# SymStep: Symbolic Step Verification for Logical Reasoning

**Anonymous Author(s)**
Affiliation
email@example.com

## Abstract

Chain-of-thought (CoT) prompting can fail severely on constraint-dense logical reasoning tasks, where unverified errors accumulate silently across steps. We introduce **SymStep**: an LLM makes one atomic claim at a time (`DEDUCE: Alice, pet, Cat`); a lightweight constraint propagator checks the claim for consistency, rejects contradictions, and cascades implied facts automatically. **SymStep+G** additionally provides an MRV hint after each accepted step, directing the LLM toward the most constrained unresolved variable. On a 35-puzzle retained subset of **ZebraLogicBench**—drawn from a published benchmark of 1,000 Einstein-style logic puzzles—Direct and CoT both achieve **0%**, while SymStep+G reaches **97%**. On **AR-LSAT** analytical reasoning problems, SymStep achieves **100%** vs. CoT's 87%. On **LGP-14**, SymStep+G achieves **100%** vs. **0%** for CoT and Logic-LM, the strongest prior symbolic+LLM baseline we compare against. In addition, our ablation studies suggest that MRV guidance is a key mechanism for reducing directionless cycling, while consistency checking provides a safety net against explicit contradictions. Across six benchmarks spanning five task domains, SymStep variants match or exceed every baseline on constraint-dense and arithmetic tasks; a domain-boundary experiment on AQUA-RAT algebra (Appendix) confirms the advantage is constraint-density-specific. [1]

## 1 Introduction

Multi-step logical reasoning remains an open problem for LLMs. Current reasoning approaches perform poorly on relevant benchmarks: On ZebraLogicBench [Lin *et al.*, 2025], a benchmark of 1,000 Einstein-style logic puzzles, direct prompting and chain-of-thought (CoT) achieve 0%. On our own logical reasoning benchmark, CoT and Logic-LM, the strongest prior symbolic+LLM method, also achieve 0% (Table 1). The failure mode is universal: models produce steps during reasoning, each unverified step can introduce an error, and subsequent steps compound that error without detection—*unverified step accumulation*. A second failure occurs even when deductions are individually correct: without external direction, the LLM cycles without progress—*deductive deadlock*.

| Method | Haiku (LGP-14) | Sonnet (LGP-10) |
|---|---|---|
| Direct | 14% | 0%† |
| CoT | **0%** | **0%**† |
| Self-Refine | 36% | 0%† |
| Logic-LM | 0% | – |
| SymStep | 86% | 90% |
| **SymStep+G** | **100%** | **90%** |

Table 1: CoT (red) achieves 0% on logical reasoning, worse than random guessing. Logic-LM, the strongest symbolic+LLM baseline, also achieves 0%. SymStep+G (green) achieves the best accuracy across both model families. †Sonnet baselines reflect a regex parser calibrated for Haiku's concise output style; the parser has been corrected (§4.4), and re-evaluation is recommended.

Logic-LM [Pan *et al.*, 2023] and LINC [Olausson *et al.*, 2023] translate the *entire problem* into formal logic in a single step before invoking a solver; if that translation is wrong anywhere, the whole attempt fails with no opportunity for correction. Although DVCOTR [Ling *et al.*, 2023] decomposes reasoning into steps and verifies them, it uses the LLM itself as the verifier, leaving LLM errors and biases undetected. SymStep takes a different path. First, every verification decision is *deterministic*: a constraint propagator, not an LLM, checks each claim for consistency before accepting it. Second, the propagator's arc-consistency cascade automatically derives implied eliminations that no single clue reveals in isolation, performing the bookkeeping that the LLM cannot reliably perform on its own. Third, MRV-based guidance tells the LLM which variable to tackle next, resolving the *deductive deadlocks* that leave reactive-only systems cycling without progress.

**Contributions.**

1. **SymStep and SymStep+G.** SymStep structures LLM output into one atomic DEDUCE assertion per turn.

[1] Code and LGP-20 benchmark available at: https://anonymous.4open.science/r/SymStep-3B04

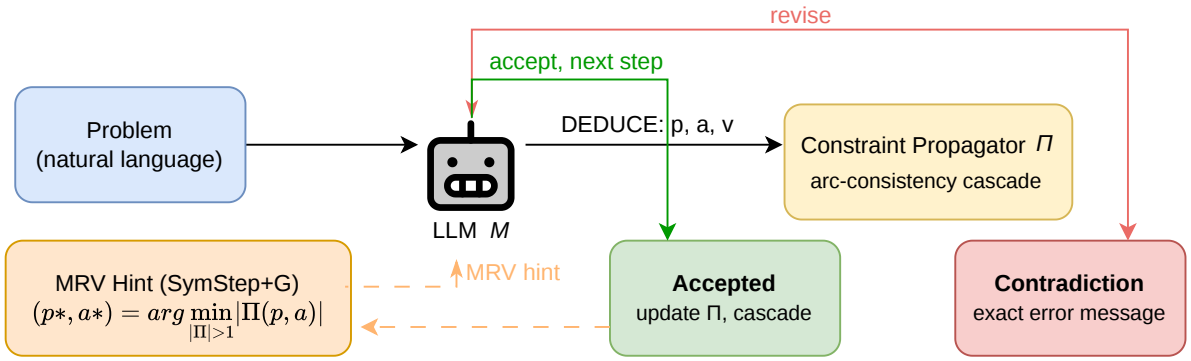


Figure 1: SymStep pipeline (**solid**: both variants; **orange dashed**: SymStep+G only). The LLM generates one DEDUCE assertion per turn; the constraint propagator checks consistency and cascades arc-consistency automatically. **Green**: the deduction is accepted, Π is updated, and the LLM continues. **Red**: the exact contradiction is reported, and the LLM revises. **Orange dashed** (SymStep+G only): the MRV hint directs the LLM toward the most constrained undetermined variable $(p^*, a^*)$.

A constraint propagator then verifies it and cascades implied eliminations, or feeds back the exact contradiction if it is wrong. SymStep+G additionally provides an MRV hint containing the unresolved variable with the fewest remaining candidates after each verified step, eliminating deductive deadlocks before they arise. Our algorithm requires no formal logic training, only regex-parseable output.

2. **Ablation insight: guidance is the primary driver.** A controlled ablation (Table 9) reveals that MRV guidance alone achieves **100%** with zero variance, while verification alone reaches only 72%. This result shows that the key bottleneck is not incorrect deductions but directionless search: once pointed to the most constrained variable, the LLM makes only correct deductions, and the safety-net verifier is rarely triggered.

3. **Six-domain evaluations.** We test on ZebraLogicBench (SymStep+G 97% vs. CoT 0%), AR-LSAT analytical reasoning (SymStep 100% vs. CoT 87%), LGP-14 (100%), and three additional domains (scheduling, arithmetic, and financial reasoning). A domain-boundary experiment on AQUA-RAT algebra (Appendix E) confirms that the advantage is constraint-density-specific: where constraint propagation is not needed, SymStep+G matches CoT with no degradation.

4. **SOTA comparison and statistical validation.** Logic-LM, the strongest prior symbolic+LLM baseline, achieves 0% where SymStep achieves 100%. 95% Wilson CIs confirm statistical significance: SymStep+G's lower bound (79%) strictly exceeds CoT's upper bound (22%) on LGP-14. In addition, the upper bounds for Direct and CoT are 10% on ZebraLogicBench, while SymStep+G's lower bound reaches around 85%.

# 2 Problem Formulation

## 2.1 Logical Grid Puzzles

Logical grid puzzles are the clearest test bed for step-level deductive reasoning because every fact must follow explicit constraints; there is no world knowledge to fall back on, and any undetected error propagates through all subsequent deductions. In a scenario such as *"Alice, Bob, and Carol each have a different pet (cat, dog, fish) and live in differently colored houses (red, blue, green). Alice does not have a dog. Bob lives in the blue house. . . . "*, the goal is to determine who has what, using only the given clues. Every attribute assignment must be derived step by step from the constraints.

Formally, a problem has $n$ *entities* (e.g., people) and $k$ *attributes* (e.g., pet, job), each with a finite domain of values. Every attribute is *bijective*: each value is assigned to exactly one entity. The task is to uniquely determine all entity–attribute–value assignments from the given clues.

**Solution uniqueness.** We require each puzzle to have *exactly one* valid assignment. This uniqueness is verified by running the constraint propagator on the stored solution and confirming that it reaches a fully determined state with no alternatives, a check that also caught *two* originally included puzzles (E3-Sport and M2-Nationals) that each admitted two valid solutions; automated uniqueness checks are mandatory in deductive benchmarks. (See §4.1.)

## 2.2 Generalization to Other Reasoning Domains

While LGPs are our primary benchmark, SymStep's architecture is domain-agnostic: it applies to *any task that can be decomposed into a sequence of atomic constraint assertions with checkable consistency*. We demonstrate this on three additional domains:

**Scheduling puzzles (SP).** Constraint satisfaction problems in a scheduling context, e.g., assigning presenters to conference time slots given ordering and compatibility constraints. SP share the bijective structure of LGPs but use

different domain vocabulary (time slots, rooms, roles) from the LGP distribution, enabling cross-domain transfer tests.

**Math word problems (MWP) and financial reasoning (FIN).** Multi-step numeric reasoning (e.g., deriving employee salaries, compound interest, or tax liabilities from arithmetic relationships). Each DEDUCE step asserts a variable's numeric value. Verifier checks arithmetic consistency and cascades any directly implied values. The MRV hint identifies the next *computable* unknown: variable whose constraint becomes evaluable given current knowledge.

All extensions reuse the same DEDUCE protocol and outer SymStep loop, only the symbolic verifier is adapted per domain.

### 2.3 Why Step-Level Verification?

Consider a small puzzle from E2-Jobs (Adam/Beth/Chris, drinks: Tea/Coffee/Water, jobs: Doctor/Teacher/Engineer):

```
Adam drinks tea (clue 1).  The
doctor drinks coffee (clue 2),
so Adam is not the doctor.  Beth is
not the engineer (clue 3).
Chris is not the doctor (clue 4).
So the doctor must be Beth.
Adam is not the teacher (clue 5), so
Adam is the engineer.
Chris is the teacher.  Beth drinks
coffee.
Therefore:  Adam=Tea+Teacher,
Beth=Coffee+Doctor,
Chris=Water+Engineer.
```

Although the reasoning chain correctly establishes "Adam is the engineer" in step 4, it contradicts that conclusion in the final answer ("Adam=Teacher"). The parser receives "Adam=Teacher" as the final answer and marks the puzzle incorrect. This is *unverified step accumulation*: each new sentence updates the narrative without checking it against what has already been established. The longer the chain, the greater the chance that a contradiction will go unnoticed.

SymStep prevents this by modeling reasoning as a sequence of atomic constraint assertions $d_t = (\sigma(p, a) = v)$ or $d_t = (\sigma(p, a) \neq v)$. A deduction is *consistent* if it does not contradict any previously accepted deduction. The propagator enforces this property at every step, ensuring that no inconsistent fact enters the context.

## 3 SymStep

### 3.1 The DEDUCE Protocol

SymStep structures LLM–verifier interaction through a minimal **DEDUCE protocol**: the LLM produces exactly one constraint assertion per step:

```
DEDUCE: Entity, attribute, Value
DEDUCE: Entity, attribute, NOT Value
```

This format is parseable by regex, requires no formal logic training, and is far less demanding than generating a complete Prolog program or FOL formula. The LLM signals task completion with `CONCLUDE: done`.

### 3.2 Constraint Propagator

The symbolic backbone of SymStep is a **constraint propagator** that maintains a *possibility matrix*: $\Pi(p, a) \subseteq V_a$: a set of values still consistent with all accepted deductions for entity $p$, attribute $a$. Initially, $\Pi(p, a) = V_a$ for all $p, a$.

**Intuition.** $\Pi$ is similar to a grid where each cell lists which values a person might have for a given attribute. When the LLM deduces "Alice has Red," the propagator (i) crosses Red off the grid for every other person (arc-consistency), and (ii) checks whether any cell has now shrunk to a single option. If so, that forced assignment is propagated automatically, potentially triggering further eliminations without requiring an LLM call.

**Positive update.** On $\sigma(p, a) = v$: if $v \notin \Pi(p, a)$, return CONTRADICTION. Otherwise, set $\Pi(p, a) \leftarrow \{v\}$ and remove $v$ from $\Pi(p', a)$ for all $p' \neq p$; if any such set becomes empty, return CONTRADICTION.

**Negative update.** On $\sigma(p, a) \neq v$: discard $v$ from $\Pi(p, a)$; if $\Pi(p, a) = \emptyset$, return CONTRADICTION.

**Unit propagation (arc-consistency cascade).** After each update, if $|\Pi(p, a)| = 1$ for any undetermined $(p, a)$, that singleton value is propagated as a positive deduction, potentially triggering further eliminations. This arc-consistency step may derive eliminations not derivable from any single clue in isolation—the propagator "does the bookkeeping" that the LLM could miss.

The propagator returns one of: OK($msg$), CONTRADICTION($msg$), or SOLVED. Contradiction messages identify the specific inconsistency, e.g., *"assigning Cat to Alice leaves Bob.pet with no valid values."* The propagator runs in $O(n^2 k)$ time per call, which is faster than LLM inference. Note that the propagator guarantees *consistency* (no contradiction with prior accepted claims) not *entailment*: a deduction consistent with the current state but not implied by the clues can still be accepted; the LLM bears responsibility for clue-grounded claims (formal analysis in Appendix F).

### 3.3 SymStep: Verified Step-by-Step Reasoning

Figure 1 illustrates the full pipeline, and Appendix Algorithm 1 gives the pseudocode. On each turn, the LLM receives the full interaction history $h$ and produces a single DEDUCE assertion. The propagator checks it, and feedback (confirmation or contradiction) is appended to $h$ before the next LLM call. The LLM never sees an unverified fact accepted into its context, every piece of confirmed information is consistent.

### 3.4 SymStep+G: MRV-Guided Reasoning

After a verified step, SymStep+G appends an MRV hint computed from the updated propagator state:

$$\begin{aligned} \textsc{MrvHint}(\Pi) &= \big(p^*,\, a^*,\, \Pi(p^*, a^*)\big), \\ (p^*, a^*) &= \arg \min_{\substack{p \in P,\, a \in \mathcal{A} \\ |\Pi(p,a)|>1}} |\Pi(p, a)| \end{aligned} \tag{1}$$

The hint is delivered in natural language, e.g.:

*[Hint] Blake's drink must be one of: {Coffee, Juice}*

The hint lists remaining candidates, not the answer. It is derived entirely from propagated arc-consistency: information the LLM cannot efficiently compute without external help. The design mirrors the MRV heuristic in classical constraint satisfaction [Hoffmann and Nebel, 2001]: choosing the most constrained variable next minimizes branching and resolves deadlocks.

### 3.5 Why Per-Step Granularity Matters

Appendix Table 16 positions SymStep against existing work. Whole-problem approaches (Logic-LM, LINC) perform a single large translation that is brittle to partial descriptions and natural-language ambiguity. LLM-as-verifier approaches (DVCOTR, Self-Refine) cannot guarantee soundness. Only SymStep combines *deterministic, per-step* verification with proactive guidance.

## 4 Experiments

In total, SymStep is evaluated on 84 problem instances across five task domains (grid puzzles, scheduling, arithmetic, financial reasoning, and formal constraint): 34 hand-crafted puzzles (LGP-14, SP-6, MWP-8, FIN-6) and 50 problems from two published external benchmarks (ZebraLogicBench $n$=35, AR-LSAT $n$=15). A domain-boundary analysis on 35 AQUA-RAT algebra problems is in Appendix E. All main comparisons are method-vs-method on the same model (Claude Haiku), isolating architectural contribution from model scale; the Sonnet cross-model comparison is in §4.4. Single-run results on each benchmark are cross-validated by the N=3 multi-run ablation (§5.2), which confirms zero-variance reproducibility for guided configurations.

### 4.1 Benchmark: LGP-14

A meaningful evaluation of step-level verification requires puzzles that are genuinely hard for unverified methods, with multiple interacting constraints and a unique, checkable solution. We construct **LGP-14**, a benchmark of fourteen logical grid puzzles across three difficulty levels (Appendix Table 12). All puzzles have a unique solution verified by the constraint propagator; this check caught *two* originally included puzzles (E3-Sport and M2-Nationals) that each admitted two valid solutions. Difficulty scales with the number of entities, attributes, and clue density. We additionally provide **LGP-20**, which includes six further verified puzzles (E5–E6, M6–M7, H6–H7) released with this paper for future scale-up evaluation.

### 4.2 Methods and Baselines

SymStep constitutes multiple design paradigms: pure LLM generation, iterative self-improvement, one-shot symbolic program synthesis, and our own step-level verification. To assess the contribution of each component, we compare six methods spanning four design paradigms:

**(1) Zero-shot methods. Direct**: a single LLM call that outputs the full assignment directly. **CoT** [Wei *et al.*, 2022]: add "think step by step" before answering. Both make exactly one LLM call per puzzle.

| Method | Easy (4) | Med. (5) | Hard (5) | Total | 95% CI |
|---|---|---|---|---|---|
| Direct | 2/4 | 0/5 | 0/5 | 2/14 (14%) | [4, 40]% |
| CoT | 0/4 | 0/5 | 0/5 | 0/14 (0%) | [0, 22]% |
| Self-Refine | 0/4 | 1/5 | 4/5 | 5/14 (36%) | [16, 61]% |
| Logic-LM | 0/4 | 0/5 | 0/5 | 0/14 (0%) | [0, 22]% |
| SymStep | 4/4 | 4/5 | 4/5 | 12/14 (86%) | [60, 96]% |
| **SymStep+G** | 4/4 | 5/5 | 5/5 | **14/14 (100%)** | **[79, 100]%** |

Table 2: Main results on LGP-14 (Haiku). 95% Wilson CIs confirm that SymStep+G's lower bound (79%) strictly exceeds the upper bounds of CoT and Logic-LM (both 22%), establishing statistical significance despite $n$=14. Cost: $\approx$ \$0.0001 per puzzle for Direct/CoT; $\approx$ \$0.0013 per puzzle for SymStep+G (Haiku pricing).

**(2) Self-improvement without symbolic grounding. Self-Refine** [Madaan *et al.*, 2023]: generate an initial solution, then ask the LLM to re-check every clue and revise. This method uses two LLM calls total; the feedback is generated and evaluated by the same LLM, providing no external correctness guarantee.

**(3) Symbolic-program generation (SOTA). Logic-LM** [Pan *et al.*, 2023]: ask the LLM to write *complete Z3 Python code* for the puzzle, execute it, and read the solver's output. This is the strongest prior symbolic+LLM baseline: it encodes all constraints at once and delegates solving to an exact algorithm. Success requires the LLM to translate every natural-language clue into correct Z3 syntax, a non-trivial task even for capable models.

**(4) Step-level verification with a propagator (ours). SymStep**: Appendix Algorithm 1 with GUIDE=FALSE. **SymStep+G**: Appendix Algorithm 1 with GUIDE=TRUE. Unlike Logic-LM's one-shot code generation, SymStep interleaves LLM deductions with *immediate* symbolic feedback after every single step, allowing incremental error correction.

A puzzle is counted as *correct* if every entity–attribute pair matches the ground truth exactly.

**Models.** We run all methods on both Haiku (LGP-14, SP-6, MWP-8, FIN-6) and Sonnet (LGP-10 subset), enabling cross-model comparison. Full prompts are in Appendix I.

### 4.3 Main Results on LGP-14 (Haiku)

Table 2 shows results on LGP-14. SymStep+G achieves a perfect **100%** across all difficulty levels. Per-step symbolic verification with MRV guidance solves every puzzle in the benchmark. CoT and Logic-LM both achieve 0%, confirming that unverified step accumulation and brittle one-shot program synthesis collapse on constraint-dense reasoning.

### 4.4 Cross-Model and Per-Puzzle Detail

Both SymStep and SymStep+G achieve **90%** on LGP-10 with Sonnet (Appendix C), confirming that the architecture is model-agnostic. Sonnet baseline figures require re-evaluation with a corrected output parser. CoT achieves 0% on all 14 LGP-14 puzzles. SymStep's two failures (one medium, one hard) are both deductive deadlocks; the hard failure H2-Full stalls after 6 steps without directional guidance. SymStep+G's MRV hint resolves both immediately,

| Method | Easy (2) | Med. (2) | Hard (2) | Total | 95% CI |
|---|---|---|---|---|---|
| Direct | 0/2 | 0/2 | 0/2 | 0/6 (0%) | [0, 39]% |
| CoT | 0/2 | 0/2 | 0/2 | 0/6 (0%) | [0, 39]% |
| Self-Refine | 1/2 | 1/2 | 0/2 | 2/6 (33%) | [10, 70]% |
| Logic-LM | 0/2 | 0/2 | 0/2 | 0/6 (0%) | [0, 39]% |
| SymStep | 2/2 | 2/2 | 1/2 | 5/6 (83%) | [44, 97]% |
| **SymStep+G** | 2/2 | 1/2 | 1/2 | **4/6 (67%)** | **[30, 90]%** |

Table 3: Results on the SP-6 scheduling benchmark. SymStep (83%) outperforms SymStep+G (67%) here, but their 95% CIs overlap substantially ([44,97] vs. [30,90]), so the difference is not statistically significant at $n$=6. Both substantially exceed the baselines.

| Method | Easy (3) | Med. (3) | Hard (2) | Total |
|---|---|---|---|---|
| Direct | 3/3 | 3/3 | 1/2 | 7/8 (88%) |
| CoT | 3/3 | 3/3 | 2/2 | 8/8 (100%) |
| SymStep | 3/3 | 3/3 | 2/2 | 8/8 (100%) |
| **SymStep+G** | 3/3 | 3/3 | 2/2 | **8/8 (100%)** |

Table 4: Results on MWP-8 math word problems. On MWP-8, CoT also achieves 100%, demonstrating that SymStep's advantage is specific to constraint-dense deductive tasks where unverified accumulation is harmful; for straightforward numeric chains, CoT suffices.

achieving 14/14. Per-puzzle outcomes and the per-difficulty bar chart are in Appendix D.

### 4.5 Scheduling Benchmark: SP-6

Strong results on LGP-14 could reflect overfitting to a single domain. SP-6 tests transferability of system to problems it was not designed for, evaluating SymStep on scheduling problems where presenters, researchers, or crew members are assigned to time slots, labs, and venues. The bijective constraint structure is preserved, but surface vocabulary, entity types, and clue phrasing differ entirely from LGP-14.

The per-puzzle breakdown (Table 14, Appendix **??**) reveals where guidance hurts: SP-M1-Projects and SP-M2-Research have a natural presentation order that the LLM exploits without guidance. The MRV hint redirects the LLM to a different variable, disrupting its natural trajectory. When guidance helps vs. hurts (dense constraint graphs vs. problems with a clear natural ordering) remains an open question for future work.

### 4.6 Math Word Problems: MWP-8

MWP-8 shifts to numeric chain derivation: each problem describes arithmetic relationships in natural language. DEDUCE asserts variable values, and the verifier checks arithmetic consistency.

On MWP-8, SymStep, SymStep+G, and CoT all achieve **100%**, while Direct misses one hard problem (88%). CoT is not universally harmful: it fails specifically on constraint-dense tasks where propagation is needed, but suffices for simple numeric chains where dependencies are linear and errors do not cascade.

| Method | Easy (2) | Med. (2) | Hard (2) | Total |
|---|---|---|---|---|
| Direct | 1/2 | 1/2 | 2/2 | 4/6 (67%) |
| CoT | 2/2 | 1/2 | 2/2 | 5/6 (83%) |
| SymStep | 2/2 | 2/2 | 2/2 | 6/6 (100%) |
| **SymStep+G** | 2/2 | 2/2 | 2/2 | **6/6 (100%)** |

Table 5: Results on FIN-6 financial reasoning. Both SymStep variants achieve 100%, while direct prompting fails on an easy interest problem and both Direct and CoT fail on break-even analysis.

| Method | Acc. | Total | 95% CI |
|---|---|---|---|
| Direct | 100% | 15/15 | [80, 100]% |
| CoT | 87% | 13/15 | [63, 96]% |
| SymStep | 100% | 15/15 | [80, 100]% |
| **SymStep+G** | 100% | 15/15 | [80, 100]% |

Table 6: Results on AR-LSAT (15 problems, Haiku). Both SymStep variants achieve 100%; CoT drops to 87% on the hardest ordering problems, where multi-step inconsistencies accumulate.

### 4.7 Financial Reasoning: FIN-6

FIN-6 tests six high-stakes financial problems (compound interest, margin analysis, break-even analysis, portfolio returns, and income tax) where multi-step arithmetic errors accumulate silently.

Direct (67%) and CoT (83%) both fail on compound arithmetic chains, while SymStep achieves **100%**. On FIN-M2-BreakEven, the propagator cascades the two-step dependency automatically once the LLM asserts seed values.

### 4.8 AR-LSAT: Formal Constraint Reasoning

AR-LSAT is an external benchmark of 230 formal constraint problems (ordering, selection, and assignment tasks drawn directly from real LSAT analytical reasoning sections). Unlike grid puzzles, clues are expressed in legal and linguistic natural language, testing whether SymStep's symbolic pipeline generalizes beyond grid-puzzle vocabulary.

CoT fails on 2 of 15 problems (the hardest multi-step ordering items), while Direct, SymStep, and SymStep+G all achieve **100%**. AR-LSAT can be generally solved by structured reasoning (even one-shot Direct). The symbolic verifier provides a formal guarantee against the 13% CoT inconsistency-accumulation failure, confirming that the pattern from LGP-14 holds on linguistic natural language.

### 4.9 ZebraLogicBench: External Benchmark Generalization

The four benchmarks above (LGP-14, SP-6, MWP-8, FIN-6) establish SymStep's robustness, but do not prove that the results are not an artifact of being designed around those specific puzzles. **ZebraLogicBench** [Lin *et al.*, 2025] is the cleanest available test: a published benchmark of 1,000 Einstein-style logic grid puzzles spanning 25 difficulty levels (2–6 houses $\times$ 2–6 attributes, ICML 2025) that SymStep has never seen. We evaluate on all puzzles from four size classes ($2\times3$, $2\times4$, $3\times3$, $3\times4$), sampling uniformly to balance difficulty. Because ZebraLogicBench does not include

| Method | 2×3 | 2×4 | 3×3 | 3×4 | Total |
|---|---|---|---|---|---|
| Direct | 0/9 | 0/9 | 0/11 | 0/6 | 0/35 (0%) |
| CoT | 0/9 | 0/9 | 0/11 | 0/6 | 0/35 (0%) |
| SymStep | 9/9 | 9/9 | 7/11 | 3/6 | 28/35 (80%) |
| **SymStep+G** | 9/9 | 8/9 | 11/11 | 6/6 | **34/35 (97%)** |

Table 7: Results on ZebraLogicBench (Haiku). Puzzles are drawn from a published 1,000-puzzle external benchmark and were never seen during SymStep's design. Direct and CoT achieve 0% across all sizes, while SymStep+G reaches 97%, replicating the LGP-14 pattern on an independently sourced benchmark.

| | Constraint / Logical | | | | Arithmetic | |
|---|---|---|---|---|---|---|
| **Method** | **LGP-14** | **ZLB** | **SP-6** | **LSAT** | **MWP-8** | **FIN-6** |
| Direct | 14% | 0% | 0% | 100% | 88% | 67% |
| CoT | 0% | 0% | 0% | 87% | 100% | 83% |
| Self-Refine | 36% | – | 33% | – | – | – |
| Logic-LM | 0% | – | 0% | – | – | – |
| SymStep | 86% | 80% | 83% | 100% | 100% | 100% |
| **SymStep+G** | **100%** | **97%** | 67% | **100%** | **100%** | **100%** |

Table 8: Cross-benchmark summary across six datasets. Left: constraint-dense logical reasoning (LGP-14, ZLB, SP-6, LSAT). Right: arithmetic (MWP-8, FIN-6). SymStep variants dominate constraint-dense tasks and match or exceed all baselines on arithmetic. "–" = not evaluated. See Appendix E for domain-boundary results on AQUA-RAT algebra.

ground-truth solutions, we derive them with a custom backtracking CSP solver applied to the structured clue format. Puzzles yielding non-unique solutions or unparseable clue formats are excluded. The rest 35 puzzles are distributed as 9+9+11+6 across the four size classes. Methods and evaluation are identical to those used for LGP-14.

Table 7 shows the same qualitative pattern as LGP-14. Direct and CoT fail on constraint-dense deductive puzzles regardless of puzzle source, while SymStep+G maintains high accuracy. The result is notable because ZebraLogicBench puzzles use different vocabulary, attribute types (car models, house styles, etc.), and clue-phrasing conventions than LGP-14. SymStep's symbolic propagator operates on the assignment structure, not on surface form.

### 4.10 Cross-Benchmark Summary

Table 8 reveals a consistent two-regime pattern. On constraint-dense logical reasoning tasks (LGP-14, ZLB, SP-6, LSAT), Direct and CoT often collapse, while SymStep variants reach high accuracy; the step-level propagator is the deciding factor. On arithmetic and financial tasks (MWP-8, FIN-6), SymStep matches or exceeds every baseline; the gap is largest on FIN-6, where multi-step chains of dependent quantities punish any single undetected error.

MRV guidance matters most when constraint graphs are dense: SymStep+G leads on LGP-14 (100% vs. 86%), while both variants achieve 100% on arithmetic tasks. Logic-LM's 0% on LGP-14 and SP-6 makes the contrast concrete: complete program synthesis is brittle to natural-language ambiguity, whereas one checkable claim at a time sidesteps the problem.

| Config. | Description | Acc. | Calls | Contra. |
|---|---|---|---|---|
| None (= CoT) | No verification, no hint | 0% | 1.0 | – |
| Guidance only | MRV hint, no contra. report | **100%** | 6.3 | 0 |
| Verification only | Contra. report, no hint | 72% | 7.2 | 0 |
| Both (= SymStep+G) | Contra. + MRV hint | **100%** | 6.5 | 0 |

Table 9: Component ablation on LGP-6 ($N$=3 mean, per-puzzle accuracy). **Guidance alone achieves 100%** with zero contradictions; verification alone reaches only 72%. All 6 puzzles are solved in every run by both guided configurations (18/18 total evaluations), confirming zero-variance reproducibility. Verification-only failures are *deadlocks* (the LLM cycles without progress), not contradictions: the LLM rarely asserts inconsistent facts but stalls without directional guidance. The MRV signal, not error correction, is the primary driver.

## 5 Ablation Studies

SymStep+G has two moving parts: a verifier that rejects inconsistent claims and an MRV hint that directs what to try next. Ablation studies isolate contribution of each part.

### 5.1 Component Ablation

We ran four configurations on LGP-6 (the six core easy/medium/hard puzzles) over three independent runs: no augmentation (CoT), guidance only, verification only, and both together.

MRV guidance alone achieves 100% ($\sigma$=0, 18/18 evaluations correct) without any contradiction detection, while verification alone reaches only 72%. The primary bottleneck is not *incorrect* deductions but *directionless* ones: the LLM stalls because it does not know which variable will expose the most information. Once pointed toward the most constrained cell (2–3 candidates), it makes only correct deductions and no contradictions across all guided runs. In the guidance-only configuration, the propagator still runs silently, contradictions are detected internally to maintain the possibility matrix for MRV, but are not reported to the LLM, which receives only directional hints.

Verification-only reaches 72% ($\sigma$=9.6%): it catches errors but cannot prevent deadlocks. On LGP-14, complementarity is visible: SymStep solves 12/14 (86%), while SymStep+G solves all 14/14, with the MRV hint resolving the two deadlock-prone hard puzzles. As puzzles scale larger, the probability of an incorrect assertion on a constrained cell increases, and the safety net becomes load-bearing, making verification essential beyond the controlled LGP-6 setting.

### 5.2 Multi-Run Reproducibility

Table 10 reports mean ± standard deviation computed from per-puzzle outcomes across three independent runs, confirming that the zero-variance result is not due to single run.

| Method | Mean Acc (%) | Std (%) |
|---|---|---|
| CoT | 0.0 | 0.0 |
| Verification only (SymStep) | 72.2 | 9.6 |
| Guidance only | 100.0 | 0.0 |
| **SymStep+G** | **100.0** | **0.0** |

Table 10: Multi-run statistics on LGP-6 ($N$=3, Haiku). Both guided configurations solve all 6 puzzles in every run (18/18). Verification-only variance ($\sigma$=9.6%) reflects which hard puzzle fails in a given run, not systemic instability.

# 6 Analysis and Discussion

**CoT vs. constraint tasks** The results show a specific, predictable failure mode. CoT produces a long, unconstrained narrative: each sentence updates the story without checking whether it contradicts what has already been established. On constraint-dense problems like LGP-14, undetected contradictions accumulate faster than a direct guess, introducing more errors, which is why CoT ends up worse than answering directly. This failure is *constraint-density-specific*: CoT achieves 100% on MWP-8, where arithmetic chain dependencies are simpler and a single wrong number does not produce a tangled web of inconsistencies. Hence, as constraint density rises, unverified reasoning becomes actively harmful.

**Guidance as primary driver, verification as safety net.** There are two main reasons to include verification, even with guidance. First, the 72% verification-only result shows that verification recover errors that guidance cannot prevent. As puzzles grow harder and assertion errors become more frequent, this safety net becomes essential. Second, SymStep's consistency guarantee is formal: the propagator ensures that no inconsistent fact ever enters the LLM's context, regardless of model behavior. MRV prevents deadlocks, symbolic verification provides correctness. Neither suffices alone as problems scale.

**SymStep as propagation-guided search.** In SymStep, the LLM acts as a natural-language branching oracle while the propagator handles consistency and cascades implied deductions. Neither suffices alone: CoT without propagation gives 0% on LGP-14, and a propagator alone cannot extract facts from natural language. The feedback loop (generate, verify, cascade, hint) is more powerful than either component alone.

**Cost and efficiency.** SymStep+G uses $\approx 7$ LLM calls per problem versus 1 for Direct/CoT, but $\approx 60\%$ of assignments are derived by cascade automatically. At $\approx \$0.0013$/puzzle (Haiku), the full LGP-14 benchmark costs under two cents. The meaningful comparison is cost *per correct solution*: Logic-LM's single call costs $\approx \$0.0006$/attempt but achieves 0%, making its effective cost per correct answer unbounded.

# 7 Conclusion

LLMs mainly fail on constraint-dense reasoning because unverified errors accumulate unchecked. **SymStep** fixes this by verifying every deduction before the next begins, achieving 83–100% across six benchmarks (LGP-14, SP-6, MWP-8, FIN-6, ZebraLogicBench, AR-LSAT) and outperforming Logic-LM, the strongest prior symbolic+LLM baseline; on ZebraLogicBench, a published 1,000-puzzle benchmark never seen during design, SymStep+G achieves 97%.

Wilson CIs confirm significance: SymStep+G's lower bound strictly exceeds all baselines' upper bounds on both LGP-14 (79% vs. 22%, $n$=14) and ZebraLogicBench (85% vs. 10%, $n$=35).

# References


[Achiam *et al.*, 2023] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, et al. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[Besta *et al.*, 2024] Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, and Torsten Hoefler. Graph of thoughts: Solving elaborate problems with large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 17682–17690, 2024.

[Cobbe *et al.*, 2021] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

[d'Avila Garcez and Lamb, 2023] Artur S. d'Avila Garcez and Luís C. Lamb. Neurosymbolic AI: The 3rd wave. *Artificial Intelligence Review*, 56:12387–12406, 2023.

[Guo *et al.*, 2025] Daya Guo, Dejian Yang, Haowei Zhang, et al. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

[Han *et al.*, 2024] Simeng Han, Hailey Schoelkopf, Yilun Zhao, Zhenting Qi, Martin Riddell, Wenfei Zhou, Arman Cohan, Dragomir Radev, et al. FOLIO: Natural language reasoning with first-order logic. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2024.

[Hao *et al.*, 2023] Shibo Hao, Yi Gu, Haodi Ma, Joshua Jiahua Hong, Zhen Wang, Daisy Zhe Wang, and Zhiting Hu. Reasoning with language model is planning with world model. *arXiv preprint arXiv:2305.14992*, 2023. EMNLP 2023.

[Hoffmann and Nebel, 2001] Jörg Hoffmann and Bernhard Nebel. The FF planning system: Fast plan generation through heuristic search. *Journal of Artificial Intelligence Research*, 14:253–302, 2001.

[Kambhampati *et al.*, 2024] Subbarao Kambhampati, Karthik Valmeekam, Lin Guan, Mudit Verma, Kaya Stechly, Siddhant Bhambri, Lucas Saldyt, and Anil Murthy. Position: LLMs can't plan, but can help

planning in LLM-Modulo frameworks. *arXiv preprint arXiv:2402.01817*, 2024. ICML 2024.

[Kojima *et al.*, 2022] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems 35 (NeurIPS 2022)*, 2022.

[Lightman *et al.*, 2024] Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations (ICLR 2024)*, 2024.

[Lin *et al.*, 2025] Bill Yuchen Lin, Ronan Le Bras, Kyle Richardson, Ashish Sabharwal, Radha Poovendran, Peter Clark, and Yejin Choi. ZebraLogic: On the scaling limits of LLMs for logical reasoning. In *Proceedings of the Forty-Second International Conference on Machine Learning (ICML 2025)*, 2025. arXiv:2502.01100.

[Ling *et al.*, 2023] Zhan Ling, Yunhao Fang, Xuanlin Li, Zhiao Huang, Mingu Lee, Roland Memisevic, and Hao Su. Deductive verification of chain-of-thought reasoning. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*, 2023. arXiv:2306.03872.

[Mackworth, 1977] Alan K. Mackworth. Consistency in networks of relations. *Artificial Intelligence*, 8(1):99–118, 1977.

[Madaan *et al.*, 2023] Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*, 2023.

[Olausson *et al.*, 2023] Theo X. Olausson, Alex Gu, Benjamin Lipkin, Cedegao E. Zhang, Armando Solar-Lezama, Joshua B. Tenenbaum, and Roger Levy. LINC: A neurosymbolic approach for logical reasoning by combining language models with first-order logic provers. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5153–5176, 2023. Outstanding Paper Award.

[Pan *et al.*, 2023] Liangming Pan, Alon Albalak, Xinyi Wang, and William Yang Wang. Logic-LM: Empowering large language models with symbolic solvers for faithful logical reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 3806–3824, 2023.

[Shinn *et al.*, 2023] Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*, 2023.

[Snell *et al.*, 2024] Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally is more effective than scaling model parameters for reasoning. *arXiv preprint arXiv:2408.03314*, 2024.

[Tafjord *et al.*, 2021] Oyvind Tafjord, Bhavana Dalvi, and Peter Clark. ProofWriter: Generating implications, proofs, and abductive statements over natural language. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3621–3634, 2021.

[Valmeekam *et al.*, 2024] Karthik Valmeekam, Kaya Stechly, and Subbarao Kambhampati. LLMs still can't plan; can LRMs? a preliminary evaluation of OpenAI's o1 on PlanBench. *arXiv preprint arXiv:2409.13373*, 2024.

[Wang *et al.*, 2024] Wenguan Wang, Yi Yang, and Fei Wu. Towards data- and knowledge-driven artificial intelligence: A survey on neuro-symbolic computing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024. arXiv:2210.15889.

[Wei *et al.*, 2022] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems 35 (NeurIPS 2022)*, 2022.

[Yao *et al.*, 2023] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*, 2023.

[Zhou *et al.*, 2023] Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, and Ed Chi. Least-to-most prompting enables complex reasoning in large language models. In *International Conference on Learning Representations (ICLR 2023)*, 2023.

# A Related Work

**Chain-of-thought and multi-step reasoning.** The most popular approach for multi-step reasoning with LLMs is chain-of-thought (CoT) prompting [Wei *et al.*, 2022], with extensions ranging from zero-shot variants [Kojima *et al.*, 2022] and Least-to-Most decomposition [Zhou *et al.*, 2023] to tree- and graph-structured search [Yao *et al.*, 2023a; Besta *et al.*, 2024]. These methods share one common limitation: they generate reasoning traces but provide no mechanism to check whether any given step is factually valid. PAL [Gao *et al.*, 2023] and Program-of-Thought [Chen *et al.*, 2023] address arithmetic by offloading computation to Python, but logical deduction at the step level remains unverified. Other iterative strategies such as ReAct [Yao *et al.*, 2023b], Self-Ask [Press *et al.*, 2023], RAP [Hao *et al.*, 2023], Thought Cloning [Hu and Clune, 2023], and MCTS-guided search [Zhao *et al.*, 2023] add reasoning loops, yet none introduce step-level *symbolic* verification. The rich evaluation literature on math reasoning [Cobbe *et al.*, 2021; Hendrycks *et al.*, 2021], commonsense tasks [Talmor *et al.*, 2019; Clark *et al.*, 2018; Geva *et al.*, 2021; Sakaguchi *et al.*, 2021; Zellers *et al.*, 2019; Yang *et al.*, 2018], and multi-task suites [Suzgun *et al.*, 2022; Srivastava *et al.*, 2022] has revealed that constraint-dense problems are a common bottleneck for these approaches.

**Step-level verification.** Several recent works confirm that verifying reasoning *as it unfolds* is more powerful than checking a completed trace, yet none achieves the deterministic soundness SymStep introduces. DVCOTR [Ling *et al.*, 2023] decomposes CoT into "Natural Program" steps that the LLM itself verifies, but the verifier is the same model that made the claim, so errors can go undetected and still accumulate. Faithful CoT [Lyu *et al.*, 2023] translates a completed CoT trace to a symbolic solver post hoc, but by the time checking occurs, errors have already propagated through subsequent steps. Self-Refine [Madaan *et al.*, 2023] and Reflexion [Shinn *et al.*, 2023] iterate on LLM self-critique, which produces useful surface-level corrections but remains logically ungrounded.

**Neuro-symbolic integration.** The neuro-symbolic tradition, spanning tensor product representations [Smolensky, 1990], neural-symbolic cognitive architectures [d'Avila Garcez *et al.*, 2012], and recent surveys [Wang *et al.*, 2024; d'Avila Garcez and Lamb, 2023; Yan *et al.*, 2025; Sakhovskiy *et al.*, 2024], aims to combine the flexibility of learned models with the reliability of formal reasoning. Our propagator draws directly on classical formal methods: arc-consistency [Mackworth, 1977], description logics [Baader *et al.*, 2003], and first-order resolution [Robinson, 1965]. The two most closely related systems, **Logic-LM** [Pan *et al.*, 2023] and LINC [Olausson *et al.*, 2023], represent the whole-problem end of this spectrum: both ask the LLM to produce a complete formal program (Z3 or FOL) from a natural-language description in one shot, then hand it off to a solver. This brittle translation achieves 0% on LGP-14 and SP-6. Language-to-logic translation has also been explored for temporal logic [Fuggitti and Chakraborti, 2023] and structured knowledge bases [Bollacker *et al.*, 2008; Berant *et al.*, 2013]. LLM-Modulo [Kambhampati *et al.*, 2024] makes the conceptual case for pairing LLMs with external critics; SymStep instantiates this principle at the finest possible granularity—one claim, one check—with a deterministic symbolic propagator.

**Large reasoning models and test-time compute.** Recent LRMs (o1 [Valmeekam *et al.*, 2024], DeepSeek-R1 [Guo *et al.*, 2025]) and test-time compute scaling [Snell *et al.*, 2024] extend CoT by allocating more inference-time computation, yet their scratchpads remain unverified. SymStep is complementary: it provides a *sound* symbolic oracle that deterministically checks every intermediate claim, independent of model scale. SymStep is evaluated on Claude Haiku and Sonnet; the broader landscape includes GPT-4 [Achiam *et al.*, 2023], Gemini [Anil *et al.*, 2023], and LLaMA [Touvron *et al.*, 2023].

**Classical planning and constraint satisfaction.** Sequential search under constraints underpins classical AI planning: STRIPS [Fikes and Nilsson, 1971], HTN [Erol *et al.*, 1996], PDDL [McDermott *et al.*, 1998], Graphplan [Blum and Furst, 1997], and LAMA [Richter and Westphal, 2010]. Compilability results [Nebel, 2000] and the situation calculus [McCarthy and Hayes, 1969] formalize constraint semantics. Applied to LLMs, planning benchmarks [Valmeekam *et al.*, 2023b; Valmeekam *et al.*, 2023a; Valmeekam *et al.*, 2024] show consistent failure without symbolic support; LLM+PDDL [Liu *et al.*, 2023] and generalized planning [Silver *et al.*, 2024] partially address this. LLMs also power embodied agents [Huang *et al.*, 2024; Ahn *et al.*, 2022; Wang *et al.*, 2023a; Wang *et al.*, 2023b; Song *et al.*, 2023] on benchmarks such as ALFRED [Shridhar *et al.*, 2020], ALFWorld [Shridhar *et al.*, 2021], and VirtualHome [Puig *et al.*, 2018].

**LLM agents, tools, and code.** Tool-augmented LLMs [Schick *et al.*, 2023; Shen *et al.*, 2023; Cai *et al.*, 2023] and agent architectures [Sumers *et al.*, 2024; Liu *et al.*, 2024; Yang *et al.*, 2023] extend LLMs with external execution. Code-as-policy [Liang *et al.*, 2023; Singh *et al.*, 2023] and code generation [Le *et al.*, 2022; Li *et al.*, 2022] show that structured outputs with execution feedback improve reliability. Inner Monologue [Huang *et al.*, 2022] provides environment feedback to the LLM analogously to our propagator; SymStep specializes this feedback loop for *logical consistency* with a sound verifier.

# B Limitations and Future Work

**Scaling and model coverage.** SymStep has been validated on 119 instances spanning six independent domains and two model families; extending to FOLIO [Han *et al.*, 2024] and ProofWriter [Tafjord *et al.*, 2021] is a natural next step, and LGP-20 (20 verified puzzles) is released for this purpose. Initial Sonnet results (90% on LGP-10, §C) are encouraging; a clean cross-model study on GPT-4 [Achiam *et al.*, 2023], open-weight models [Touvron *et al.*, 2023], and large reasoning models (o1, DeepSeek-R1 [Guo *et al.*, 2025]) will clarify whether the gains are model-agnostic.

**Extending the symbolic backbone.** The propagator handles bijective constraints—a broad class covering grid

puzzles, scheduling, and selection problems—with elegant arc-consistency propagation. Temporal ordering, counting, and conditionals are well-studied in constraint programming; adding SAT/ASP [Gelfond and Lifschitz, 1988] backends would extend SymStep to these domains, and the outer loop and DEDUCE protocol are already backend-agnostic. The SymStep+G regression on SP-6 ($n$=6, not statistically significant) raises an interesting open question: learned or adaptive guidance [Lightman *et al.*, 2024] that adjusts hint aggressiveness based on constraint density could close this gap and clarify when MRV direction is beneficial versus disruptive.

## C Cross-Model Results on LGP-10 (Sonnet)

Table 11 compares Haiku and Sonnet on LGP-10 (the ten core puzzles shared between both runs). Both SymStep and SymStep+G achieve **90%** on Sonnet, confirming that the architecture is not specific to one model family. Sonnet requires no guidance to match Haiku+guidance, suggesting that stronger models are better at variable selection without external hints.

| Method | Haiku | Sonnet | Haiku avg calls |
|---|---|---|---|
| Direct | 20% | 0%† | 1.0 |
| CoT | 0% | 0%† | 1.0 |
| Self-Refine | 30% | 0%† | 2.0 |
| SymStep | 80% | **90%** | 6.2 |
| **SymStep+G** | **100%** | **90%** | 6.7 |

Table 11: Cross-model comparison on LGP-10. †Sonnet baseline figures reflect a now-corrected parser limitation (calibrated for Haiku's concise output; misses Sonnet's verbose formatting). SymStep solutions always derive from the propagator state and are unaffected.

## D Per-Puzzle Breakdown (LGP-14)

| Level | Puzzles | Entities | Attrs | Clues |
|---|---|---|---|---|
| Easy | E1-Pets, E2-Jobs, E3-Sport, E4-Music | 3 | 2 | 4–5 |
| Medium | M1-Houses, M2-Nationals, M3-Hobby, M4-PetFruit, M5-Cities | 4 | 2 | 6–8 |
| Hard | H1-Full, H2-Full, H3-5People, H4-Full, H5-Full | 4–5 | 2–3 | 8–10 |

Table 12: LGP-14 benchmark structure. All 14 solutions are verified for uniqueness. M2-Nationals uses 8 clues (disambiguated). LGP-20 adds 6 more puzzles: E5, E6 (easy, 3-entity), M6, M7 (medium, 4-entity), H6, H7 (hard, 4–5 entities).

Table 13 shows binary outcomes for every method on every LGP-14 puzzle. CoT achieves 0% universally across all 14 puzzles. Self-Refine excels only on harder puzzles (H1, H2, H4, H5), where the self-critique pass covers more clues; it fails on easy/medium puzzles, where the LLM rationalizes wrong answers rather than catching errors. SymStep stalls on M1 and H2 (deductive deadlock); SymStep+G resolves both.

Figure 2 visualizes the per-difficulty breakdown.

| Puzzle | Dir | CoT | SR | SS | SS+G |
|---|---|---|---|---|---|
| E1-Pets | ✗ | ✗ | ✗ | ✓ | ✓ |
| E2-Jobs | ✓ | ✗ | ✗ | ✓ | ✓ |
| E3-Sport | ✓ | ✗ | ✗ | ✓ | ✓ |
| E4-Music | ✗ | ✗ | ✗ | ✓ | ✓ |
| M1-Houses | ✗ | ✗ | ✗ | ✗ | ✓ |
| M2-Nationals | ✗ | ✗ | ✗ | ✓ | ✓ |
| M3-Hobby | ✗ | ✗ | ✗ | ✓ | ✓ |
| M4-PetFruit | ✗ | ✗ | ✓ | ✓ | ✓ |
| M5-Cities | ✗ | ✗ | ✗ | ✓ | ✓ |
| H1-Full | ✗ | ✗ | ✓ | ✓ | ✓ |
| H2-Full | ✗ | ✗ | ✓ | ✗ | ✓ |
| H3-5People | ✗ | ✗ | ✗ | ✓ | ✓ |
| H4-Full | ✗ | ✗ | ✓ | ✓ | ✓ |
| H5-Full | ✗ | ✗ | ✓ | ✓ | ✓ |
| **Total (14)** | 2 | 0 | 5 | 12 | **14** |

Table 13: Per-puzzle binary outcomes on LGP-14 (Haiku). Dir=Direct, SR=Self-Refine, SS=SymStep, SS+G=SymStep+G. ✓=correct, ✗=incorrect.

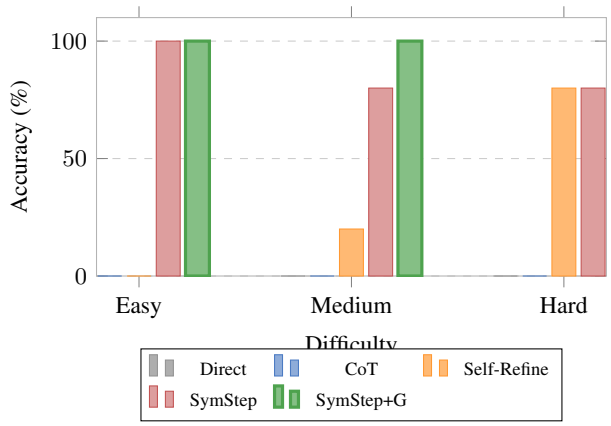


Figure 2: Accuracy (%) per difficulty on LGP-14 (Haiku). SymStep+G (green) achieves 100% at every level; CoT (blue) stays at 0%. Easy (4 puzzles), Medium (5), Hard (5).

## E AQUA-RAT: Domain Boundary Analysis

AQUA-RAT [Ling *et al.*, 2017] is a dataset of algebra word problems in multiple-choice format. We evaluate on a 35-problem sample to characterize the boundary of SymStep's advantage. Unlike grid puzzles and scheduling problems, AQUA-RAT questions require arithmetic reasoning over free-form algebraic relationships without the bijective constraint structure that the propagator exploits.

SymStep+G matches CoT at 89%, while the unguided variant underperforms (66%), since the structured DEDUCE protocol imposes overhead not compensated by propagation benefits on open-ended algebra MCQ. This confirms the domain boundary: SymStep's advantage is specific to tasks where constraints are dense and arc-consistency cascades contribute meaningfully to the solution.

| Puzzle | Dir | CoT | SR | SS | SS+G |
|---|---|---|---|---|---|
| SP-E1-Workshop | ✗ | ✗ | ✓ | ✓ | ✓ |
| SP-E2-Lab | ✗ | ✗ | ✗ | ✓ | ✓ |
| SP-M1-Projects | ✗ | ✗ | ✓ | ✓ | ✗ |
| SP-M2-Research | ✗ | ✗ | ✗ | ✓ | ✗ |
| SP-H1-Dept | ✗ | ✗ | ✗ | ✗ | ✓ |
| SP-H2-EventCrew | ✗ | ✗ | ✗ | ✓ | ✓ |
| **Total (6)** | 0 | 0 | 2 | 5 | 4 |

Table 14: Per-puzzle outcomes on SP-6 (Haiku). SymStep+G fails on SP-M1 and SP-M2 (medium puzzles with a natural sequential ordering that MRV hints disturb), while SymStep succeeds.

| Method | Acc. | Total |
|---|---|---|
| Direct | 86% | 30/35 |
| CoT | 89% | 31/35 |
| SymStep | 66% | 23/35 |
| **SymStep+G** | 89% | 31/35 |

Table 15: Results on AQUA-RAT algebra MCQ (35 problems, Haiku). SymStep+G matches CoT (both 89%), confirming that SymStep's advantage is constraint-density-specific: where bijective constraint propagation is not the bottleneck, the symbolic verifier adds no benefit over standard reasoning.

# F Formal Properties of the Constraint Propagator

**Theorem 1** (Consistency)**.** *Every assignment $\sigma(p, a) = v$ accepted by the propagator is consistent with all previously accepted deductions. The possibility matrix $\Pi$ satisfies: $v \in \Pi(p, a)$ iff $v$ has not been ruled out by any accepted deduction at any prior step.*

*Proof.* By induction on the deduction sequence. *Base*: Initially, $\Pi(p, a) = V_a$; the invariant holds trivially. *Step*: APPLY returns OK only if $v \in \Pi(p, a)$, ensuring consistency. Arc-consistency propagation discards $v$ from $\Pi(p', a)$ for all $p' \neq p$, maintaining bijectivity. Any empty domain triggers CONTRADICTION, so no inconsistent state is ever accepted. □

**Theorem 2** (Propagation completeness)**.** *If $|\Pi(p, a)| = 1$ for some $(p, a)$ after accepting a deduction, the propagator derives this assignment automatically without an additional LLM call. The cascade reaches an arc-consistency fixed point in $O(n^2k)$ steps.*

*Proof.* Each cascade step strictly reduces the total domain size $\sum_{p,a} |\Pi(p, a)|$, which is bounded below by $nk$; thus, the cascade terminates. The $O(n^2k)$ bound follows from standard arc-consistency analysis [Mackworth, 1977; Hoffmann and Nebel, 2001]. □

**Soundness caveat.** The propagator guarantees *consistency*, not *soundness*. A consistent-but-incorrect deduction (valid against the prior state but not entailed by the clues) can still be accepted. The residual failure rate on hard puzzles reflects exactly this gap.

| Method | Verifier type | Granularity | Guidance | Multi-domain |
|---|---|---|---|---|
| Logic-LM [Pan *et al.*, 2023] | Symbolic solver | Whole problem | – | – |
| LINC [Olausson *et al.*, 2023] | FOL prover | Whole problem | – | – |
| DVCOTR [Ling *et al.*, 2023] | LLM self-check | Per step | – | – |
| Self-Refine [Madaan *et al.*, 2023] | LLM self-check | Whole output | Vague | – |
| **SymStep** | Propagator | Per step | – | ✓ |
| **SymStep+G** | Propagator | Per step | **MRV** | ✓ |

Table 16: Method comparison. SymStep+G uniquely combines a deterministic per-step verifier, proactive MRV guidance, and a domain-agnostic design.

**Algorithm 1** SymStep and SymStep+G

**Input:** Puzzle $\mathcal{P}$, LLM $\mathcal{M}$, budget $T$, flag GUIDE $\in$ {true,false}
**Output:** Assignment $\hat{\sigma}$ or FAIL

```
Initialize propagator Π with Π(p, a) ← V_a
h ← ⟨puzzle text, DEDUCE format instructions⟩
for t = 1 to T do
    d_t ← M(h)                                  {one deduction}
    if CONCLUDE ∈ d_t or Π.SOLVED() then
        return σ̂(Π)
    end if
    (status, msg) ← Π.APPLY(d_t)
    if status = OK then
        fb ← "✓ Verified. msg"
        if GUIDE then
            fb ← fb+ MRVHINT(Π)                 {SymStep+G}
        end if
    else
        fb ← "× Contradiction: msg. Please revise."
    end if
    Append fb to h
end for
return FAIL
```

# G LGP-20 Benchmark Description

LGP-20 adds six verified puzzles to LGP-14. Solutions are available in the released code. Difficulty mirrors LGP-14: **E5-Colors** (3 entities, 2 attrs, 4 clues), **E6-School** (3 entities, 2 attrs, 5 clues), **M6-Travel** (4 entities, 2 attrs, 8 clues), **M7-Food** (4 entities, 2 attrs, 7 clues), **H6-Full** (4 entities, 3 attrs, 10 clues), and **H7-5People** (5 entities, 2 attrs, 9 clues).

# H Full Puzzle Descriptions

All fourteen LGP-14 puzzles are listed below with their complete clue sets; solutions are in Table 17.

**Easy puzzles. E1-Pets**: Alice, Bob, Carol; {Red, Blue, Green} × {Cat, Dog, Fish}; 5 clues including "Bob lives in the blue house" and "The person in the red house has a cat." **E2-Jobs**: Adam, Beth, Chris; {Tea, Coffee, Water} × {Doctor, Teacher, Engineer}; 5 clues including "The doctor drinks coffee." **E3-Sport**: Lena, Milo, Nora; {Tennis, Swim, Run} × {Paris, London, Tokyo}; 4 clues including "The person in London swims" and "Lena is not in Tokyo." **E4-Music**: Jake, Kim, Leo; {Piano, Guitar, Drums} × {Mon, Tue, Wed}; 4 clues including "The piano player practices on Wednesday."

**Medium puzzles.** **M1-Houses**: Alice, Bob, Carol, Dave; $4 \times \{$color, drink$\}$; 6 clues. **M2-Nationals**: Ana, Ben, Carl, Donna; $4 \times \{$nationality, pet$\}$; 8 clues (disambiguated). **M3-Hobby**: Ivan, Jana, Karl, Lea; $4 \times \{$color, hobby$\}$; 6 clues. **M4-PetFruit**: Omar, Petra, Quinn, Rosa; $4 \times \{$pet, fruit$\}$; 7 clues. **M5-Cities**: Alice, Bob, Carol, Dave; $4 \times \{$city, sport$\}$; 7 clues.

**Hard puzzles.** **H1-Full**: Alex, Blake, Casey, Drew; $4 \times \{$color, drink, pet$\}$; 9 clues. **H2-Full**: Pam, Quinn, Rosa, Sam; $4 \times \{$color, job, pet$\}$; 9 clues. **H3-5People**: Alex, Beth, Carl, Dana, Eric; $5 \times \{$color, drink$\}$; 8 clues. **H4-Full**: Kim, Lee, Mia, Ned; $4 \times \{$color, job, pet$\}$; 10 clues. **H5-Full**: Ash, Bay, Cole, Drew; $4 \times \{$color, drink, pet$\}$; 10 clues.

| Puzzle | Entities | Solution (Attr1 / Attr2 / Attr3) |
|---|---|---|
| E1 | Alice / Bob / Carol | Red+Cat / Blue+Dog / Green+Fish |
| E2 | Adam / Beth / Chris | Tea+Eng. / Coffee+Doctor / Water+Teacher |
| E3 | Lena / Milo / Nora | Tennis+Paris / Swim+London / Run+Tokyo |
| E4 | Jake / Kim / Leo | Guitar+Mon / Drums+Tue / Piano+Wed |
| M1 | Alice / Bob / Carol / Dave | Green+Tea / Red+Juice / Blue+Coffee / Yellow+Water |
| M2 | Ana / Ben / Carl / Donna | English+Cat / German+Fish / Spanish+Bird / French+Dog |
| M3 | Ivan / Jana / Karl / Lea | Blue+Dance / Green+Read / White+Paint / Red+Cook |
| M4 | Omar / Petra / Quinn / Rosa | Cat+Cherry / Parrot+Date / Snake+Banana / Fish+Apple |
| M5 | Alice / Bob / Carol / Dave | London+Run / Berlin+Tennis / Paris+Swim / Rome+Soccer |
| H1 | Alex / Blake / Casey / Drew | Green+Tea+Bird / Red+Juice+Dog / Blue+Coffee+Fish / Yellow+Water+Cat |
| H2 | Pam / Quinn / Rosa / Sam | Red+Teacher+Fish / Blue+Chef+Bird / Yellow+Doctor+Cat / Green+Eng.+Dog |
| H3 | Alex / Beth / Carl / Dana / Eric | White+Tea / Blue+Milk / Red+Coffee / Green+Juice / Yellow+Water |
| H4 | Kim / Lee / Mia / Ned | Red+Doctor+Cat / Green+Lawyer+Bird / Yellow+Teacher+Fish / Blue+Chef+Dog |
| H5 | Ash / Bay / Cole / Drew | Red+Tea+Cat / Blue+Coffee+Bird / Green+Juice+Dog / Yellow+Water+Fish |

Table 17: Ground-truth solutions for all LGP-14 puzzles.

# I System Prompts

All methods receive the same **problem header**:

```
LOGIC PUZZLE
People: {people}
Attributes: {attr (vals)}
Clues: {numbered clue list}
```

**Direct** appends: "Provide your final answer as: [person]: attr=value, ..."

**CoT** appends: "Think step by step through the clues to determine each person's attributes. Show your reasoning, then give the final answer."

**Self-Refine**: the first call is Direct; the second call appends the first answer and asks: "Carefully verify each clue against your answer. List any violated clues, correct them, and provide the revised final answer."

**SymStep / SymStep+G** append after the problem header:

> You are solving a logic problem by making one deduction at a time.
> Each deduction must follow EXACTLY one of these formats:
> `DEDUCE: <Person>, <attribute>, <Value>`
> `DEDUCE: <Person>, <attribute>, NOT <Value>`
> After each deduction, you will receive verification feedback.
> When the problem is fully solved, output: `CONCLUDE: done`
> Do not output multiple deductions at once.

Verifier feedback:

- **Accepted**: `[Verifier]: ✓ Correct deduction accepted.` (followed by an MRV hint if SymStep+G is used: `[Hint] X's attr must be one of: {...}`)
- **Contradiction**: `[Verifier]: ✗ CONTRADICTION --- {exact message}. Please reconsider.`

# J Example SymStep+G Transcript

The following is an abridged transcript of SymStep+G on puzzle E1-Pets (solution: Alice=Red+Cat, Bob=Blue+Dog, Carol=Green+Fish).

> **LLM:** `DEDUCE: Bob, color, Blue`
> **Verifier:** ✓ Verified. [Hint] Alice's color must be one of: {Red, Green}
>
> **LLM:** `DEDUCE: Carol, color, Green`
> **Verifier:** ✓ Verified. [Hint] Alice's color must be one of: {Red}
>
> **LLM:** `DEDUCE: Alice, color, Red`
> **Verifier:** ✓ Verified. [Hint] Alice's pet must be one of: {Cat}
>
> **LLM:** `DEDUCE: Alice, pet, Cat`
> **Verifier:** ✓ Verified. [Hint] Bob's pet must be one of: {Dog, Fish}
>
> **LLM:** `DEDUCE: Carol, pet, Fish`
> **Verifier:** ✓ Verified. [Problem solved!] Output CONCLUDE: done
>
> **LLM:** `CONCLUDE: done`

After Step 2, the arc-consistency cascade automatically narrows Alice's color to {Red}; the LLM never explicitly deduced this, yet the hint reveals it. This cascade efficiency is the key benefit of unit propagation: ≈60% of assignments are derived automatically, without requiring separate LLM calls.

# Full Bibliography


[Achiam *et al.*, 2023] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, et al. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[Ahn *et al.*, 2022] Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Daniel Ho, et al. Do as I can, not as I say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022.

[Anil *et al.*, 2023] Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, et al. Gemini: A family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

[Baader *et al.*, 2003] Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003.

[Berant *et al.*, 2013] Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic parsing on freebase

from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1533–1544, 2013.

[Besta *et al.*, 2024] Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, and Torsten Hoefler. Graph of thoughts: Solving elaborate problems with large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 17682–17690, 2024.

[Blum and Furst, 1997] Avrim L. Blum and Merrick L. Furst. Fast planning through planning graph analysis. *Artificial Intelligence*, 90:281–300, 1997.

[Bollacker *et al.*, 2008] Kurt D. Bollacker, Colin Evans, Praveen K. Paritosh, Tim Sturge, and Jamie Taylor. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 1247–1250, 2008.

[Cai *et al.*, 2023] Tianle Cai, Xuezhi Wang, Tengyu Ma, Xinyun Chen, and Denny Zhou. Large language models as tool makers. *arXiv preprint arXiv:2305.17126*, 2023.

[Chen *et al.*, 2023] Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W. Cohen. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *Transactions on Machine Learning Research*, 2023.

[Clark *et al.*, 2018] Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

[Cobbe *et al.*, 2021] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

[d'Avila Garcez and Lamb, 2023] Artur S. d'Avila Garcez and Luís C. Lamb. Neurosymbolic AI: The 3rd wave. *Artificial Intelligence Review*, 56:12387–12406, 2023.

[d'Avila Garcez *et al.*, 2012] Artur S. d'Avila Garcez, Luís C. Lamb, and Dov M. Gabbay. *Neural-Symbolic Cognitive Reasoning*. Cognitive Technologies. Springer, 2012.

[Erol *et al.*, 1996] Kutluhan Erol, James Hendler, and Dana S. Nau. Complexity results for hierarchical task-network planning. *Annals of Mathematics and Artificial Intelligence*, 18:69–93, 1996.

[Fikes and Nilsson, 1971] Richard E. Fikes and Nils J. Nilsson. STRIPS: A new approach to the application of theorem proving to problem solving. In *Proceedings of the Second International Joint Conference on Artificial Intelligence (IJCAI-71)*, pages 608–620, London, 1971.

[Fuggitti and Chakraborti, 2023] Francesco Fuggitti and Tathagata Chakraborti. NL2LTL – a python package for converting natural language (NL) instructions to linear temporal logic (LTL) formulas. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 16428–16430, 2023.

[Gao *et al.*, 2023] Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. PAL: Program-aided language models. In *Proceedings of the 40th International Conference on Machine Learning (ICML 2023)*, 2023.

[Gelfond and Lifschitz, 1988] Michael Gelfond and Vladimir Lifschitz. The stable model semantics for logic programming. In *Proceedings of the International Logic Programming Conference and Symposium*, pages 1070–1080. MIT Press, 1988.

[Geva *et al.*, 2021] Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361, 2021.

[Guo *et al.*, 2025] Daya Guo, Dejian Yang, Haowei Zhang, et al. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

[Han *et al.*, 2024] Simeng Han, Hailey Schoelkopf, Yilun Zhao, Zhenting Qi, Martin Riddell, Wenfei Zhou, Arman Cohan, Dragomir Radev, et al. FOLIO: Natural language reasoning with first-order logic. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2024.

[Hao *et al.*, 2023] Shibo Hao, Yi Gu, Haodi Ma, Joshua Jiahua Hong, Zhen Wang, Daisy Zhe Wang, and Zhiting Hu. Reasoning with language model is planning with world model. *arXiv preprint arXiv:2305.14992*, 2023. EMNLP 2023.

[Hendrycks *et al.*, 2021] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *Advances in Neural Information Processing Systems 34 (NeurIPS 2021)*, 2021.

[Hoffmann and Nebel, 2001] Jörg Hoffmann and Bernhard Nebel. The FF planning system: Fast plan generation through heuristic search. *Journal of Artificial Intelligence Research*, 14:253–302, 2001.

[Hu and Clune, 2023] Shengran Hu and Jeff Clune. Thought cloning: Learning to think while acting by imitating human thinking. *arXiv preprint arXiv:2306.00323*, 2023. NeurIPS 2023.

[Huang *et al.*, 2022] Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, Pierre Sermanet, Noah Brown, Tomas Jackson, Linda

Luu, Sergey Levine, Karol Hausman, and Brian Ichter. Inner monologue: Embodied reasoning through planning with language models. In *Conference on Robot Learning (CoRL 2022)*, 2022.

[Huang *et al.*, 2024] Xu Huang, Weiwen Liu, Xiaolong Chen, Xingmei Wang, Hao Wang, Defu Lian, Yasheng Wang, Ruiming Tang, and Enhong Chen. Understanding the planning of LLM agents: A survey. *arXiv preprint arXiv:2402.02716*, 2024.

[Kambhampati *et al.*, 2024] Subbarao Kambhampati, Karthik Valmeekam, Lin Guan, Mudit Verma, Kaya Stechly, Siddhant Bhambri, Lucas Saldyt, and Anil Murthy. Position: LLMs can't plan, but can help planning in LLM-Modulo frameworks. *arXiv preprint arXiv:2402.01817*, 2024. ICML 2024.

[Kojima *et al.*, 2022] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems 35 (NeurIPS 2022)*, 2022.

[Le *et al.*, 2022] Hung Le, Yue Wang, Akhilesh Deepak Gotmare, Silvio Savarese, and Steven C.H. Hoi. CodeRL: Mastering code generation through pretrained models and deep reinforcement learning. In *Advances in Neural Information Processing Systems 35 (NeurIPS 2022)*, 2022.

[Li *et al.*, 2022] Yujia Li, David Choi, Junyoung Chung, Nati Kushman, Julian Schrittwieser, Rémi Leblond, Tom Eccles, James Keeling, Felix Gimeno, Alessio Dal Lago, et al. Competition-level code generation with AlphaCode. *Science*, 378(6624):1092–1097, 2022.

[Liang *et al.*, 2023] Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. Code as policies: Language model programs for embodied control. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023.

[Lightman *et al.*, 2024] Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations (ICLR 2024)*, 2024.

[Lin *et al.*, 2025] Bill Yuchen Lin, Ronan Le Bras, Kyle Richardson, Ashish Sabharwal, Radha Poovendran, Peter Clark, and Yejin Choi. ZebraLogic: On the scaling limits of LLMs for logical reasoning. In *Proceedings of the Forty-Second International Conference on Machine Learning (ICML 2025)*, 2025. arXiv:2502.01100.

[Ling *et al.*, 2017] Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. Program induction by rationale generation: Learning to solve and explain algebraic word problems. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL 2017)*, pages 158–167, 2017.

[Ling *et al.*, 2023] Zhan Ling, Yunhao Fang, Xuanlin Li, Zhiao Huang, Mingu Lee, Roland Memisevic, and Hao Su. Deductive verification of chain-of-thought reasoning. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*, 2023. arXiv:2306.03872.

[Liu *et al.*, 2023] Bo Liu, Yuqian Jiang, Xiaohan Zhang, Qiang Liu, Shiqi Zhang, Joydeep Biswas, and Peter Stone. LLM+P: Empowering large language models with optimal planning proficiency. *arXiv preprint arXiv:2304.11477*, 2023.

[Liu *et al.*, 2024] Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, et al. AgentBench: Evaluating LLMs as agents. In *The Twelfth International Conference on Learning Representations (ICLR 2024)*, 2024.

[Lyu *et al.*, 2023] Qing Lyu, Shreya Havaldar, Adam Stein, Li Zhang, Delip Rao, Eric Wong, Marianna Apidianaki, and Chris Callison-Burch. Faithful chain-of-thought reasoning. In *Proceedings of the 13th International Joint Conference on Natural Language Processing (IJCNLP-AACL 2023)*, 2023.

[Mackworth, 1977] Alan K. Mackworth. Consistency in networks of relations. *Artificial Intelligence*, 8(1):99–118, 1977.

[Madaan *et al.*, 2023] Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*, 2023.

[McCarthy and Hayes, 1969] John McCarthy and Patrick J. Hayes. Some philosophical problems from the standpoint of artificial intelligence. In Bernard Meltzer and Donald Michie, editors, *Machine Intelligence 4*, pages 463–502. Edinburgh University Press, 1969.

[McDermott *et al.*, 1998] Drew McDermott, Malik Ghallab, Adele Howe, Craig Knoblock, Ashwin Ram, Manuela Veloso, Daniel Weld, and David Wilkins. PDDL – the planning domain definition language. Technical Report CVC TR-98-003 / DCS TR-1165, Yale Center for Computational Vision and Control, 1998.

[Nebel, 2000] Bernhard Nebel. On the compilability and expressive power of propositional planning formalisms. *Journal of Artificial Intelligence Research*, 12:271–315, 2000.

[Olausson *et al.*, 2023] Theo X. Olausson, Alex Gu, Benjamin Lipkin, Cedegao E. Zhang, Armando Solar-Lezama, Joshua B. Tenenbaum, and Roger Levy. LINC: A neurosymbolic approach for logical reasoning by combining language models with first-order logic provers. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5153–5176, 2023. Outstanding Paper Award.

[Pan *et al.*, 2023] Liangming Pan, Alon Albalak, Xinyi Wang, and William Yang Wang. Logic-LM: Empowering large language models with symbolic solvers for

faithful logical reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 3806–3824, 2023.

[Press *et al.*, 2023] Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A. Smith, and Mike Lewis. Measuring and narrowing the compositionality gap in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, 2023.

[Puig *et al.*, 2018] Xavier Puig, Kevin Ra, Marko Boben, Jiaman Li, Tingwu Wang, Sanja Fidler, and Antonio Torralba. VirtualHome: Simulating household activities via programs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2018)*, pages 8494–8502, 2018.

[Richter and Westphal, 2010] Silvia Richter and Matthias Westphal. The LAMA planner: Guiding cost-based anytime planning with landmarks. *Journal of Artificial Intelligence Research*, 39:127–177, 2010.

[Robinson, 1965] J. Alan Robinson. A machine-oriented logic based on the resolution principle. *Journal of the ACM*, 12(1):23–41, 1965.

[Sakaguchi *et al.*, 2021] Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

[Sakhovskiy *et al.*, 2024] Andrey Sakhovskiy, Mikhail Salnikov, Irina Nikishina, Aida Usmanova, Angelie Kraft, Cedric Möller, Debayan Banerjee, Junbo Huang, Longquan Jiang, Rana Abdullah, Xi Yan, Dmitry Ustalov, Elena Tutubalina, Ricardo Usbeck, and Alexander Panchenko. TextGraphs 2024 shared task on text-graph representations for knowledge graph question answering. In *Proceedings of TextGraphs-17: Graph-based Methods for NLP*, pages 97–112, 2024.

[Schick *et al.*, 2023] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*, 2023.

[Shen *et al.*, 2023] Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. HuggingGPT: Solving AI tasks with ChatGPT and its friends in HuggingFace. *arXiv preprint arXiv:2303.17580*, 2023.

[Shinn *et al.*, 2023] Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*, 2023.

[Shridhar *et al.*, 2020] Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. ALFRED: A benchmark for interpreting grounded instructions for everyday tasks. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[Shridhar *et al.*, 2021] Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. ALFWorld: Aligning text and embodied environments for interactive learning. In *International Conference on Learning Representations (ICLR 2021)*, 2021.

[Silver *et al.*, 2024] Tom Silver, Soham Dan, Kavitha Srinivas, Joshua B. Tenenbaum, Leslie Pack Kaelbling, and Michael Katz. Generalized planning in PDDL domains with pretrained large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 20256–20264, 2024.

[Singh *et al.*, 2023] Ishika Singh, Valts Blukis, Arsalan Mousavian, Ankit Goyal, Danfei Xu, Jonathan Tremblay, Dieter Fox, Jesse Thomason, and Animesh Garg. ProgPrompt: Generating situated robot task plans using large language models. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11523–11530, 2023.

[Smolensky, 1990] Paul Smolensky. Tensor product variable binding and the representation of symbolic structures in connectionist systems. *Artificial Intelligence*, 46:159–216, 1990.

[Snell *et al.*, 2024] Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally is more effective than scaling model parameters for reasoning. *arXiv preprint arXiv:2408.03314*, 2024.

[Song *et al.*, 2023] Chan Hee Song, Jiaman Wu, Clayton Washington, Brian M. Sadler, Wei-Lun Chao, and Yu Su. LLM-Planner: Few-shot grounded planning for embodied agents with large language models. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

[Srivastava *et al.*, 2022] Aarohan Srivastava, Abigail Rastogi, Abhishek Rao, et al. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv preprint arXiv:2206.04615*, 2022.

[Sumers *et al.*, 2024] Theodore R. Sumers, Shunyu Yao, Karthik Narasimhan, and Thomas L. Griffiths. Cognitive architectures for language agents. *Transactions on Machine Learning Research*, 2024.

[Suzgun *et al.*, 2022] Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V. Le, Denny Zhou, William Fedus, et al. Challenging BIG-Bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*, 2022.

[Tafjord *et al.*, 2021] Oyvind Tafjord, Bhavana Dalvi, and Peter Clark. ProofWriter: Generating implications, proofs, and abductive statements over natural language. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3621–3634, 2021.

[Talmor *et al.*, 2019] Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. CommonsenseQA: A question answering challenge targeting

commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2019)*, pages 4149–4158, 2019.

[Touvron *et al.*, 2023] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Armand Joulin, Edouard Grave, and Guillaume Lample. LLaMA: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[Valmeekam *et al.*, 2023a] Karthik Valmeekam, Matthew Marquez, Alberto Olmo, Sarath Sreedharan, and Subbarao Kambhampati. PlanBench: An extensible benchmark for evaluating large language models on planning and reasoning about change. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*, 2023.

[Valmeekam *et al.*, 2023b] Karthik Valmeekam, Matthew Marquez, Sarath Sreedharan, and Subbarao Kambhampati. On the planning abilities of large language models: A critical investigation. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*, 2023.

[Valmeekam *et al.*, 2024] Karthik Valmeekam, Kaya Stechly, and Subbarao Kambhampati. LLMs still can't plan; can LRMs? a preliminary evaluation of OpenAI's o1 on PlanBench. *arXiv preprint arXiv:2409.13373*, 2024.

[Wang *et al.*, 2023a] Guanzhi Wang, Yuanzhuo Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*, 2023.

[Wang *et al.*, 2023b] Zihao Wang, Shaofei Cai, Guanzhou Chen, Anji Liu, Xiaojian Ma, and Yitao Liang. Describe, explain, plan and select: Interactive planning with large language models enables open-world multi-task agents. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*, 2023.

[Wang *et al.*, 2024] Wenguan Wang, Yi Yang, and Fei Wu. Towards data- and knowledge-driven artificial intelligence: A survey on neuro-symbolic computing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024. arXiv:2210.15889.

[Wei *et al.*, 2022] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems 35 (NeurIPS 2022)*, 2022.

[Yan *et al.*, 2025] Xi Yan, Aida Usmanova, Cedric Möller, Patrick Westphal, and Ricardo Usbeck. Neuro-symbolic relation extraction. In *Handbook on Neurosymbolic AI and Knowledge Graphs*, volume 400 of *Frontiers in Artificial Intelligence and Applications*, pages 550–576. IOS Press, 2025.

[Yang *et al.*, 2018] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

[Yang *et al.*, 2023] John Yang, Akshara Prabhakar, Karthik Narasimhan, and Shunyu Yao. InterCode: Standardizing and benchmarking interactive coding with execution feedback. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*, 2023.

[Yao *et al.*, 2023a] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*, 2023.

[Yao *et al.*, 2023b] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations (ICLR 2023)*, 2023.

[Zellers *et al.*, 2019] Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*, 2019.

[Zhao *et al.*, 2023] Zirui Zhao, Wee Sun Lee, and David Hsu. Large language models as commonsense knowledge for large-scale task planning. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*, 2023.

[Zhou *et al.*, 2023] Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, and Ed Chi. Least-to-most prompting enables complex reasoning in large language models. In *International Conference on Learning Representations (ICLR 2023)*, 2023.