PSPACE-completeness of Pulling Blocks to Reach a Goal

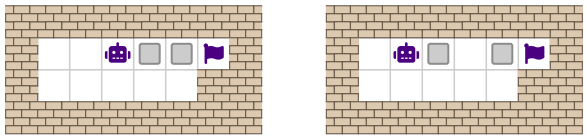
Joshua Ani* Sualeh Asif* Erik D. Demaine* Yevhenii Diomidov*
Dylan Hendrickson* Jayson Lynch* Sarah Scheffler† Adam Suhl‡

Abstract. We prove PSPACE-completeness of all variations of pulling-block path-planning puzzles (reaching a goal with or without forced pulls, of arbitrary strength, with or without gravity) that include fixed blocks or walls, with the exception of Pull?-1FG (strength 1, fixed blocks, with gravity) for which we only show NP-hardness.

In the Pull series of path-planning problems [3,4], the goal is to navigate an agent from a given starting square to a given target square within a rectangular board featuring impassable but pullable 1×1 blocks. We study several different variants of Pull, which can be combined in arbitrary combination:

1. **Strength:** In Pull-k, the agent can pull an unbroken horizontal or vertical line of up to k pullable blocks at once. In Pull-$*$, the agent can pull arbitrarily many blocks at once.
2. **Fixed blocks/walls:** In Pull-F, the board may have fixed 1×1 blocks that cannot be traversed or pulled. In the more general Pull-W, the board may have fixed thin (1×0) walls.
3. **Optional/forced pulls:** In Pull!, every agent motion that can also pull blocks must pull as many as possible (as in many video games where the player input is just a direction). In Pull?, the agent can choose whether and how many blocks to pull. (The latter is traditionally called Pull in the literature, but we use the explicit "?" to indicate optionality.)
4. **Gravity:** In Pull-G, all blocks fall maximally downward after each agent move (like gravity).

Table 1 summarizes our and known results for all variants that include fixed blocks or walls: we prove PSPACE-completeness for any strength, with optional or forced pulls, and with or without gravity, with the exception of Pull?-1FG for which we only show NP-hardness. The only previous result for this problem family is NP-hardness for Pull?-k even without fixed blocks [4]. In some cases, our results are stronger than the best known results for the corresponding Push (pushing-block) problem; see [3]. More complex variants PullPull (where pulled blocks slide maximally), PushPull (where blocks can be pushed and pulled), and Storage Pull (where the goal is to place multiple blocks into desired locations) are also known to be PSPACE-complete [3].



(a) Initial state **(b)** A strength-1 move

Figure 1: A pulling-block puzzle. The robot is the agent, the flag is the goal square, the light gray blocks can be moved, and the bricks are fixed in place. Robot and flag icons from Font Awesome under CC BY 4.0 License.

Our reductions are from two problems: Asynchronous Nondeterministic Constraint Logic (NCL) [2,5] and planar 1-player motion planning [1]. Without gravity, Figure 2 shows our NCL gadgets. With gravity, we reduce from the motion-planning framework [1]. For optional-pulling PSPACE-hardness (with $k \geq 2$), we use the locking 2-toggle gadget in Figure 3. For forced pulling, we introduce a new gadget, a *self-closing door*, and show it can simulate a locking 2-toggle; see Figure 4. For the one remaining case of Pull?-1FG, we show NP-hardness by constructing a crossing XOR gadget (not shown here).

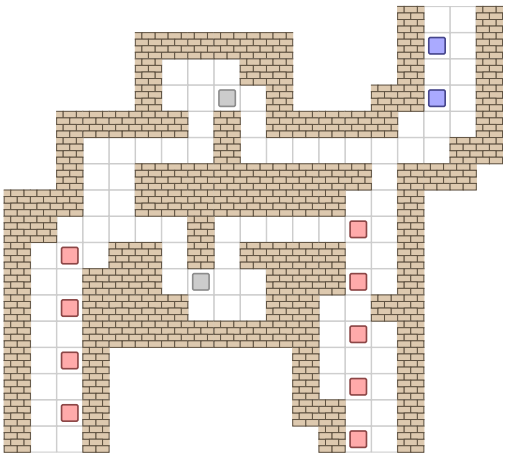
*Massachusetts Institute of Technology, Cambridge, MA, USA
†Boston University, Boston, MA, USA
‡Algorand, Boston, MA, USA

References

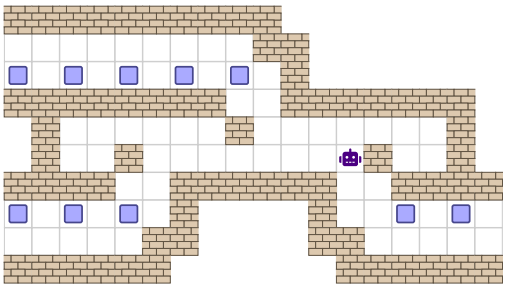
[1] Erik D. Demaine, Dylan H. Hendrickson, and Jayson Lynch. A general theory of motion planning complexity: Characterizing which gadgets make games hard. arXiv:1812.03592, 2018.

Problem	Gravity	Forced	Strength	Features	Hardness	Previous Best
PULL?-kF	no	no	$k \geq 1$	fixed blocks	PSPACE-complete	NP-hard [4]
PULL?-$*$F	no	no	∞	fixed blocks	PSPACE-complete	NP-hard [4]
PULL!-kF	no	yes	$k \geq 1$	fixed blocks	PSPACE-complete	—
PULL!-$*$F	no	yes	∞	fixed blocks	PSPACE-complete	—
PULL?-1FG	yes	no	$k = 1$	fixed blocks	NP-hard	—
PULL?-1WG	yes	no	$k = 1$	thin walls	PSPACE-complete	—
PULL?-kFG	yes	no	$k \geq 2$	fixed blocks	PSPACE-complete	—
PULL?-$*$FG	yes	no	∞	fixed blocks	PSPACE-complete	—
PULL!-kFG	yes	yes	$k \geq 1$	fixed blocks	PSPACE-complete	—
PULL!-$*$FG	yes	yes	∞	fixed blocks	PSPACE-complete	—

Table 1: New and known results for PULL variants. We omit PULL-W hardness implied by PULL-F hardness.



(a) NCL AND gadget for PULL-kF for $k \geq 1$. Also works for W variant and PULL-$*$.



(b) NCL OR gadget for PULL-kF for $k \geq 1$ Also works for W variant and PULL-$*$.

Figure 2: Gadgets for the reduction from asynchronous NCL to PULL-kF.

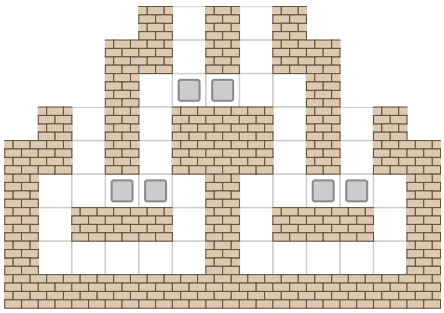
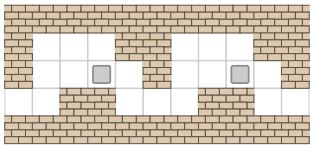
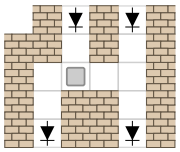


Figure 3: Construction of a locking 2-toggle (in the state where only the right tunnel is traversable, and traversing lets you optionally flip), which by [1] shows PSPACE-completeness of PULL?-kFG for $k \geq 2$. Also works for PULL?-kWG for $k \geq 1$ and for PULL?-$*$WG.



(a) Re-usable one-way "diode" gadget, denoted in Figure 4b.



(b) This self-closing door can be "opened" from the right tunnel, and forced to "close" when traversing the left tunnel.

Figure 4: Gadgets for PULL!-1FG.

[2] Robert A. Hearn and Erik D. Demaine. *Games, Puzzles, and Computation.* A K Peters/CRC Press, 2009.

[3] André G. Pereira, Marcus Ritt, and Luciana S. Buriol. Pull and PushPull are PSPACE-complete. *Theoretical Computer Science*, 628:50–61, 2016.

[4] Marcus Ritt. Motion planning with pull moves. arXiv:1008.2952, 2010.

[5] Giovanni Viglietta. Partial searchlight scheduling is strongly PSPACE-complete. In *Proceedings of the 25th Canadian Conference on Computational Geometry*, Waterloo, Canada, August 2013.