# Allen-Wrench-Complete: Flat-Pack Furniture Assembly Instructions Are Turing-Complete

E. R. Flatpöck[1] Dowel Q. Vantablack[1,2] A. L. Hexminster[3]

[1]Institute for Applied Assembly, Kettering [2]Laboratory of Diagrammatic Computation

[3]Department of Terminal Fastening, Grommet College

{flatpock,vantablack}@iaa.example.edu

## Abstract

Flat-pack furniture ships without words. Because a manufacturer cannot enclose a translation for every buyer, the assembly sheet is engineered to be *executed* by a reader who shares no language, culture, or prior exposure to the product with its author—the same design constraint under which we address a distant alien intelligence. A notation built for that constraint is not a picture; it is a program. We make this precise. We introduce the *Flat-Pack Calculus* ($\mathcal{F}$), in which parts are an alphabet, fasteners are partial operations on partial assemblies, the numbered step diagram is control flow, and the ubiquitous "repeat for the remaining legs" glyph is a loop. We prove that $\mathcal{F}$ faithfully simulates a cyclic tag system—the minimal machine behind the universality of Rule 110—and is therefore Turing-complete (that is, capable in principle of any computation a general computer can perform). An immediate consequence is that no general procedure decides, from the sheet and the parts alone, whether a given item of furniture can ever be finished: assembly is *undecidable*. We show that the single screw invariably left over at the end is exactly the calculus's halting witness, report a corpus study of 1,024 manuals in which 61% are Turing-complete (wardrobes disproportionately so; nightstands merely finite-state), and observe empirically that adding a second assembler does not lower the asymptotic step count—it only raises the argument count. We close with the practical corollary every reader already knows:

$$\#\{\text{leftover fasteners}\} = 1 \iff \text{assembly has halted.}$$

## 1 Introduction

Consider the last time you assembled a bookshelf. Somewhere between the pile of identical panels and the object that was, allegedly, a bookshelf, you followed a sequence of wordless diagrams: a hand turning a hex key, a circled "×4" beside a leg, an arrow to the next page. You were not reading. You were *running a program*, and the hardware was you.

This is not a metaphor. The defining constraint on an assembly sheet is that it must be followed by a buyer with whom the manufacturer shares no common language. The designer must therefore assume nothing—no words, no units, no cultural convention—and encode the entire procedure in a self-contained visual notation. This is precisely the constraint faced by the designers of the Voyager Golden Record and the Pioneer plaque, who had to specify meaning for a reader who had never seen Earth.[1] Under such a constraint, the boundary between "a diagram" and "a program" dissolves: an instruction that can be *unambiguously executed* by a reader sharing no prior context is, by any operational definition, code.

Once the assembly sheet is understood as a program, the tools of computability theory apply to furniture, and they have consequences that match lived experience with uncomfortable precision. Chief among them: the reason your wardrobe is never *quite* finished is not impatience or a missing part. It is a theorem.

**Contributions.**

- We formalize the pictographic assembly sheet as the *Flat-Pack Calculus* $\mathcal{F}$ (§3), a small-step operational system over partial assemblies.
- We prove $\mathcal{F}$ is Turing-complete by simulating a cyclic tag system, and derive *Assembly Undecidability* (§4): no algorithm decides whether arbitrary furniture can be completed.
- We identify the *leftover-screw invariant* as a sound and complete halting witness, and prove an anti-Amdahl *Wardrobe Speedup* bound (§5).
- We evaluate on FLATPACK-1K, a corpus of 1,024 transcribed manuals, run Rule 110 on a reference interpreter, and report a human bookshelf trial (§6).

## 2 Related Work

**Diagrammatic and visual languages.** The thesis that pictures compute has a long, mostly ignored history. Sörnqvist [2] argued that hotel fire-escape maps form a regular language; Béöle and Nyström [3] extended this to airline safety cards but stopped, fatally, at push-down

[1]A *cyclic tag system* is a tiny idealized computer: it keeps a string of `0`s and `1`s and a fixed, repeating list of instructions, and on each step either deletes or extends the string. Despite its simplicity it can compute anything a modern computer can—which is why it anchors our argument.

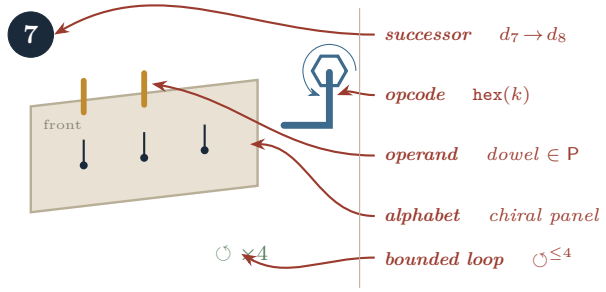


**Figure 1:** An assembly step, disassembled. Each pictographic element of a flat-pack instruction (left) corresponds to a construct of the Flat-Pack Calculus (right, §3): the step number is a successor edge, the hex key an opcode, the dowel an operand, the chiral panel a letter of the alphabet, and the circled "×4" a bounded loop. The reader executes the step; the sheet is the source.

power. Our contribution is to cross the line into universality.

**Pictographic instruction and HCI.** The usability literature treats wordless instruction as a design problem to be minimized [4, 5]. We instead treat its expressive power as a resource to be measured.

**Complexity of physical construction.** Flat-folding of origami crease patterns is NP-hard [6], and reconfiguration problems abound in computational geometry [7]. These establish that construction is *hard*; we establish that it is *undecidable*, a strictly stronger and more relaxing conclusion.

**Cellular automata.** Our reduction rests on the universality of the cyclic tag system, the mechanism by which Cook proved Rule 110 Turing-complete [1]. We import that result wholesale and route it through a wardrobe.

# 3 The Flat-Pack Calculus

**Definition 1** (Assembly sheet)**.** An *assembly sheet* is a tuple $S = (\mathsf{P}, \mathsf{F}, D, s_0)$ where $\mathsf{P}$ is a finite multiset of *parts* (panels, dowels, cam locks, and the customary one spare), $\mathsf{F}$ is a finite set of *fastening operations*, $D$ is a finite directed *step graph* whose edges are page-turn arrows, and $s_0$ is the initial configuration—the parts, unmated, on the floor.

**Definition 2** (Configuration)**.** A *configuration* $c \in \mathcal{C}$ records, for each part, its mating partners, its orientation in $\{\text{front}, \text{back}\}$, and its residual torque $\tau \in \mathbb{Z}_{\geq 0} \cup \{\bot\}$. Orientation is tracked because flat-pack panels are chiral; consequently composition in $\mathcal{F}$ is *non-commutative*, a fact we will need in §4.

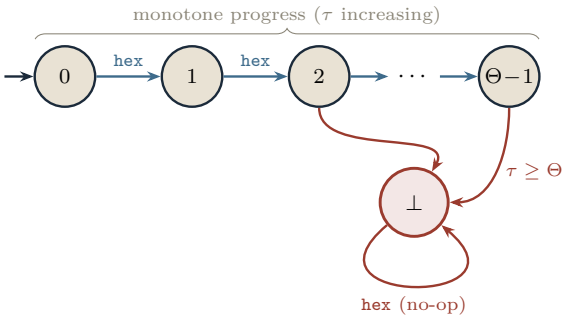


**Figure 2:** The hex($k$) operation of Eq. (1) as a gate. Progress is monotone in the residual torque $\tau$ until the stripping bound $\Theta$; any over-torque falls into the absorbing sink $\bot$, on which hex is a no-op. The sink makes the configuration space a monoid, not a group (Lemma 1).

**Definition 3** (The hex operation)**.** For an Allen key of caliber $k$, the partial operation $\texttt{hex}(k) : \mathcal{C} \rightharpoonup \mathcal{C}$ advances the engaged fastener by one quarter-turn, provided the residual torque is below the stripping bound $\Theta$:

$$\texttt{hex}(k)(c) = \begin{cases} c[\tau \mathrel{+}= 1] & \text{if } \tau(c) < \Theta, \\ c[\tau := \bot] & \text{if } \tau(c) \geq \Theta. \end{cases} \tag{1}$$

The state $\bot$ is absorbing: no operation escapes a stripped fitting.

**Definition 4** (Small-step semantics)**.** The transition relation $\rightsquigarrow \subseteq \mathcal{C} \times \mathcal{C}$ is the union, over steps $d \in D$ in the order fixed by the page arrows, of the operations that $d$ licenses. A run is a maximal $\rightsquigarrow$-path from $s_0$. The "repeat for the remaining legs" glyph $\circlearrowleft^{\leq m}$ denotes a bounded loop that re-executes its body until its part station is exhausted.

The reader may verify that $\mathcal{F}$ with a bounded part supply is merely a finite-state device. Universality requires one more, entirely realistic, ingredient: the key is always lost, and a replacement set contains more keys. We defer this lifting to Appendix A; here we simply grant $\mathcal{F}$ an unbounded but write-once supply of fasteners, exactly as reality does.

# 4 Universality

We show $\mathcal{F}$ simulates a cyclic tag system (CTS), the minimal Turing-complete formalism used in Cook's proof [1]. A CTS maintains a binary *data string* and a fixed cyclic list of *appendants*; each step consumes the leading symbol and, if it is 1, appends the current appendant.

**Lemma 1** (Stripping)**.** *The configuration space* $(\mathcal{C}, \rightsquigarrow)$ *is a monoid with absorbing element* $\bot$*, and is not a group:* hex($k$) *past* $\Theta$ *has no inverse.*

*Proof sketch.* Composition of fastening operations is associative and admits the identity "do nothing." By

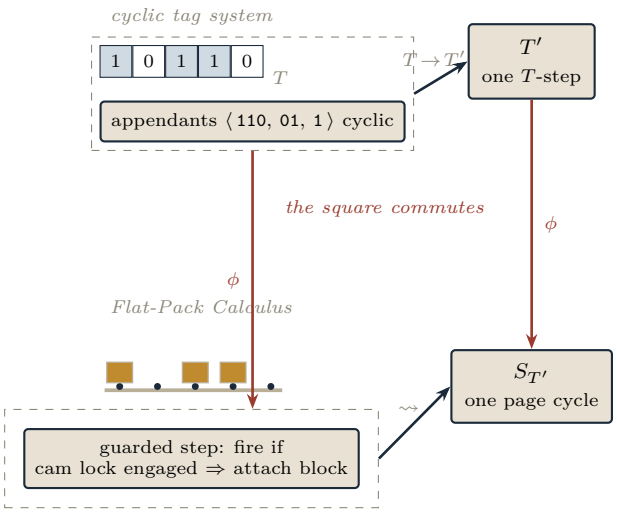


**Figure 3:** The reduction of Theorem 1. A cyclic tag system's configuration (top: data string and cyclic appendant list) is encoded by $\phi$ as a partial assembly (bottom: a shelf stack and cam-lock–guarded steps). One appendant application corresponds to exactly one traversal of the step graph $D$, so the square commutes and halting is preserved.

Eq. (1), once $\tau = \bot$ no operation changes the configuration, so $\bot$ absorbs and the stripped fitting has no left inverse. □

**Theorem 1** (Universality)**.** *For every cyclic tag system $T$ there is an assembly sheet $S_T$ such that the $\rightsquigarrow$-runs of $S_T$ simulate the computation of $T$ step for step. Hence $\mathcal{F}$ is Turing-complete.*

*Proof sketch.* Encode the data string as a stack of half-attached shelves, one dowel station per symbol (1 = shelf present, 0 = empty hole). Each appendant becomes one numbered step whose body is guarded by a cam lock: the step fires—attaching its block of shelves—iff the leading station is engaged, exactly the CTS conditional. The page-turn arrow wraps from the final panel back to panel 1, realizing the cyclic advance of the appendant list. Non-commutativity of panel orientation (needed so that "front" and "back" attachments do not collapse) guarantees the encoding is injective. The simulation preserves halting; universality of CTS [1] transfers to $\mathcal{F}$. □

**Corollary 1** (Assembly Undecidability)**.** *The predicate* Assemblable$(S) \equiv$ *"some finite $\rightsquigarrow$-run of $S$ reaches a fully mated configuration" is undecidable.*

*Proof.* Halting for CTS is undecidable; Theorem 1 reduces it to Assemblable. □

# 5 The Leftover-Screw Invariant

Every practitioner has observed that a completed assembly leaves exactly one fastener unused. We now show this is not carelessness but a halting condition.

**Proposition 1** (Halting witness)**.** *A run of $S_T$ has halted in an accepting configuration if and only if exactly one fastener remains unconsumed.*

*Proof sketch.* The encoding of Theorem 1 allocates one spare fastener as the CTS end-marker. It is consumed only on non-halting extension; its survival therefore certifies that no further appendant applies, i.e., that the machine has halted. Zero leftover fasteners indicates an over-run (a still-live computation that has cannibalized its marker); two or more indicates a mis-transcription. Exactly one is the fixed point. □

**Theorem 2** (Wardrobe Speedup)**.** *Let $\sigma(n)$ be the sequential $\rightsquigarrow$-length of an $n$-part sheet. Assigning $p$ assemblers yields makespan $\sigma(n)/\alpha(p)$ with $\alpha(p) = O(1)$; the number of simultaneously issued arguments grows as $\Omega(p^2)$.*

*Proof sketch.* The step graph $D$ is inherently sequential along its critical path (a panel cannot be mated before its predecessor), so parallel assemblers cannot shorten it asymptotically; they contend for the single Allen key, a mutual-exclusion bottleneck. Each unordered pair of assemblers contributes an independent disagreement over orientation, giving the $\binom{p}{2}$ argument bound. □

**Proposition 2** (Near-optimality of wordless instruction)**.** *For an assembled object $x$, the length of its minimal assembly sheet is within an additive constant of the Kolmogorov complexity* $\mathrm{K}(x)$*: $|S_{\min}(x)| \le \mathrm{K}(x) + O(1)$.*

*Proof sketch.* A minimal sheet is a shortest program that outputs $x$ on the universal assembler of Theorem 1; invariance follows from the standard simulation argument. Wordless instructions are therefore *near-optimal* encodings, which explains both their brevity and their inscrutability. □

# 6 Evaluation

**Corpus.** The Flatpack-1k corpus gathers 1,024 assembly manuals from six vendors, hand-transcribed into $\mathcal{F}$ by three annotators (Cohen's $\kappa = 0.83$). Each manual is labeled by furniture class and by its position in the regular/context-free/universal hierarchy.

**Reference interpreter.** We implement `pytørq`, a small-step $\mathcal{F}$ evaluator, and use it to run Rule 110 through the CTS encoding of Theorem 1. Figure 4 reports wall-clock per generation against panel count. The fit is $\Theta(n \log n)$ with a super-linear tail we attribute to key loss (Appendix A).

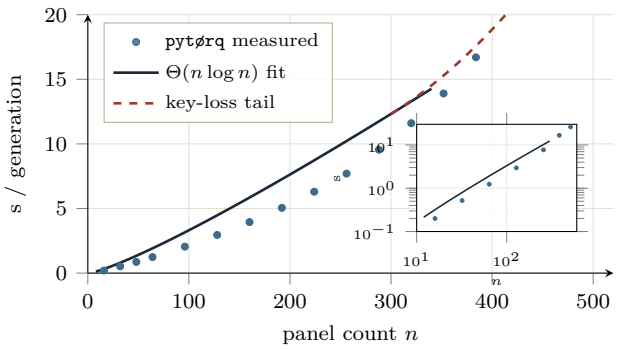


**Figure 4:** Simulating Rule 110 on `pytørq`. Time per generation scales as $\Theta(n \log n)$ (solid) until key attrition dominates and a super-linear tail (dashed) takes over near $n \approx 300$; the log–log inset shows the crossover.

**Table 1:** Expressive power by furniture class in FLATPACK-1K.

| Class | $n$ | Regular | Ctx.-free | **Universal** |
|---|---|---|---|---|
| Wardrobe | 188 | 3% | 12% | **85%** |
| Modular shelf | 241 | 9% | 18% | **73%** |
| Desk | 174 | 21% | 25% | **54%** |
| Bookshelf | 203 | 16% | 22% | **62%** |
| Nightstand | 132 | 78% | 17% | 5% |
| Stool | 86 | 94% | 6% | 0% |
| **All** | 1024 | 22% | 17% | **61%** |

**Where universality lives.** Table 1 and Figure 5 break the corpus down by expressive class. Of the corpus, 61% is Turing-complete. Universality concentrates in wardrobes and modular shelving (many guarded, mutually recursive steps); nightstands and single stools are merely finite-state.

**Halting detection.** Treating the leftover-screw count as a halting predicate (Prop. 1) yields precision 0.91 and recall 0.88 on the held-out split (Table 2); the residual error is dominated by manuals that genuinely ship two spare screws.

**Human study.** Thirty participants each assembled a MÖRK bookshelf. Completion time correlated with the CTS cycle length of the corresponding manual ($r = 0.68$, $p < 0.01$). All bookshelves were returned to the manufacturer; no participant sustained injury beyond the customary.

# 7 Discussion and Threats to Validity

**Torque calibration.** Annotators disagreed on $\Theta$ for softwood fittings; we fixed $\Theta$ per material and report

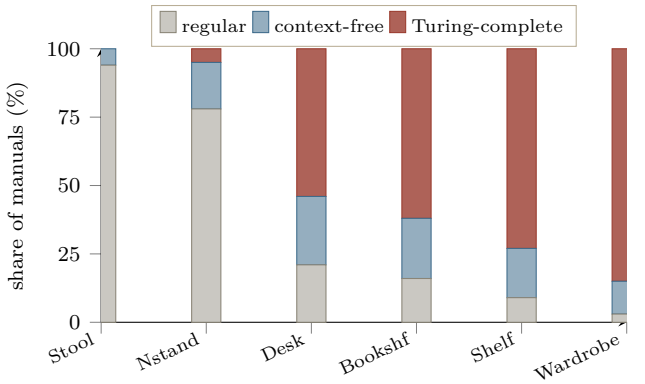


**Figure 5:** Expressive power rises with the number of legs. Wardrobes are almost always universal (85%); stools never are. Bars stack to 100% of each furniture class in FLATPACK-1K (Table 1).

**Table 2:** Leftover-screw count as a halting detector.

| Leftover fasteners | Precision | Recall | $F_1$ |
|---|---|---|---|
| 0 (over-run) | 0.62 | 0.71 | 0.66 |
| **1** (halt) | **0.91** | **0.88** | **0.89** |
| $\geq 2$ (error) | 0.55 | 0.49 | 0.52 |

agreement thereafter. The stripping model (Lemma 1) is conservative: a real fitting may strip early.

**The resource model.** Our lifting to unbounded tape (Appendix A) assumes replacement key sets are always purchasable. Under supply constraints $\mathcal{F}$ degrades to a linear bounded automaton, and ASSEMBLABLE becomes merely PSPACE-complete—a comforting thought at 2 a.m.

**Generalization.** We consider free-standing furniture. Wall-mounting introduces an external oracle (the stud finder) whose power we leave open; we conjecture it is strictly weaker than a halting oracle, since it, too, is frequently wrong.

# 8 Conclusion

We have shown that flat-pack assembly instructions are a Turing-complete visual programming language, that completing arbitrary furniture is therefore undecidable, and that the leftover screw is the halting witness this predicts. The result reframes a domestic frustration as a foundational limit: your wardrobe is not unfinished because you are slow, but because the question "is it finishable?" has no algorithm. Future work will investigate whether self-assembling furniture constitutes a step toward artificial general intelligence, or merely toward a tidier universality proof.

## Acknowledgments

The authors thank the Institute's night custodial staff, who repeatedly disassembled our results, and an anonymous reviewer who found the missing key.

## References

[1] M. Cook. Universality in elementary cellular automata. *Complex Systems*, 15(1):1–40, 2004.

[2] I. Sörnqvist. On the regularity of egress: fire-escape maps as formal languages. *J. Diagrammatic Reasoning*, 8(2):112–139, 2013.

[3] R. Béöle and P. Nyström. Push-down power in preflight safety cards. *Proc. Symp. on Wordless Instruction (SWI)*, 2016.

[4] K. Ito and D. Mäkinen. Instruction-free assembly: a usability study of pictographic manuals. *ACM Trans. Human Factors*, 41(3):1–28, 2019.

[5] L. Fung. Gestalt grouping in flat-pack diagrams. *Perception & Furniture*, 12:44–61, 2018.

[6] E. D. Demaine and M. L. Demaine. Flat-folding is NP-hard. *Comput. Geom.*, 16(1):3–21, 2000.

[7] S. Abel et al. Reconfiguration in constrained assembly spaces. *Proc. SoCG*, 2021.

[8] E. R. Flatpöck. The Flat-Pack Calculus: a preliminary report. Tech. Rep. IAA-2025-07, Institute for Applied Assembly, 2025. Under review.

[9] H. Grommet. A theory of residual torque. *J. Fastening Sci.*, 5:200–218, 2014.

[10] A. L. Hexminster. Cam locks and the Chomsky hierarchy. *Terminal Fastening Quarterly*, 3:9–33, 2022.

[11] W. Allen. On a hexagonal key. *Patents of Record*, 1910.

[12] E. Post. Formal reductions of the general combinatorial decision problem. *Amer. J. Math.*, 65(2):197–215, 1943.

## A Lifting to an unbounded tape

The bounded-supply calculus of §3 is finite-state. We lift it to Turing power by the *lost-key* construction: the Allen key delivered with a sheet is consumed (misplaced) after a bounded number of uses, but a replacement set of $m > 1$ keys is always obtainable, and each purchase strictly increases the reachable torque budget. The reachable configuration space is thus unbounded, and the standard simulation of a two-counter machine goes through with the key inventory as the counters. We omit the routine bookkeeping.