COMPILED WORLDS: WHAT ADAPTATION COSTS ON TRANSFORMED ARC-AGI-3 GAMES

Anonymous authors
Paper under double-blind review

ABSTRACT

Completion on familiar games does not show how an agent adapts when a game changes, or what the adaptation costs. We introduce *Compiled Worlds*, a protocol that edits official ARC-AGI-3 game programs into paired variants (colour permutations, action relabellings, resampled layouts and rule mutations), serves them through the unmodified toolkit, and attaches executable solution witnesses to layouts and mutations. We evaluate four harnesses with a common backbone, GLM-5.3, and selected configurations with three further backbones. Because every variant has an original, the protocol separates what a harness costs on a familiar level from what a change adds. On the original first levels of the five games with layout variants, Tycho with GLM-5.3 already uses nine times the calls of a compile-by-intervention harness (CbI) with the same backbone. Measured against its own originals, Tycho pays mainly for new layouts, 3.3 times its original first-level calls and more on all eight layouts it completes, whereas CbI's premium is at most 1.4 on every tier; each fails five of thirteen layouts. First-level rule mutations cost about as much as the original levels, and each model-building configuration evaluated on them with an API backbone solves at least four of five. Across 33 paired first runs, colour permutation does not change completion measurably. Finally, ten of CbI's twelve zero-completion first runs with GLM-5.3 reach full agreement with the logged transitions, so trace agreement does not ensure completion.

1 INTRODUCTION

An agent may solve a familiar game yet struggle when its colours, controls or layout change. Even when it succeeds, adaptation may require substantially more interaction or computation. ARC-AGI-3 makes these questions concrete: agents must infer rules and goals through interaction, while the public games also serve as development environments. Their scores leave unresolved which capabilities transfer and at what cost (ARC Prize Foundation, 2026a).

In each game, the agent observes a grid, selects an action and sees the resulting frame and level progress. It must discover both the dynamics and the completion condition through interaction. A *harness* supplies the model's prompts, memory, tools and stopping rules. Compiled Worlds supplies transformed environments and measurements. Alongside three published harnesses we evaluate CbI, a compile-by-intervention harness that we developed before this study on four of the eight evaluated games (Section 4), as an instrumented reference: its logs record every compiled model, trace test and goal proposal. It is evaluated under the same protocol, and the contribution is the protocol and the comparative evidence, not CbI.

A useful evaluation must relate changes in performance to changes in the task. Private and semi-private evaluations test performance on unseen games; Compiled Worlds complements them with paired interventions on accessible programs: which changes alter completion, and at what resource cost? A performance drop measures sensitivity, not evidence of contamination by itself.

Compiled Worlds makes these comparisons by transforming the official game programs. Each Python game program specifies its sprites, levels and transition logic. A colour permutation changes what the agent sees while preserving the dynamics; a rule edit changes the dynamics on a familiar layout (Figure 1). Layout resampling supplies a third comparison: new positions under the original rules. The `arc-agi` package (v0.9.9; ARC Prize Foundation, 2026c) serves each version through

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

the standard interface, so the same harness can play both members of a pair. Search supplies executable solution witnesses and reference action counts for layout and rule variants.

We make three contributions.

1. **A program-based evaluation protocol.** Compiled Worlds generates colour permutations (T1), action relabellings (T1b), first-level layout variants (T2) and rule mutations (T4), with transformation audits, determinism checks and search-based solution certificates (Section 3). Generator coverage and the evaluated subset are specified separately.
2. **A model-matched harness comparison.** We evaluate the ARC Prize benchmarking harness, Tycho, Twin and CbI with GLM-5.3, plus selected configurations with three additional backbones (Section 4). Common model identity permits comparisons of complete harness configurations under their stated budgets.
3. **Measurements of adaptation and its cost.** Paired completion against measured run-to-run variation, each harness's cost relative to its own original levels, and execution traces that distinguish model-building failures from unsuccessful goal proposals (Section 5).

The central finding is that paired originals separate what a harness costs on a familiar level from what a change adds. (i) With GLM-5.3, Tycho uses about nine times CbI's calls already on the original first levels; on the six layouts both complete the ratio is 11.7 against 5.8 on their originals, so the layout change doubles a gap that the harnesses bring with them. (ii) Measured against its own originals, Tycho pays mainly for new layouts (3.3 times its first-level calls), CbI at most 1.4 times on any tier, and both fail five of thirteen layouts. (iii) First-level rule mutations cost about as much as the originals and saturate completion for the model-building configurations with API backbones; colour permutation and action relabelling do not change completion measurably. (iv) A transition model that matches every logged transition can still fail, through goal proposals or unseen states. These findings concern eight games, first-level layout and rule variants, and the stated budgets.

2 Related work

ARC-AGI-3 systems. ARC-AGI-3 extends the static grid puzzles of ARC-AGI-1/2 (Chollet, 2019; Chollet et al., 2025) to interaction. Tycho (Lehmann et al., 2026) separates an actor from a builder that writes and repairs a programmatic world model; its orchestrator reports 100 relative human action efficiency (RHAE) with GPT-5.6 Sol and Opus 5, under a budget of 3,500 language-model calls per game. Twin (Skoutnev et al., 2026) reports 179/183 levels with executable `step(grid, action)` and `goal_reached` models that must replay every logged transition before an action is scored, and Schema (Zeng et al., 2026) reports 98.98 RHAE with backtested executable models. OPINE-World (Courtis et al., 2026) and executable world models (Rodionov, 2026) explore object-centric and single-agent variants; AERA (Liew, 2026) motivates trivial-policy controls. AVO (Chen et al., 2026) and SwarmOS (INT21 AI, 2026) report 100 RHAE with general-purpose agent systems, each completing all 183 levels. We also evaluate the standard direct-play configuration of the ARC Prize benchmarking harness (ARC Prize Foundation, 2026b), distinct from the provider-adapter configuration behind GPT-6 Astra's reported 99.9 semi-private score (ARC Prize Foundation, 2026d).

Generalisation and memorisation in interactive benchmarks. Overfitting to fixed environments is well documented in deep RL (Zhang et al., 2018; Farebrother et al., 2018); CoinRun and Procgen (Cobbe et al., 2019; 2020) introduced procedurally generated levels precisely to separate training levels from test levels, and Kirk et al. (2023) survey the resulting notion of zero-shot generalisation. Schema Networks (Kansky et al., 2017) evaluated Breakout variants (paddle offsets, wall positions, new goals) to test whether a learned causal model transfers, the Arcade Learning Environment exposes game modes and difficulties as variants of the same game (Machado et al., 2018), and HackAtari introduces controlled variations of Atari games (Delfosse et al., 2024). For static ARC tasks, RE-ARC generates new examples from per-task generator programs (Hodel, 2024) and ConceptARC organises tasks around concepts (Moskvichev et al., 2023). Our tiers bring this tradition to interactive ARC-AGI-3 games by transforming the official programs rather than sampling a fresh distribution, which allows paired comparisons against the public games used during development.

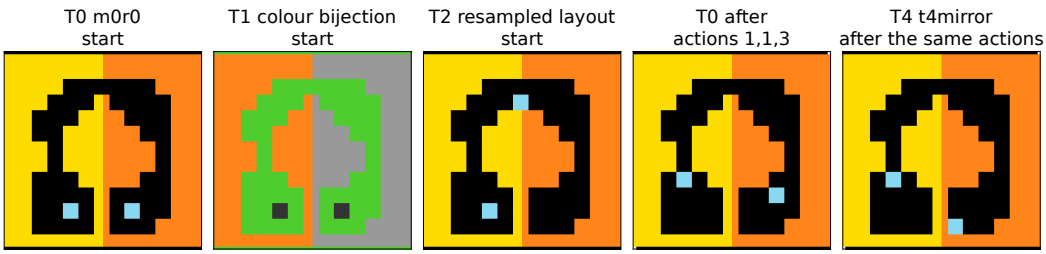


Figure 1: One game (`m0r0`) across the ladder, rendered with the official palette. Left to right: the T0 start frame; the T1 version (a colour bijection of every rendered frame, dynamics untouched); a T2 version (one small sprite moved to a new position, certified solvable by breadth-first search with the same reference length); and the T0 game versus the T4 `t4mirror` mutation after the same three actions, where the mirrored twin has moved along the other axis. Every version is served through the toolkit as an ordinary game.

Contamination in LLM evaluation. GSM1k (Zhang et al., 2024) evaluates fresh problems resembling an established benchmark, GSM-Symbolic (Mirzadeh et al., 2025) instantiates templates of familiar problems with new values and clauses, and counterfactual tasks (Wu et al., 2024) compare default and altered versions of familiar tasks; contamination audits (Sainz et al., 2023) examine whether benchmark exposure compromises evaluation. Compiled Worlds brings controlled variation to interactive tasks by transforming the programs that generate them. CbI checks predicted transitions by execution, whereas LLM-as-a-Verifier (Kwok et al., 2026) scores them from a judge's token logits; agreement on a finite trace does not establish correctness on unseen states.

3 Compiled Worlds

3.1 The official format as a generator

Games define sprites, levels and a `step` method, executing locally at thousands of frames per second. The toolkit loads a Python module and `metadata.json` from `environment_files/<game>/<tag>/`; metadata give the game id and per-level human baselines. We rewrite the source into a new version tag. Baselines are retained unless the search reference changes, when they are scaled by the reference ratio. Each run uses a private directory containing only its version, preventing the toolkit's newest-prefix resolution from selecting another game (Appendix M).

3.2 Tiers

T0 (reference). The eight original games in the evaluation are four games from our earlier development work (`m0r0`, `tu93`, `ar25`, `tr87`) and four further games whose first level yields a search certificate (`cd82`, `cn04`, `ft09`, `lp85`; Section 3.3); `ls20` also certifies but was not in the plan. They establish reference performance under our backbones and budgets.

T1 (colour bijection). A subclass wraps the original game and applies a fixed permutation of the sixteen colours to every rendered frame; the dynamics program is untouched. The permutation is drawn per version from a seeded generator. An audit plays the same 40 random actions on the original and on the version and checks that every observed frame is the permutation of the original frame; 24 of 25 public games pass this audit. The exception, `bp35`, has reset-dependent hidden state and is excluded from certificate tiers.

T1b (action relabelling). For the 16 games controlled by the four arrow actions, a derangement of the action ids $\{1, 2, 3, 4\}$ is applied before the engine sees the action. The audit verifies exactness under the inverse map. A system that has learned "action 1 moves up" from the public games must re-derive the mapping.

T2 (certified layout re-sampling). An abstract syntax tree (AST) editor identifies the sprites of a game's first level, computes their areas, and moves a random subset of the small sprites (area at most 25% of the grid) to new positions. A candidate is accepted only if the start frame differs from the original, the level is deterministic, and breadth-first search (BFS) certifies the level solvable with a search reference within $[0.5\times, 3\times]$ of the original's; the human baseline is scaled by the reference ratio. T2 versions are single-level games. Seeds are tried in order and the first three accepted per game are kept (fewer where fewer of the first twelve seeds pass), which yields thirteen evaluated layouts: three each for `m0r0`, `cd82`, `cn04` and `lp85` and one for `ft09`. The first accepted layout per game formed the original plan; the other eight were added under the same rule and run on Tycho and CbI with GLM-5.3.

T4 (rule mutation). A registry lists, per game and tag, a description and a set of exact-once source edits. A mutation is accepted only if it (i) diverges from the original during a 40-action random rollout, (ii) is deterministic, and (iii) has a BFS certificate. The accepted mutations are listed verbatim in Appendix K: `m0r0` mirror axes swapped (reference 21 vs 15) and mirror removed (20); `cd82` wrap-around instead of clamping (2 vs 5); `cn04` counter-clockwise rotation (12 vs 14); `lp85` inverted button semantics (5 vs 5). Six further candidates (three for `tu93`, one for `ar25`, two for `ft09`) were rejected for lacking a certificate or an observed divergence; all are listed with the accepted ones.

T3 (recombination). Embedding one game's mechanic in another's world would require a compositional engine. We reserve T3 for this future extension; it is not part of the matrix.

3.3 Certificates

We check determinism by independently replaying the same action sequence twice and comparing the frames. We then search engine snapshots with BFS, subject to depth, node and time limits. First-level sweeps use a depth cap of 24 actions; mutations allow deeper searches. The search enumerates discrete actions and, where supported, sprite-centre click targets with an optional coarse grid. Nine public games yield a first-level solution within 120 s. Click-only games yield no certificates at these caps and are absent from T2 and T4.

A returned sequence witnesses solvability in the official engine; its length, the *search reference*, sets T2/T4 efficiency and baseline scaling. Because the search merges states by rendered frame, a shortest-path reading would require merged states to behave identically and the enumerated actions to cover every relevant choice; we do not treat the reference as a global optimum. Similarly, the determinism and transformation audits check sampled trajectories only.

4 Experimental setup

4.1 Systems

Table 1 lists the systems and budgets. The plain harness uses the ARC Prize benchmarking repository in its standard mode with an OpenAI-compatible model configuration. Tycho uses its orchestrator policy; Twin uses its Codex-CLI mode with a custom provider. CbI compiles an executable model from probes, tests it against logged transitions, repairs failures, proposes goal predicates, plans over the model and executes the resulting plans. It does not read pre-existing replays. We built and debugged CbI on six public games (`m0r0`, `tr87`, `tu93`, `ar25`, `su15`, `bp35`) before this study; four of them are among the eight evaluated games, and `cd82` and `cn04` were each started once in an earlier evaluation that was stopped. Its prompts and code name no game, but those four games are familiar to its developers, as all public games are to the published harnesses, which report results on them. We include CbI because its logs expose each step, and its runs follow the same matrix, isolation and reporting rules as the published harnesses. All four systems use GLM-5.3 (Z.ai, 2026) through Z.ai. Additional lanes evaluate the plain harness and CbI with local Qwen3.8-27B (Qwen Team, 2026) (4-bit, Ollama, one RTX 3090); Twin and CbI with GPT-6 (OpenAI, 2026) at medium reasoning effort through a Codex subscription; and the plain harness, Tycho and CbI with DeepSeek V4.1 Flash (DeepSeek-AI, 2026) at high effort. Coverage varies by lane because of cost and usage limits.

Table 1: Systems, backbones and budgets. GLM-5.3 starts at *high* reasoning effort. Wall-time caps are 5 h by default and 10 h for GPT-6; quota waits and resumptions are documented in Appendix M. Patches cover backbone routing, environment directories, Tycho's required text-only messages, and plain-harness effort fallback after empty or timed-out answers (Appendix L).

System	What it represents	Backbone(s)	Budget per game
ARC Prize benchmarking harness	official standard direct play	GLM-5.3; DeepSeek V4.1 Flash; Qwen3.8-27B	$1.5\times$ human baseline actions per level
Tycho (orchestrator)	published 100-RHAE world-model system	GLM-5.3; DeepSeek V4.1 Flash	300 LLM calls; its own cut-off of $5\times$ human baseline actions per level
Twin	published 179/183 replay-validated system	GLM-5.3; GPT-6 (medium)	$2\times$ human total actions (subset)
CbI	instrumented reference harness	GLM-5.3; GPT-6 (medium); DeepSeek V4.1 Flash; Qwen3.8-27B	60 LLM calls; 1000 actions
Trivial controls	random (3 seeds), each single action repeated, blind click sweep	none	120 actions per policy

Deviations from published configurations. The published harnesses do not run here as their authors configured them. Tycho runs with vision disabled, because the endpoint accepts only text, with 300 calls instead of its published 3,500, and on GLM-5.3 or DeepSeek V4.1 Flash instead of the GPT-5.6 Sol and Opus 5 backbones behind its 100-RHAE result; Twin, which re-sends its whole thread on every tool call, runs on a subset of games. Results therefore characterise these configurations, not the published systems at their reported settings. Calibration on `m0r0` confirms non-trivial progress for each scaffolded system (Table 10, Appendix J).

4.2 Matrix, isolation and metrics

A *version* is one original or transformed game. A *lane* fixes the harness and backbone; a *cell* pairs a lane with a version, and may have repeated runs. Each lane executes one cell at a time. The full plan has 32 distinct cells per lane: 8 games $\times$ {T0, T1}, 6 T1b, 5 T2 and 5 T4; Tycho and CbI with GLM-5.3 additionally run the eight further layouts, and Tycho repeats all five mutations and five layouts once. Each cell gets a private environment directory holding exactly one version, and, for the plain harness, a dedicated local server. Each run records completed levels, actions, model calls, tokens, wall time and status. Appendix A maps reported counts to their run populations and gives a reuse procedure.

Metrics. (i) Level completion per tier. (ii) Paired completion changes from T0 to T1 and T1b, measuring transformation sensitivity without attributing its cause. (iii) *Efficiency*: on T0, T1 and T1b, actions per completed level against the human baseline using an RHAE-style score; on T2 and T4, against the search reference (a reference-normalised score, ROAE $= \min(1, \text{reference}/\text{agent})^2$ per completed level). (iv) For CbI, compile success, trace-test pass rate and repairs per level.

Trivial controls. All 40 versions are also played by eleven model-free policies (random with three seeds, each of the seven actions repeated, a blind click sweep), 120 actions each; we report the best trivial policy as a control, since completion of a level that it also clears is weak evidence of rule induction (Liew, 2026). A random policy clears one version, the `m0r0` layout with seed 8, in 20 actions (search reference 8); no trivial policy clears any other level. Liew (2026) reports every public game reachable by such strategies, but on our 40 versions no single first action or single action repeated 300 times completes a level; the steps reported there are game-over states (Appendix H).

Repeats. Each run samples a stochastic policy; selected cells have independent repeats. The plain harness repeats the T0/T1 pairs of three games (`lp85`, `cn04`, `m0r0`); Tycho with GLM-5.3 repeats all five mutations, the first five layouts and two colour-tier losses once; CbI with GLM-5.3 repeats its original `lp85` run, three colour-tier cells (`cd82`, `cn04`, `lp85`) and two layouts once. Apart from

Table 2: Adaptation premium with GLM-5.3, first runs: answered LLM calls spent before completing the first level of a variant, relative to the same harness's calls on the original game's first level. *total*: ratio of summed calls over completed variants; *more*: variants with more calls than their original; *failed*: first level not completed (for T1 and T1b: completed on the original only). `tr87` is excluded because neither harness completed its original first level; CbI's `lp85` original took one call, which inflates its `lp85` ratios.

	Tycho				CbI			
Change	total	median	more	failed	total	median	more	failed
colour permutation (T1)	1.24	1.36	6/7	0	1.42	1.14	3/5	2
action relabelling (T1b)	0.74	1.30	3/5	0	0.93	0.93	0/4	1
layout (T2)	3.26	3.66	8/8	5	1.29	1.12	5/8	5
rule mutation (T4)	1.09	1.41	4/5	0	0.53	0.64	1/4	1

Tycho's mutations and layouts, repeats were chosen after inspecting first runs, mostly where levels were lost or harnesses disagreed, so they are not a random sample. Tables average over repeats per version and state the count.

What is held fixed. Within each lane, model configuration, nominal budgets and isolation are fixed across tiers (retry and resume policies: Appendix M); across systems only model identity is controlled, so the comparison measures complete harness configurations, not equal compute.

5 RESULTS

Table 3 gives levels completed per lane and tier, averaged within versions over repeats (every cell: Appendix N). First-run comparisons in the text, the cost and premium analyses and the paired test exclude repeats. Direct play stays near the floor on every tier, so the comparisons concern the scaffolded harnesses.

5.1 BASELINE COST AND ADAPTATION COST

Because every variant has an original, a harness can be compared with itself. For each variant whose first level a harness completes, we divide the answered LLM calls it spends before completing that level by the calls it spent on the original game's first level in its T0 first run. Table 2 summarises these *premiums* for Tycho and CbI with GLM-5.3, which differ before any change: on the original first levels of the five games with layout variants, Tycho spends 277 calls and CbI 30.

Only layouts raise Tycho's cost consistently: its eight completed layouts need 3.3 times the calls of their originals' first levels in total, and more on every one of them. Colour permutation raises its first-level calls by a quarter (six of seven games), rule mutations change them little in total (1.09) and action relabelling lowers them (0.74). CbI's premiums stay between 0.5 and 1.4 on every tier; its ratios below one on mutations come from `m0r0`, one of its development games (Section 4). Both harnesses fail five of thirteen layouts; on the other tiers Tycho fails no first level and CbI at most two. Each ratio rests on one run of the original, and calls on one version vary by up to 6.6 times between runs (Section 5.2), so single ratios are indicative.

5.2 RULE MUTATIONS SATURATE COMPLETION (T4)

Tycho with GLM-5.3 solves each of the five mutated levels, at 1.25 to 15.5 times the search reference (first-run mean ROAE 0.38) and 30 to 126 calls, and solves all five again in a second run, with call counts moving by up to 6.6 times between runs (41 and 149 on the mirror swap, 126 and 19 on the wrap-around). With DeepSeek V4.1 Flash it solves the same five at 9 to 37 actions and 25 to 142 calls. CbI with GLM-5.3 clears four of the five (not the `cd82` wrap-around) at 1 to 6 calls each and a first-run mean ROAE of 0.17. With GPT-6, Twin and CbI each solve all five in a few minutes, Twin at one to two times the search reference and CbI at one to three calls. Direct play with GLM-5.3 clears only the two-action `cd82` wrap-around, and CbI with Qwen3.8-27B none.

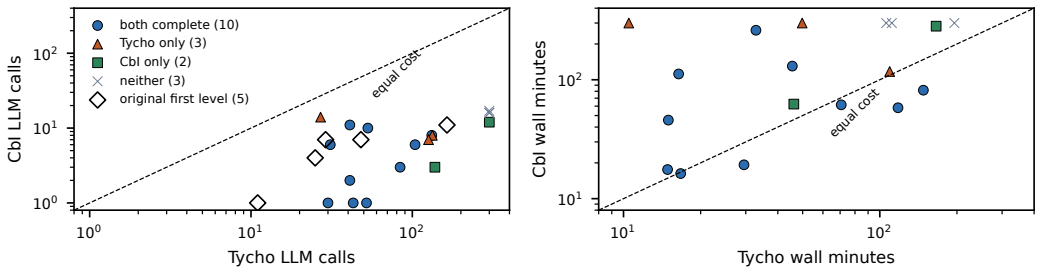


Figure 2: Tycho versus CbI with GLM-5.3 on the same 18 layout and mutation versions (first runs, one point per version, marker by which harness completed the level), costs up to completion or, for failures, for the whole run. Left: answered LLM calls, with the original first levels of the five games (diamonds) as a baseline; CbI uses fewer calls on all 18 versions and on all five originals. Right: wall minutes; CbI takes longer on 13 of 18.

Completion on this tier is at or near its ceiling for the five model-building configurations with API backbones, and the total premium is at most 1.09 (Table 2): these mutations separate harnesses from direct play, not from each other. As single source edits to the first level of a familiar game, they test whether a harness revises a rule it has just observed, not deeper or multi-level changes, which the same registry can express.

5.3 Layouts: the change that raises cost (T2)

With GLM-5.3, Tycho completes three of the first five layouts on first runs (`cn04`, `lp85`, `ft09`), using 2.8 to 5.3 times the calls of the corresponding original first level. Repeats recover both failures (62 and 157 calls) and clear the other three again. CbI completes four of the first five, including both Tycho misses (`m0r0` in 3 calls, `cd82` in 12). On the eight further layouts Tycho completes five and CbI four (`m0r0` and `lp85`; its `cd82` and `cn04` runs end at the 5-hour wall cap after 14 to 17 answered calls). Over all thirteen each completes eight, six in common, one of which (`m0r0` seed 8) a random policy also clears. With GPT-6, CbI completes all five planned layouts and Twin four (`cd82` partial); Tycho with DeepSeek V4.1 Flash completes two and direct play none.

Equal completion counts hide unequal costs, but much of the difference is present before any change (Figure 2). Up to completion, the six layouts both harnesses complete take Tycho 457 calls, 29.4M tokens and 432 minutes, and CbI 39 calls, 3.1M tokens and 609 minutes. On the original first levels of the same games Tycho already uses 5.8 times CbI's calls (139 versus 24); the layouts raise the ratio to 11.7 because Tycho's premium on them (3.3) is twice CbI's (1.6). Fewer calls do not imply shorter runtime or better action efficiency: CbI takes longer on 13 of the 18 layout and mutation versions, and its completed layouts score a pooled ROAE of 0.05 against Tycho's 0.23 (Table 6, Appendix C). Recorded totals, including failures, are in Table 5 (Appendix B). The pattern fits layout-dependent model construction but does not separate familiarity from difficulty.

5.4 Presentation changes: completion within run variation (T1, T1b)

Colour permutation (T1). Tycho with GLM-5.3 completes 24 levels on the eight original games and 20 on their colour-permuted versions in first runs: it loses two levels on `tu93` and one each on `ar25`, `cd82` and `lp85`, and gains one on `tr87`. The `tu93` and `lp85` losses follow about three and two times the calls on an earlier level (133 against 43 on level 3; 81 against 40 on level 2); `ar25` spends 1.2 to 1.4 times the calls on three of its first four levels before reaching the call cap, and `cd82` stops at its action cutoff on level 2 (Figure 5, Appendix E). Repeats recover the `cd82` loss (2/6, matching T0) and retain the `lp85` one (3/8 versus 5/8). CbI with GLM-5.3 completes 15 and 17 levels in first runs. With GPT-6 both harnesses retain completion on `lp85` and `cd82` but lose levels on `cn04`: Twin from 4 to 3 in partial runs, CbI from 4 to 0, the goal-proposal failure of Section 5.5. Direct play with GLM-5.3 moves in both directions near the floor (2.17 versus 1.83 levels after averaging repeats; on `lp85` 1.7 versus 0.3 over three runs each).

Table 3: Levels completed / levels available per lane and tier, summed over the games of the tier (T0 and T1: eight games; T1b: six; T4: five single-level games; T2: thirteen single-level layouts for Tycho and CbI with GLM-5.3 and the trivial controls, five for other lanes); means over independent repeats where marked (nr). Last column: paired T0→T1 levels over the games with both cells.

System	Backbone	T0	T1	T1b	T2	T4	T0→T1
Plain harness	GLM-5.3	2.17/55 (12r)	1.83/55 (12r)	0/41	0/5	1/5	2.17→1.83
Tycho	GLM-5.3	24/55	20/55 (10r)	13/41	9/13 (18r)	5/5 (10r)	24→20
Twin	GLM-5.3	10/20	12/20	–	–	–	10→12
CbI	GLM-5.3	14.5/55 (9r)	18/55 (11r)	10/41	8/13 (14r)	4/5	14.5→18
Twin	GPT-6 medium	22/26	17/20	–	4/5	5/5	18→17
CbI	GPT-6 medium	21/26	20/26	–	5/5	5/5	21→20
Plain harness	DeepSeek V4.1 Flash	1/26	1/26	–	–	–	1→1
Tycho	DeepSeek V4.1 Flash	7/26	7/26	–	2/5	5/5	7→7
CbI	DeepSeek V4.1 Flash	4/20	1/20	–	–	–	4→1
Plain harness	Qwen3.8-27B	0/55	0/55	1/41	0/5	0/5	0→0
CbI	Qwen3.8-27B	0/55	0/55	0/41	1/5	0/5	0→0
Trivial (best of 11)	none	0/55	0/55	0/41	1/13	0/5	0→0

Paired analysis. Pooling the first runs of the seven scaffolded lanes gives 33 lane–game pairs: 11 lose levels under permutation, 16 are unchanged and 6 gain, for 99 versus 94 levels (Table 8, Appendix G). An exact two-sided sign test over the 17 untied pairs gives $p = 0.33$; Tycho with GLM-5.3 alone (four losses, one gain) gives $p = 0.38$. Pairs sharing a game are dependent and the test detects only large effects, so the data do not establish a drop; they do not show that none exists. Of the twelve multi-level versions with a repeated run, nine differ between their first two runs by one or two levels, although these repeats were chosen after inspecting first runs and may overstate typical variation. The clearer effect of colour permutation is on cost (Table 2).

Action relabelling (T1b). Under a derangement of the four arrow actions Tycho completes 13 levels on the six relabelled games against 15 on the same games with the original actions, losing one level on `tu93` and `ar25`, with completion-conditioned efficiency of 1.02 (versus 1.08 across T0). CbI with GLM-5.3 completes 10 against 11, at similar efficiency (0.67 versus 0.65 across T0). Over these twelve pairs four lose, seven are unchanged and one gains ($p = 0.38$), and neither harness's first-level calls rise consistently (Table 2). Direct play clears none of the relabelled games with GLM-5.3 and one level (`cd82`, in 11 actions) with Qwen3.8-27B; no trivial policy clears any.

5.5 Trace agreement does not ensure completion

Where Tycho exhausts its calls while building a model, CbI can match observed transitions without finding a successful goal: on permuted `cd82`, 46 planned sequences execute as predicted without a level-up (a repeat completes both levels, so the effect is stochastic). GPT-6 CbI shows the same pattern on `cn04` (4/6 original levels, none permuted), consistent with goal proposals leaning on familiar colour meanings. Permuted `cn04` with GLM-5.3 fails differently: its model matches the logged transitions, but plans break on mispredicted unseen transitions.

CbI's call composition and trace tests provide a broader diagnostic (Figure 3; Table 7, Appendix F). Best-so-far trace agreement reaches 1.0 within about five calls on T0, T1 and T1b; every zero-completion first run on those tiers reaches full agreement at some point. Across all tiers, ten of the twelve zero-completion first runs reach full agreement; the exceptions are one layout (0.92) and one mutation (0.74). Full agreement is a best-so-far observation on a changing test set, not a property of the final model; the inspected logs, not the statistic, distinguish goal failures from errors on unseen transitions.

5.6 Failure taxonomy

We identify four recurring, potentially overlapping failure patterns. (i) *First-level budget exhaustion*: direct play reaches the action limit on most GLM-5.3 games and nearly all Qwen3.8-27B games. (ii) *Unfinished model construction*: Tycho spends its remaining calls on one level (e.g. 109 on permuted `cd82` level 2) until its $5\times$-baseline action cutoff or the call cap. (iii) *Trace agreement without progress* (Section 5.5). (iv) *Mode-switch failures*: `m0r0` level 3 stops each GLM configuration that

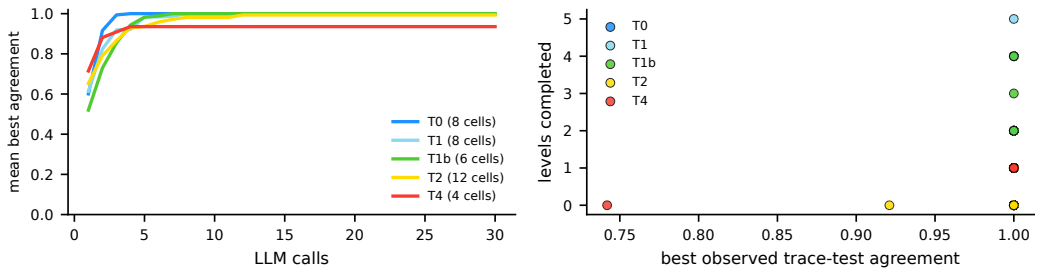


Figure 3: CbI with GLM-5.3, verification as a graded signal. Left: best-so-far agreement over trace tests, against LLM calls, averaged over first-run cells with available traces (cell counts in the legend). Tests use the transitions logged at that point; these sets change during play. Traces contain tests for 38 of 40 first runs, including all twelve zero-completion runs. Right: best observed agreement against levels completed per cell. Agreement of 1.0 with no completed level shows that passing an observed trace test is insufficient for success.

reaches it; only the GPT-6 configurations complete it. The local backbone, Qwen3.8-27B, completes two levels across 64 cells (Appendix I).

6 Discussion and Limitations

Interpretation of the transformations. T1 and T1b test sensitivity to colour and action labels; a drop may reflect learned conventions, exploration cost, budget limits or familiarity, and does not alone prove memorisation. These tiers are diagnostic interventions, not a matched scale of difficulty. Familiarity also differs across harnesses: CbI completes all five `m0r0` variants but 7 of the 13 variants of games it was not developed on, and Tycho 4 of 5 and 9 of 13; no evaluated game is held out for every harness. The variants are generated from public programs; this design neither establishes absence from backbone training data nor certifies an untouched development holdout.

Coverage and uncertainty. T2 and T4 cover only first levels for which bounded search finds solutions; click-only games are absent, and the T1 audit excludes `bp35`. Thirteen layouts span five games; ten Tycho mutation runs repeat five tasks. Lane coverage and repeats are uneven (usage caps, timeouts, restarts; Appendix M). We report no confidence intervals on completion: levels within a game are dependent and repeats are few. Completion-conditioned efficiency excludes failures, and the search-reference filter limits solution length, not agent difficulty.

7 Conclusion

Compiled Worlds pairs each transformed ARC-AGI-3 game with its original under the standard interface, and these comparisons show what completion alone hides. Much of the call gap between Tycho and CbI with GLM-5.3 is present on the original first levels; relative to its own originals, Tycho needs 3.3 times the calls on resampled layouts, where all of its first-level failures and five of CbI's nine occur (Table 2), whereas first-level rule mutations cost about as much as the originals and colour and action changes do not measurably change completion. Ten of CbI's twelve zero-completion first runs with GLM-5.3 matched every logged transition at some point: a verified transition model is not a solved game.

For evaluation practice, these results suggest reporting completion, the cost of reaching it, and that cost relative to the same agent on the original game; pairing makes the third measurable, which evaluation on unseen games alone cannot. Trace agreement is then a diagnostic rather than a success criterion, and trivial controls should count only levels the engine reports as completed, not game-over or error states (Appendix H). Beyond these eight games and first-level variants, the same generator supports harder tests: mutations that span later levels and click-only games, recombined mechanics, repeated runs, and the published backbones and budgets.

ETHICS STATEMENT

The study involves no human subjects and no personal data; human baselines are the per-level action counts distributed with the official games. Compiled Worlds modifies game programs obtained through the ARC-AGI-3 toolkit, whose Python packages are MIT-licensed. The generator, seeds, rule-mutation registry and certificates regenerate every variant from a locally installed copy of the games; the generated game files themselves are released only as far as the terms of the original game files permit. One evaluated harness, CbI, was developed by the authors. To limit this conflict of interest, CbI runs under the same matrix, isolation and reporting rules as the published harnesses, its failures are reported in the same detail, and its development games are disclosed (Section 4). All backbones were accessed through their providers' interfaces (Z.ai, DeepSeek, a Codex subscription for GPT-6) or run locally (Qwen3.8-27B).

AI USE STATEMENT

Generative AI tools were used in this work as follows. (i) Language-model coding assistants were used to implement the game generator, the certificate search, the cell runner and matrix driver, the patches to the published harnesses, the analysis scripts, and the figure and table generators; the T4 rule mutations were specified by the authors and their code was written with the same assistance. (ii) Language-model assistants took part in refining the tier protocol and the metrics, in interpreting the per-cell results, in searching and checking the literature, and in drafting and editing the text, title and keywords. (iii) Language models are also the *object* of study: GLM-5.3, Qwen3.8-27B, DeepSeek V4.1 Flash and GPT-6 act as the backbones of the evaluated systems, and their outputs are the measured data, not assistance to the authors. Generative AI tools were not used to alter experimental outcomes; every number in the tables is regenerated from the released run logs by the released scripts. The authors reviewed the AI-assisted code and text, re-ran the certificate and summary scripts whose outputs appear in the paper, and take responsibility for the final content of this work, including text, claims and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

The supplementary material contains the generator with its seeds and rule registry, which regenerate every per-version environment from the official game files; the certificates; the patches to the three published harnesses; the cell runner and matrix driver; the CbI harness; one record per run; and the Tycho event logs and CbI session logs from which every table and figure except Figure 1 is regenerated by the included scripts (Figure 1 renders the game files, which are not redistributed). Section 3 defines the tiers and Appendix K lists every rule mutation verbatim; Section 4 and Appendix L give the pinned commits, model configurations and the exact behavioural changes made to the external harnesses; Appendix M records the usage caps, timeouts and restart rules that shaped the runs; Appendix N lists every cell. Complete logs, including frames and model responses, will be released with the code.

REFERENCES

ARC Prize Foundation. ARC-AGI-3: A new challenge for frontier agentic intelligence. arXiv preprint arXiv:2603.24621, 2026a. URL https://arxiv.org/abs/2603.24621.

ARC Prize Foundation. ARC-AGI-3 benchmarking harness. https://github.com/arcprize/arc-agi-3-benchmarking, 2026b.

ARC Prize Foundation. arc-agi: ARC-AGI toolkit, version 0.9.9. https://pypi.org/project/arc-agi/, 2026c.

ARC Prize Foundation. OpenAI's GPT-6 Astra on ARC-AGI-3. https://arcprize.org/blog/astra, 2026d.

Terry Chen, Yeyin (Eva) Zhu, Zhifan Ye, Jean-Francois Puget, and Humphrey Shi. NVIDIA AVO reaches 100% on ARC-AGI-3, demonstrating a frontier-level general-purpose architecture for long-horizon autonomous agents. NVIDIA Technical Blog, 2026.

François Chollet. On the measure of intelligence. *arXiv preprint arXiv:1911.01547*, 2019.

François Chollet, Mike Knoop, Gregory Kamradt, Bryan Landers, and Henry Pinkard. ARC-AGI-2: A new challenge for frontier AI reasoning systems. *arXiv preprint arXiv:2505.11831*, 2025.

Karl Cobbe, Oleg Klimov, Chris Hesse, Taehoon Kim, and John Schulman. Quantifying generalization in reinforcement learning. In *International Conference on Machine Learning*, 2019.

Karl Cobbe, Chris Hesse, Jacob Hilton, and John Schulman. Leveraging procedural generation to benchmark reinforcement learning. In *International Conference on Machine Learning*, 2020.

David Courtis, Wenhao Li, and Scott Sanner. OPINE-World: Programmatic world modeling with ontology-error-prioritized interactive exploration for ARC-AGI-3. *arXiv preprint arXiv:2607.01531*, 2026.

DeepSeek-AI. DeepSeek-V4.1-Flash. Accessed through the DeepSeek API under the model name `deepseek-v4-flash`; `https://api-docs.deepseek.com/quick_start/pricing`, 2026.

Quentin Delfosse, Jannis Blüml, Bjarne Gregori, and Kristian Kersting. HackAtari: Atari learning environments for robust and continual reinforcement learning. *arXiv preprint arXiv:2406.03997*, 2024.

Jesse Farebrother, Marlos C. Machado, and Michael Bowling. Generalization and regularization in DQN. *arXiv preprint arXiv:1810.00123*, 2018.

Michael Hodel. Addressing the abstraction and reasoning corpus via procedural example generation. *arXiv preprint arXiv:2404.07353*, 2024.

INT21 AI. Pushing GPT-5.6-Sol from 13.3% to 100% on ARC-AGI-3 public. `https://int21.ai/insights/pushing-gpt-5-6-sol-to-100-on-arc-agi-3-public/`, 2026.

Ken Kansky, Tom Silver, David A. Mély, Mohamed Eldawy, Miguel Lázaro-Gredilla, Xinghua Lou, Nimrod Dorfman, Szymon Sidor, Scott Phoenix, and Dileep George. Schema networks: Zero-shot transfer with a generative causal model of intuitive physics. In *International Conference on Machine Learning*, 2017.

Robert Kirk, Amy Zhang, Edward Grefenstette, and Tim Rocktäschel. A survey of zero-shot generalisation in deep reinforcement learning. *Journal of Artificial Intelligence Research*, 76, 2023.

Jacky Kwok, Shulu Li, Pranav Atreya, Yuejiang Liu, Yixing Jiang, Chelsea Finn, Marco Pavone, Ion Stoica, and Azalia Mirhoseini. LLM-as-a-verifier: A general-purpose verification framework. *arXiv preprint arXiv:2607.05391*, 2026.

Jens Lehmann, Andrei Aioanei, and Sahar Vahdati. Tycho: Active abstraction with programmatic world models for ARC-AGI-3. *arXiv preprint arXiv:2607.28287*, 2026.

Keong Han Liew. Explore before you solve: The speed–depth trade-off in epistemic agents for ARC-AGI-3. *arXiv preprint arXiv:2605.25931*, 2026.

Marlos C. Machado, Marc G. Bellemare, Erik Talvitie, Joel Veness, Matthew Hausknecht, and Michael Bowling. Revisiting the arcade learning environment: Evaluation protocols and open problems for general agents. *Journal of Artificial Intelligence Research*, 61, 2018.

Seyed Iman Mirzadeh, Keivan Alizadeh, Hooman Shahrokhi, Oncel Tuzel, Samy Bengio, and Mehrdad Farajtabar. GSM-Symbolic: Understanding the limitations of mathematical reasoning in large language models. In *International Conference on Learning Representations*, 2025.

Arsenii Kirillovich Moskvichev, Victor Vikram Odouard, and Melanie Mitchell. The ConceptARC benchmark: Evaluating understanding and generalization in the ARC domain. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL `https://openreview.net/forum?id=8ykyGbtt2q`.

OpenAI. GPT-6 Astra. Model `gpt-6-astra`, accessed through the Codex CLI, 2026.

Qwen Team. Qwen3.8-27B. Open-weight model, https://huggingface.co/Qwen/Qwen3.8-27B; served locally with Ollama at 4-bit quantisation, 2026.

Sergey Rodionov. Executable world models for ARC-AGI-3 in the era of coding agents. *arXiv preprint arXiv:2605.05138*, 2026. Accepted at AGI-2026.

Oscar Sainz, Jon Campos, Iker García-Ferrero, Julen Etxaniz, Oier Lopez de Lacalle, and Eneko Agirre. NLP evaluation in trouble: On the need to measure LLM data contamination for each benchmark. In *Findings of the Association for Computational Linguistics: EMNLP*, 2023.

Alexy Skoutnev, Kirill Acharya, Gaston Longhitano, Madeleine Udell, Kevin Ellis, and Iddo Drori. Twin: Playing an unknown game with a test-time digital twin. *arXiv preprint arXiv:2608.14490*, 2026.

Zhaofeng Wu, Linlu Qiu, Alexis Ross, Ekin Akyürek, Boyuan Chen, Bailin Wang, Najoung Kim, Jacob Andreas, and Yoon Kim. Reasoning or reciting? Exploring the capabilities and limitations of language models through counterfactual tasks. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2024.

Z.ai. GLM-5.3. Model glm-5.3, accessed through the Z.ai API; https://docs.z.ai/guides/llm/glm-5.3, 2026.

Guanning Zeng, Jiani Wang, Wenjie Ma, Shaofeng Yin, Chenyang Wang, Shichen Liu, Angjoo Kanazawa, Wode Ni, Xiuyu Li, Andrea Zanette, and Haiwen Feng. Frontier models with our harness achieve ∼99% on ARC-AGI-3 public. Schema project page, https://schema-harness.github.io, 2026.

Chiyuan Zhang, Oriol Vinyals, Rémi Munos, and Samy Bengio. A study on overfitting in deep reinforcement learning. *arXiv preprint arXiv:1804.06893*, 2018.

Hugh Zhang, Jeff Da, Dean Lee, Vaughn Robinson, Catherine Wu, Will Song, Tiffany Zhao, Pranav Raja, Charlotte Zhuang, Dylan Slack, Qin Lyu, Sean Hendryx, Russell Kaplan, Michele Lunati, and Summer Yue. A careful examination of large language model performance on grade school arithmetic. In *Advances in Neural Information Processing Systems (Datasets and Benchmarks Track)*, 2024.

A READING AND REUSING THE EVALUATION

Units and denominators. The evaluation starts from eight original games. A transformed version retains a game's identity but changes a specified component. T0, T1 and T1b retain multiple levels; each T2 or T4 version contains one level. Repeated runs measure stochastic variation on the same version, not additional task diversity. The generator's coverage (24 colour-audited games, 16 arrow-controlled games and six games with accepted layouts) is broader than the evaluated matrix; the accepted `ls20` layout is not evaluated.

Table 4: Populations behind the headline counts. Different aggregations answer different questions; repeats do not enlarge the set of distinct tasks.

Quantity	Included observations	Interpretation
Layout completion: 8/13 each	First runs of Tycho and CbI with GLM-5.3 on 13 layouts from five games	Equal counts, six successes in common; a random policy also clears one of the six
Layout costs: 457 vs 39 calls	Those six shared first-run successes, up to completion (recorded totals: 457 vs 55)	Conditional comparison; omits failures
Adaptation premium, e.g. Tycho layouts 3.26	Variants completed in first runs, each against the same harness's first-run calls on the original first level	Within-harness ratio; one run per original
Paired T0→T1: 11/16/6	33 lane–game first-run pairs from seven scaffolded lanes	Sign test $p = 0.33$; pairs sharing a game are dependent
Full costs: 13 vs 12 completed	All 18 first-run layout and mutation versions per harness	Includes failures and stopping-rule effects
Mutation repeatability: 10/10	Two Tycho GLM-5.3 runs on each of five mutations	Five tasks, ten successful runs
Trace diagnostic: 10/12	Twelve zero-completion CbI GLM-5.3 first runs, all with trace tests	Full agreement reached at some point in ten runs
Tier table	Available runs, averaged within version, then summed	Version-weighted completion; e.g., Tycho T2 is 9/13 after repeats, versus 8/13 on first runs
Efficiency table	Completed-level observations pooled over runs	Repeated successes contribute repeatedly; failures are excluded

Reusing the protocol. A new evaluation should (1) record the game sources, generator settings, ordered seed rule, acceptance checks and rejected candidates; (2) record the harness revision, model configuration, observation format, budgets and retry rules before the comparison; (3) serve each accepted version in a separate environment directory and run the same declared configuration on its original and transformed versions; (4) retain every result, including failures, partial runs and repeats, under distinct run identifiers; and (5) report paired completion and resources, with successful-subset costs explicitly separated from totals including failures. Applying this procedure requires an adapter to the game interface, not a learned world model: direct-play and model-free agents are included in the present evaluation. Certificate search is performed by the evaluator, separately from the agent's model fitting.

Artifact map and interpretation. The released `worlds/paper_tables.py` aggregates `runs/cells/*/*/result.json`; `worlds/cost_table.py` adds per-call token records for CbI; `worlds/cbi_calls.py` and `worlds/fig_trace.py` read first-run session logs; `worlds/premium.py` computes calls, tokens and minutes up to completion and the premiums, and `worlds/paired_stats.py` the paired test. The trace statistic takes the maximum observed test agreement per run; two successful runs have no test events and are absent from the figure. A new user should preserve these distinctions when recomputing summaries. The protocol reports sensitivity and resource tradeoffs, without a binary test for memorisation, a calibrated pass threshold or a claim to replace private-set evaluation.

B MATCHED COSTS ON THE T2 AND T4 VERSIONS

Table 5: First runs of Tycho and CbI with GLM-5.3 on all thirteen re-sampled layouts and five rule mutations, successes and failures alike: levels completed, total recorded actions, LLM calls, tokens (Tycho: input including cache reads plus output; CbI: prompt plus completion) and wall minutes. Calls and tokens are not interchangeable across harnesses (Tycho makes many short calls, CbI few long ones), so calls, tokens and wall time are reported separately. These are recorded totals: Tycho stops at a win, whereas CbI did not register the level count of these single-level versions and kept acting, and on five successes calling, after the win (22 of 86 recorded calls on its twelve successes); CbI's recorded requests also include retried empty answers (Appendix M). The main text counts answered calls, tokens and minutes up to completion.

		Tycho					CbI				
Tier	Version	lvl	act.	calls	Mtok	min	lvl	act.	calls	Mtok	min
T2	m0r0-t2l0s007	0	109	138	6.1	46	1	66	3	0.2	62
T2	m0r0-t2l0s008	1	12	53	2.2	46	1	141	14	0.7	142
T2	m0r0-t2l0s012	1	25	104	2.2	71	1	72	6	0.3	62
T2	cd82-t2l0s001	0	134	300	26.6	166	1	86	14	1.1	282
T2	cd82-t2l0s002	0	92	300	19.5	106	0	72	18	1.6	300
T2	cd82-t2l0s003	0	77	300	75.7	195	0	110	19	1.8	300
T2	cn04-t2l0s001	1	47	133	5.2	109	0	39	8	0.6	117
T2	cn04-t2l0s002	1	16	27	1.3	10	0	79	17	1.0	300
T2	cn04-t2l0s003	0	120	300	18.7	111	0	72	17	1.3	300
T2	lp85-t2l0s002	1	23	43	1.6	17	1	32	5	0.2	28
T2	lp85-t2l0s006	1	9	41	2.1	33	1	67	16	1.4	268
T2	lp85-t2l0s009	1	20	84	7.3	118	1	46	6	0.3	63
T2	ft09-t2l0s012	1	35	132	14.0	148	1	47	8	0.4	82
T4	m0r0-t4mirror	1	27	41	2.7	15	1	61	2	0.1	46
T4	m0r0-t4same	1	25	52	1.5	30	1	51	1	0.1	19
T4	cd82-t4wrap	1	31	126	4.3	50	0	45	12	0.5	300
T4	cn04-t4rotccw	1	20	31	1.8	16	1	40	6	0.4	112
T4	lp85-t4flip	1	9	30	1.2	15	1	31	5	0.2	48
all	(18 versions)	13	831	2235	194.0	1302	12	1157	177	12.2	2832

C EFFICIENCY OF COMPLETED LEVELS

Table 6: Efficiency of completed levels: RHAE-style $\min(1.15, (\text{human}/\text{agent})^2)$ on T0/T1/T1b, ROAE $\min(1, \text{reference}/\text{agent})^2$ on T2/T4, with agent actions counted up to completion; mean over completed levels, pooling first runs and repeats (count in parentheses). First-run means quoted in the text differ slightly (Tycho T4 0.38; CbI T4 0.17).

System	Backbone	T0	T1	T1b	T2	T4
Plain harness	GLM-5.3	0.99 (6L)	1.02 (3L)	–	–	0.04 (1L)
Tycho	GLM-5.3	1.08 (24L)	1.00 (25L)	1.02 (13L)	0.23 (13L)	0.35 (10L)
Twin	GLM-5.3	0.92 (10L)	0.86 (12L)	–	–	–
CbI	GLM-5.3	0.65 (16L)	0.76 (24L)	0.67 (10L)	0.05 (9L)	0.17 (4L)
Twin	GPT-6 medium	1.15 (22L)	1.15 (17L)	–	0.36 (4L)	0.62 (5L)
CbI	GPT-6 medium	0.91 (21L)	0.87 (20L)	–	0.14 (5L)	0.25 (5L)
Plain harness	DeepSeek V4.1 Flash	1.00 (1L)	1.12 (1L)	–	–	–
Tycho	DeepSeek V4.1 Flash	1.04 (7L)	1.04 (7L)	–	0.35 (2L)	0.28 (5L)
CbI	DeepSeek V4.1 Flash	0.64 (4L)	1.15 (1L)	–	–	–
Plain harness	Qwen3.8-27B	–	–	1.15 (1L)	–	–
CbI	Qwen3.8-27B	–	–	–	0.02 (1L)	–
Trivial (best of 11)	none	–	–	–	0.16 (1L)	–

D COMPLETION FRACTIONS

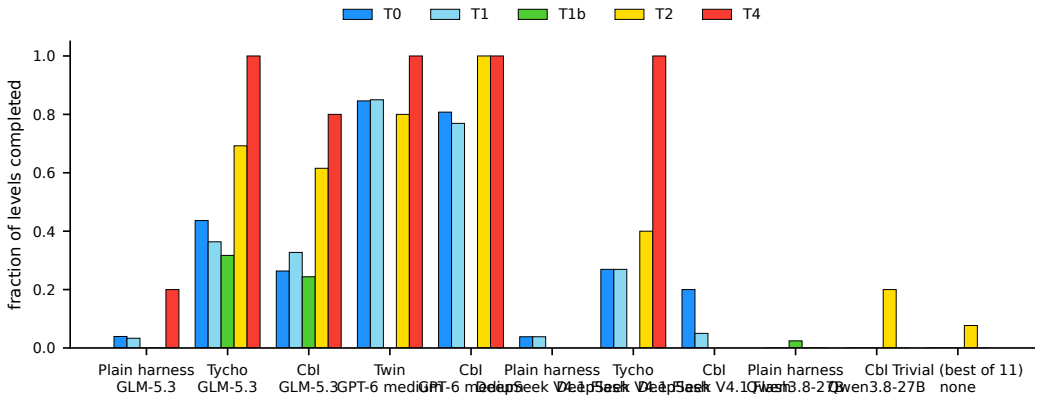


Figure 4: Fraction of available levels completed per lane and tier. T2 and T4 are single-level games certified solvable by search; coverage differs by lane and tier (Table 3); T2 covers thirteen layouts for Tycho, CbI with GLM-5.3 and the trivial controls, and five elsewhere.

E CALLS PER LEVEL UNDER THE COLOUR BIJECTION

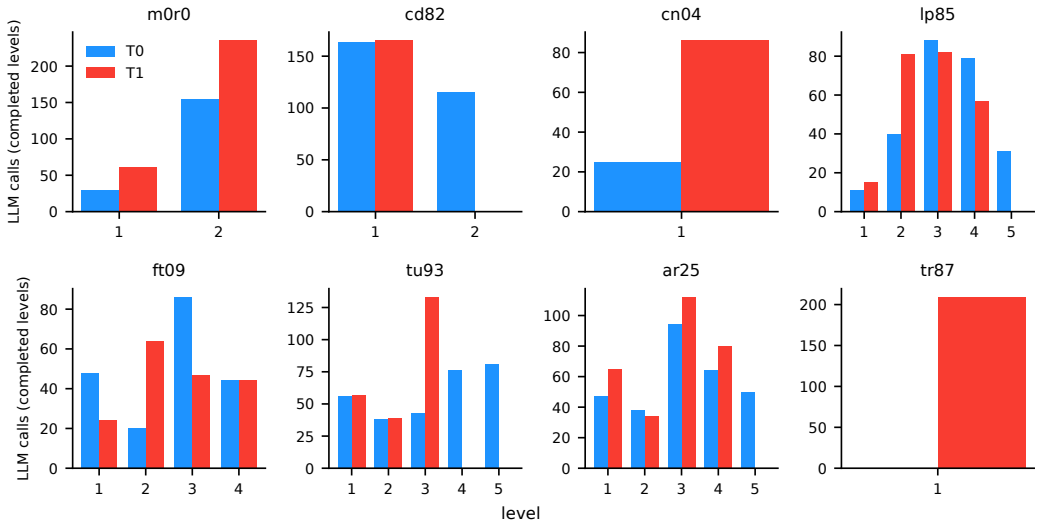


Figure 5: Tycho with GLM-5.3: LLM calls spent on each completed level, original game (T0, blue) and colour-permuted version (T1, red). A missing red bar is a level not completed on the permuted version within the 300-call budget.

F CbI call composition

Table 7: CbI with GLM-5.3, first runs: mean calls per cell by role, plans executed, and plans whose goal predicate was reached, per tier.

Tier	cells	levels	compile	repair	goal	probe	plans run	reached predicate
T0	8	1.9	3.1	11.1	5.2	0.8	11.2	6.1
T1	8	2.1	2.8	11.5	4.5	0.2	14.5	10.2
T1b	6	1.7	2.5	8.8	2.8	0.8	10.5	6.0
T2	13	0.6	1.9	7.8	0.8	0.1	3.0	1.7
T4	5	0.8	1.4	2.4	0.2	0.2	1.0	0.2

G Paired completion under colour permutation

Table 8: First runs only, per scaffolded lane: games with both a T0 and a T1 cell, levels completed on each, and the number of games with fewer, equal and more levels under permutation. Produced by `worlds/paired_stats.py` from the same records as Table 3.

System	Backbone	games	T0	T1	fewer	equal	more
Tycho	GLM-5.3	8	24	20	4	3	1
CbI	GLM-5.3	8	15	17	2	4	2
Twin	GLM-5.3	3	10	12	0	2	1
Tycho	DeepSeek V4.1 Flash	4	7	7	1	2	1
CbI	DeepSeek V4.1 Flash	3	4	1	2	1	0
Twin	GPT-6 medium	3	18	17	1	2	0
CbI	GPT-6 medium	4	21	20	1	2	1
All scaffolded lanes		33	99	94	11	16	6

Run-to-run variation is taken from every multi-level version with at least two runs (twelve versions: Tycho and CbI with GLM-5.3 and the plain harness): the first two runs differ by 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1 and 2 levels.

H Trivial strategies reported by Liew (2026)

Liew (2026) reports that every public game is reachable by a non-intelligent strategy: a single blind action, a single action repeated 50–200 times, or ACTION6 with null coordinates, which `arc-agi` 0.9.8 returned as a win there. We test these strategies with `arc-agi` 0.9.9, the version used for every run, and count a level only when the engine reports `levels_completed` > 0 (Table 9; `worlds/aera_probe.py`). None completes a level on any evaluated version. The step counts reported for the repeated actions are the steps at which the game ends in the game-over state, and the null-coordinate click raises an exception inside the engine, for which the toolkit returns no frame. The harness parsers count levels from the engine's `levels_completed`, so a step that raises cannot add one; CbI refuses a click without integer coordinates; and no run log records a step error.

Table 9: Model-free checks of the strategies in Liew (2026) with `arc-agi` 0.9.9. The eight repeated-action strategies are `tu93` ACTION1×50, `re86` ACTION1×100, `tr87` ACTION1×128, `ka59` ACTION6 at (32,32)×100, `ls20` ACTION2×129, `sc25` ACTION6 at (24,48)×52, `g50t` ACTION1×130 and `wa30` ACTION1×200.

Strategy	Versions	Result
Every single first action: ACTION1–5, ACTION7, and ACTION6 at each of the 64×64 cells	40 evaluated	no level completed
Each single action repeated for 300 steps: ACTION1–5, ACTION7, and ACTION6 at (32,32), (24,48) and (0,0)	40 evaluated	no level completed
The eight repeated-action strategies reported there	8 public games	game over at exactly the reported step; no level completed
ACTION6 with null coordinates	25 public games	engine error on 18, for which no frame is returned; no effect on 7
ACTION6 without coordinates	40 evaluated	engine error on the `m0r0` and `cn04` versions

I Limits of the evaluated local backbone

Qwen3.8-27B (4-bit, one RTX 3090) completes two levels across 64 cells: direct play clears `cd82` level 1 under action relabelling in 11 actions (search reference 5), and CbI clears the resampled first level of `lp85` in 33 actions (reference 5). Neither success is reproduced by a trivial policy, and neither system clears the corresponding original level. Both configurations sit near the floor: the CbI lane spends about 23 GPU hours and 1,789 calls for one level, under a 24.5k-token request budget. Trace agreement alone does not isolate whether the bottleneck is goal inference, planning or unseen dynamics. We did not evaluate larger local models or other quantisations.

J Calibration on the reference game

Table 10: T0 calibration on `m0r0` with GLM-5.3 (six levels; human medians 30, 111, 202, ...). Actions are per level.

System	Levels	Actions per level	LLM calls	Outcome
Plain harness	0/6	45 (cap)	45	level-1 budget exhausted; identical on T1
Twin	2/6	21, 49, 177+	–	stalled on the mode-switch level; killed by the usage cap
Tycho	2/6	15, 84, 56+	300	call cap on level 3; 10.4M input tokens, 3.2 h
CbI	2/6	58, 175, 174+	42	5-hour cap on the mode-switch level

K Rule mutations (T4), verbatim

Table 11 lists the generated T2 versions and the accepted and rejected T4 candidates. The registry entries follow verbatim (identifiers are the obfuscated names in the official sources; lines folded at 92 characters are marked with a trailing backslash). Baselines of the reference games (actions per level, human median): `ar25` 32,50,75,37,89,159,233,73; `cd82` 55,8,41,21,23,23; `cn04` 29,54,85,300,208,113; `ft09` 43,12,23,28,65,37; `lp85` 17,38,31,16,41,60,26,159; `m0r0` 30,111,203,26,500,237; `tr87` 54,58,40,45,71,146; `tu93` 19,16,34,42,123,80,14,23,111.

Table 11: Search outcomes for generated versions: reference lengths for the generated and original levels, scaled human baseline, and acceptance status. A solution witness does not establish global optimality. The one accepted `ls20` layout was not evaluated.

Tier	Game	Version	Reference	Original reference	Baseline	Note
T2	cd82	t2l0s001	5	5	55	3 sprites moved
T2	cd82	t2l0s002	5	5	55	2 sprites moved
T2	cd82	t2l0s003	5	5	55	4 sprites moved
T2	cn04	t2l0s001	16	14	33	2 sprites moved
T2	cn04	t2l0s002	17	14	35	1 sprites moved
T2	cn04	t2l0s003	21	14	44	1 sprites moved
T2	ft09	t2l0s012	4	4	43	7 sprites moved
T2	lp85	t2l0s002	5	5	17	4 sprites moved
T2	lp85	t2l0s006	5	5	17	5 sprites moved
T2	lp85	t2l0s009	5	5	17	4 sprites moved
T2	ls20	t2l0s005	13	13	22	4 sprites moved
T2	m0r0	t2l0s007	11	15	22	1 sprites moved
T2	m0r0	t2l0s008	8	15	16	2 sprites moved
T2	m0r0	t2l0s012	8	15	16	2 sprites moved
T4	ar25	t4jump	–	–	–	no certificate
T4	cd82	t4wrap	2	5	22	accepted
T4	cn04	t4rotccw	12	14	25	accepted
T4	ft09	t4back	4	4	–	no divergence
T4	ft09	t4far	4	4	–	no divergence
T4	lp85	t4flip	5	5	17	accepted
T4	m0r0	t4mirror	21	15	42	accepted
T4	m0r0	t4same	20	15	40	accepted
T4	tu93	t4any	18	18	–	no divergence
T4	tu93	t4invert	–	18	–	no certificate
T4	tu93	t4jump	–	18	–	no certificate

ar25 `t4jump` (rejected: no certificate). the selected piece moves two cells per arrow instead of one; original: one cell

```
--- original
gndsqmkhxkd = -1
+++ mutated
gndsqmkhxkd = -2

--- original
gndsqmkhxkd = 1
+++ mutated
gndsqmkhxkd = 2

--- original
qlqwojsdxcs = -1
+++ mutated
qlqwojsdxcs = -2

--- original
qlqwojsdxcs = 1
+++ mutated
qlqwojsdxcs = 2
```

cd82 `t4wrap` (accepted). selection wraps around the 3x3 position grid (modulo 3) instead of clamping at the edges; original: arrows clamp at the border

```
--- original
if geaclnfpnn == 1:
    wdtucaweg, tivucwuqw = (max(0, dfsddohkn - 1), miipdppxi)
elif geaclnfpnn == 2:
    wdtucaweg, tivucwuqw = (min(2, dfsddohkn + 1), miipdppxi)
elif geaclnfpnn == 3:
    wdtucaweg, tivucwuqw = (dfsddohkn, max(0, miipdppxi - 1))
elif geaclnfpnn == 4:
    wdtucaweg, tivucwuqw = (dfsddohkn, min(2, miipdppxi + 1))
+++ mutated
```

```
if geaclnfpnn == 1:
    wdtucaweg, tivucwuqw = ((dfsddohkn - 1) % 3, miipdppxi)
elif geaclnfpnn == 2:
    wdtucaweg, tivucwuqw = ((dfsddohkn + 1) % 3, miipdppxi)
elif geaclnfpnn == 3:
    wdtucaweg, tivucwuqw = (dfsddohkn, (miipdppxi - 1) % 3)
elif geaclnfpnn == 4:
    wdtucaweg, tivucwuqw = (dfsddohkn, (miipdppxi + 1) % 3)
```

cn04 `t4rotccw` (accepted). ACTION5 rotates the selected piece COUNTER-clockwise (original: clockwise 90 degrees)

```
--- original
self.xseexqzst.rotate(90)
+++ mutated
self.xseexqzst.rotate(-90)
```

ft09 `t4back` (rejected: no divergence). a click cycles the affected cells BACKWARDS through the colour palette; original: forwards

```
--- original
kNa = (kNa + 1) % len(self.gqb)
+++ mutated
kNa = (kNa - 1) % len(self.gqb)
```

ft09 `t4far` (rejected: no divergence). the click pattern acts on cells 8 apart instead of 4 (the pattern's neighbourhood is stretched to twice the spacing); original: spacing 4

```
--- original
cAw = (self.blr.x + (ybc * 4), self.blr.y + (lga * 4))
+++ mutated
cAw = (self.blr.x + (ybc * 8), self.blr.y + (lga * 8))
```

lp85 `t4flip` (accepted). button direction inverted: buttons tagged R now shift their line the other way (original: R means rightward)

```
--- original
kypahuoyom = True if weskkxahis == "R" else False
+++ mutated
kypahuoyom = False if weskkxahis == "R" else True
```

m0r0 `t4mirror` (accepted). mirror axes swapped: the right-hand twin now mirrors VERTICAL motion (dx, -dy) and the lower twin mirrors HORIZONTAL motion (-dx, dy); original: right mirrors horizontal, lower mirrors vertical

```
--- original
elif "toljda-rivmdg" in dtktmyjjtsa:
    urouxrqoiz.append((ssebydbziot, -tbbgvkvoptv, rjcrgldrrqz))
elif "boweok-leklkn" in dtktmyjjtsa:
    urouxrqoiz.append((ssebydbziot, tbbgvkvoptv, -rjcrgldrrqz))
+++ mutated
elif "toljda-rivmdg" in dtktmyjjtsa:
    urouxrqoiz.append((ssebydbziot, tbbgvkvoptv, -rjcrgldrrqz))
elif "boweok-leklkn" in dtktmyjjtsa:
    urouxrqoiz.append((ssebydbziot, -tbbgvkvoptv, rjcrgldrrqz))
```

m0r0 `t4same` (accepted). no mirroring: all twins move in the SAME direction as the pressed arrow (original: partners mirror one axis)

```
--- original
elif "toljda-rivmdg" in dtktmyjjtsa:
    urouxrqoiz.append((ssebydbziot, -tbbgvkvoptv, rjcrgldrrqz))
elif "boweok-leklkn" in dtktmyjjtsa:
    urouxrqoiz.append((ssebydbziot, tbbgvkvoptv, -rjcrgldrrqz))
elif "boweok-rivmdg" in dtktmyjjtsa:
    urouxrqoiz.append((ssebydbziot, -tbbgvkvoptv, -rjcrgldrrqz))
+++ mutated
elif "toljda-rivmdg" in dtktmyjjtsa:
    urouxrqoiz.append((ssebydbziot, tbbgvkvoptv, rjcrgldrrqz))
elif "boweok-leklkn" in dtktmyjjtsa:
    urouxrqoiz.append((ssebydbziot, tbbgvkvoptv, rjcrgldrrqz))
elif "boweok-rivmdg" in dtktmyjjtsa:
    urouxrqoiz.append((ssebydbziot, tbbgvkvoptv, rjcrgldrrqz))
```

tu93 `t4any` (rejected: no divergence). win when ANY agent stands on an exit (original: ALL agents must stand on exits)

```
--- original
if ttauyadveo and all([any([libujymozt.x == exit.x and libujymozt.y == exit.y for exit in \
        onherbnfxo]) for libujymozt in ttauyadveo]):
+++ mutated
if ttauyadveo and any([any([libujymozt.x == exit.x and libujymozt.y == exit.y for exit in \
        onherbnfxo]) for libujymozt in ttauyadveo]):
```

tu93 `t4invert` (rejected: no certificate). maze passability inverted: cells the mask marks as floor (2) are now blocked and all other cells are walkable; original: only mask==2 cells are walkable

```
--- original
i = jpbsiftilway - hwthhtvyki
                cjjzphgqvu = gdcqnkdomzf
                if i >= 0 and cjjzphgqvu >= 0 and (i < height) and (cjjzphgqvu < width) and \
        (ymsjtadvsk.pixels[i, cjjzphgqvu] == 2):
+++ mutated
i = jpbsiftilway - hwthhtvyki
                cjjzphgqvu = gdcqnkdomzf
                if i >= 0 and cjjzphgqvu >= 0 and (i < height) and (cjjzphgqvu < width) and \
        (ymsjtadvsk.pixels[i, cjjzphgqvu] != 2):

--- original
i = jpbsiftilway + hwthhtvyki
                cjjzphgqvu = gdcqnkdomzf
                if i >= 0 and cjjzphgqvu >= 0 and (i < height) and (cjjzphgqvu < width) and \
        (ymsjtadvsk.pixels[i, cjjzphgqvu] == 2):
+++ mutated
i = jpbsiftilway + hwthhtvyki
                cjjzphgqvu = gdcqnkdomzf
                if i >= 0 and cjjzphgqvu >= 0 and (i < height) and (cjjzphgqvu < width) and \
        (ymsjtadvsk.pixels[i, cjjzphgqvu] != 2):

--- original
i = jpbsiftilway
                cjjzphgqvu = gdcqnkdomzf - hwthhtvyki
                if i >= 0 and cjjzphgqvu >= 0 and (i < height) and (cjjzphgqvu < width) and \
        (ymsjtadvsk.pixels[i, cjjzphgqvu] == 2):
+++ mutated
i = jpbsiftilway
                cjjzphgqvu = gdcqnkdomzf - hwthhtvyki
                if i >= 0 and cjjzphgqvu >= 0 and (i < height) and (cjjzphgqvu < width) and \
        (ymsjtadvsk.pixels[i, cjjzphgqvu] != 2):

--- original
i = jpbsiftilway
                cjjzphgqvu = gdcqnkdomzf + hwthhtvyki
                if i >= 0 and cjjzphgqvu >= 0 and (i < height) and (cjjzphgqvu < width) and \
        (ymsjtadvsk.pixels[i, cjjzphgqvu] == 2):
+++ mutated
i = jpbsiftilway
                cjjzphgqvu = gdcqnkdomzf + hwthhtvyki
                if i >= 0 and cjjzphgqvu >= 0 and (i < height) and (cjjzphgqvu < width) and \
        (ymsjtadvsk.pixels[i, cjjzphgqvu] != 2):
```

tu93 `t4jump` (rejected: no certificate). the player moves TWO grid steps per arrow instead of one (the landing cell must still be walkable; the cell jumped over is not checked); original: one step

```
--- original
i = jpbsiftilway - hwthhtvyki
+++ mutated
i = jpbsiftilway - 2 * hwthhtvyki

--- original
i = jpbsiftilway + hwthhtvyki
+++ mutated
i = jpbsiftilway + 2 * hwthhtvyki

--- original
cjjzphgqvu = gdcqnkdomzf - hwthhtvyki
+++ mutated
cjjzphgqvu = gdcqnkdomzf - 2 * hwthhtvyki

--- original
cjjzphgqvu = gdcqnkdomzf + hwthhtvyki
+++ mutated
cjjzphgqvu = gdcqnkdomzf + 2 * hwthhtvyki
```

L PATCHES TO THE PUBLISHED HARNESSES

The external repositories are pinned to Tycho `f68912a`, Twin `b9b937c` and the ARC Prize benchmarking harness `1aa78da`, with the following diffs (released with the code): Tycho gains `TYCHO_TEXT_ONLY` (image message parts are replaced by a text note, required because the endpoint accepts only text parts) and `TYCHO_ENV_DIR` (environment-directory override); Twin gains `ARC_CODEX_EXTRA_READ` (an extra readable path for the sandboxed CLI); the benchmarking harness gains three model configurations (GLM-5.3 high; DeepSeek V4.1 Flash high; Qwen3.8-27B via a local OpenAI-compatible server) and one behavioural change: after an empty or timed-out answer, a retry steps the reasoning effort down (high, medium, low) instead of repeating the request, because GLM-5.3 at high effort can spend its whole output budget reasoning and return nothing (the harness otherwise aborts the game after four empty answers). CbI applies the same policy. No prompt, policy or budget logic is changed otherwise.

M OPERATIONAL FINDINGS

The following constraints affected runtime, coverage and restart policy.

1. **Usage cap.** The coding endpoint enforces a 5-hour usage window; when it is reached every request fails with a message that states the reset time. Twin's calibration run was killed by it after 250 actions. The matrix runner recognises the message, rolls the affected cell back, waits for the stated reset and retries; Tycho's in-process retry budget is raised to six hours; CbI's backend waits without charging the call budget.
2. **Token cost of whole-thread tool calls.** Twin re-sends the conversation on every tool call, so one game consumed most of a usage window; Twin therefore runs on a subset.
3. **Text-only message parts.** The OpenAI-compatible endpoint rejects image parts (HTTP 400); Tycho is run with vision disabled and a patch that replaces image parts by a text note.
4. **Version resolution.** The toolkit's remote wrapper resolves a game id to the newest version of its prefix, so a shared server would have served the original game when a generated version was requested. Every plain-harness cell gets a dedicated server over a one-version directory; the other systems get a one-version environment directory.
5. **Reasoning length.** At high effort GLM-5.3 spends 30–80k output tokens on a compile request (15–21 minutes) and 16–65k on goal or repair requests. Requests with a 32k output limit returned empty answers (the whole budget went to reasoning); non-streamed responses were closed by the server after about 15 minutes. The backend streams with a 131k output limit.
6. **Local backbone.** Ollama serves a request without an explicit context at a small default (the harness prompts are about 15k tokens), and two lanes using different context sizes reload the model between requests and abort generations mid-stream (17 of 22 answers empty). The local lanes run one at a time with one model variant (49k context, fully on the GPU), and CbI runs Qwen3.8 with thinking disabled: with thinking enabled, most answers to our goal prompts were truncated (no answer after a few thousand characters of thinking), while the same prompts without thinking answered in about 80 s. The server also serves each request with 24.5k tokens of context, and a digit-grid frame costs about 4k tokens, so for the local lane CbI renders full frames run-length encoded and includes at most three transition diffs per prompt (the GLM lanes use twelve); this is the one prompt-level difference between backbones.
7. **Session resume.** A local-engine session can only be resumed by replaying its logged actions; a compiled model that raises an exception during goal evaluation or transition prediction must be routed to repair rather than allowed to end the run; it did twice before the loop caught every model exception.
8. **Reasoning exhaustion.** Late in a game, goal prompts of about 20k tokens made GLM-5.3 at high effort reason until the 131k output budget was spent and return nothing (26 minutes per attempt). The backend falls back to medium and then low effort when an answer is empty for that reason, and logs the effort of every call; the local backend retries once with thinking disabled.

9. **Loop limits.** CbI's default cap of 12 planning cycles per process ended the calibration at 30 of 60 calls; matrix cells lift it so that the call budget, the wall cap and two loop rules (stop after 60 tried goal predicates on one level; stop after three consecutive compile answers without code) are the only limits. Qwen3.8 often proposes many goal predicates per call, few of which are plannable; the goal-predicate limit can therefore end a run before the call budget is exhausted. The CbI T0 cell for `m0r0` is the calibration session resumed under the cell's limits, so its wall time spans several restarts.
10. **Codex plan window.** Twin driven by GPT-6 through a Codex subscription played 83 actions of `lp85` (seven levels) in 23 minutes and then hit the plan's 5-hour usage limit; one Twin game consumes a window. Retries are seeded with the previous window's world model (Twin's `--memory-from`) and capped at three windows per cell, after which the best attempt is kept as a partial record. The plan also has a weekly limit, which Twin and CbI exhausted in two days; the GPT-6 rows therefore cover a handful of games.
11. **Single-level stopping and request counts (CbI).** CbI's session did not register the level count of single-level versions, so after a win it kept acting and, on five of its twelve layout and mutation successes with GLM-5.3, calling the model: 22 of the 86 requests recorded on those successes came after the win. Recorded requests also include retried empty answers and, with GPT-6, requests refused at the Codex usage limit (14 of the 15 recorded on the `m0r0` layout, which one answered call solved). Main-text costs therefore count answered calls, tokens and minutes up to completion; per-cell and cost tables keep the recorded totals.
12. **Per-action cost of direct play.** With GLM-5.3 at high effort the plain harness takes about two minutes per action (123 s mean on `m0r0`), so a game whose level-1 budget is exhausted without a level-up costs $1.5\times$ baseline $\times$ 2 min; Qwen3.8-27B on the RTX 3090 takes 157 s per action when it has the GPU to itself.

N PER-CELL RESULTS

Plain harness, GLM-5.3.

Version	Tier	Levels	Actions per level	Calls	Wall (min)	Status
ar25-0c556536	T0	0/8	48	–	61	done
ar25-t1r001	T1	1/8	22,76	–	202	done
ar25-t1a001	T1b	0/8	48	–	43	done
cd82-fb555c5d	T0	0/6	83	–	231	done
cd82-t1r001	T1	0/6	83	–	132	done
cd82-t1a001	T1b	0/6	83	–	267	done
cd82-t2l0s001	T2	0/1	83	–	236	done
cd82-t4wrap	T4	1/1	10,1	–	30	done
cn04-2fe56bfb	T0	0/6	44	–	58	done
cn04-2fe56bfb (run 2)	T0	1/6	33,82	–	263	done
cn04-t1r001	T1	1/6	33,82	–	196	done
cn04-t1r001 (run 2)	T1	0/6	44	–	48	done
cn04-t1a001	T1b	0/6	44	–	84	done
cn04-t2l0s001	T2	0/1	50	–	69	done
cn04-t4rotccw	T4	0/1	38	–	134	done
ft09-0d8bbf25	T0	0/6	65	–	121	done
ft09-t1r001	T1	0/6	58	–	300	partial
ft09-t2l0s012	T2	0/1	65	–	145	done
lp85-305b61c3	T0	2/8	7,10,48	–	257	done
lp85-305b61c3 (run 2)	T0	2/8	6,13,48	–	246	done
lp85-305b61c3 (run 3)	T0	1/8	22,58	–	201	done
lp85-t1r001	T1	0/8	26	–	26	done
lp85-t1r001 (run 2)	T1	1/8	13,58	–	260	done
lp85-t1r001 (run 3)	T1	0/8	26	–	22	done
lp85-t2l0s002	T2	0/1	26	–	36	done
lp85-t4flip	T4	0/1	26	–	124	done
m0r0-492f87ba	T0	0/6	45	–	–	done
m0r0-492f87ba (run 2)	T0	0/6	45	–	14	done
m0r0-t1r001	T1	0/6	45	–	92	done
m0r0-t1r001 (run 2)	T1	0/6	45	–	77	done
m0r0-t1a001	T1b	0/6	45	–	21	done
m0r0-t2l0s007	T2	0/1	33	–	46	done
m0r0-t4mirror	T4	0/1	63	–	34	done
m0r0-t4same	T4	0/1	60	–	44	done
tr87-cd924810	T0	0/6	81	–	82	done
tr87-t1r001	T1	0/6	81	–	84	done
tr87-t1a001	T1b	0/6	81	–	141	done
tu93-0768757b	T0	0/9	18	–	300	partial
tu93-t1r001	T1	0/9	27	–	300	partial
tu93-t1a001	T1b	0/9	29	–	191	done

Tycho, GLM-5.3.

Version	Tier	Levels	Actions per level	Calls	Wall (min)	Status
ar25-0c556536	T0	5/8	22,19,57,23,28,1	300	259	done
ar25-t1r001	T1	4/8	28,17,91,21,1	300	164	done
ar25-t1a001	T1b	4/8	21,19,43,40,1	300	172	done
cd82-fb555c5d	T0	2/6	67,23,1	300	194	done
cd82-t1r001	T1	1/6	60,39,0	274	184	done
cd82-t1r001 (run 2)	T1	2/6	21,21,41	300	190	done
cd82-t1a001	T1b	2/6	30,15,20	300	234	done
cd82-t2l0s001	T2	0/1	134	300	166	done
cd82-t2l0s001 (run 2)	T2	1/1	78,0	157	94	done
cd82-t2l0s002	T2	0/1	92	300	106	done
cd82-t2l0s003	T2	0/1	77	300	195	done
cd82-t4wrap	T4	1/1	31,0	126	50	done
cd82-t4wrap (run 2)	T4	1/1	7,0	19	11	done
cn04-2fe56bfb	T0	1/6	15,76	300	181	done
cn04-t1r001	T1	1/6	21,64	300	260	done
cn04-t1a001	T1b	1/6	26,40	300	226	done
cn04-t2l0s001	T2	1/1	47,0	133	109	done
cn04-t2l0s001 (run 2)	T2	1/1	84,0	197	98	done
cn04-t2l0s002	T2	1/1	16,0	27	10	done
cn04-t2l0s003	T2	0/1	120	300	111	done
cn04-t4rotccw	T4	1/1	20,0	31	16	done
cn04-t4rotccw (run 2)	T4	1/1	11,0	29	16	done
ft09-0d8bbf25	T0	4/6	12,6,19,20,23	300	210	done
ft09-t1r001	T1	4/6	10,6,13,20,88	300	138	done
ft09-t2l0s012	T2	1/1	35,0	132	148	done
ft09-t2l0s012 (run 2)	T2	1/1	78,0	162	158	done
lp85-305b61c3	T0	5/8	7,15,23,15,10,3	300	257	done
lp85-t1r001	T1	4/8	8,19,30,13,1	300	161	done
lp85-t1r001 (run 2)	T1	3/8	11,20,96,4	300	177	done
lp85-t2l0s002	T2	1/1	23,0	43	17	done
lp85-t2l0s002 (run 2)	T2	1/1	12,0	30	63	done
lp85-t2l0s006	T2	1/1	9,0	41	33	done
lp85-t2l0s009	T2	1/1	20,0	84	118	done
lp85-t4flip	T4	1/1	9,0	30	15	done
lp85-t4flip (run 2)	T4	1/1	12,0	41	19	done
m0r0-492f87ba	T0	2/6	15,84,56	300	0	done
m0r0-t1r001	T1	2/6	20,115,1	300	96	done
m0r0-t1a001	T1b	2/6	26,45,50	277	300	done
m0r0-t2l0s007	T2	0/1	109,0	138	46	done
m0r0-t2l0s007 (run 2)	T2	1/1	14,0	62	33	done
m0r0-t2l0s008	T2	1/1	12,0	53	46	done
m0r0-t2l0s012	T2	1/1	25,0	104	71	done
m0r0-t4mirror	T4	1/1	27,0	41	15	done
m0r0-t4mirror (run 2)	T4	1/1	59,0	149	49	done
m0r0-t4same	T4	1/1	25,0	52	30	done
m0r0-t4same (run 2)	T4	1/1	41,0	133	53	done
tr87-cd924810	T0	0/6	112	242	109	partial
tr87-t1r001	T1	1/6	62,20	300	172	done
tr87-t1a001	T1b	0/6	142	300	95	done
tu93-0768757b	T0	5/9	17,16,19,17,36,1	300	199	done
tu93-t1r001	T1	3/9	19,14,23,12	300	300	done
tu93-t1a001	T1b	4/9	23,14,26,21,6	300	156	done

Twin, GLM-5.3.

Version	Tier	Levels	Actions per level	Calls	Wall (min)	Status
cd82-fb555c5d	T0	0/6	583	–	300	partial
cd82-t1r001	T1	2/6	57,33,41	–	158	partial
lp85-305b61c3	T0	8/8	7,177,34,22,44,36,9,13	–	230	done
lp85-t1r001	T1	8/8	7,11,16,16,21,21,9,62	–	300	done
m0r0-492f87ba	T0	2/6	22,50,177	–	–	done
m0r0-t1r001	T1	2/6	111,264,617	–	300	partial

CbI, GLM-5.3.

Version	Tier	Levels	Actions per level	Calls	Wall (min)	Status
ar25-0c556536	T0	2/8	41,46,267	19	300	done
ar25-t1r001	T1	2/8	82,21,26	22	300	done
ar25-t1a001	T1b	3/8	51,54,100	19	300	done
cd82-fb555c5d	T0	2/6	69,74,139	27	300	done
cd82-t1r001	T1	0/6	487	28	300	done
cd82-t1r001 (run 2)	T1	2/6	114,45,20	20	300	done
cd82-t1a001	T1b	1/6	73,78	16	300	done
cd82-t2l0s001	T2	1/1	86	14	282	done
cd82-t2l0s002	T2	0/1	72	18	300	done
cd82-t2l0s003	T2	0/1	110	19	300	done
cd82-t4wrap	T4	0/1	45	12	300	done
cn04-2fe56bfb	T0	1/6	30,36	24	300	done
cn04-t1r001	T1	0/6	52	14	300	done
cn04-t1r001 (run 2)	T1	1/6	31,43	16	300	done
cn04-t1a001	T1b	0/6	57	14	300	done
cn04-t2l0s001	T2	0/1	39	8	117	done
cn04-t2l0s002	T2	0/1	79	17	300	done
cn04-t2l0s003	T2	0/1	72	17	300	done
cn04-t4rotccw	T4	1/1	40	6	112	done
ft09-0d8bbf25	T0	2/6	62,24,44	20	300	done
ft09-t1r001	T1	4/6	57,14,15,19,107	20	300	done
ft09-t2l0s012	T2	1/1	47	8	82	done
lp85-305b61c3	T0	2/8	10,166,20	17	300	done
lp85-305b61c3 (run 2)	T0	1/8	14,21	10	197	done
lp85-t1r001	T1	5/8	9,40,32,28,26,18	20	300	done
lp85-t1r001 (run 2)	T1	4/8	10,36,24,22,15	15	300	done
lp85-t2l0s002	T2	1/1	10,22	5	28	done
lp85-t2l0s006	T2	1/1	45,22	16	268	done
lp85-t2l0s009	T2	1/1	17,29	6	63	done
lp85-t4flip	T4	1/1	9,22	5	48	done
m0r0-492f87ba	T0	2/6	58,175,174	42	300	done
m0r0-t1r001	T1	2/6	46,45,305	26	300	done
m0r0-t1a001	T1b	2/6	53,61,39	20	300	done
m0r0-t2l0s007	T2	1/1	66	3	62	done
m0r0-t2l0s007 (run 2)	T2	1/1	111	6	71	done
m0r0-t2l0s008	T2	1/1	106,35	14	142	done
m0r0-t2l0s012	T2	1/1	72	6	62	done
m0r0-t4mirror	T4	1/1	58,3	2	46	done
m0r0-t4same	T4	1/1	51	1	19	done
tr87-cd924810	T0	0/6	80	17	300	done
tr87-t1r001	T1	0/6	74	15	300	done
tr87-t1a001	T1b	0/6	208	17	300	done
tu93-0768757b	T0	4/9	26,23,27,30,18	21	300	done
tu93-t1r001	T1	4/9	70,28,25,50,4	21	300	done
tu93-t1a001	T1b	4/9	48,27,27,20,38	19	300	done

Twin, GPT-6 medium.

Version	Tier	Levels	Actions per level	Calls	Wall (min)	Status
cd82-fb555c5d	T0	6/6	17,6,17,14,13,16	–	17	done
cd82-t1r001	T1	6/6	20,6,17,14,13,16	–	23	done
cd82-t2l0s001	T2	0/1	22	–	–	partial
cd82-t4wrap	T4	1/1	4	–	2	done
cn04-2fe56bfb	T0	4/6	16,32,46,32	–	–	partial
cn04-t1r001	T1	3/6	14,42,28,11	–	–	partial
cn04-t2l0s001	T2	1/1	29	–	15	done
cn04-t4rotccw	T4	1/1	14	–	5	done
ft09-t2l0s012	T2	1/1	12	–	4	done
lp85-305b61c3	T0	8/8	8,8,16,12,10,20,7,11	–	19	done
lp85-t1r001	T1	8/8	8,8,16,13,11,20,6,11	–	19	done
lp85-t2l0s002	T2	1/1	6	–	2	done
lp85-t4flip	T4	1/1	6	–	2	done
m0r0-492f87ba	T0	4/6	20,35,59,11,2	–	7	partial
m0r0-t2l0s007	T2	1/1	19	–	7	done
m0r0-t4mirror	T4	1/1	33	–	5	done
m0r0-t4same	T4	1/1	20	–	3	done

CbI, GPT-6 medium.

Version	Tier	Levels	Actions per level	Calls	Wall (min)	Status
cd82-fb555c5d	T0	6/6	133,25,36,22,21,24	29	259	done
cd82-t1r001	T1	6/6	254,13,40,46,51,24	38	321	done
cd82-t2l0s001	T2	1/1	40	8	13	done
cd82-t4wrap	T4	1/1	22	2	3	done
cn04-2fe56bfb	T0	4/6	23,74,51,228,66	51	600	done
cn04-t1r001	T1	0/6	341	33	299	done
cn04-t2l0s001	T2	1/1	97	29	377	done
cn04-t4rotccw	T4	1/1	21	3	5	done
ft09-t2l0s012	T2	1/1	9	1	1	done
lp85-305b61c3	T0	5/8	9,14,18,18,28,10	27	300	done
lp85-t1r001	T1	8/8	9,10,20,14,15,45,7,17	17	42	done
lp85-t2l0s002	T2	1/1	9	1	1	done
lp85-t4flip	T4	1/1	9	1	1	done
m0r0-492f87ba	T0	6/6	44,44,97,50,75,99	35	326	done
m0r0-t1r001	T1	6/6	23,178,75,36,68,227	26	91	done
m0r0-t2l0s007	T2	1/1	27	15	267	done
m0r0-t4mirror	T4	1/1	55	1	1	done
m0r0-t4same	T4	1/1	29	1	1	done

Trivial controls.

Version	Tier	Levels	Actions per level	Calls	Wall (min)	Status
ar25-0c556536	T0	0/8	120	0	0	done
ar25-t1r001	T1	0/8	120	0	0	done
ar25-t1a001	T1b	0/8	120	0	0	done
cd82-fb555c5d	T0	0/6	120	0	0	done
cd82-t1r001	T1	0/6	120	0	0	done
cd82-t1a001	T1b	0/6	120	0	0	done
cd82-t2l0s001	T2	0/1	120	0	0	done
cd82-t2l0s002	T2	0/1	120	0	0	done
cd82-t2l0s003	T2	0/1	120	0	0	done
cd82-t4wrap	T4	0/1	120	0	0	done
cn04-2fe56bfb	T0	0/6	120	0	0	done
cn04-t1r001	T1	0/6	120	0	0	done
cn04-t1a001	T1b	0/6	120	0	0	done
cn04-t2l0s001	T2	0/1	120	0	0	done
cn04-t2l0s002	T2	0/1	120	0	0	done
cn04-t2l0s003	T2	0/1	120	0	0	done
cn04-t4rotccw	T4	0/1	120	0	0	done
ft09-0d8bbf25	T0	0/6	120	0	0	done
ft09-t1r001	T1	0/6	120	0	0	done
ft09-t2l0s012	T2	0/1	120	0	0	done
lp85-305b61c3	T0	0/8	120	0	0	done
lp85-t1r001	T1	0/8	120	0	0	done
lp85-t2l0s002	T2	0/1	120	0	0	done
lp85-t2l0s006	T2	0/1	120	0	0	done
lp85-t2l0s009	T2	0/1	120	0	0	done
lp85-t4flip	T4	0/1	120	0	0	done
m0r0-492f87ba	T0	0/6	120	0	0	done
m0r0-t1r001	T1	0/6	120	0	0	done
m0r0-t1a001	T1b	0/6	120	0	0	done
m0r0-t2l0s007	T2	0/1	120	0	0	done
m0r0-t2l0s008	T2	1/1	20	0	0	done
m0r0-t2l0s012	T2	0/1	120	0	0	done
m0r0-t4mirror	T4	0/1	120	0	0	done
m0r0-t4same	T4	0/1	120	0	0	done
tr87-cd924810	T0	0/6	120	0	0	done
tr87-t1r001	T1	0/6	120	0	0	done
tr87-t1a001	T1b	0/6	120	0	0	done
tu93-0768757b	T0	0/9	120	0	0	done
tu93-t1r001	T1	0/9	120	0	0	done
tu93-t1a001	T1b	0/9	120	0	0	done

Plain harness, DeepSeek V4.1 Flash.

Version	Tier	Levels	Actions per level	Calls	Wall (min)	Status
cd82-fb555c5d	T0	0/6	83	–	92	done
cd82-t1r001	T1	1/6	52,13	–	81	done
cn04-2fe56bfb	T0	0/6	44	–	47	done
cn04-t1r001	T1	0/6	44	–	20	done
lp85-305b61c3	T0	1/8	17,58	–	54	done
lp85-t1r001	T1	0/8	26	–	17	done
m0r0-492f87ba	T0	0/6	45	–	42	done
m0r0-t1r001	T1	0/6	45	–	43	done

Tycho, DeepSeek V4.1 Flash.

Version	Tier	Levels	Actions per level	Calls	Wall (min)	Status
cd82-fb555c5d	T0	1/6	74,12	300	56	done
cd82-t1r001	T1	0/6	99	300	68	done
cd82-t2l0s001	T2	0/1	60	300	67	done
cd82-t4wrap	T4	1/1	25,0	139	26	done
cn04-2fe56bfb	T0	0/6	86	300	48	done
cn04-t1r001	T1	1/6	37,22	300	72	done
cn04-t2l0s001	T2	0/1	123	300	64	done
cn04-t4rotccw	T4	1/1	26,0	103	19	done
ft09-t2l0s012	T2	0/1	85	300	69	done
lp85-305b61c3	T0	4/8	12,11,26,15,6	300	46	done
lp85-t1r001	T1	4/8	10,11,16,14,4	300	52	done
lp85-t2l0s002	T2	1/1	8,0	44	8	done
lp85-t4flip	T4	1/1	9,0	25	3	done
m0r0-492f87ba	T0	2/6	30,38,21	300	41	done
m0r0-t1r001	T1	2/6	31,36,2	300	50	done
m0r0-t2l0s007	T2	1/1	20,0	100	21	done
m0r0-t4mirror	T4	1/1	37,0	142	23	done
m0r0-t4same	T4	1/1	27,0	75	16	done

CbI, DeepSeek V4.1 Flash.

Version	Tier	Levels	Actions per level	Calls	Wall (min)	Status
cd82-fb555c5d	T0	1/6	133,26	61	165	done
cd82-t1r001	T1	0/6	129	62	150	done
lp85-305b61c3	T0	3/8	9,144,23,3	54	300	done
lp85-t1r001	T1	1/8	9,11	60	42	done
m0r0-492f87ba	T0	0/6	211	62	86	done
m0r0-t1r001	T1	0/6	54	60	64	done

Plain harness, Qwen3.8-27B.

Version	Tier	Levels	Actions per level	Calls	Wall (min)	Status
ar25-0c556536	T0	0/8	48	–	56	done
ar25-t1r001	T1	0/8	48	–	29	done
ar25-t1a001	T1b	0/8	48	–	66	done
cd82-fb555c5d	T0	0/6	83	–	70	done
cd82-t1r001	T1	0/6	83	–	37	done
cd82-t1a001	T1b	1/6	11,13	–	23	done
cd82-t2l0s001	T2	0/1	83	–	69	done
cd82-t4wrap	T4	0/1	33	–	33	done
cn04-2fe56bfb	T0	0/6	44	–	26	done
cn04-t1r001	T1	0/6	44	–	33	done
cn04-t1a001	T1b	0/6	44	–	38	done
cn04-t2l0s001	T2	0/1	50	–	32	done
cn04-t4rotccw	T4	0/1	38	–	25	done
ft09-0d8bbf25	T0	0/6	65	–	124	done
ft09-t1r001	T1	0/6	65	–	131	done
ft09-t2l0s012	T2	0/1	65	–	133	done
lp85-305b61c3	T0	0/8	26	–	28	done
lp85-t1r001	T1	0/8	26	–	15	done
lp85-t2l0s002	T2	0/1	26	–	26	done
lp85-t4flip	T4	0/1	26	–	23	done
m0r0-492f87ba	T0	0/6	45	–	28	done
m0r0-t1r001	T1	0/6	45	–	136	done
m0r0-t1a001	T1b	0/6	45	–	29	done
m0r0-t2l0s007	T2	0/1	33	–	19	done
m0r0-t4mirror	T4	0/1	63	–	38	done
m0r0-t4same	T4	0/1	60	–	40	done
tr87-cd924810	T0	0/6	81	–	77	done
tr87-t1r001	T1	0/6	81	–	50	done
tr87-t1a001	T1b	0/6	81	–	70	done
tu93-0768757b	T0	0/9	29	–	31	done
tu93-t1r001	T1	0/9	29	–	47	done
tu93-t1a001	T1b	0/9	29	–	23	done

CbI, Qwen3.8-27B.

Version	Tier	Levels	Actions per level	Calls	Wall (min)	Status
ar25-0c556536	T0	0/8	145	60	41	done
ar25-t1r001	T1	0/8	224	60	41	done
ar25-t1a001	T1b	0/8	118	49	37	done
cd82-fb555c5d	T0	0/6	216	60	47	done
cd82-t1r001	T1	0/6	210	60	53	done
cd82-t1a001	T1b	0/6	166	60	49	done
cd82-t2l0s001	T2	0/1	171	60	49	done
cd82-t4wrap	T4	0/1	193	60	54	done
cn04-2fe56bfb	T0	0/6	206	60	45	done
cn04-t1r001	T1	0/6	155	60	44	done
cn04-t1a001	T1b	0/6	233	60	41	done
cn04-t2l0s001	T2	0/1	192	60	46	done
cn04-t4rotccw	T4	0/1	203	60	45	done
ft09-0d8bbf25	T0	0/6	236	60	27	done
ft09-t1r001	T1	0/6	260	60	21	done
ft09-t2l0s012	T2	0/1	152	60	37	done
lp85-305b61c3	T0	0/8	93	60	58	done
lp85-t1r001	T1	0/8	171	60	59	done
lp85-t2l0s002	T2	1/1	33,35	25	16	done
lp85-t4flip	T4	0/1	70	60	58	done
m0r0-492f87ba	T0	0/6	122	16	16	done
m0r0-t1r001	T1	0/6	253	60	36	done
m0r0-t1a001	T1b	0/6	219	60	44	done
m0r0-t2l0s007	T2	0/1	134	60	39	done
m0r0-t4mirror	T4	0/1	606	19	84	done
m0r0-t4same	T4	0/1	203	60	39	done
tr87-cd924810	T0	0/6	152	60	41	done
tr87-t1r001	T1	0/6	160	60	40	done
tr87-t1a001	T1b	0/6	156	60	40	done
tu93-0768757b	T0	0/9	124	60	42	done
tu93-t1r001	T1	0/9	138	60	46	done
tu93-t1a001	T1b	0/9	131	60	45	done