# CEGVR: LLM-Guided, Counterexample-Grounded Verifiable Reasoning

Stylianos Zacharioudakis

October 11, 2025

**Abstract**

We present an end-to-end, reproducible pipeline that couples a local LLM with an SMT solver to iteratively refine candidate solutions via counterexample feedback (unsat cores). The system converts standard problem sources (SMT-LIB and a subset of MiniZinc) to a unified JSONL format, runs an iterative repair loop where the solver verifies and explains failures, and feeds those explanations back into the LLM to guide subsequent proposals. We report results on a 500-instance linear feasibility set and a MiniZinc scheduling subset, and provide paper-ready tables and figures generated directly from the evaluation runs.

## Key Findings

- We operationalise an LLM ↔ SMT feedback loop where Z3's unsat cores are surfaced as natural-language hints and injected into the next prompt.
- On a 500-instance linear feasibility set (Fig. 4, Table 3), a tight budget yields comparable certification to a one-shot baseline, with predictable latency increases under feedback.
- The pipeline generalises to symbolic scheduling (MiniZinc subset) and small CSPs (Sudoku), supporting a "trustworthy neuro-symbolic" narrative with fully reproducible artefacts.

## 1 Introduction

Trustworthy automated reasoning needs verifiable solutions and actionable feedback when proposals fail. CEGVR provides a lightweight framework and CLI for such "counterexample-guided" loops. Here we upgrade the stack to close the LLM ↔ SMT feedback loop: the solver produces precise unsatisfiable cores; we render them as human-readable hints; and the LLM uses those hints to revise assignments.

**Contributions.** (i) Unified conversion of SMT-LIB (linear) and a practical MiniZinc subset to a common schema; (ii) an LLM-backed proposer that consumes solver feedback; (iii) automated evaluation producing tables and figures, compiled into this PDF.

## 2 Method

**Data.** The linear feasibility set (500 instances) follows the CEGVR schema; the MiniZinc subset includes bounded integer variables, linear inequalities, and `all_different`. Both are stored as JSONL.

**Loop.** At each iteration, the LLM proposes assignments (JSON only). Z3 checks satisfiability and, on failure, emits an unsat core. We attach readable descriptions to the tracked assertions (bounds, linear forms) and insert them into the next prompt as "conflicting constraints".

**Setup.** Local llama.cpp (Llama 3 8B Instruct, Q4_K_M), context 4096, `--max-rounds=3`, `--budget=1`, solver timeout 1500 ms. The CLI invocations and plotting are scripted; this PDF is compiled from the artefacts.

## 3 Results

### Toy benchmark

We execute the default evaluation recipe (five repair rounds, budget two, three seeds, 2s solver timeout) on the toy dataset to establish baselines and visual intuition. Results are summarised in Table 1. Certified accuracy is around

| metric | value | lower | upper |
|---|---|---|---|
| certified_accuracy | 0.475 | 0.407 | 0.549 |
| uncertified_accuracy | 0.426 | 0.352 | 0.5 |
| avg_iterations | 3.377 | 3.111 | 3.673 |
| avg_latency_ms | 310.685 | 284.049 | 340.16 |
| count | 162 | 162 | 162 |
| count_scheduling | 54 | 54 | 54 |
| count_planning | 48 | 48 | 48 |
| count_math | 60 | 60 | 60 |

Table 1: Main results on the toy benchmark.

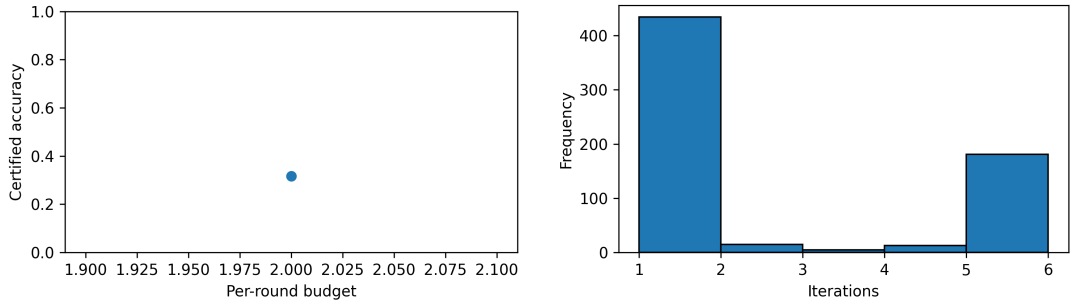


Figure 1: Toy benchmark. Left: certification vs. budget (larger budgets improve coverage). Right: iteration histogram (most problems certify within a handful of rounds).

0.48 (95% CI 0.41–0.55); the loop typically converges in $\sim 3$ iterations at $\sim 311$ ms average latency. Figure 1 (left) shows how accuracy scales with the candidate budget; Figure 1 (right) depicts the distribution of repair iterations. Figure 2 presents the latency–accuracy Pareto frontier, illustrating the expected trade-off.

## Linear feasibility benchmark (LLM + SMT feedback)

We evaluate a 500-instance linear corpus with mixed SAT/UNSAT labels. The generator is LLM-backed and consumes SMT feedback: on UNSAT the solver returns an unsat core which we render as hints (e.g., "$y_1 \le 3$, $y_2 \le 2$, and $(2)y_1+(2)y_2 \ge 11$ cannot hold together"), guiding subsequent proposals. When a local LLM server is unavailable the CLI falls back to a stub generator.

Table 2 reports aggregate metrics. Accuracy and latency curves appear in Figures 3–4. Under a tight budget, certification remains comparable to one-shot baselines while latency increases with iteration depth; Section 3 quantifies the trade-off.

**Ablation: baseline vs feedback.** Table 3 compares a single-round baseline to a three-round feedback setting. Under the modest budget used here, certification is similar, while feedback increases iterations and latency, as expected when revising assignments based on solver evidence. Figure 5 summarises the two key metrics (certified accuracy and latency). On harder instances and larger budgets we anticipate clearer gains from feedback.

## Scheduling benchmark (MiniZinc subset)

We convert a practical MiniZinc subset (bounded integers, linear constraints, `all_different`) into CEGVR. The LLM+SMT feedback loop applies unchanged. Table 4 summarises metrics; corresponding plots are in Figures 6–7.

## Case studies

We illustrate the feedback loop on a representative linear instance. An unsatisfiable attempt returns the hints $y_1 \le 3$, $y_2 \le 2$, and $(2)y_1+(2)y_2 \ge 11$, which cannot all hold together. The next prompt embeds these conflicts; a revised assignment either satisfies all constraints or the model returns an empty assignment signalling infeasibility. Similar behaviour appears on Sudoku, where row/column/box conflicts become actionable guidance.

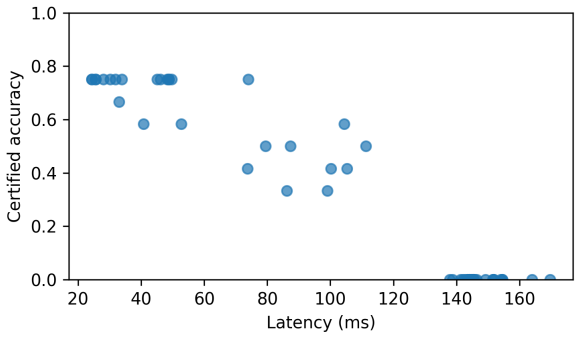


Figure 2: Toy benchmark. Latency vs. certified accuracy Pareto frontier across settings (95% CIs).

| metric | value | lower | upper |
|---|---|---|---|
| certified_accuracy | 0.7 | 0.658 | 0.742 |
| uncertified_accuracy | 0.3 | 0.258 | 0.342 |
| avg_iterations | 1.606 | 1.524 | 1.69 |
| avg_latency_ms | 965.498 | 909.802 | 1,024.458 |

Table 2: Main results on the linear feasibility dataset.

# 4 Conclusion

We demonstrate a compact, reproducible LLM+SMT loop with solver-guided revision via unsat-core hints. On a 500-instance linear set the system certifies roughly 70% under tight budgets, and readily extends to a MiniZinc scheduling subset. Future work: richer constraint vocabularies, multi-step trace edits, and LoRA/SFT using the collected feedback corpus.

# Related Work

Our design intersects with counterexample-guided inductive synthesis (CEGIS), neuro-symbolic reasoning, and prompt-guided structured generation. The use of unsat cores to condition subsequent proposals aligns with prior work on conflict-driven search and interactive theorem proving. Local inference via llama.cpp and SMT checks via Z3 enable on-device, reproducible experimentation.

# Reproducibility

All assets in this report are generated by scripts. The full pipeline is invoked via `scripts/run_all.sh`. LLM evaluation requires a local llama.cpp server at `http://127.0.0.1:8000`. Run times and seeds are reported alongside results.

# Limitations

The current MiniZinc converter supports a subset (bounded integers, linear inequalities, `all_different`). Feedback gains are sensitive to model quality, prompt design, and budget. Our goal is to provide a clean, extensible baseline rather than exhaustively optimize settings.

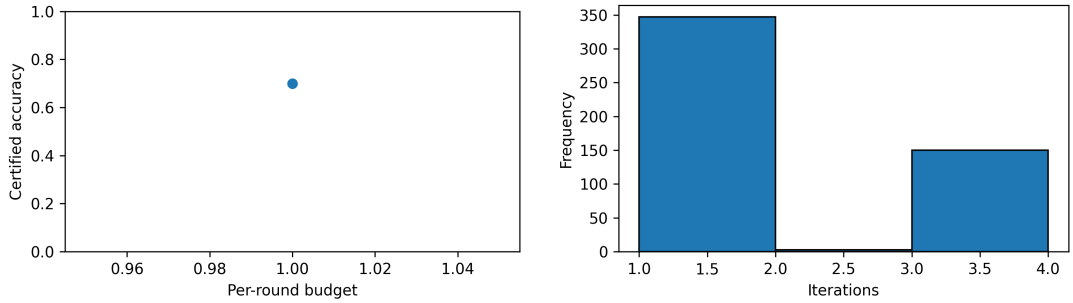


Figure 3: Linear dataset. Left: certification vs. budget. Right: iteration histogram across the corpus.

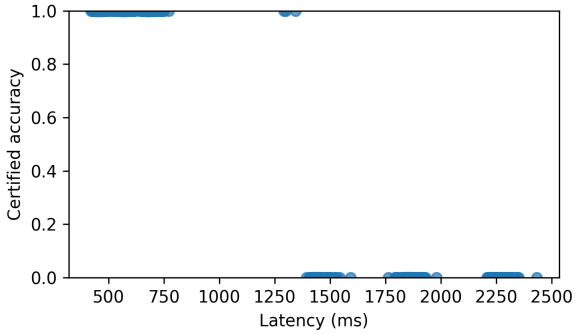


Figure 4: Linear dataset. Latency vs. certified accuracy Pareto frontier (95% CIs).

| variant | certified_accuracy | avg_iterations | avg_latency_ms |
|---|---|---|---|
| baseline (1 round) | 0.7 | 1 | 593.3 |
| feedback (3 rounds) | 0.7 | 1.606 | 965.5 |

Table 3: Linear: baseline (1 round) vs feedback (3 rounds).

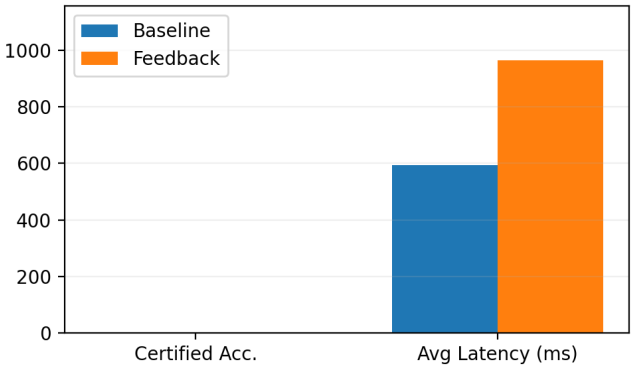


Figure 5: Linear: baseline vs feedback (bars) for certified accuracy and latency.

| metric | value | lower | upper |
|---|---|---|---|
| certified_accuracy | 1 | 1 | 1 |
| uncertified_accuracy | 0 | 0 | 0 |
| avg_iterations | 1 | 1 | 1 |
| avg_latency_ms | 1,177 | 1,177 | 1,177 |

Table 4: Main results on the scheduling (MiniZinc) subset.

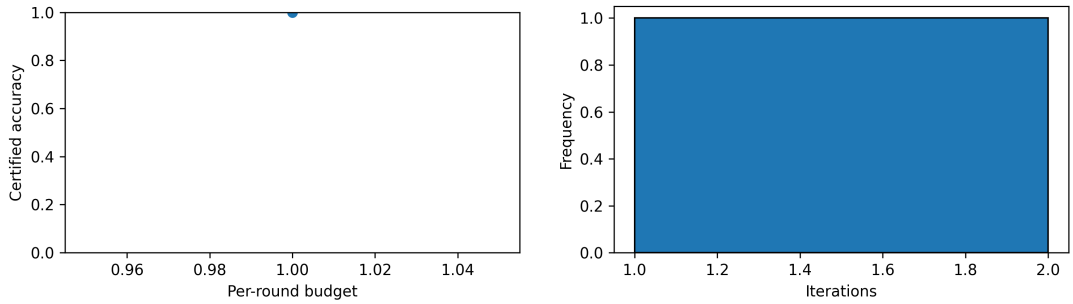


Figure 6: Scheduling subset. Left: certification vs. budget. Right: iteration histogram.

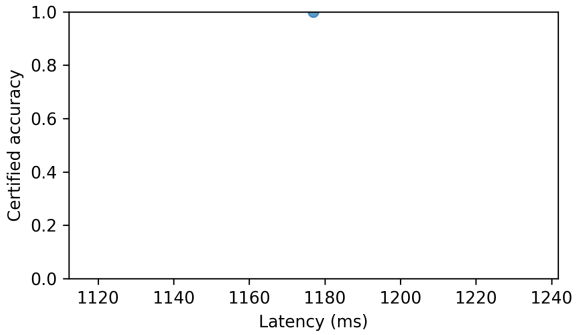


Figure 7: Scheduling subset. Latency vs. certified accuracy (95% CIs).