the frozen set F_{core}, and the deeper cells lie on the background segment fixed by the blocking separator at $r_{\text{bot}}(x)$ (Appendix C.3). Since that separator survives to Stage 6, a target hole at y_1 raises the pre-read quota by 1 and forces that working-row cell to be empty in E.
- Offsets $2, 10$ (sampling columns X_{samp}): Stage 5 reads them from the bottom before reading the intact marker at $M + 4$, after Stage 4 has cleared all non-street rows below M. By Steps 2–4, every street cell below $M + 4$ on these columns outside y_1 and $y_1 - 2$ (other than the already-consumed bottom-frame cell $(x, 2D)$) holds a box in E (background boxes are fixed, and any deeper bottom wire bar crossing x_c keeps its boxes with its three holes within three cells of its own trains). Thus on offset 10 only $(10, y_1)$ can be a hole of E below $M + 4$, while on offset 2 only $(2, y_1)$ and the permanently empty private-branch cell $(2, y_1 - 2) \in N$ can be holes of E below $M + 4$; hence a target hole at $(10, y_1)$ (quota 1)

or $(2, y_1)$ (quota 2) forces that cell to be a hole in E.

- Offset 4 (private column): certified empty in Appendix C.3 (Stage 3's four bottom reads below its inter-level separator).
- Offset 0 (middle train): by Step 4, its $g_i + h_i = 4$ holes strictly above y_1 are exactly paid by Stage 2 and Stage 4's first reads, and its half-depth separator below is still required in Stage 6. Thus when $(0, y_1)$ is a hole in B_τ, Stage 4's extra top read of 1 hole must consume $(0, y_1)$.

Similarly, when B_τ has 2 holes at an outer end ($l^* = 2$ or $r^* = 2$), the bar corridor meets the second hole from the train, so Stage 1's pre-read on that bar corridor forces $l \geq 2$ or $r \geq 2$ in E.

Now let $I_G = [l - \kappa + 1, p]$ and $I_J = [p + 3, 13 - r]$ with $l, r \geq 0$ and $l - \kappa \leq p \leq 11 - r$.

- *Bit $\gamma_R = [r \leq 1]$:* Since $p \leq 11 - r \leq 11$, cell 12 lies strictly right of $I_G = [l - \kappa + 1, p]$, so when cell 12 is certified empty it cannot lie in I_G or any box run (including $[p + 1, p + 2]$) and must lie in I_J. If $\gamma_R^* = 1$, cell 12 is certified empty, so $13 - r \geq 12$, i.e., $r \leq 1$; if $\gamma_R^* = 0$, the bar-corridor pre-read gives $r \geq 2$.
- *Bit $\gamma_L = [l \leq 1]$:* Set $x_L = 2 - \kappa$ (-2 for OR, 0 for AND). Since I_J begins at $p + 3 \geq l - \kappa + 3 \geq 3 - \kappa > x_L$, cell x_L can lie only in I_G. If $\gamma_L^* = 1$, cell x_L is certified empty, so $l - \kappa + 1 \leq 2 - \kappa$, i.e., $l \leq 1$; if $\gamma_L^* = 0$, the bar-corridor pre-read gives $l \geq 2$.
- *Bit $\gamma_H = [p \geq 6]$:* Any certified empty cell x excludes $p \in \{x - 2, x - 1\}$ (since the short bar occupies $[p + 1, p + 2]$). If $\gamma_H^* = 1$, certified holes $2, 4, 6$ exclude $[0, 5]$. For $p \in [-2, -1]$: in an OR, $(0, y_1)$ is certified empty and excludes $[-2, -1]$; in an AND, if $l \geq 2$ then $p \geq l - 2 \geq 0$ automatically, while if $l \leq 1$ then $\gamma_L^* = 1$ certifies $(0, y_1)$ and excludes $[-2, -1]$. Finally, $p \in [-4, -3]$ requires $l - \kappa \leq -3$, which occurs only for OR ($\kappa = 4$) with $l \leq 1$; then $\gamma_L^* = 1$ certifies -2 and excludes $[-4, -3]$. Hence $p \geq 6$. If $\gamma_H^* = 0$, certified holes $8, 10$ exclude $[6, 9]$; in canonical B_τ we also have $\gamma_R^* = 0$, so the bar-corridor pre-read forces $r \geq 2$ and hence $p \leq 11 - r \leq 9$. Hence $p \leq 5$.
- *Non-emptiness of I_G and I_J:* If $\gamma_H^* = 1$, then $p \geq 6$, so certified hole 2 lies in I_G and certified hole 12 lies in I_J. If $\gamma_H^* = 0$, then $p \leq 5$, so certified hole 8 lies in I_J. For I_G: if $\gamma_L^* = 1$, then $x_L \in I_G$; if $\gamma_L^* = 0$ and $\gamma_H^* = 0$, the vertex must be a satisfied OR (an AND with $\gamma_H^* = \gamma_L^* = 0$ violates its constraint), so $l \geq 2$, $p \geq l - 4 \geq -2$, and $(0, y_1)$ is certified empty; since I_J starts at $p + 3 \geq 1$, cell 0 cannot lie in I_J and must lie in I_G.

Hence both I_G and I_J are nonempty in E, proving that the three runs on the working row never merged during π.

Five cut-off shapes and first-violation coverage (Step 6). At an ordinary wire or outer AND/OR end, let $a \in \{1, 2, 3, 4\}$ be the number of holes at the train end (as in Appendix B.3: $a \geq 3$ means the train points in, while $a \leq 2$ confines the train gap to offsets $\{0, 1\}$) and h the bar-end hole count. The five cut-off conditions (each paired with "the player is outside the cut-off region"; Figure 29) are:

$\mathcal{T}_{\text{pkt}}$ *Ordinary pocket* ($a \leq 2$ and $h \leq 1$): the train gap lies in offsets $\{0, 1\}$ and the bar has at most 1 hole, reaching neither the train corridor nor the bar corridor.

$\mathcal{T}_{\text{rel}}$ *Relay gap* ($p \geq 6$ and $r \geq 2$): the nonempty gap $I_J = [p + 3, 13 - r] \subseteq [9, 11]$ misses every public column $(-2, 6, 8, 12)$, the private branch (4), and the middle train (0).

$\mathcal{T}_{\text{OR}}$ *OR core* ($l \geq 2$, $p \leq 5$, $v \leq 1$): the nonempty core $I_G = [l - 3, p] \subseteq [-1, 5]$ misses every public column; $I_G \neq \emptyset$ gives $p \geq l - 3 \geq -1$, ensuring $0 \notin I_J$; and $v \leq 1$ disconnects V from N. If $4 \in I_G$, then N is reachable only through I_G (so $I_G \cup N$ is cut off); otherwise N is separated from I_G.

$\mathcal{T}_{\text{mid}}$ *AND core without middle* ($p \le 5$, $v \le 1$): since an AND core has left end $l - 1 \ge -1$, $I_G \subseteq [-1, 5]$ is cut off by the exact same argument as $\mathcal{T}_{\text{OR}}$.

$\mathcal{T}_{\text{left}}$ *AND core without left* ($p \le 5$, $l \ge 2$): $I_G = [l - 1, p] \subseteq [1, 5]$, so the left bar occupies cell 0, blocking V and N from reaching I_G through 0 even if $v \ge 2$. If $4 \in I_G$, then N (and V if $v \ge 2$) joins the cut-off region because its working-row exit returns only to I_G and cell 0 is occupied; if $4 \notin I_G$, then I_G is separated from N and V.

In every case, restoring any violated inequality requires enlarging a gap bordering the cut-off region (increasing a, h, or v, decreasing l or r, or shifting p across $5 \leftrightarrow 6$), which requires pushing a train, bar, or short bar *away* from the cut-off region—possible only by pushing from inside it; pushes from outside can only push those runs further in. Thus each condition is invariant while the player remains outside.

Every first violation at a vertex u (with the player pushed to the far end of the reversed train, hence outside u) hits one of these five shapes:

- *Wire:* neither train points in ($a_1, a_2 \le 2$). The two bar ends sum to 3 holes, so at least one end has $h \le 1$, hitting $\mathcal{T}_{\text{pkt}}$.
- *OR:* all three trains point away ($a_l, a_r \le 2$ and $v \le 1$). If $l \le 1$ or $r \le 1$, that outer pocket hits $\mathcal{T}_{\text{pkt}}$. Otherwise $l, r \ge 2$; then $p \ge 6$ hits $\mathcal{T}_{\text{rel}}$ and $p \le 5$ hits $\mathcal{T}_{\text{OR}}$.
- *AND:* the heavy train points away ($a_r \le 2$) and at least one light train points away ($a_l \le 2$ or $v \le 1$). If $r \le 1$, the right pocket hits $\mathcal{T}_{\text{pkt}}$. Otherwise $r \ge 2$; if $p \ge 6$, this hits $\mathcal{T}_{\text{rel}}$. For $p \le 5$: if $v \le 1$, this hits $\mathcal{T}_{\text{mid}}$; if $a_l \le 2$, then $l \le 1$ hits $\mathcal{T}_{\text{pkt}}$ on the left and $l \ge 2$ hits $\mathcal{T}_{\text{left}}$.

Step 5 decodes a final state satisfying every vertex constraint and $\gamma_H^* = \gamma_R^*$, which avoids all five shapes—contradiction.

D Details for PushPush-kS and PushPush-$*$

This appendix records the exact coordinates, walkability proof, door transition verification, exhaustive wrong-push case analysis, and seven-round quota accounting omitted from Section 6.

D.1 Coordinates, room layout, and walkability

Let an instance of directed symmetric self-closing doors have L locations $\{0, \dots, L-1\}$, d doors $(s_0, t_0, s_1, t_1, \text{bit})$, start location v_{start}, and target location v_{target}. We first assign local row coordinates starting at $g = 0$ (north partition of the first room) and then apply a global vertical shift.

Door rooms. For each door $i \in \{1, \dots, d\}$ with north partition g_i and south partition $g_i + 90$, the six base slots lie at

$$g_i + 10, \quad g_i + 30, \quad g_i + 40, \quad g_i + 50, \quad g_i + 60, \quad g_i + 80,$$

with valid output sides $+1, -1, -1, +1, +1, -1$ respectively, and input-pin sides ε_3 equal to the negatives $-1, +1, +1, -1, -1, +1$. Each slot row joins Y_P; the initial stored box sits at $(c, g_i + 30)$ if bit $= 0$ and at $(c, g_i + 60)$ if bit $= 1$. Four internal wires connect $(g_i + 39) \to (g_i + 80)$, $(g_i + 79) \to (g_i + 60)$, $(g_i + 51) \to (g_i + 10)$, and $(g_i + 11) \to (g_i + 30)$; external wires link $(g_i + 61)$ to an input of t_0, $(g_i + 29)$ to an input of t_1, an exit of s_0 to $(g_i + 40)$ (when $s_0 \ne v_{\text{target}}$), and an exit of s_1 to $(g_i + 50)$ (when $s_1 \ne v_{\text{target}}$; without loss of generality v_{target} has no outgoing passages). In the storage construction of Section 6.3, we

also add the cleanup slot $E_i = g_i + 70$ to Y_P with valid output row $E_i - 1 = g_i + 69$ (side -1) and input-pin side $\varepsilon_3 = +1$.

Location and junction rooms. A junction room with $\ell \geq 0$ inputs and $r_{\text{out}} \in \{0, 1, 2\}$ exits has north partition g, south partition $g' = g + 20 + 8\ell$, and input rows $g + 10 + 8j$ $(0 \leq j < \ell)$, all placed in Y_P with input-pin side $\varepsilon_3 = +1$. If $r_{\text{out}} \geq 1$, it places a north cap at $g + 2 \in Y_{\text{cap}}$ with valid output row $g + 3$ (side $+1$); if $r_{\text{out}} = 2$, it also places a south cap at $g' - 2 \in Y_{\text{cap}}$ with valid output row $g' - 3$ (side -1). For each location v:

- its primary room has one input per door passage ending at v, plus one extra input row holding the initial signal box if $v = v_{\text{start}}$;
- if $v = v_{\text{target}}$, it has $r_{\text{out}} = 1$ (north cap only) and its exit $z = g + 3$ is the target parking row;
- if $v \neq v_{\text{target}}$ and v has q outgoing door passages, we set $r_{\text{out}} = \min(2, q)$ and, when $q > 2$, recursively split the q destinations into two balanced halves, wiring a half of size 1 directly to its destination and feeding any half of size ≥ 2 into a 1-input, 2-output auxiliary junction room until each leaf has at most 2 destinations.

In the storage construction, we append the d cleanup wires $(c, z) \to (c, E_1)$ and $(c, E_i - 1) \to (c, E_{i+1})$ $(1 \leq i < d)$, and when $d > 0$ the signal ends at the final signal cell $(c, E_d - 1)$; the target parking row is still z. If N is the encoding length of the source instance, a location of out-degree $q \geq 2$ adds $q - 2$ auxiliary rooms, so the binary choice trees add at most $2d$ auxiliary junction rooms; the total number of rooms, input rows, and wires is $O(N)$, and the final partition coordinate $g_{\max}$ is $O(N)$.

Global dimensions and wire pins. Let n be the total number of wires. Set

$$\text{shift} = 20n{+}15, \qquad P = 30n{+}40, \qquad c = P{-}1, \qquad w_0 = P{+}6, \qquad h_0 = g_{\max}{+}2\,\text{shift}{+}1,$$

so $\mu = w_0 - 2 - P = 4$, and increase every room row coordinate by shift. We also add $g \pm 1$ to Y_P for every $g \in Y_{\text{part}}$. For wire $i \in \{0, \ldots, n-1\}$ connecting shifted output row r to shifted input row z' with sides $\varepsilon_1, \varepsilon_3 \in \{-1, +1\}$, define its three private columns and two private rows by

$$x_{i,1} = 30i + 8, \qquad x_{i,2} = 30i + 15, \qquad x_{i,3} = 30i + 24,$$

$$y_{i,1} = \text{level}(2i,\ \varepsilon_1 = +1), \qquad y_{i,2} = \text{level}(2i + 1,\ \varepsilon_3 = -1), \qquad \varepsilon_2 = \text{sign}(y_{i,2} - y_{i,1}),$$

where $\text{level}(j, \text{true}) = g_{\max} + \text{shift} + 10 + 10j$ (south of all rooms) and $\text{level}(j, \text{false}) = 5 + 10j$ (north of all rooms). Notice that $\varepsilon_1 = +1$ puts $y_{i,1}$ south of all rooms ($\text{sign}(y_{i,1} - r) = +1 = \varepsilon_1$) and $\varepsilon_1 = -1$ puts $y_{i,1}$ north of all rooms ($\text{sign}(y_{i,1} - r) = -1 = \varepsilon_1$); likewise $\varepsilon_3 = -1$ puts $y_{i,2}$ south of all rooms ($\text{sign}(z' - y_{i,2}) = -1 = \varepsilon_3$) and $\varepsilon_3 = +1$ puts $y_{i,2}$ north of all rooms ($\text{sign}(z' - y_{i,2}) = +1 = \varepsilon_3$). Because $0 \leq 2i < 2i + 1 < 2n$, all private rows lie in $[5,\ 20n - 5]$ or $[g_{\max} + \text{shift} + 10,\ h_0 - 16]$, separated by at least 10 rows from each other, at least 9 rows from every room and rack row, and at least 5 rows from the frame rows 0 and $h_0 - 1$. All $6n$ wire pins

$$p_1 = (x_{i,1} - 1, r), \quad p_2 = (x_{i,1}, y_{i,1} + \varepsilon_1), \quad p_3 = (x_{i,3} + 1, y_{i,1}),$$

$$p_4 = (x_{i,3}, y_{i,2} + \varepsilon_2), \quad p_5 = (x_{i,2} - 1, y_{i,2}), \quad p_6 = (x_{i,2}, z' + \varepsilon_3)$$

therefore have pairwise distinct x-coordinates in $\{7, \ldots, 30n - 5\}$ and pairwise distinct y-coordinates disjoint from $Y_P \cup Y_{\text{part}} \cup Y_{\text{cap}}$ (and, without the cleanup wires, also disjoint from the target parking row; on the storage board, when $d > 0$, only the first cleanup wire's pin p_1 lies on the target parking row). The initial player stands at the lower right corner $(w_0 - 1, h_0 - 1)$, and the board has width $O(N)$, height $O(N)$, and area $O(N^2)$.

Walkability. At any point during canonical play (and at the start and end), every occupied column belongs to one of the pairwise separated two-column bands $(x_{i,1}-1, x_{i,1})$, $(x_{i,2}-1, x_{i,2})$, $(x_{i,3}, x_{i,3}+1)$, or (c, P), and row 0 is completely empty. For any column x, at least one of x, $x-1$, or $x+1$ is an entirely empty column (the nearest *lane* $\lambda(x)$). To walk from any empty cell (x, y) to any target pushing cell (u, v), the player walks horizontally at most one step to $(\lambda(x), y)$, north to $(\lambda(x), 0)$, horizontally along row 0 to $(\lambda(u), 0)$, south to $(\lambda(u), v)$, and horizontally at most one step to (u, v). Every support cell required along the canonical trajectory is empty:

- inside a room, two active slots on column c are separated by at least 8 rows, so the vertical pushing cell behind the pushed box is empty; to push a signal west from an output row r, the player stands at (P, r), which is empty since $r \notin Y_P$;
- at each of the six turns of wire i, the pin that just stopped the signal lies in the direction of the *incoming* slide, whereas the next slide is perpendicular, so the player stands at $(x_{i,1}, r-\varepsilon_1)$, $(x_{i,1}-1, y_{i,1})$, $(x_{i,3}, y_{i,1}-\varepsilon_2)$, $(x_{i,3}+1, y_{i,2})$, $(x_{i,2}, y_{i,2}-\varepsilon_3)$, and $(x_{i,2}-1, z')$ respectively—all empty because no other box shares their column or row;
- after parking at (c, z), the same five-segment route reaches (w_0-1, h_0-1) via column w_0-1.

D.2 Door transitions and exhaustive wrong-push case analysis

Door room transitions. Under the backbone invariant, a door room with north partition g and south partition $g+90$ contains one stored box on column c at a slot $m \in \{10, 30, 40, 50, 60, 80\}$ (relative to g). When a signal arrives at an input slot $e \neq m$, sliding one box onto the other produces the following outcomes:

In e	Stored m	Push $m \to e$ (store e)	Push $e \to m$ (store m)	Outcome
40 (check α)	30 (α)	emit $39 \to 80$	emit 31 (dead)	Passage 0, step 1
80 (high)	40	emit $79 \to 60$	emit 41 (dead)	Passage 0, step 2
60 (β)	80	emit $61 \to t_0$	emit $79 \to 60$ (retry)	Passage 0, commit
40 (check α)	60 (β)	emit 41 (dead)	emit 59 (dead)	Blocked (wrong state)
50 (check β)	60 (β)	emit $51 \to 10$	emit 59 (dead)	Passage 1, step 1
10 (low)	50	emit $11 \to 30$	emit 49 (dead)	Passage 1, step 2
30 (α)	10	emit $29 \to t_1$	emit $11 \to 30$ (retry)	Passage 1, commit
50 (check β)	30 (α)	emit 49 (dead)	emit 31 (dead)	Blocked (wrong state)

Notice that all four reachable dead rows $(31, 41, 49, 59)$ lie among the input-pin sides $z'+\varepsilon_3$ of door slots, and that the retry at step 3 of either passage preserves the stored staging box and re-enters the exact same commit slot, so every reachable arrival occurs at $e \neq m$. On the storage board, $(c, E_i) = (c, g_i+70)$ is empty prior to cleanup, so vertical slides between $40, 60, 80$ pass through it without stopping; no normal or wrong move inside the door room can stop at (c, E_i).

Exhaustive wrong-push analysis (proof of Lemma 19). Suppose a trajectory π obeys the backbone invariant and has not yet placed a box at (c, z) (hence no cleanup wire has fired on the storage board). Because no two wire pins share a row or column and every room without the signal is immobile, every wire pin other than the one aligned with the active signal has no second box on its row or column (so pushing it hits the frame, violating the backbone invariant). The remaining pushes that are not listed below (pushing the signal away from its next pin, pushing the signal together with the pin it rests on, pushing a signal east from an output row, pushing two adjacent boxes of a room vertically, or pushing a rack box, partition, or cap) slide a box into the frame or a forbidden cell or move the backbone,

so they violate the backbone invariant. At any canonical turn along wire i (or at a dead-row emission from a room), we list every other possible first wrong push and its stopping cell:

1. *Reverse-pushing $p_1 = (x_{i,1} - 1, r)$ east toward (c, r):* p_1 stops at $(c - 1, r)$. Column $c - 1$ holds no other box, row r holds no third box (since (P, r) is empty), and the signal remains vertically adjacent to the stored box or cap on column c. Neither box on column c can separate vertically without hitting a forbidden cell or moving a frozen end-cap (violating the backbone invariant), and the horizontal pair $\{(c - 1, r), (c, r)\}$ has no stopper to its east or west. No further legal push exists.

2. *Reverse-pushing $p_2 = (x_{i,1}, y_{i,1} + \varepsilon_1)$ toward $(x_{i,1}, r)$:* p_2 stops at $(x_{i,1}, r + \varepsilon_1)$. Because ε_1 points away from the room box adjacent to (c, r), row $r + \varepsilon_1$ is completely empty (even on c and P). The signal stays pinned against $p_1 = (x_{i,1} - 1, r)$ on row r (no third box on row r), and column $x_{i,1}$ has no third box. No further legal push exists.

3. *Reverse-pushing $p_3 = (x_{i,3} + 1, y_{i,1})$ west toward $(x_{i,1}, y_{i,1})$, or $p_5 = (x_{i,2} - 1, y_{i,2})$ east toward $(x_{i,3}, y_{i,2})$:* The pin stops at $(x_{i,1} + 1, y_{i,1})$ (resp. $(x_{i,3} - 1, y_{i,2})$). Columns $x_{i,1} + 1$ and $x_{i,3} - 1$ are completely empty, rows $y_{i,1}$ and $y_{i,2}$ have no third box, and the signal stays pinned against its vertical pin p_2 (resp. p_4) with no third box on column $x_{i,1}$ (resp. $x_{i,3}$). No further legal push exists.

4. *Reverse-pushing $p_4 = (x_{i,3}, y_{i,2} + \varepsilon_2)$ toward $(x_{i,3}, y_{i,1})$:* p_4 stops at $(x_{i,3}, y_{i,1} + \varepsilon_2)$. If $\varepsilon_2 \neq \varepsilon_1$, row $y_{i,1} + \varepsilon_2$ is completely empty and no further push exists. If $\varepsilon_2 = \varepsilon_1$, row $y_{i,1} + \varepsilon_1$ contains exactly one other box, namely $p_2 = (x_{i,1}, y_{i,1} + \varepsilon_1)$. The only possible next push slides either $(x_{i,3}, y_{i,1} + \varepsilon_1)$ west onto p_2 (landing at $(x_{i,1} + 1, y_{i,1} + \varepsilon_1)$) or p_2 east onto $(x_{i,3}, y_{i,1} + \varepsilon_1)$ (landing at $(x_{i,3} - 1, y_{i,1} + \varepsilon_1)$). In both outcomes, the two boxes sit adjacent on row $y_{i,1} + \varepsilon_1$ with no third box, the displaced box lies in empty column $x_{i,1} + 1$ or $x_{i,3} - 1$, column $x_{i,1}$ has no other box, and on column $x_{i,3}$ the original signal stays pinned at $(x_{i,3}, y_{i,1})$ against $p_3 = (x_{i,3} + 1, y_{i,1})$ (either alone on column $x_{i,3}$ or vertically adjacent to $(x_{i,3}, y_{i,1} + \varepsilon_1)$ with no third box). No further push exists, and column c is untouched.

5. *Reverse-pushing $p_6 = (x_{i,2}, z' + \varepsilon_3)$ toward $(x_{i,2}, y_{i,2})$:* p_6 stops at $(x_{i,2}, y_{i,2} + \varepsilon_3)$. The analysis is identical to Case 4: if $\varepsilon_3 \neq \varepsilon_2$, row $y_{i,2} + \varepsilon_3$ is empty; if $\varepsilon_3 = \varepsilon_2$, it may interact once with $p_4 = (x_{i,3}, y_{i,2} + \varepsilon_2)$, stranding the pair in empty column $x_{i,3} - 1$ or $x_{i,2} + 1$ while the signal stays pinned at $(x_{i,2}, y_{i,2})$ against $p_5 = (x_{i,2} - 1, y_{i,2})$. Column c is untouched.

6. *Emitting onto a dead row $z' + \varepsilon_3$ from a door room (or sliding into a capless end):* Sliding into a capless end hits $(c, g \pm 1)$, violating the backbone invariant immediately. On a dead row $z' + \varepsilon_3$, either no box lies west of c (so pushing west hits the boundary), or the sole box west of c is the final pin $p_6 = (x_{j,2}, z' + \varepsilon_3)$ of the wire feeding slot z'. Pushing the signal west onto p_6 lands it at $(x_{j,2} + 1, z' + \varepsilon_3)$ in the empty column $x_{j,2} + 1$, while reverse-pushing p_6 east lands it at $(c - 1, z' + \varepsilon_3)$ in the empty column $c - 1$. In either outcome, row $z' + \varepsilon_3$ has no third box, the remaining box in the door room sits alone against column P (or vertically adjacent to the trapped signal with no third box), and no further push can move any box into (c, z) or into any cleanup slot (c, E_i).

In every case, a contiguous run of boxes alone on a line cannot be pushed even at unbounded strength without hitting the frame (or a forbidden partition neighbour on column c), and adjacent boxes have no space between them for a reverse push. Thus no continuation obeying the backbone invariant can ever reach (c, z) or any (c, E_i), completing the proof of Lemma 19.

D.3 Seven-round quota accounting and verification

On the work board R of Section 6.1, of size $w_0 \times h_0$ with rack column P, $\mu = w_0 - 2 - P = 4$, and $Y_{\mathrm{occ}} = Y_{\mathrm{part}} \cup Y_{\mathrm{cap}} \cup \{z\}$, the seven-round program $\mathcal{P}_7$ has exact quotas:

$$q_x^{(1)} = h_0 \cdot \mathbf{1}[x \in \{0, w_0 - 1\}], \qquad q_y^{(2)} = (w_0 - 2) \cdot \mathbf{1}[y \in \{0, h_0 - 1\}],$$
$$q_x^{(3)} = 0, \qquad q_x^{(5)} = 0,$$
$$q_y^{(4)} = \mu \cdot \mathbf{1}[y \in Y_P] + (\mu + 1) \cdot \mathbf{1}[y \in Y_{\mathrm{occ}}], \quad q_y^{(6)} = (P - 1) \cdot \mathbf{1}[\exists g \in Y_{\mathrm{part}} : \ |y - g| = 1],$$

and for every $x \in \{1, \ldots, w_0 - 2\}$, $q_x^{(7)}$ equals the number of cells (x, y) with $1 \le y \le h_0 - 2$ that are holes in B_{can} and were not deleted in Rounds 4 or 6 (i.e., $x \le w_0 - 2 - q_y^{(4)}$ and $x > q_y^{(6)}$).

Exact cuts, fullness, and state independence. In any canonical parked configuration B_{can}, the sets Y_P and $Y_{\mathrm{occ}} = Y_{\mathrm{part}} \cup Y_{\mathrm{cap}} \cup \{z\}$ are pairwise disjoint, and every wire pin avoids $Y_P \cup Y_{\mathrm{occ}}$ and every row $g \pm 1$ ($g \in Y_{\mathrm{part}}$):

- Rounds 1 and 2 delete every cell of the empty frame (exact cuts at the board edge).
- In Round 4, for every $y \in Y_P$, $(P, y) \in B_{\mathrm{can}}$ and columns $P + 1, \ldots, w_0 - 2$ are empty, so the rightmost hole run has exact length $\mu = q_y^{(4)}$. For every $y \in Y_{\mathrm{occ}}$, $(P, y) \notin B_{\mathrm{can}}$ and $(c, y) \in B_{\mathrm{can}}$, so the rightmost hole run has exact length $\mu + 1 = q_y^{(4)}$.
- In Round 6, for every $g \in Y_{\mathrm{part}}$, row $g \pm 1$ lies in Y_P (so Round 4 deleted columns $P + 1, \ldots, w_0 - 2$), $(P, g \pm 1) \in B_{\mathrm{can}}$, and columns $1, \ldots, c$ are all empty, so the leftmost hole run has exact length $P - 1 = q_{g \pm 1}^{(6)}$.
- Round 7 deletes every remaining hole in each column x. Because each remaining hole of $R \setminus B_{\mathrm{can}}$ is counted once in Round 7 and every cell deleted in Rounds 1–6 is a hole of $R \setminus B_{\mathrm{can}}$, the seven rounds sum to $|R \setminus B_{\mathrm{can}}| = w_0 h_0 - |B_0|$ (the program is full) and accept $R \setminus B_{\mathrm{can}}$ with exact run cuts.
- Finally, changing any door i's state swaps the box between (c, α_i) and (c, β_i). Since $\alpha_i, \beta_i \in Y_P \setminus (Y_{\mathrm{occ}} \cup \{g \pm 1 : g \in Y_{\mathrm{part}}\})$, we have $q_{\alpha_i}^{(4)} = q_{\beta_i}^{(4)} = \mu$ (stopping east of column P) and $q_{\alpha_i}^{(6)} = q_{\beta_i}^{(6)} = 0$. Hence both (c, α_i) and (c, β_i) survive into Round 7 on column c, so $q_c^{(7)}$ (and every other quota) is independent of the final door states.

E Details for Push-kFS

This appendix records the exact coordinates, phase assignment, AND fan-out, and single-box macros for the k-independent walled board of Section 7.1. Coordinates have x increasing rightward and y increasing downward.

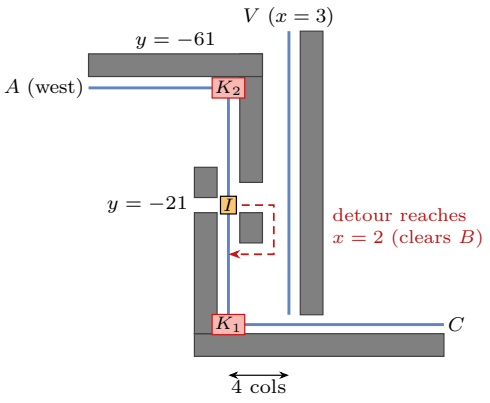
E.1 Gadget coordinates, phase encoding, and the AND fan-out

All AND and OR cores are placed only by translation; routing utilities may be rotated and reflected. Semi-infinite wall strips below indicate port templates truncated at matching splices (with at least 16-cell straight buffer segments and 3-cell overlap preserving all 1-cell neighbourhoods).

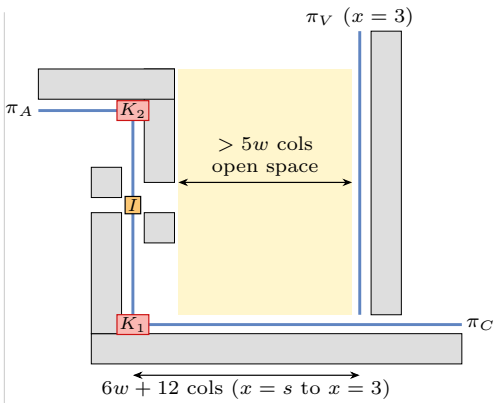
Gadget	Fixed walls	Normal box slots ($q \in \{0,1\}$ or (a,b,c))
Line	$y \in \{1,2\}$	$(2j+q, 0)$
Corner	$\{x \le 2,\ y \in \{-2,-1\}\}$ $\cup\{x \in \{1,2\},\ y \ge -2\}$	$F = (-2+q, 0)$, $U = (0, 1+q)$; $K = (0,0)$
Shifter	$\{y \in \{1,2\}\}$ $\cup\{x \in [2,4],\ y \in \{-2,-1\}\}$	$D = (q,0)$, $E = (5+q,0)$
Flip	$\{y \in \{1,2\},\ x \le 0$ or $x \in \{2,3\}\}$ $\cup\{y \in \{-2,-1\},\ x \in \{0,1\}$ or $x \ge 3\}$	$H = (-2+q,0)$, $I = (3q,0)$, $J = (4+q,0)$ passage: $(1,2) \to \cdots \to (2,-2)$
OR	$\{y \in \{1,2\}\} \cup \{x \in \{5,6\},\ y \le -1\}$	$D = (1-a,0)$, $A = (3-a,0)$, $C = (8-c,0)$, $M = (10-c,0)$, $V = (4,-1-b)$, $E = (4,-3-b)$
AND (prot.)	$\{x \ge -3,\ y \in \{1,2\}\}$ $\cup\{x \in \{-3,-2\},\ y \le 2\}$ $\cup\{x \in \{4,5\},\ y \le -1\}$	$D = (1-a,0)$, $A = (3-a,0)$, $C = (7-c,0)$, $M = (9-c,0)$, $V = (3,-1-b)$, $E = (3,-3-b)$, $U = (-1,-1-a)$, $S = (-1,-3-a)$; $K = (-1,0)$

Here $a, b \in \{0,1\}$ record inflow on the two inputs and $c \in \{0,1\}$ records *outflow* on the third port, so legal NCL states satisfy $c \le a \wedge b$ (AND) and $c \le a \vee b$ (OR). The protected AND simply glues a concave corner transformed by $(x,y) \mapsto (-1-x,-y)$ onto AND's left input, identifying the corner's F box with $D = (1-a,0)$ and replacing the leftward tail by the upward A track at $x = -1$.

AND fan-out and global layout. Because the upward A track ($x = -1$) sits only four columns west of V ($x = 3$), every AND core uses a fixed finite fan-out template (Figure 48(a)): along $x = -1$ upward, place ordinary boxes at $y = -1-a, -3-a, \ldots, -19-a$, a vertical flip transformed by $(x,y) \mapsto (-1-y, -21-x)$ with middle box $I = (-1, -21-3a)$ and $J = (-1, -25-a)$, ordinary boxes up to $y = -59-a$, and a standard concave corner at $K_2 = (-1,-61)$ turning A westward. On the V side, the flip's detour reaches only $x = 2$, leaving V's track ($x = 3$) and wall ($x = 4, 5$) untouched; the upper corner's top wall ends at $x = 1$, also clearing V. Every AND and OR core therefore exposes three separated straight tails pointing west, north, and east.



(a) Push-kFS AND fan-out ($w = 2$)



(b) Push-kS finite AND core (7 blocks $\ge w \times w$)

Figure 48. Separating AND's narrow vertical inputs into west, north, and east ports. (a) In Push-kFS ($w = 2$), a fixed flip at $y = -21$ and upper corner at $y = -61$ turn A west while keeping the flip detour at $x \le 2$, clear of V at $x = 3$. (b) In Push-kS ($w = k+1$), moving the protective corner far west to $x = s = -6w - 9$ yields a seven-rectangle finite core (each block $\ge w \times w$) with more than $5w$ open columns between the A branch and V.

For the global board, compute a planar orthogonal grid drawing of the cubic graph G [15], compress its distinct x- and y-coordinates to integer ranks (keeping polynomial area), and scale by a large constant λ (chosen so every straight segment has length at least 60). Inside

each vertex box, match the cyclic order of the three incident edges by permuting AND's two light inputs or OR's three inputs (without rotating the core) and wiring the three tails with one of finitely many constant-size orthogonal templates (also with segment lengths at least 60). Insert a flip whenever wall sides mismatch between consecutive bends, and insert one extra pair of flips on a straight segment of every edge: each flip gives the player a crossing of the edge, and the second flip restores the wall side. Finally, surround the bounding box by a 20-cell empty margin and place the player in the outer margin.

Per-edge phase encoding and check of (C1)–(C3). Orient each edge $e = uv$ arbitrarily and write $q = 1$ for $v \to u$. At every port with outward unit vector d and non-inflow slot X_{out}, the boundary slot is $X(\iota) = X_{\text{out}} + \iota d$ (where $\iota = 1$ means inflow). At u this is $X_{\text{out},u} + qd_u$, and at v it is $X_{\text{out},v} + (1-q)d_v = (X_{\text{out},v} + d_v) + q(-d_v)$, matching the path tangent at both ends for all port combinations. At the port where e leaves the vertex box of u, the box in this boundary slot is the *outer port box* $X_{e,u}$ read by the decoding of Section 7.2. Every utility propagates $q \in \{0,1\}$ forward (and $1-q$ when traversed in reverse); on each straight segment (between consecutive bends or ports) whose endpoint reference-slot distance is odd, insert one phase shifter (gap 5) and fill the remaining even distance at spacing 2, avoiding duplicate boxes at splices. Thus every NCL orientation ρ uniquely defines a complete normal configuration B_ρ.

Every track cell in a straight line, phase shifter, or flip borders a wall on at least one side, so its only arcs in D_W stay on its line; concave corners add only arcs into the sink K. Because all logic gates share one orientation, the only cross-track arcs between live tracks are $(3,-1) \to (3,0)$ (AND) and $(4,-1) \to (4,0)$ (OR), both vertical-to-horizontal; no cycle can mix directions, giving (C2). Every box's phase-0 and phase-1 slots lie in the same bidirectional straight SCC (including flip's I from 0 to 3), giving (C1). Intra-track target spacings are 2 (lines, splices, D/A, V/E, C/M), ≥ 4 (A/C in AND and in OR), 5 (phase shifter), and $(2,4)$ or $(4,2)$ (flip), with no target at K, giving (C3).

E.2 Single-box macros

In the macros below, every push moves one box; $I \to^3$ denotes three consecutive single pushes of I. Whenever an edge reverses from $v \to u$ to $u \to v$, we execute the lose-inflow macro at u, sweep the utilities and straight segments along the edge from u toward v (pushing leading boxes ahead of trailing boxes on each straight segment), and end with the gain-inflow macro at v. This propagation order prepares the exact outer neighbour on each side of the active utility ($P_1 = (-4,0), P_2 = (0,4)$ for a corner; $P_1 = (-2,0), P_2 = (8,0)$ for a phase shifter; $P_1 = (-4,0), P_2 = (7,0)$ for a flip).

Utility / transition	Forward ($q: 0 \to 1$)	Reverse ($q: 1 \to 0$)
Concave corner	$U\downarrow,\ F\to$	$F\leftarrow,\ U\uparrow$
Phase shifter	$E\to,\ D\to$	$D\leftarrow,\ E\leftarrow$
Flip (atomic)	$J\to,\ I\to^3,\ H\to$	$H\leftarrow,\ I\leftarrow^3,\ J\leftarrow$
AND / OR transition	AND macro	OR macro
$a: 0 \to 1$	$U\uparrow,\ D\leftarrow,\ A\leftarrow$	$D\leftarrow,\ A\leftarrow$
$a: 1 \to 0$	$A\to,\ D\to,\ U\downarrow$	$A\to,\ D\to$
$b: 0 \to 1$ $(a=1)$	$E\uparrow,\ V\uparrow$	$E\uparrow,\ V\uparrow$
$b: 0 \to 1$ $(a=0)$	$E\uparrow,\ A\to,\ V\uparrow,\ A\leftarrow$	$E\uparrow,\ V\uparrow$
$b: 1 \to 0$	$V\downarrow,\ E\downarrow$	$V\downarrow,\ E\downarrow$
$c: 0 \to 1$	$C\leftarrow,\ M\leftarrow$	$C\leftarrow,\ M\leftarrow$
$c: 1 \to 0$	$M\to,\ C\to$	$M\to,\ C\to$

Between macros, every flip stands in a full normal phase ($I \in \{0, 3\}$) with its crossing passage $(1, 2) \to \cdots \to (2, -2)$ open; since the planar dual is connected and every edge has its extra pair of flips, the player can walk between all outer face regions. During a flip macro, the player never needs to cross that flip while $I \in \{1, 2\}$: in the forward macro all three pushing cells are reached from above, walking between the passage column $x = 2$ and the open region $x \le -1$ along row $y = -3$ around the island $\{0, 1\} \times \{-2, -1\}$, and in the reverse macro they are reached from below, walking between the passage column $x = 1$ and the open region $x \ge 4$ along row $y = 3$ around the island $\{2, 3\} \times \{1, 2\}$. Inside AND and OR, the only non-trivial interior entries are:

- $A \leftarrow$ needs $(4, 0)$: in AND, $a = 0$ forces $c = 0$, so the player enters from the right at $(6, -1)$; in OR, either $c = 0$ (enter via $(7, -1)$) or $c = 1, a = 0$ forces $b = 1$ (enter via $(4, -1)$).
- $V \uparrow$ needs $(3, 0)$ (AND) or $(4, 0)$ (OR): in AND with $a = 0$ (forcing $c = 0$), push A temporarily from 3 to 4 so the player stands at $(3, 0)$ to push $V \uparrow$, then enter from $(6, -1)$ to push A back to 3; in AND with $a = 1$ (forcing $c = 0$ since $b = 0$), enter $(3, 0)$ from the right via $(6, -1) \to (6, 0) \to (5, 0) \to (4, 0) \to (3, 0)$; in OR with $a = 1$, enter via $(3, -1) \to (3, 0) \to (4, 0)$, while in OR with $a = 0$ (forcing $c = 0$ since $b = 0$), enter from the right via $(7, -1) \to \cdots \to (4, 0)$.
- $C \to$ needs $(5, 0)$ (AND, where $c = 1$ forces $a = b = 1$, entering via $(3, -1)$) or $(6, 0)$ (OR, where $c = 1$ forces $a = 1$ or $b = 1$, entering via $x = 3$ or $x = 4$).

F Details for Push-kS

Here we summarize the width-w utility and finite core coordinates ($w = k+1 \ge 2$), the macro routes through thick tunnels, and the fixed-core global composition used in Section 7.2.

F.1 Parameterized utilities, core rectangles, normal slots, and macro routes

Let m be the smallest odd integer with $m \ge \max(3, w)$. The width-w routing utilities have virtual walls (cells of W, which hold the frozen boxes B_W in the instance; unions of $\ge w \times w$ rectangles) and normal box slots ($q \in \{0, 1\}$):

Utility	Virtual walls	Normal box slots ($q \in \{0, 1\}$)
Line	$1 \le y \le w$	$(2j + q, 0)$
Corner	$\{x \le w,\ -w \le y \le -1\}$ $\cup\{1 \le x \le w,\ y \ge -w\}$	$F = (-2 + q, 0)$, $U = (0, 1 + q)$; $K = (0, 0)$
Shifter	$\{1 \le y \le w\}$ $\cup\{2 \le x \le m + 1,\ -w \le y \le -1\}$	$D = (q, 0)$, $E = (m + 2 + q, 0)$
Flip	$\{1 \le y \le w :\ x \le 0$ or $x \in [2, w + 1]\}$ $\cup\{-w \le y \le -1 :\ x \in [2 - w, 1]$ or $x \ge 3\}$	$H = (-w + q, 0)$, $I = (3q, 0)$, $J = (w + 2 + q, 0)$; passage: $(1, w) \to \cdots \to (1, 0) \to (2, 0) \to \cdots \to (2, -w)$

In a thick flip, the two cavity zones are $[2 - w, 0] \times \{0\}$ and $[3, w + 1] \times \{0\}$, giving the cascade $H_x \ge 1 - w \Rightarrow I_x \ge 2 \Rightarrow J_x \ge w + 3$ of Lemma 25.

For the finite AND core (Figure 48(b)), set

$$L = 6w + 8, \qquad s = -6w - 9, \qquad f = 3w + 9, \qquad g = -6w - 21,$$

where L is the protective-chain length of Section 7.2 and $s = -L - 1$, with $(a, b, c) = (0, 0, 0)$ outer port slots $\pi_A = (s - 2w - 7, g)$, $\pi_V = (3, -8w - 31)$, $\pi_C = (3w + 13, 0)$, and tail wall ends $\eta_A = \pi_{A,x} - 2w - 10$, $\eta_V = \pi_{V,y} - 2w - 10$, $\eta_C = \pi_{C,x} + 2w + 10$. Its virtual walls are

the union of seven closed integer rectangles, each of width and height at least w (with the two flip islands exactly $w \times w$):

Wall rectangle	x-range	y-range
Bottom wall	$[s-w,\, \eta_C]$	$[1,\, w]$
Protective corner / lower A left wall	$[s-w,\, s-1]$	$[-f,\, w]$
Flip left island ($w \times w$)	$[s-w,\, s-1]$	$[-f-w-1,\, -f-2]$
Flip right island ($w \times w$)	$[s+1,\, s+w]$	$[-f-1,\, -f+w-2]$
Upper A right wall	$[s+1,\, s+w]$	$[g-w,\, -f-3]$
West-tail roof	$[\eta_A,\, s+w]$	$[g-w,\, g-1]$
V right wall / AND roof	$[4,\, w+3]$	$[\eta_V,\, -1]$

The protective corner sits at $K_1 = (s, 0)$, the vertical flip at $(s, -f)$ via $(x, y) \mapsto (s-y, -f-x)$, and the upper corner turning A west at $K_2 = (s, g)$. The A branch's rightmost wall sits at $x = s + w = -5w - 9$, leaving more than $5w$ open columns west of V at $x = 3$. For a legal state (a, b, c) $(c \le a \wedge b)$, the normal working boxes of the finite AND core are:

- west tail: $(s-1-2j-a,\, g)$ for $0 \le j \le w+3$ ($j = 0$ is F_2);
- upper vertical tail: $(s,\, g+2+2j-a)$ for $0 \le j \le w+4$ ($j = 0$ is U_2, $j = w+4$ is flip's J);
- flip middle box: $I = (s, -f-3a)$;
- lower vertical tail: $(s,\, -f+w+2j-a)$ for $0 \le j \le w+4$ ($j = 0$ is flip's H, $j = w+4$ is U_1);
- protective horizontal chain: $(s+2+2j-a,\, 0)$ for $0 \le j \le 3w+4$ ($j = 0$ is F_1, $j = 3w+4$ is D);
- core A: $(3-a, 0)$; V branch: $(3, -1-2j-b)$ $(0 \le j \le 4w+15)$; C branch: $(w+5+2j-c, 0)$ $(0 \le j \le w+4)$.

The finite OR core uses just two rectangles—bottom wall $[\eta_A, \eta_C] \times [1, w]$ and right wall/roof $[5, w+4] \times [\eta_V, -1]$—with $(a, b, c) = (0, 0, 0)$ outer port slots $\pi_A = (-2w-7, 0)$, $\pi_V = (4, -2w-13)$, $\pi_C = (3w+14, 0)$ (and η_A, η_V, η_C defined by the same formulas as above) and normal boxes $(1-2j-a, 0)$ $(0 \le j \le w+4)$, $A = (3-a, 0)$, $(4, -1-2j-b)$ $(0 \le j \le w+6)$, and $(w+6+2j-c, 0)$ $(0 \le j \le w+4)$. Both cores fit inside a box of side $200(w+3)$. Condition (C1) holds because every working box moves between its phase-0 and phase-1 slots inside its own straight track. For (C2), the argument of Appendix E.1 carries over: every track cell of a thick utility borders a wall on at least one side, corners add only arcs into their sink K, and the open region of the AND core between the A branch and the V track (and the open region left of V in OR) holds no box and only sends arcs into tracks. As all cores share one orientation, the only arcs between box-bearing tracks are $(3, -1) \to (3, 0)$ (AND) and $(4, -1) \to (4, 0)$ (OR), both from a vertical to a horizontal track, so every box-bearing component is straight. For (C3), all intra-track normal spacings are at least 2: 2 on lines and splices, $\ge w+2$ for the A/C pair of AND and of OR, $m+2 \ge 5$ in a phase shifter, and $(w, w+2)$ or $(w+2, w)$ in a thick flip.

Macro routes through thick tunnels. The single-box macros follow the same push sequences as Appendix E.2 (with AND's a-transition sweeping the west tail, upper tail, I three times, lower tail, and the $3w+5$ boxes of the protective chain in order before A, and reversing for $a : 1 \to 0$). Every thickened roof only lengthens an empty straight walk without blocking any pushing cell:

- In a phase shifter, the player enters under the $m \times w$ roof from $(1, -1)$ (forward, walking east to $(m+1, 0)$ to push $E \to$) or from $(m+2, -1)$ (reverse, walking west to $(2, 0)$ to push $D \leftarrow$).
- In a thick flip (forward $q : 0 \to 1$), the player enters from above via $(2, -1)$ to reach $(w+1, 0)$ for $J \to$, detours around the $w \times w$ upper island to $(1-w, -1)$ to walk to

$(-1, 0)$ and push $I \to^3$ (finishing at the open crossing $(2, 0)$), then pushes $H \to$ from $(-w - 1, 0)$; the reverse macro mirrors this from below via $(1, 1)$ and $(w + 2, 1)$.

- In AND ($x_{\text{in}} = w+4$) and OR ($x_{\text{in}} = w+5$), the right tunnel entry is $(x_{\text{in}}, -1) \to (x_{\text{in}}, 0)$, open whenever $c = 0$. When $b : 0 \to 1$ in AND with $a = 0$ (forcing $c = 0$), the player pushes A from 3 to 4, pushes $V \uparrow$ from $(3, 0)$, and walks from $(x_{\text{in}}, -1)$ through the empty tunnel to $(5, 0)$ to restore A to 3; in OR, $V \uparrow$ needs $(4, 0)$, reached via $(3, -1) \to (3, 0) \to (4, 0)$ when $a = 1$ and via $(x_{\text{in}}, -1) \to \cdots \to (4, 0)$ when $a = 0$ (which forces $c = 0$). For $C \to (x_{\text{in}} \to x_{\text{in}} + 1$, needing $(x_{\text{in}} - 1, 0))$, $c = 1$ guarantees $a = b = 1$ in AND (entering via $(3, -1)$) and $a = 1$ or $b = 1$ in OR (entering via $x = 3$ or $x = 4$).

F.2 Fixed-core routing, offset adapters, and global walkability

When embedding the cores into a planar orthogonal grid drawing of G, we must *never scale the unit-grid cores* (since working boxes stay 1×1); instead, we scale only the external routing geometry around each fixed core.

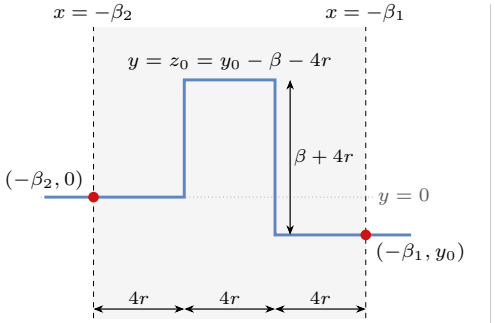
Let β_0 bound both finite cores and their local pushing cells inside the square $\mathcal{C}_{\beta_0} = [-\beta_0, \beta_0]^2$, set the routing clearance to $r = 100(w + 3)$, and extend the three straight tails (west, north, east) to $\mathcal{C}_\beta$ with $\beta \geq 4r + 4\beta_0$. First we fix routing templates. The tails of every core leave through its west, north, and east sides, while the three edges at a vertex of the drawing leave it in three of the four directions, so there are 12 cases: four choices of the three outer sides and three cyclic-order-preserving bijections from the tails to them. In each case, since the bijection preserves the cyclic order, the annulus between the squares of radius 1 and 10 contains three disjoint curves joining the matched side midpoints, with radial ends; replacing them by nearby orthogonal grid polylines and scaling gives an integer N_0, independent of w and G, and for each case a template of three disjoint simple orthogonal polylines from the edge midpoints of $\mathcal{C}_{N_0}$ to the matched edge midpoints of $\mathcal{C}_{10N_0}$, with radial ends and pairwise distance at least 1.

Now set $\mu = 100(\beta + r + 1)$, $\beta_2 = N_0\mu$, and $\beta_1 = \beta_2 - 12r$, and extend each core's three straight tails to $\mathcal{C}_{\beta_1}$. The west tail lands at $(-\beta_1, y_0)$ with $|y_0| \leq \beta$, which may not sit at the edge midpoint $(-\beta_2, 0)$. To snap it to $(-\beta_2, 0)$ without creating short bends when $|y_0|$ is small, attach the explicit five-segment overshoot-and-return *offset adapter* with $z_0 = y_0 - \beta - 4r$ (Figure 49(a)):

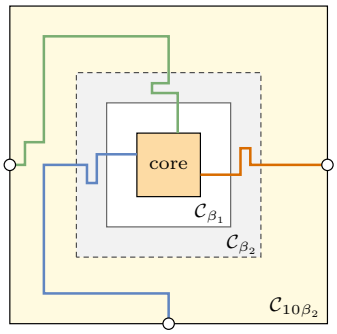
$$(-\beta_1, y_0) \to (-\beta_1 - 4r, y_0) \to (-\beta_1 - 4r, z_0) \to (-\beta_1 - 8r, z_0) \to (-\beta_1 - 8r, 0) \to (-\beta_2, 0).$$

Its five segments have lengths $4r$, $\beta + 4r$, $4r$, $-z_0 \geq 4r$, and $4r$, and all non-adjacent segments sit at distance at least $4r$. Rotated copies handle the north and east tails inside disjoint $12r$-wide strips outside $\mathcal{C}_{\beta_1}$. From the three edge midpoints of $\mathcal{C}_{\beta_2}$, attach the template of the vertex's case scaled by μ to reach the desired edge midpoints of $\mathcal{C}_{10\beta_2}$ (Figure 49(b)), with every segment length and clearance at least $4r$.

Finally, take a rank-compressed planar orthogonal grid drawing of G [15] and scale its coordinates by $\Delta = 100\beta_2 + 100r$. Place each core and its routing square $\mathcal{C}_{10\beta_2}$ at its vertex, realize every 90° bend by a thick concave corner, insert a thick flip whenever wall sides mismatch, place two extra thick flips on one long segment of each edge, and insert a thick phase shifter on each straight segment whose endpoint reference-slot distance is odd (every segment has length at least $4r$, which easily accommodates it), splicing all straight tails with overlap at least $w + 3$ so every wall block remains a union of $\geq w \times w$ rectangles. Finally, surround the bounding box by an empty outer margin of width r and place the initial player p_0 in this margin. Because w and all routing constants are fixed, the resulting board has polynomial area and is constructed in polynomial time.



(a) Five-segment offset adapter (all clearances $\geq 4r$)



(b) Fixed-core routing inside $\mathcal{C}_{10\beta_2}$

Figure 49. Fixed-core global composition for Push-kS without scaling the unit-grid cores. (a) The five-segment overshoot-and-return offset adapter snaps a tail at $(-\beta_1, y_0)$ to $(-\beta_2, 0)$ with every segment length and non-adjacent clearance at least $4r$, even when $|y_0|$ is small. (b) Concentric squares $\mathcal{C}_{\beta_1} \subset \mathcal{C}_{\beta_2} \subset \mathcal{C}_{10\beta_2}$ join the unscaled core's west, north, and east tails to any three outer ports in the same cyclic order.

Global walkability and outer-port claim propagation. Because G is 3-connected, every face boundary is a simple cycle and every face retains a connected inner empty region outside the thin routing strips, linked to the outer entrances of every utility and gate core. As in Appendix E.2, between macros every flip is in a normal phase ($I \in \{0, 3\}$) with its crossing passage open, and every edge carries its extra pair of flips, so the player can walk between all faces; during a flip macro the player stays on one side of that flip (by the routes above), so its passage, blocked while $I \in \{1, 2\}$, is never needed.

For soundness, Section 7.2 proved the NCL threshold inequalities on the near-core boxes (U_1, E, M for AND; D, E, M for OR). Along each vertex branch from a near-core box to its outer port box on the boundary of $\mathcal{C}_{10\beta_2}$ (in particular along AND's A-branch $U_1 \to H \to J \to U_2 \to F_2 \to X_{e_A,u}$, using the contrapositive of the upper corner contract at $U_2 \to F_2$), composing Part 1 of Lemma 25 yields near claim $\Rightarrow$ outer claim for every port. Because NCL constraints are monotone in inflow, every vertex satisfies its constraint using the outer-port claims $i(e, u)$ alone, completing the proof of Theorem 26.