Mechanics of Learned Reasoning 1: TempoBench, A Benchmark for Interpretable Deconstruction of Reasoning System Performance

NIKOLAUS HOLZER, Columbia University, USA
WILLIAM FISHELL, Columbia University, USA
BAISHAKHI RAY, Columbia University, USA
MARK SANTOLUCITO, Columbia University, Barnard College, USA

ACM Reference Format:
Nikolaus Holzer, William Fishell, Baishakhi Ray, and Mark Santolucito. 2025. Mechanics of Learned Reasoning 1: TempoBench, A Benchmark for Interpretable Deconstruction of Reasoning System Performance. 1, 1 (November 2025), 26 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

Abstract

Large Language Models (LLMs) are increasingly excelling and outpacing human performance on many tasks. However, to improve LLM reasoning, researchers either rely on ad-hoc generated datasets or formal mathematical proof systems such as the Lean proof assistant. Whilst ad-hoc generated methods can capture the decision chains of real-world reasoning processes, they may encode some inadvertent bias in the space of reasoning they cover; they also cannot be formally verified. On the other hand, systems like Lean can guarantee verifiability, but are not well-suited to capture the nature of agentic decision chain-based tasks. This creates a gap both in performance for functions such as business agents or code assistants, and in the usefulness of LLM reasoning benchmarks, whereby these fall short in reasoning structure or real-world alignment. We introduce TempoBench, the first formally grounded and verifiable diagnostic benchmark that parametrizes difficulty to systematically analyze how LLMs perform reasoning. TempoBench uses two evaluation benchmarks to break down reasoning ability. First, temporal trace evaluation (TTE) tests the ability of an LLM to understand and simulate the execution of a given multi-step reasoning system. Subsequently, temporal causal evaluation (TCE) tests an LLM's ability to perform multi-step causal reasoning and to distill cause-and-effect relations from complex systems. We find that models score 65.6% on TCE-normal, and 7.5% on TCE-hard. This shows that state-of-the-art LLMs clearly understand the TCE task but perform poorly as system complexity increases. Our code is available at our GitHub repository.

1 Introduction

The ability to reason over temporal traces and causality is a core reasoning task in many industry applications. AWS deploys automata-based reasoning commercially to automate the analysis of

Authors' Contact Information: Nikolaus Holzer, holzer@cs.columbia.edu, Columbia University, USA; William Fishell, wf2322@columbia.edu, Columbia University, USA; Baishakhi Ray, rayb@cs.columbia.edu, Columbia University, USA; Mark Santolucito, msantolu@barnard.edu, Columbia University, Barnard College, USA.

ACM XXXX-XXXX/2025/11-ART
https://doi.org/10.1145/nnnnnnn.nnnnnnn

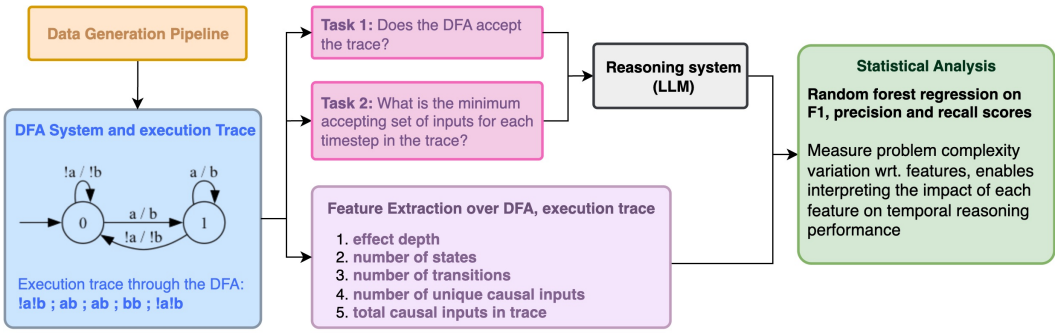


Fig. 1. Overview of the TempoBench framework. TempoBench includes 5 key features for modeling temporal problem difficulty and uses them to conduct rigorous statistical analysis of reasoning performance. TempoBench consists of two tasks: Temporal Trace Evaluation (TTE) and Temporal Causality Evaluation (TCE)

.

permissions, such as access control lists [4]. Causality is also used for root cause analysis, for example, in microservices failures [27, 36], showing that many real-world reasoning problems reduce to temporal causality-like problems [46]. Furthermore, these types of problems remain highly relevant in natural language settings as well, where temporal reasoning is prevalent, especially in agentic systems and workflows [5, 10, 47, 52].

In this work, we introduce TempoBench, focusing on temporal reasoning, a cornerstone capability of agentic workflows, complex long-horizon reasoning, and a core component of Artificial General Intelligence (AGI) [8]. It is essential to have benchmarks that go beyond leaderboards and isolate components of temporal reasoning, such as problem complexity, information density, and horizon depth, to examine the impact of various confounding factors. TempoBench is a diagnostically focused **formally grounded synthesis framework** that isolates and quantifies, with statistical significance, the structural factors that make reasoning tasks difficult. When generating tasks, we can control synthesis parameters to ensure the tasks have a quantifiable measure of reasoning difficulty. There is a commonly held belief that the difficulty of a reasoning system scales strictly with its size. Our findings add nuance to this belief; specifically, we find that reasoning over systems with larger problem spaces and more logical transitions does increase complexity. However, our results also demonstrate that, in many cases, larger systems with more transitions have a higher density of connections between states, making it easier for LLMs to track chains of reasoning through them.

TempoBench contains two tasks shown in Figure 1: Temporal Trace Evaluation (TTE), which requires an LLM to determine whether a given automaton accepts a trace of inputs, and Temporal Causality Evaluation (TCE), in which the LLM retroactively examines the system's behavior and determines the necessary set of counterfactuals for a given output. Figure 3 illustrates one of the reasoning tasks present in TempoBench.

Current benchmarks cannot isolate the underlying difficulty space of their tasks in a deterministic and verified framework. TempoBench deterministically generates reasoning tasks with a verifiable optimal solution for each task. We check that LLMs can find the optimal solution for these tasks. Improving on the interpretability of agent reasoning, through a framework like TempoBench, will be essential in improving agent performance and increasing the adoption rate of LLM agents in real-world deployments [40]. To the best of our knowledge, we are the first to provide a dataset

of exclusively formally verified synthetic temporal reasoning problems that vary in structural complexity in a controlled way.

Our key contributions in this work are:

(1) **Verifiable temporal causality in complex real-world systems.** We provide the first dataset of fully verified temporal reasoning problems for LLM evaluation, based on real-world systems.
(2) **End-to-end controllable synthesis framework** for generating data and evaluating LLMs that leverages reactive synthesis and causality analysis
(3) **Empirical evaluation of LLM reasoning limits.** We demonstrate that even state-of-the-art LLMs exhibit negative scaling with increasing problem complexity, revealing systematic gaps in current reasoning abilities.

2 Related Work

2.1 LLM Reasoning Systems

Recent advances in LLM-powered reasoning agents have driven research on benchmarking and LLM reasoning capabilities. Methods such as Chain of Thought (CoT) [49], self-consistency [47], and tree or graph-structured reasoning [5, 52] improve accuracy on reasoning benchmarks by leveraging intermediate steps or tool use [10]. Other reasoning approaches leverage reinforcement learning (RL) to teach LLMs reasoning semantics and procedures, such as human feedback [35], or reasoning traces collected over synthetic algorithmic coding problems [17, 20, 21]. While such benchmarks are useful for holistic model evaluations, they are less amenable to dissecting the structure of reasoning or the factors that determine task difficulty. Evaluations often rely on aggregate accuracy or qualitative trace inspection rather than systematic analysis of reasoning complexity. In contrast, TempoBench grounds reasoning evaluation in formal, parameterized systems, enabling quantitative study of how reasoning performance scales with structural problem features.

2.2 Real-world performance of LLM agents

Despite their successes in agentic benchmarks, an increasing number of empirical studies highlight the lack of trust in LLM reasoning systems as a primary hindrance to widespread adoption in commercial applications [16, 31, 40]. Empirical studies report that concerns about reliability, and the subsequent need to review code generated through LLM agents manually, are slowing developers down [40], harming productivity and AI agent adoption. These works underscore the need to explore structural reasoning behavior and identify interpretable failure modes in LLM reasoning, making it easier to identify areas of improvement and enabling targeted training towards better, more reliable reasoning models.

2.3 Reasoning system benchmarks

An increasing number of benchmarks focus on causal reasoning agents. These include mathematical problem reasoning [14], code reasoning over constrained competition style problems or directory level issues [9, 30], challenging human-designed logic puzzles [11, 12, 42] or generated quasi-formal benchmarks that incorporate verifiable structures [22]. Yet, before this work, there was no verifiable causal reasoning benchmark for temporal tasks.

2.4 Temporal Reasoning Benchmarks

Temporal reasoning benchmarks are difficult to curate because of the complexity of extracting temporal transitions from real-world systems and of verifying ground-truth causal relationships.

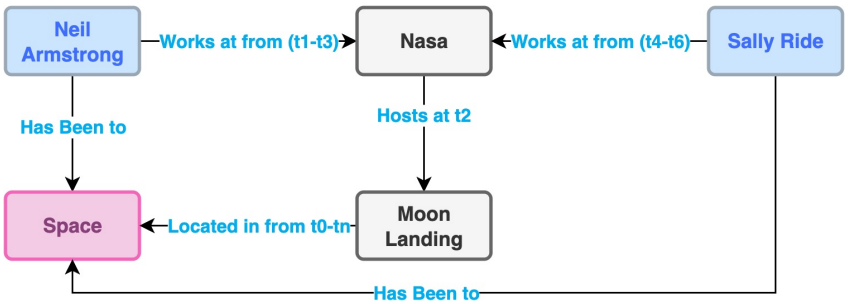


Fig. 2. Sample knowledge graph showcasing relationship inference. In this case, asked to determine who has been to the moon, the LLM is highly likely to have prior knowledge of this fact, showing a deficiency of temporal benchmarks.

State-of-the-art benchmarks are generated through three primary methods: LLM-generated synthetic temporal knowledge graphs, assigning temporal structures to randomly generated graphs, and human-curated data sets [6, 23, 44, 45, 48, 50]. Temporal knowledge graphs, as shown in Figure 2, are a widely used approach for representing relationships among entities over time. They are easy to construct and check, forming the basis for benchmarks such as TGQA [50]. These graphs are generated from real-world or synthetic LLM data. Despite their popularity, they present four limitations for evaluating temporal reasoning:

(1) **Graphs built from real-world data often enable models to rely on prior knowledge**, making it difficult to isolate an LLM's temporal reasoning ability [23].
(2) **Relationships are overly simplistic**, an edge between two nodes denotes a relation, but real systems often involve multiple interacting inputs and outputs that are not all causally linked.
(3) **Task difficulty is hard to quantify**, as it depends simultaneously on linguistic, symbolic, and temporal complexity.
(4) **Synthetic data generated by LLMs lacks verifiability** and typically requires human validation.

TestofTime (ToT) [23] is a graph-based benchmark designed to mitigate these issues. ToT uses randomly generated graphs where edges represent temporal relationships drawn from a predefined set. The resulting structures are verifiable and can be tuned via parameters such as graph size and edge count to create more challenging settings. However, ToT still faces key limitations:

(1) **Causal relationships cannot be inferred** from the graph construction.
(2) **The graph structures are not representative of real-world systems**, such as Mealy machines.
(3) **Performance remains dependent on the symbolic interpretation** of abstract graph structures.

TempoBench addresses these challenges by generating temporal traces from automata synthesized from formal specifications describing real-world systems—such as arbiters and controllers [43]. Temporal logic yields temporally rich, verifiable data. Furthermore, causal relationships are explicitly synthesized for outputs, ensuring that causal inputs are formally validated. TempoBench uses the HOA file structure, a structured, interpretable formalism, to reduce reliance on pure symbolic reasoning. Ultimately, these formalisms provide more insight into the difficulty of the variant temporal-reasoning tasks our benchmark addresses.

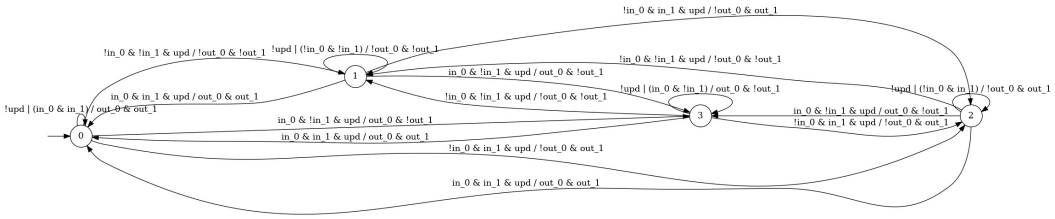
3 TempoBench

TempoBench is a benchmark for assessing an LLM's ability to perform key temporal reasoning tasks. We use reactive systems specified by temporal logics such as linear temporal logic (LTL) and synthesized using reactive synthesis [38] to examine various system behaviors and tease out the temporal aspects of complex systems (See appendix A.1 for more information on reactive synthesis and LTL). The benchmark and tool are available on GitHub[1].

3.1 Task Formulation

To rigorously assess temporal reasoning, TempoBench focuses on two core tasks: Temporal Trace Evaluation (TTE), which measures a model's ability to determine whether a sequence satisfies temporal constraints, and Temporal Causality Evaluation (TCE), which tests its ability to infer causal dependencies over time.

(a) Computer-generated visualisation of the system



(b) Visualisation of the inputs generating a trace over the system and the causal explanations of the effect

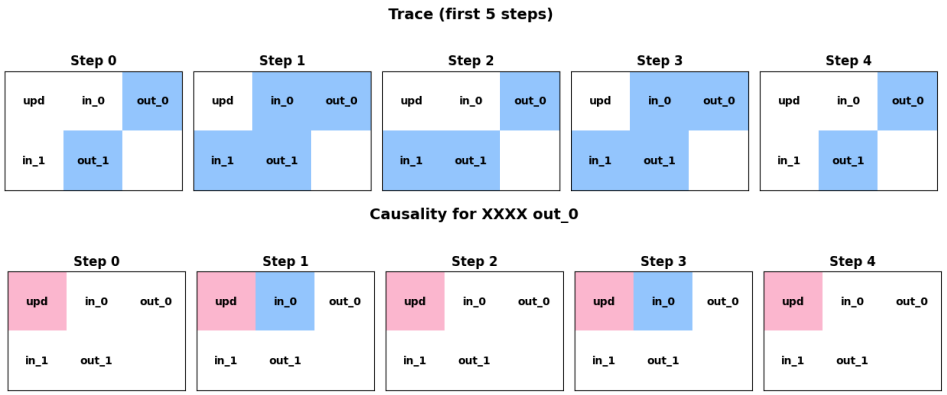


Fig. 3. Sample visualization of a tempo-bench problem. This example shows a trace through a system and a causal explanation of what caused `out_0` at step 4. A correct solution on this benchmark identifies the causal effects of `XXXX out_0`. Light pink for negative constraints (-1). White for neutral (0). Light blue for positive constraints (+1)

(1) **Temporal Trace Evaluation (TTE)**. Given a finite-state machine $A = (Q, \Sigma, \delta, q_0, F)$ and a finite trace π, determine whether π is accepted by A. To do so, walk through the transitions of the state machine, and at each time step T_i, verify whether the set of inputs and outputs $I \cup O$ is accepted by A. This task unifies elements of runtime verification [32] and world modeling [19].

[1] https://github.com/nik-hz/tempobench

The model must determine whether the observed behavior satisfies the temporal constraints imposed by the system; it must also reason over the system's transition dynamics sufficiently to interpret the current state and navigate subsequent transitions within that structured world. The TTE task is a good measure of how well models perform retrieval from their inputs, as they must accurately parse and apply the HOA's state transitions. See Appendix D.1 for the HOA file representation and its automaton counterpart. As such, we use the TTE task as a litmus test of how well the models understand the HOA file format for state machines and natural language representations of their traces. Bad performance on this task suggests that the models are confused at the language level and that problem difficulty is biased by complex representations rather than by structural indicators.

(2) **Temporal Causality Evaluation (TCE).** Given a finite-state machine $A = (Q, \Sigma, \delta, q_0, F)$, a finite trace π generated from A, and an effect $e \in AP^O$ that occurs at some time $T_i \in \pi$ (where AP^O is the set of outputs $\in \Sigma$), generate the causes $c \in AP^I$ at each $T \leq T_i$ that were necessary for e to occur at T_i. At its core, this task tests the model's ability to reason causally through time. (for more information on temporal causality and synthesis of temporal causality see Appendix A.2).

3.2 Data Generation

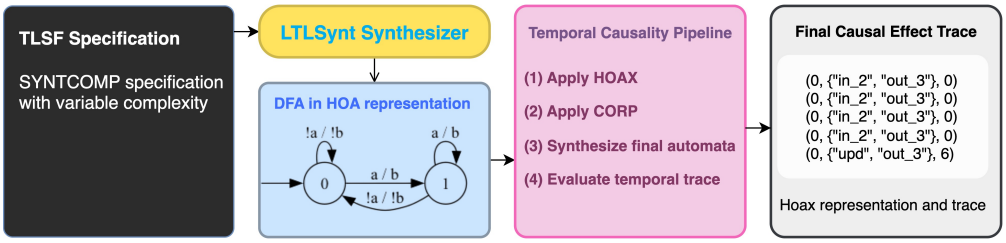


Fig. 4. Pipeline flowchart for data generation in TempoBench. This flowchart illustrates the creation of a formal controller and then the extraction of key data needed to solve problems 1 and 2. A more detailed pipeline visualization is provided in Appendix 13

TempoBench is built on an end-to-end pipeline for generating data for the TTE and TCP tasks (Fig. 4). We use the SYNTCOMP benchmark [43], a large standardized collection of reactive systems specified in temporal logic. Each system is expressed in LTL using the TLSF format [28], which clearly distinguishes inputs, outputs, and their temporal behavior.

(1) **Synthesizing Controllers.** TLSF files are synthesized into controllers using the LTLsynt tool [39] from Spot, producing HOA representations that preserve input–output separation and faithfully implement the temporal logic specification. These HOA controllers provide ideal structures for executing arbitrarily complex temporal logic formulas.

(2) **Generating Finite Traces.** By generating numerous finite traces π, we obtain high-quality, formally validated data for our temporal reasoning tasks. We use the HOAX tool [18] to generate random traces from synthesized systems. The HOAX tool walks through the HOA file, choosing random legal transitions at each time step and recording the history for a specified length T steps.

(3) **Dataset Construction.** This approach allows us to efficiently collect large numbers of correct system executions, supporting the scalable creation of high-quality datasets for TempoBench. We generate an initial dataset of **4,000 TTE traces** and **20,000 TCE traces** using a standard laptop CPU with minimal cost and time.

Listing 1 HOAX generated finite trace over atomic propositions. The first line shows the structure of the tuples. $AP = \{\texttt{in\_0}, \texttt{in\_1}, \texttt{in\_2}, \texttt{in\_3}, \texttt{out\_0}, \texttt{out\_1}, \texttt{out\_2}, \texttt{out\_3}, \texttt{upd}\}$

```
# (current state, {inputs and outputs}, next state)
(0, {"in_2", "in_0", "out_2", "out_1", "out_0", "in_1", "out_3"}, 0)
(0, {"in_2", "in_0", "out_2", "out_1", "out_0", "out_3"}, 0)
(0, {"upd", "out_1", "in_3", "in_1", "out_3"}, 6)
(6, {"in_0", "out_1", "in_3", "in_1", "out_3"}, 6)
(6, {"upd", "in_0", "out_1", "out_0", "in_1"}, 13)
(13, {"upd", "in_2", "in_0", "out_2", "out_1", "out_0", "in_1"}, 15)
```

Consider a finite trace generated from an arbiter as shown in Listing 1. The output illustrates which APs are true at each step and the transitions taken by the controller. This output, alongside the HOA, is used directly for the TTE task. After generating a trace and HOA, a temporal causality controller can be extracted for each output $AP^O \in \pi$.

Given **Causal Inputs:** $\phi_i,\ \phi_j \subseteq AP^I$ and **Effect:** $\mathcal{T}(\phi_o),\ \phi_o \subseteq AP^O,\ \mathcal{T} \in \langle G, F, U, X\rangle$, the synthesis of temporal causality is the synthesis of the minimum controller over the causal inputs that guarantees the monitored effect occurs [15].

Algorithm 1 Causal Output Reconstruction for Finite Trace

Require: Finite trace π generated by controller A

1: **for** each time step T_i in π **do**
2: Extract the output proposition ϕ_o at time T_i
3: Define the *effect:* Effect $\leftarrow XX \cdots X_{T_i}\, \phi_o$
4: CORP$(A, \pi,$ Effect$)$
5: **end for**

We are interested in explaining each output generated by the synthesized controllers. We use the tool Causes for Omega-regular properties (CORP) [24], which is designed to synthesize temporal causality for the various outputs in each trace. Using Algorithm 1, we extract a causal automaton that describes the inputs at each timestep up to T_i that are required for the effect to be observed (See example A.1 and Fig. 12 for more details of the synthesis of temporal causality given a specific output).

3.3 Evaluation Metrics

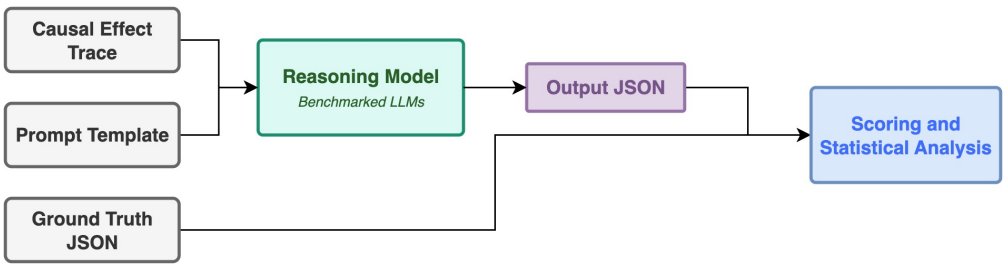


Fig. 5. Pipeline flowchart for the evaluation harness that we use to score reasoning model performance

TempoBench provides a formally guaranteed benchmark set with deterministic ground truth, enabling fine-grained and rigorous statistical analysis through well-defined measures such as

precision, recall, and F_1 scores.

$$\text{Precision} = \frac{TP}{TP + FP} \qquad \text{Recall} = \frac{TP}{TP + FN} \qquad F_1 = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

To evaluate temporal trace acceptance and temporal causality, we define true positives, false positives, and false negatives for each task. In trace acceptance, a true positive occurs when the model correctly predicts the next state via the correct input–output pair; false positives and negatives reflect incorrect transitions or missing valid ones. In temporal causality, predicted causal relations are treated as positives and compared to the provable causal structure. True positives capture correctly identified causal inputs. In contrast, false positives and negatives reflect over- and under-prediction, respectively. These definitions enable balanced evaluation of reasoning accuracy beyond simple correctness, highlighting how LLMs interpret temporal dependencies within automata.

TempoBench exposes five features that let us scale problem difficulty: **effect depth**, the time step at which the effect occurs within the trace; **system states**, the total number of states in the automaton; **transition count**, the number of transitions in the system; **causal inputs count**, the total number of causal inputs throughout the trace (counting repeated occurrences of the same atomic proposition); and **unique inputs in trace**, the number of distinct input propositions appearing in the trace. We can freely set thresholds for classifying problems as normal or hard. To make TempoBench Hard, we select the top n problems with the highest value for each of these features at generation time. These features allow us to condition evaluation on problem structure, such as LLM performance on large state systems. Examining failure modes in light of this analysis helps identify the primary sources of sample difficulty and pinpoint areas for improving temporal reasoning in LLMs.

4 Experimental Setup

4.1 Sample Selection

We use TLSF specifications from SYNTCOMP [29]. Where applicable, we modify specifications to increase the number of atomic propositions, ensuring they remain synthesizable within 5 minutes on our hardware, thereby increasing the diversity and complexity of our problem set and using the pipeline described in Section 3.2.

4.2 LLM selection and prompting

We use TempoBench to dissect reasoning performance in several well-known LLMs. We benchmark GPT-4o-mini [33], GPT-4o [34], Claude-3.5-sonnet [2], Claude-sonnet-4.5 [3], and Qwen3-coder-plus [51]. These models are accessed via their corresponding APIs. Testing the models on 800 samples, 400 for each of the TTE and TCE tasks, we evaluate each model using one-shot prompting [7]. We include an example of a CoT [49] solution strategy in the prompt and evaluate the models using the evaluation metrics detailed in Section 3. Our inputs take the form of JSON objects shown in Listing 2.

Listing 2 Sample JSON for the causal effect determination task

```
# Sample of input to the LLM, describing the atomic propositions of a system
# "aps": ["g", "r"]

# JSON Ground Truth
gt_sample_json = {
      "XXX g": {
        "0": ["no constraints"],
        "1": ["g and r"],
        "2": ["no constraints"],
        "3": ["r"]
        }
    }
```

During evaluation, LLMs must produce exact sentences within the JSON object. Outputting in this format is unlikely to be a problem, as we did not encounter any difficulties with JSON output in experiments. Examples of our one-shot prompt, as well as the formatting, are provided in Appendix 3.

5 Results

The results are broken down into two sections. First, we compare the reasoning performance of various state-of-the-art LLMs on TempoBench. Then we perform statistical analysis on the model scores within TempoBench's feature space.

5.1 Model Results

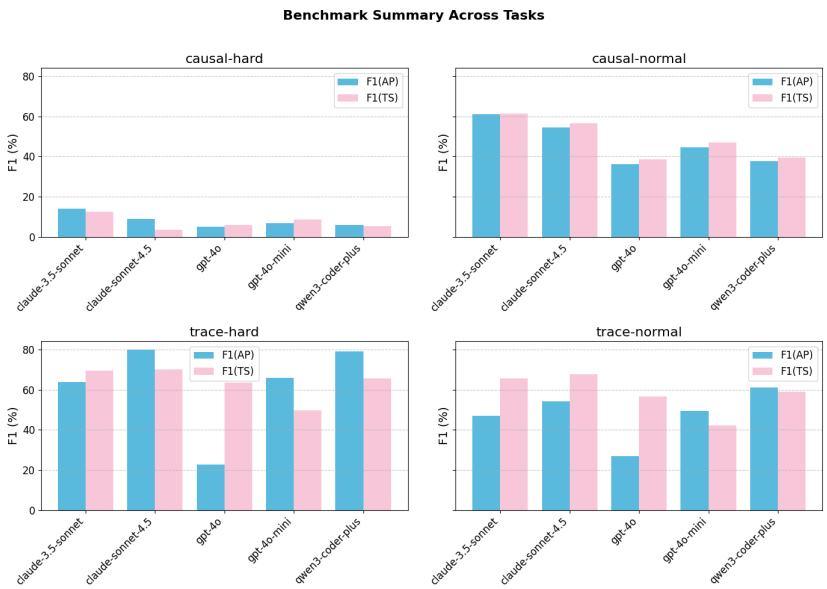


Fig. 6. Visualization of results across all benchmark tasks. TTE normal and hard, TCE normal and hard.

The F_1 scores are measured across the two benchmark tasks, split into hard and normal, at the Atomic Proposition (AP) and Timestep (TS) levels. The F_1 scores are evaluated against the ground truth, as seen in listing 2. F_1(AP) measures the performance of the model at predicting the correct APs within each timestep, allowing partial correctness within a single timestep. F_1(TS) assesses model performance by predicting the whole at each timestep.

Example Illustrating F_1(AP) and F_1(TS). Let the ground-truth trace π and model prediction $\hat{\pi}$ be:

$$\pi = [\{a, b\}, \{b, c\}, \{c\}], \qquad \hat{\pi} = [\{a, c\}, \{b\}, \{c, d\}].$$

F_1(AP) compares individual atomic propositions within each timestep. For instance, overlaps occur at $t_1 : \{a\}$, $t_2 : \{b\}$, $t_3 : \{c\}$, yielding partial matches. Thus, F_1(AP) rewards local overlap: $P = \frac{3}{4}$, $R = 1.0$, $F_1 = 0.86$. F_1(TS) considers each timestep correct only if all APs match exactly. Since none align ($\{a, b\} \neq \{a, c\}$, etc.), F_1(TS)= 0. This captures full temporal correctness rather than partial AP overlap.

Table 1. Benchmark Summary Across Tasks

Model	F_1(AP)	F_1(TS)	Task
anthropic/claude-3.5-sonnet	**14.1%**	**12.5%**	causal-hard
anthropic/claude-sonnet-4.5	8.8%	3.6%	causal-hard
openai/gpt-4o	5.1%	5.9%	causal-hard
openai/gpt-4o-mini	6.7%	8.5%	causal-hard
qwen/qwen3-coder-plus	6.0%	5.4%	causal-hard
anthropic/claude-3.5-sonnet	**61.1%**	**61.5%**	causal-normal
anthropic/claude-sonnet-4.5	54.4%	56.7%	causal-normal
openai/gpt-4o	36.2%	38.6%	causal-normal
openai/gpt-4o-mini	44.6%	47.0%	causal-normal
qwen/qwen3-coder-plus	37.7%	39.6%	causal-normal
anthropic/claude-3.5-sonnet	63.7%	69.5%	trace-hard
anthropic/claude-sonnet-4.5	**80.1%**	**70.1%**	trace-hard
openai/gpt-4o	22.6%	63.5%	trace-hard
openai/gpt-4o-mini	65.9%	49.6%	trace-hard
qwen/qwen3-coder-plus	79.2%	65.5%	trace-hard
anthropic/claude-3.5-sonnet	46.9%	65.7%	trace-normal
anthropic/claude-sonnet-4.5	54.1%	**67.7%**	trace-normal
openai/gpt-4o	27.0%	56.6%	trace-normal
openai/gpt-4o-mini	49.4%	42.2%	trace-normal
qwen/qwen3-coder-plus	**61.2%**	58.9%	trace-normal

Roughly, F_1(AP) asks how well models get mostly correct answers at most time steps, while F_1(TS) asks how well models get all the correct answers for an entire timestep. There is no strong relationship between their performance on TTE and TCP tasks, suggesting these are truly distinct temporal reasoning tasks. Table 1 shows our results across various LLMs.

Our analysis of absolute model performance demonstrates that TempoBench presents a challenging yet tractable benchmark. All evaluated models successfully solve a nontrivial portion of the problems—partially or entirely—showing that the tasks are well-calibrated in difficulty. Notably, TCE-hard proves substantially more difficult than TCE-normal or either TTE variant,

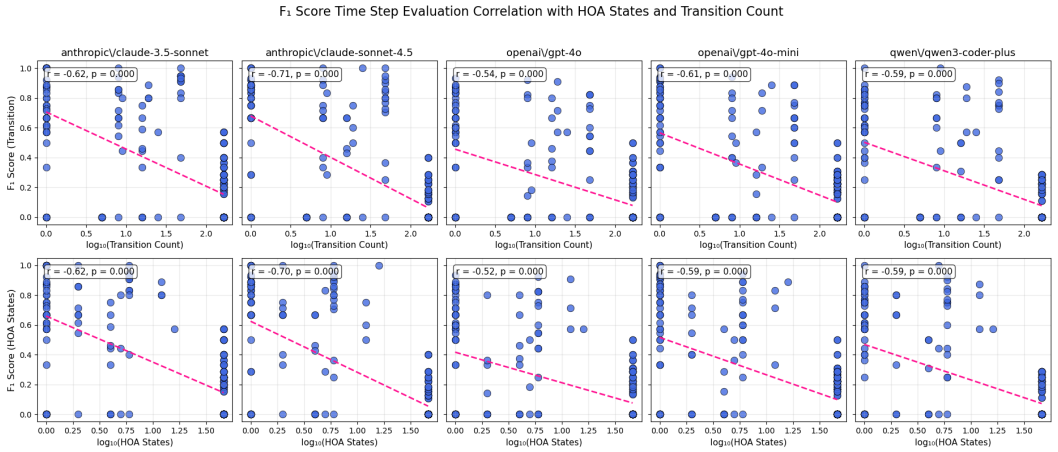


Fig. 7. Correlation scores for $\log_{10}(\texttt{transition\_count})$ and $\log_{10}(\texttt{hoa\_states})$ versus F_1(TS) scores at the temporal step, with corresponding p-values denoting statistical significance

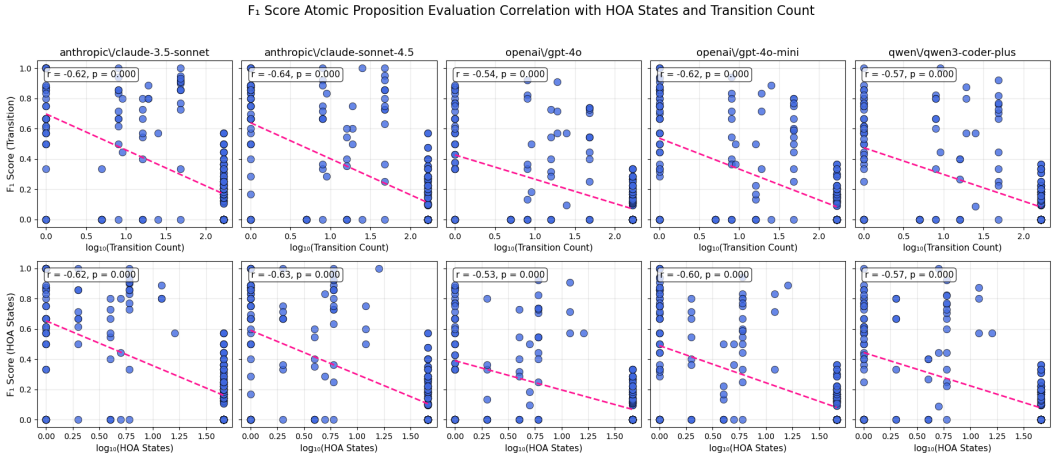


Fig. 8. Correlation scores for $\log_{10}(\texttt{transition\_count})$ and $\log_{10}(\texttt{hoa\_states})$ versus F_1(AP) scores at the Atomic Proposition step, with corresponding p-values denoting statistical significance

validating the effectiveness of our difficulty features. This trend is consistent across both F_1(AP) and F_1(TS) metrics. Interestingly, models exhibit higher scores on temporal-hard AP but lower scores on temporal-hard Timestep tasks. This discrepancy arises because hard samples contain more inputs—allowing models to achieve higher overlap scores within individual timesteps without genuinely resolving the underlying temporal dependencies.

Figures 7 and 8 show correlations between F_1(AP) scores and **transition count** and **system states** for the TCE task. Each scatter plot includes a line of best fit and correlation statistics, showing that the negative relationships between these parameters and the F_1(AP) scores are statistically significant. The p-value of $p < 0.001$ indicates that observing such a negative relationship by chance is extremely unlikely. These graphs provide a useful in-depth understanding of the specific LLM reasoning capabilities. For example, it is surprising that Claude-Sonnet-3.5 outperforms the newer Claude-Sonnet-4.5 on TCE tasks. As shown in Figure 7, F_1(AP) scores for Claude-Sonnet-4.5 exhibit

stronger negative correlations with **transition count** and **HOA state** ($R = -0.71$ and $R = -0.70$) compared to $R = -0.62$ for Claude-Sonnet-3.5, suggesting that the earlier model generalizes more robustly and is less sensitive to increases in system complexity. We hypothesize that recent LLMs are increasingly optimized for the multi-context protocol (MCP), prioritizing flexible tool and agent coordination over deep internal state modeling, which may explain their weaker performance on tasks that require reasoning over large latent state spaces [41].

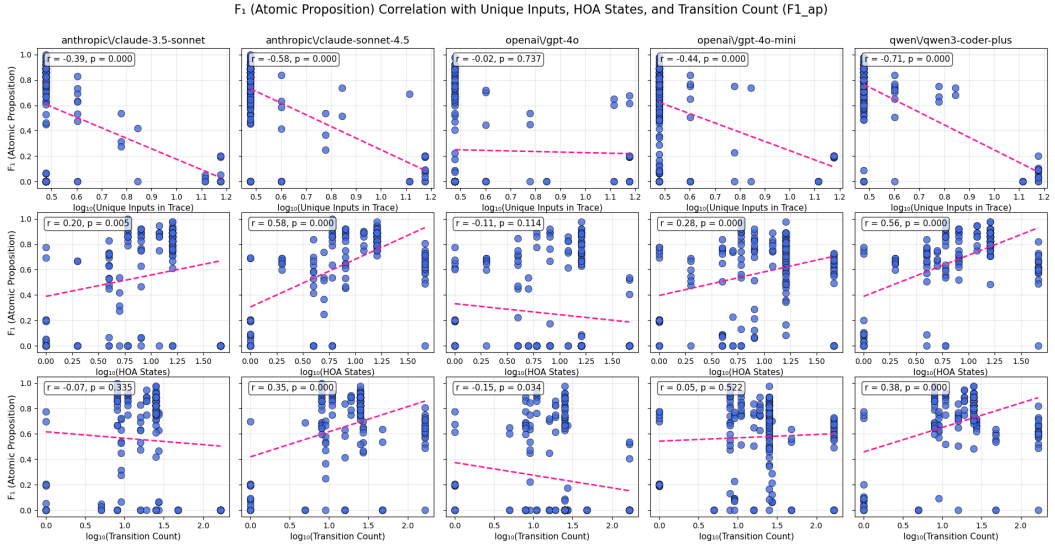


Fig. 9. Correlation scores for $\log_{10}(\texttt{unique\_inputs\_in\_trace})$, $\log_{10}(\texttt{hoa\_states})$, and $\log_{10}(\texttt{transition\_count})$ versus F_1 scores at the Atomic Proposition step, with corresponding p-values denoting statistical significance. This figure covers the TTE task.

The TTE task reveals a different sensitivity pattern across features as seen in Figure 9. There is a negative correlation between the number of unique inputs in the trace and the F_1(AP) score for all models except for GPT-4o. There is a positive correlation between F_1(AP) and the number of system states for all models except GPT-4o and Claude-Sonnet-4.5, which exhibit neutral and weak positive correlation, respectively. Surprisingly, GPT-4o performs much worse than other similar models across all of these tasks, including GPT-4o-mini, which not only outperforms GPT-4o but is also on par with newer models like Claude-Sonnet-4.5. The positive relationship between system states and F_1(AP) score illustrates that the model only needs to know its current state. Figure 9 shows that modern LLMs have become very strong at symbol mapping and retrieval, yet still struggle to perform more complex temporal reasoning. The results in figure 9 show a negative correlation with the number of inputs; the number of constraints on transitions between states mainly determines TTE difficulty. Importantly, the high performance on TTE across both hard and normal conditions indicates that all of these models can understand the HOA format. This demonstrates that while there is variation across models, the HOA file format is not a limiting factor in performing this task.

5.2 Statistical Analysis

We report **precision**, **recall**, and F_1 scores using two evaluation schemes on the Timestep and AP levels. Although results aggregate all models, per-model parameter–F_1 trends remain consistent (see Appendix D.2 for per-model SHAP plots and R^2). The results in Table 2 highlight the increased

Temporal Causality Evaluations	Benchmark Difficulty	Precision	Recall	F_1Score
Time Step Evaluation	Hard Benchmark	12.5%	5.4%	7.5%
	Normal Benchmark	62.8%	68.6%	65.6%
Atomic Proposition Evaluation	Hard Benchmark	19.9%	5.4%	8.5%
	Normal Benchmark	57.1%	62.1%	59.5%

Table 2. Temporal Causality Performance metrics comparing Time Step and Atomic Proposition evaluations across benchmark difficulties aggregated across all models

difficulty of this task for more complex systems. We further illustrate this with SHAP plots derived from **random forest** models trained to predict F_1(TS) and F_1(AP), respectively. **Random forest** regression is used because the F_1 distributions deviate from normality as shown in Appendix D making simpler models such as **Ordinary Least Squares** regression inappropriate [1]. The SHAP plots in figure 11 highlight both the relative importance of each feature and the direction of its influence on the F_1 Score. For instance, the number of system states emerges as a key feature: higher state counts generally negatively affect F_1, and when the relationship is positive, the effect is comparatively weak. Lastly, the features are listed from most important to least important. The **random forest** models for predicting F_1(AP) and F_1(TS) achieve R^2 scores of **0.646** and **0.66**. R^2 represents the amount of our F_1 scores explained by the predictors. These R^2 scores indicate that in our temporal causality task, the majority of the variation in the F_1 score is explained by the customizable parameters of the benchmark. The remainder is explained by variation in LLM performance on temporal causality. Given the strong model fit indicated by the R^2 score, the SHAP values provide meaningful insights into how increasing automaton complexity hinders LLMs' ability to perform temporal credit assignment. Features such as the number of states, transitions, and unique inputs emerge as key drivers of difficulty, aligning with intuition. In contrast, traces with more causal inputs show higher F_1 scores because a greater number of distinct causes reduces the need for the LLM to model long-range dependencies. When traces contain few inputs, the model must infer relationships from limited information, whereas traces with many causal inputs provide a denser causal structure.

Temporal Trace Evaluations	Benchmark Difficulty	Precision	Recall	F_1Score
Time Step Evaluation	Hard Benchmark	61.8%	60.3%	61.0%
	Normal Benchmark	54.3%	51.2%	52.8%
Atomic Proposition Evaluation	Hard Benchmark	60.0%	74.0%	66.3%
	Normal Benchmark	48.3%	50.2%	49.2%

Table 3. TTE Performance metrics comparing Timestep and AP Evaluation across benchmark difficulties

Table 3 reports performance on the trace acceptance task, evaluated at two levels: (1) TS, and (2) AP level. The TS results show that TTE is an easier task than TCE, demonstrating that LLMs struggle to model long-range temporal dependencies 3. Interestingly, the AP analysis reveals that the hard benchmark is less challenging than the normal one. This is because larger systems contain more atomic propositions, increasing opportunities for partial credit rather than reflecting. Yet this demonstrates that, in some temporal reasoning tasks, such as TTE, simply having larger state machines does not make the task more difficult. We forgo fitting a random forest regression

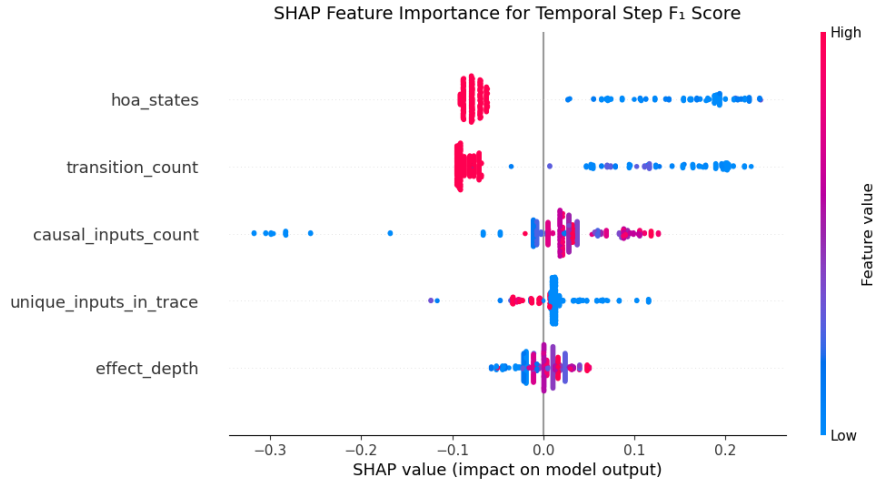


Fig. 10. SHAP Beeswarm plot representing feature importance and correlation with Temporal Causality F_1 Scores at Atomic Proposition Step

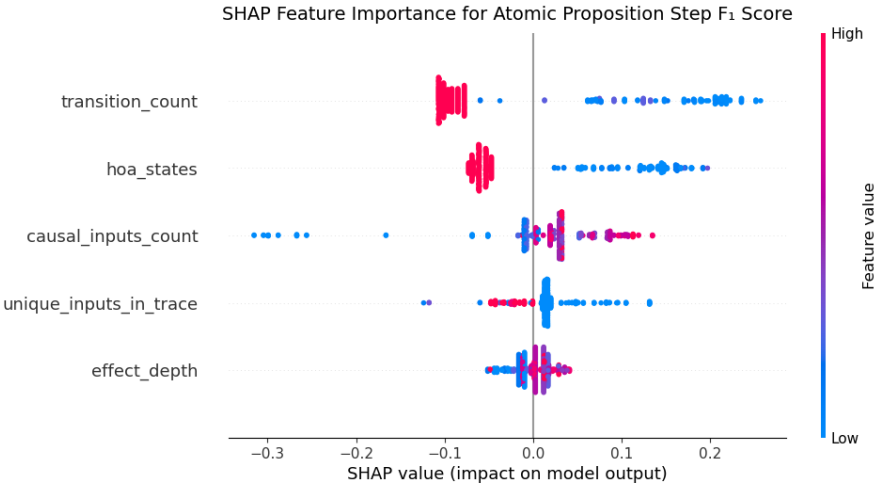


Fig. 11. SHAP Beeswarm plot representing feature importance and correlation with Temporal Causality F_1 Scores at Temporal Step

model to the F_1 scores, as this task primarily depends on only two parameters: **system states** and **transition count**. The convergence of precision and recall for the TTE task demonstrates that LLMs genuinely understand the HOA file format rather than struggling with parsing. If precision and recall diverged—with the model either over-predicting (high recall, low precision) or under-predicting (high precision, low recall)—this would indicate random guessing or parse errors rather than systematic understanding.

6 Future Work

LLMs trained on mathematical and code reasoning tasks exhibit consistently stronger general-purpose reasoning capabilities [26]. We see TempoBench as a gateway to being an effective tool in training LLMs to perform structured temporal reasoning in ways that neither ad-hoc generated nor Lean proof-based systems are able to. Tempobench offers a uniquely rich source of formally grounded temporal reasoning traces over deterministic systems, which may be able to help models generalize towards broader reasoning tasks requiring causal credit assignment, planning, and multi-step prediction. Future work would utilize TempoBench to generate training data for LLM reasoning agents, extending their temporal reasoning ability.

7 Conclusions

In summary, we present TempoBench, a generative, synthetic, and formally verified benchmark for temporal reasoning. This diagnostic-focused benchmark is created using formal methods tools for reactive program synthesis and causality analysis. In this work, we introduce TempoBench as a formally grounded benchmark for evaluating temporal causality and credit assignment, and demonstrate its utility not only as a benchmark but as a full diagnostic pipeline. TempoBench benchmarks offer critical advantages for analyzing and interpreting LLMs, including verifiability, compositional structure, and scalability to large state spaces. Unlike prior temporal reasoning benchmarks that primarily serve as leaderboards, TempoBench is explicitly designed as a diagnostic tool — enabling researchers to probe failure modes, trace causal reasoning, and study alignment with formal system dynamics, rather than merely measuring task performance. Using TempoBench, find that reasoning over systems with larger problem spaces and more logical transitions does increase complexity. However, our results also demonstrate that, in many cases, larger systems with more transitions have a higher density of connections between states, making it easier for LLMs to track chains of reasoning through them. Our results show that temporal causality is complex for LLMs, with F_1 score on TCE tasks dropping by 54.6% on average across TS and AP. By identifying the structures, such as system states and transition counts, we aim to help diagnose these system failures so that future LLMs can perform better on such temporal reasoning tasks.

References

[1] Acito, F. Ordinary least squares regression. In *Predictive analytics with KNIME: Analytics for citizen data scientists*. Springer, 2023, pp. 105–124.

[2] Anthropic. Claude-3.5 sonnet. https://www.anthropic.com/news/claude-3-5-sonnet, 2024. Accessed: 2025-10-05.

[3] Anthropic. Claude sonnet-4.5. https://www.anthropic.com/news/claude-sonnet-4-5, 2025. Accessed: 2025-10-05.

[4] Backes, J., Bolignano, P., Cook, B., Dodge, C., Gacek, A., Luckow, K., Rungta, N., Tkachuk, O., and Varming, C. Semantic-based automated reasoning for aws access policies using smt. In *2018 Formal Methods in Computer Aided Design (FMCAD)* (2018), pp. 1–9.

[5] Besta, M., Blach, N., Kubicek, A., Gerstenberger, R., Podstawski, M., Gianinazzi, L., Gajda, J., Lehmann, T., Niewiadomski, H., Nyczyk, P., and Hoefler, T. Graph of thoughts: solving elaborate problems with large language models. In *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence and Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence and Fourteenth Symposium on Educational Advances in Artificial Intelligence* (2024), AAAI'24/IAAI'24/EAAI'24, AAAI Press.

[6] Blöbaum, P., Götz, P., Budhathoki, K., Mastakouri, A. A., and Janzing, D. Dowhy-gcm: An extension of dowhy for causal inference in graphical causal models. *Journal of Machine Learning Research 25*, 147 (2024), 1–7.

[7] Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *Advances in neural information processing systems 33* (2020), 1877–1901.

[8] Bubeck, S., Chandrasekaran, V., Eldan, R., Gehrke, J., Horvitz, E., Kamar, E., Lee, P., Lee, Y. T., Li, Y., Lundberg, S., Nori, H., Palangi, H., Ribeiro, M. T., and Zhang, Y. Sparks of artificial general intelligence: Early experiments with gpt-4, 2023.

[9] Chen, M., Tworek, J., Jun, H., Yuan, Q., de Oliveira Pinto, H. P., Kaplan, J., Edwards, H., Burda, Y., Joseph, N., Brockman, G., Ray, A., Puri, R., Krueger, G., Petrov, M., Khlaaf, H., Sastry, G., Mishkin, P., Chan, B., Gray, S., Ryder, N., Pavlov, M., Power, A., Kaiser, L., Bavarian, M., Winter, C., Tillet, P., Such, F. P., Cummings, D., Plappert, M., Chantzis, F., Barnes, E., Herbert-Voss, A., Guss, W. H., Nichol, A., Paino, A., Tezak, N., Tang, J., Babuschkin, I., Balaji, S., Jain, S., Saunders, W., Hesse, C., Carr, A. N., Leike, J., Achiam, J., Misra, V., Morikawa, E., Radford, A., Knight, M., Brundage, M., Murati, M., Mayer, K., Welinder, P., McGrew, B., Amodei, D., McCandlish, S., Sutskever, I., and Zaremba, W. Evaluating large language models trained on code, 2021.

[10] Chen, W., Ma, X., Wang, X., and Cohen, W. W. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks, 2023.

[11] Chollet, F., Knoop, M., Kamradt, G., and Landers, B. Arc prize 2024: Technical report, 2025.

[12] Chollet, F., Knoop, M., Kamradt, G., Landers, B., and Pinkard, H. Arc-agi-2: A new challenge for frontier ai reasoning systems, 2025.

[13] Church, A. Application of recursive arithmetic to the problem of circuit synthesis. *Journal of Symbolic Logic 28*, 4 (1963).

[14] Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano, R., Hesse, C., and Schulman, J. Training verifiers to solve math word problems, 2021.

[15] Coenen, N., Finkbeiner, B., Frenkel, H., Hahn, C., Metzger, N., and Siber, J. Temporal causality in reactive systems. In *International symposium on automated technology for verification and analysis* (2022), Springer, pp. 208–224.

[16] Davila, N., Wiese, I., Steinmacher, I., Lucio da Silva, L., Kawamoto, A., Favaro, G. J. P., and Nunes, I. An industry case study on adoption of ai-based programming assistants. In *Proceedings of the 46th International Conference on Software Engineering: Software Engineering in Practice* (New York, NY, USA, 2024), ICSE-SEIP '24, Association for Computing Machinery, p. 92–102.

[17] DeepSeek-AI, Guo, D., Yang, D., Zhang, H., Song, J., Zhang, R., Xu, R., Zhu, Q., Ma, S., Wang, P., Bi, X., Zhang, X., Yu, X., Wu, Y., Wu, Z. F., Gou, Z., Shao, Z., Li, Z., Gao, Z., Liu, A., Xue, B., Wang, B., Wu, B., Feng, B., Lu, C., Zhao, C., Deng, C., Zhang, C., Ruan, C., Dai, D., Chen, D., Ji, D., Li, E., Lin, F., Dai, F., Luo, F., Hao, G., Chen, G., Li, G., Zhang, H., Bao, H., Xu, H., Wang, H., Ding, H., Xin, H., Gao, H., Qu, H., Li, H., Guo, J., Li, J., Wang, J., Chen, J., Yuan, J., Qiu, J., Li, J., Cai, J. L., Ni, J., Liang, J., Chen, J., Dong, K., Hu, K., Gao, K., Guan, K., Huang, K., Yu, K., Wang, L., Zhang, L., Zhao, L., Wang, L., Zhang, L., Xu, L., Xia, L., Zhang, M., Zhang, M., Tang, M., Li, M., Wang, M., Li, M., Tian, N., Huang, P., Zhang, P., Wang, Q., Chen, Q., Du, Q., Ge, R., Zhang, R., Pan, R., Wang, R., Chen, R. J., Jin, R. L., Chen, R., Lu, S., Zhou, S., Chen, S., Ye, S., Wang, S., Yu, S., Zhou, S., Pan, S., Li, S. S., Zhou, S., Wu, S., Ye, S., Yun, T., Pei, T., Sun, T., Wang, T., Zeng, W., Zhao, W., Liu, W., Liang, W., Gao, W., Yu, W., Zhang, W., Xiao, W. L., An, W., Liu, X., Wang, X., Chen, X., Nie, X., Cheng, X., Liu, X., Xie, X., Liu, X., Yang, X., Li, X., Su, X., Lin, X., Li, X. Q., Jin, X., Shen, X., Chen, X., Sun, X., Wang, X., Song, X., Zhou, X., Wang, X., Shan, X., Li, Y. K., Wang, Y. Q., Wei, Y. X., Zhang, Y., Xu, Y., Li, Y., Zhao, Y., Sun, Y., Wang, Y., Yu, Y., Zhang, Y., Shi, Y., Xiong, Y., He, Y., Piao, Y., Wang, Y., Tan, Y., Ma, Y., Liu, Y., Guo, Y., Ou, Y., Wang, Y., Gong, Y., Zou, Y., He, Y., Xiong, Y., Luo, Y., You, Y., Liu, Y., Zhou, Y., Zhu, Y. X., Xu, Y., Huang, Y., Li, Y., Zheng, Y., Zhu, Y., Ma, Y., Tang, Y., Zha, Y., Yan, Y., Ren, Z. Z., Ren, Z., Sha, Z., Fu, Z., Xu, Z., Xie, Z., Zhang, Z., Hao, Z., Ma, Z., Yan, Z., Wu, Z., Gu, Z., Zhu, Z., Liu, Z., Li, Z., Xie, Z., Song, Z., Pan, Z., Huang, Z., Xu, Z., Zhang, Z., and Zhang, Z. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025.

[18] Di Stefano, L. Execution and monitoring of hoa automata with hoax. In *International Conference on Runtime Verification* (2025), Springer, pp. 44–53.

[19] Ding, J., Zhang, Y., Shang, Y., Zhang, Y., Zong, Z., Feng, J., Yuan, Y., Su, H., Li, N., Sukiennik, N., et al. Understanding world or predicting future? a comprehensive survey of world models. *ACM Computing Surveys 58*, 3 (2025), 1–38.

[20] Ding, Y., Min, M. J., Kaiser, G., and Ray, B. Cycle: Learning to self-refine the code generation. *Proc. ACM Program. Lang. 8*, OOPSLA1 (Apr. 2024).

[21] Ding, Y., Peng, J., Min, M. J., Kaiser, G., Yang, J., and Ray, B. Semcoder: Training code language models with comprehensive semantics reasoning. In *Advances in Neural Information Processing Systems* (2024), A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, Eds., vol. 37, Curran Associates, Inc., pp. 60275–60308.

[22] Fatemi, B., Kazemi, M., Tsitsulin, A., Malkan, K., Yim, J., Palowitch, J., Seo, S., Halcrow, J., and Perozzi, B. Test of time: A benchmark for evaluating llms on temporal reasoning, 2024.

[23] Fatemi, B., Kazemi, M., Tsitsulin, A., Malkan, K., Yim, J., Palowitch, J., Seo, S., Halcrow, J., and Perozzi, B. Test of time: A benchmark for evaluating llms on temporal reasoning. *arXiv preprint arXiv:2406.09170* (2024).

[24] Finkbeiner, B., Frenkel, H., Metzger, N., and Siber, J. Synthesis of temporal causality. In *International Conference on Computer Aided Verification* (2024), Springer, pp. 87–111.

[25] Finkbeiner, B., Klein, F., Piskac, R., and Santolucito, M. Temporal stream logic: Synthesis beyond the bools. In *International Conference on Computer Aided Verification* (2019), Springer, pp. 609–629.

[26] Huan, M., Li, Y., Zheng, T., Xu, X., Kim, S., Du, M., Poovendran, R., Neubig, G., and Yue, X. Does math reasoning improve general llm capabilities? understanding transferability of llm reasoning. *arXiv preprint arXiv:2507.00432* (2025).

[27] Ikram, A., Chakraborty, S., Mitra, S., Saini, S., Bagchi, S., and Kocaoglu, M. Root cause analysis of failures in microservices through causal discovery. *Advances in Neural Information Processing Systems 35* (2022), 31158–31170.

[28] Jacobs, S., Klein, F., and Schirmer, S. A high-level ltl synthesis format: Tlsf v1. 1 (extended version). *arXiv preprint arXiv:1604.02284* (2016).

[29] Jacobs, S., Pérez, G. A., Abraham, R., Bruyère, V., Cadilhac, M., Colange, M., Delfosse, C., van Dijk, T., Duret-Lutz, A., Faymonville, P., Finkbeiner, B., Khalimov, A., Klein, F., Luttenberger, M., Meyer, K., Michaud, T., Pommellet, A., Renkin, F., Schlehuber-Caissier, P., Sakr, M., Sickert, S., Staquet, G., Tamines, C., Tentrup, L., and Walker, A. The Reactive Synthesis Competition (SYNTCOMP): 2018–2021. *International Journal on Software Tools for Technology Transfer 26*, 5 (Oct. 2024), 551–567.

[30] Jimenez, C. E., Yang, J., Wettig, A., Yao, S., Pei, K., Press, O., and Narasimhan, K. Swe-bench: Can language models resolve real-world github issues?, 2024.

[31] Klemmer, J. H., Horstmann, S. A., Patnaik, N., Ludden, C., Burton, Cordell, J., Powers, C., Massacci, F., Rahman, A., Votipka, D., Lipford, H. R., Rashid, A., Naiakshina, A., and Fahl, S. Using ai assistants in software development: A qualitative study on security practices and concerns. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security* (New York, NY, USA, 2024), CCS '24, Association for Computing Machinery, p. 2726–2740.

[32] Leucker, M., and Schallhart, C. A brief account of runtime verification. *The journal of logic and algebraic programming 78*, 5 (2009), 293–303.

[33] OpenAI. Gpt-4o mini: Advancing cost-efficient intelligence. https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/, 2024. Accessed: 2025-10-05.

[34] OpenAI. Gpt-4o ("omni"): Openai's flagship multimodal model. https://platform.openai.com/docs/models/gpt-4o, 2024. Accessed: 2025-10-05.

[35] Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P., Leike, J., and Lowe, R. Training language models to follow instructions with human feedback, 2022.

[36] Pham, L., Ha, H., and Zhang, H. Root cause analysis for microservice system based on causal inference: How far are we? ASE '24, Association for Computing Machinery.

[37] Pnueli, A. The temporal logic of programs. In *18th annual symposium on foundations of computer science (sfcs 1977)* (1977), ieee, pp. 46–57.

[38] Pnueli, A., and Rosner, R. On the synthesis of a reactive module. In *Proceedings of the 16th ACM SIGPLAN-SIGACT symposium on Principles of programming languages* (1989), pp. 179–190.

[39] Renkin, F., Schlehuber-Caissier, P., Duret-Lutz, A., and Pommellet, A. Dissecting ltlsynt. *Formal Methods in System Design 61*, 2 (2022), 248–289.

[40] Roychoudhury, A., Pasareanu, C., Pradel, M., and Ray, B. Agentic ai software engineers: Programming with trust, 2025.

[41] Schick, T., Dwivedi-Yu, J., Dessì, R., Raileanu, R., Lomeli, M., Hambro, E., Zettlemoyer, L., Cancedda, N., and Scialom, T. Toolformer: Language models can teach themselves to use tools. *Advances in Neural Information Processing Systems 36* (2023), 68539–68551.

[42] Srivastava, A., Rastogi, A., Rao, A., Shoeb, A. A. M., Abid, A., Fisch, A., Brown, A. R., Santoro, A., Gupta, A., Garriga-Alonso, A., Kluska, A., Lewkowycz, A., Agarwal, A., Power, A., Ray, A., Warstadt, A., Kocurek, A. W., Safaya, A., Tazarv, A., Xiang, A., Parrish, A., Nie, A., Hussain, A., Askell, A., Dsouza, A., Slone, A., Rahane, A., Iyer, A. S., Andreassen, A., Madotto, A., Santilli, A., Stuhlmüller, A., Dai, A., La, A., Lampinen, A., Zou, A., Jiang, A., Chen, A., Vuong, A., Gupta, A., Gottardi, A., Norelli, A., Venkatesh, A., Gholamidavoodi, A., Tabassum, A., Menezes, A., Kirubarajan, A., Mullokandov, A., Sabharwal, A., Herrick, A., Efrat, A., Erdem, A., Karakaş, A., Roberts, B. R., Loe, B. S., Zoph, B., Bojanowski, B., Özyurt, B., Hedayatnia, B., Neyshabur, B., Inden, B., Stein, B., Ekmekci, B., Lin, B. Y., Howald, B., Orinion, B., Diao, C., Dour, C., Stinson, C., Argueta, C., Ramírez, C. F., Singh, C., Rathkopf, C., Meng, C., Baral, C., Wu, C., Callison-Burch, C., Waites, C., Voigt, C., Manning, C. D., Potts, C., Ramirez, C., Rivera, C. E., Siro, C., Raffel, C., Ashcraft, C., Garbacea, C., Sileo, D., Garrette, D., Hendrycks, D., Kilman, D., Roth, D., Freeman, D., Khashabi, D., Levy, D., González, D. M., Perszyk, D., Hernandez, D., Chen, D., Ippolito, D., Gilboa, D., Dohan, D., Drakard, D., Jurgens, D., Datta, D., Ganguli, D., Emelin, D., Kleyko, D., Yuret, D., Chen, D., Tam, D., Hupkes, D., Misra, D., Buzan, D., Mollo, D. C., Yang, D., Lee, D.-H., Schrader, D., Shutova, E., Cubuk, E. D., Segal, E., Hagerman, E., Barnes, E., Donoway, E., Pavlick, E., Rodola, E., Lam, E., Chu, E., Tang, E., Erdem, E., Chang, E., Chi, E. A., Dyer, E., Jerzak, E., Kim, E., Manyasi, E. E., Zheltonozhskii, E., Xia, F., Siar, F., Martínez-Plumed, F., Happé, F., Chollet, F., Rong, F., Mishra, G., Winata, G. I., de Melo, G., Kruszewski, G., Parascandolo, G., Mariani, G., Wang, G., Jaimovitch-López, G.,

Betz, G., Gur-Ari, G., Galijasevic, H., Kim, H., Rashkin, H., Hajishirzi, H., Mehta, H., Bogar, H., Shevlin, H., Schütze, H., Yakura, H., Zhang, H., Wong, H. M., Ng, I., Noble, I., Jumelet, J., Geissinger, J., Kernion, J., Hilton, J., Lee, J., Fisac, J. F., Simon, J. B., Koppel, J., Zheng, J., Zou, J., Kocoń, J., Thompson, J., Wingfield, J., Kaplan, J., Radom, J., Sohl-Dickstein, J., Phang, J., Wei, J., Yosinski, J., Novikova, J., Bosscher, J., Marsh, J., Kim, J., Taal, J., Engel, J., Alabi, J., Xu, J., Song, J., Tang, J., Waweru, J., Burden, J., Miller, J., Balis, J. U., Batchelder, J., Berant, J., Frohberg, J., Rozen, J., Hernandez-Orallo, J., Boudeman, J., Guerr, J., Jones, J., Tenenbaum, J. B., Rule, J. S., Chua, J., Kanclerz, K., Livescu, K., Krauth, K., Gopalakrishnan, K., Ignatyeva, K., Markert, K., Dhole, K. D., Gimpel, K., Omondi, K., Mathewson, K., Chiafullo, K., Shkaruta, K., Shridhar, K., McDonell, K., Richardson, K., Reynolds, L., Gao, L., Zhang, L., Dugan, L., Qin, L., Contreras-Ochando, L., Morency, L.-P., Moschella, L., Lam, L., Noble, L., Schmidt, L., He, L., Colón, L. O., Metz, L., Şenel, L. K., Bosma, M., Sap, M., ter Hoeve, M., Farooqi, M., Faruqui, M., Mazeika, M., Baturan, M., Marelli, M., Maru, M., Quintana, M. J. R., Tolkiehn, M., Giulianelli, M., Lewis, M., Potthast, M., Leavitt, M. L., Hagen, M., Schubert, M., Baitemirova, M. O., Arnaud, M., McElrath, M., Yee, M. A., Cohen, M., Gu, M., Ivanitskiy, M., Starritt, M., Strube, M., Swędrowski, M., Bevilacqua, M., Yasunaga, M., Kale, M., Cain, M., Xu, M., Suzgun, M., Walker, M., Tiwari, M., Bansal, M., Aminnaseri, M., Geva, M., Gheini, M., T, M. V., Peng, N., Chi, N. A., Lee, N., Krakover, N. G.-A., Cameron, N., Roberts, N., Doiron, N., Martinez, N., Nangia, N., Deckers, N., Muennighoff, N., Keskar, N. S., Iyer, N. S., Constant, N., Fiedel, N., Wen, N., Zhang, O., Agha, O., Elbaghdadi, O., Levy, O., Evans, O., Casares, P. A. M., Doshi, P., Fung, P., Liang, P. P., Vicol, P., Alipoormolabashi, P., Liao, P., Liang, P., Chang, P., Eckersley, P., Htut, P. M., Hwang, P., Miłkowski, P., Patil, P., Pezeshkpour, P., Oli, P., Mei, Q., Lyu, Q., Chen, Q., Banjade, R., Rudolph, R. E., Gabriel, R., Habacker, R., Risco, R., Millière, R., Garg, R., Barnes, R., Saurous, R. A., Arakawa, R., Raymaekers, R., Frank, R., Sikand, R., Novak, R., Sitelew, R., LeBras, R., Liu, R., Jacobs, R., Zhang, R., Salakhutdinov, R., Chi, R., Lee, R., Stovall, R., Teehan, R., Yang, R., Singh, S., Mohammad, S. M., Anand, S., Dillavou, S., Shleifer, S., Wiseman, S., Gruetter, S., Bowman, S. R., Schoenholz, S. S., Han, S., Kwatra, S., Rous, S. A., Ghazarian, S., Ghosh, S., Casey, S., Bischoff, S., Gehrmann, S., Schuster, S., Sadeghi, S., Hamdan, S., Zhou, S., Srivastava, S., Shi, S., Singh, S., Asaadi, S., Gu, S. S., Pachchigar, S., Toshniwal, S., Upadhyay, S., Shyamolima, Debnath, Shakeri, S., Thormeyer, S., Melzi, S., Reddy, S., Makini, S. P., Lee, S.-H., Torene, S., Hatwar, S., Dehaene, S., Divic, S., Ermon, S., Biderman, S., Lin, S., Prasad, S., Piantadosi, S. T., Shieber, S. M., Misherghi, S., Kiritchenko, S., Mishra, S., Linzen, T., Schuster, T., Li, T., Yu, T., Ali, T., Hashimoto, T., Wu, T.-L., Desbordes, T., Rothschild, T., Phan, T., Wang, T., Nkinyili, T., Schick, T., Kornev, T., Tunduny, T., Gerstenberg, T., Chang, T., Neeraj, T., Khot, T., Shultz, T., Shaham, U., Misra, V., Demberg, V., Nyamai, V., Raunak, V., Ramasesh, V., Prabhu, V. U., Padmakumar, V., Srikumar, V., Fedus, W., Saunders, W., Zhang, W., Vossen, W., Ren, X., Tong, X., Zhao, X., Wu, X., Shen, X., Yaghoobzadeh, Y., Lakretz, Y., Song, Y., Bahri, Y., Choi, Y., Yang, Y., Hao, Y., Chen, Y., Belinkov, Y., Hou, Y., Hou, Y., Bai, Y., Seid, Z., Zhao, Z., Wang, Z., Wang, Z. J., Wang, Z., and Wu, Z. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models, 2023.

[43] SYNTCOMP. SYNTCOMP/benchmarks: Repository of benchmarks for the Reactive Synthesis Competition (SYNTCOMP). https://github.com/SYNTCOMP/benchmarks, 2025. Accessed: 2025-10-05.

[44] Tan, Q., Ng, H. T., and Bing, L. Towards benchmarking and improving the temporal reasoning capability of large language models. *arXiv preprint arXiv:2306.08952* (2023).

[45] Tan, Q., Ng, H. T., and Bing, L. Towards benchmarking and improving the temporal reasoning capability of large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (Toronto, Canada, July 2023), A. Rogers, J. Boyd-Graber, and N. Okazaki, Eds., Association for Computational Linguistics, pp. 14820–14835.

[46] Toles, M., Balwani, N., Singh, R., Rodriguez, V. G. S., and Yu, Z. Program synthesis dialog agents for interactive decision-making, 2025.

[47] Wang, X., Wei, J., Schuurmans, D., Le, Q., Chi, E., Narang, S., Chowdhery, A., and Zhou, D. Self-consistency improves chain of thought reasoning in language models, 2023.

[48] Wang, Y., and Zhao, Y. Tram: Benchmarking temporal reasoning for large language models. *arXiv preprint arXiv:2310.00835* (2023).

[49] Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., Le, Q. V., Zhou, D., et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems 35* (2022), 24824–24837.

[50] Xiong, S., Payani, A., Kompella, R., and Fekri, F. Large language models can learn temporal reasoning. *arXiv preprint arXiv:2401.06853* (2024).

[51] Yang, A., Li, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Gao, C., Huang, C., Lv, C., Zheng, C., Liu, D., Zhou, F., Huang, F., Hu, F., Ge, H., Wei, H., Lin, H., Tang, J., Yang, J., Tu, J., Zhang, J., Yang, J., Yang, J., Zhou, J., Zhou, J., Lin, J., Dang, K., Bao, K., Yang, K., Yu, L., Deng, L., Li, M., Xue, M., Li, M., Zhang, P., Wang, P., Zhu, Q., Men, R., Gao, R., Liu, S., Luo, S., Li, T., Tang, T., Yin, W., Ren, X., Wang, X., Zhang, X., Ren, X., Fan, Y., Su, Y., Zhang, Y., Zhang, Y., Wan, Y., Liu, Y., Wang, Z., Cui, Z., Zhang, Z., Zhou, Z., and Qiu, Z. Qwen3 technical report, 2025.

[52] Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T. L., Cao, Y., and Narasimhan, K. Tree of thoughts: deliberate problem solving with large language models. In *Proceedings of the 37th International Conference on Neural Information Processing Systems* (Red Hook, NY, USA, 2023), NIPS '23, Curran Associates Inc.

A Synthesis Preliminaries

A.1 Linear Temporal Logic & Reactive Systems

Reactive systems appear across a wide range of domains, from traditional hardware and software to more abstract environments such as biological systems. Temporal logics, most prominently Linear Temporal Logic (LTL) [37], provide a formal language for specifying the desired behavior of such systems over time. We consider LTL, which has the following syntax.

$$\varphi ::= \top | a | \varphi \vee \varphi | \neg\varphi | X\varphi | \varphi \mathcal{U} \varphi,$$

where $a \in AP$ is an *atomic proposition*, $\{\wedge, \neg\}$ are the common Boolean operators of *conjunction* and *negation*, respectively, and $\{X, \mathcal{U}\}$ are the *next* and *until* temporal operators, respectively.

Additional temporal operators include F(*finally*), and G (*always*), which can be derived from the syntax above.

Building on LTL specifications, reactive synthesis has extended the Church Synthesis problem [13] to automatically construct implementations specified from LTL specifications [38]. This reactive synthesis can be framed as a two-player infinite game between an environment and a system player. The set of atomic propositions is partitioned into inputs $I = \{i_0, i_1, \ldots, i_n\}$ (controlled by the environment) and outputs $O = \{o_0, o_1, \ldots, o_m\}$ (controlled by the system). At each round, the environment first assigns values to all inputs, then the system assigns values to all outputs, producing an infinite trace over $I \cup O$. The system wins if it has a strategy to choose outputs such that for every possible sequence of inputs, the resulting infinite trace satisfies the LTL specification φ. If such a winning strategy exists, the specification is realizable and a corresponding Mealy machine can be extracted. The outputted Mealy machine provides a temporal relationship between the atomic propositions I and O that inherits formal guarantees of the specification φ.

A.2 Temporal Causality

Causality in temporal systems is more complicated than the analysis of the underlying LTL specification. While LTL specifications describe relationships between inputs and outputs, temporal causality is interested in identifying the minimum causal set over a trace π which describes each of the inputs necessary for some effect E to occur. Specifically, given a system $\mathcal{T}$ with traces $\pi \in \text{Traces}(\mathcal{T})$, let $C \subseteq (2^I)^\omega$ be a cause property over the inputs and let $E \subseteq (2^O)^\omega$ be an effect property over the outputs [15]. We say that C is a cause of E on π in $\mathcal{T}$ if three conditions hold:

(1) $\pi \models C$ and $\pi \models E$
(2) under some counterfactual variation of the inputs, removing or altering C leads to a trace π' in which the effect property E no longer holds, which ensures the effect depends on the cause
(3) no strict subset $C' \subset C$ also satisfies these two conditions, guaranteeing that the cause is minimal and not redundant

Identifying the causal set for some trace is a synthesis problem that identifies, given a Trace π and system T, what inputs at each time step are necessary to produce the desired output. In our credit-assignment work, we focus on outputs at particular time steps. The resulting effects take the form

$$XX \ldots X\, AP^{o_i}.$$

Our goal is to identify which inputs at each time step $t_i \in \pi$ produce this effect E, across all traces with this counterfactual structure, and from this synthesize the minimal controller. This controller

represents the minimal causal inputs necessary to produce the desired effect at a time step t_j [24]. We use an LTL specification of a music app from the syntcomp benchmark that synthesizes a controller managing the interplay between user inputs (play/pause button presses, leaving/resuming the app) and system outputs (play/pause commands, internal control state) [25] to illustrate the difference between simple credit assignment of a single input-output pair and full causal temporal reasoning.

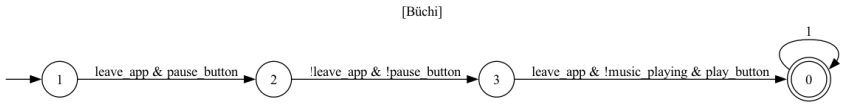


Fig. 12. Temporal Causality HOA for Trace of Music Player

Example A.1 (Music Player App). Given the specification:

$$
\begin{aligned}
\varphi = \Big(& G\big(\neg(pause\_cmd \wedge \neg(play\_cmd \vee ctrl) \leftrightarrow \\
& \quad \neg(play\_cmd \wedge \neg ctrl \leftrightarrow ctrl \wedge \neg play\_cmd) \wedge \neg pause\_cmd\big)\big) \\
& \wedge\ G\big(leave\_app \rightarrow \neg play\_button \wedge \neg pause\_button\big) \wedge G\big(\neg(play\_button \wedge pause\_button)\big) \\
& \wedge\ G\big(\neg(leave\_app \wedge resume\_app)\big) \wedge G\big(play\_cmd \rightarrow X(music\_playing\ W\ pause\_cmd)\big) \\
& \wedge\ G\big(pause\_cmd \rightarrow X(\neg music\_playing\ W\ play\_cmd)\big)\Big) \\
\rightarrow \Big(& G\big(play\_button \rightarrow play\_cmd\big) \wedge G\big(pause\_button \rightarrow pause\_cmd\big) \\
& \wedge\ G\big(pause\_cmd \rightarrow leave\_app \vee pause\_button\big) \wedge G\big(play\_cmd \rightarrow \neg leave\_app\big) \\
& \wedge\ G\big(pause\_cmd \wedge pause\_button \rightarrow \neg play\_cmd\ W\ play\_button\big) \\
& \wedge\ G\Big(music\_playing \wedge leave\_app \rightarrow \big(\neg(pause\_cmd \wedge \\
& \quad (\neg(\neg pause\_button \rightarrow play\_cmd)\ W\ (\neg pause\_button \rightarrow play\_cmd) \wedge \neg leave\_app))\ W \\
& \quad pause\_cmd \wedge (\neg(\neg pause\_button \rightarrow play\_cmd)\ W \\
& \qquad (\neg pause\_button \rightarrow play\_cmd) \wedge \neg leave\_app) \wedge leave\_app\big)\Big)\Big)
\end{aligned}
$$

with

$I = \{leave\_app, music\_playing, pause\_button, play\_button, resume\_app\},\ O = \{ctrl, pause\_cmd, play\_cmd\}$

Given φ, consider the trace

π = {pause_cmd, leave_app, pause_button; play_button, play_cmd, leave_app, play_button}[2]

and let

$$E = \text{XX play\_cmd}.$$

The instantiation of temporal causality as seen in 12 for the music player T's behavior in π demonstrates that the set of inputs required to trigger play_cmd at t_3 is much greater than the play_button input at t_2, as implied by φ. Understanding these unintuitive relationships helps develop deeper temporal reasoning about various reactive systems and enables a more efficient way to generate temporal reasoning for LLMs in training and chain-of-thought (CoT) prompting.

[2]Only positive atomic propositions are denoted.

B Data Pipeline

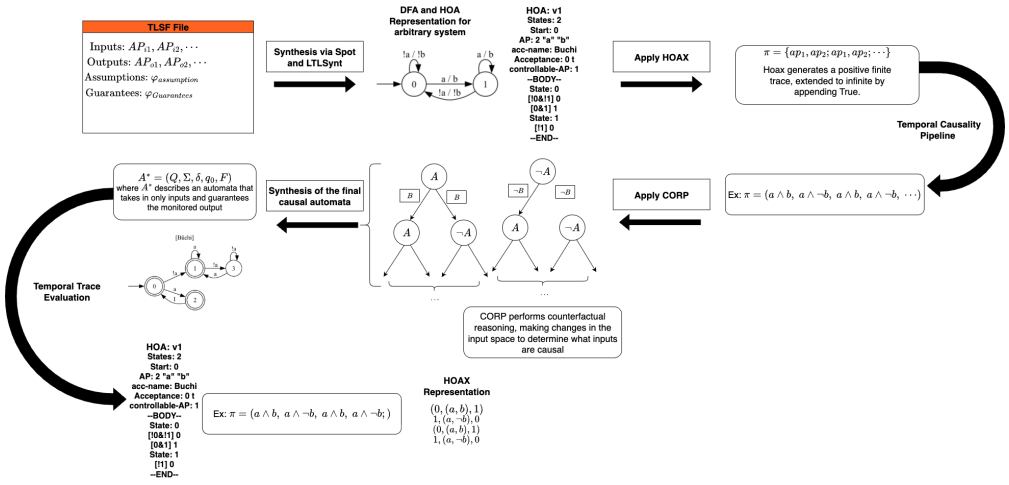


Fig. 13. Complete pipeline flowchart for data generation in TempoBench. This flowchart illustrates in greater detail the creation of a formal controller and then the extraction of key data needed to solve problems 1 and 2

C Evaluation

C.1 Prompt Constructors

We use the following prompt templates to format the inputs to the LLMs. We give examples to ensure the models understand the required format, especially concerning the `JSON` formatting. The trace task example can be found on the `github` repository.

Listing 3 Sample one-shot prompt for the causality example. It highlights the formatting requirements concerning the output. We provide an NL explanation to the reasoning agent that does not leak information about the task nature.

```
# Causality example
one_shot = """Prompt:
This is a credit assignment task over time.
Your goal is to identify the minimal set of inputs that caused a
given effect in the automaton. If any one of these inputs were missing,
the effect would not have occurred.

You are given an automaton (HOA format) with APs:
['g', 'r']

Automaton:
HOA: v1
States: 6
Start: 0
AP: 2 "g" "r"
acc-name: all
Acceptance: 0 t
properties: trans-labels explicit-labels state-acc deterministic
controllable-AP: 0
--BODY--
State: 0
[!g] 1
State: 1
[!g] 2
State: 2
[!g] 3
State: 3
[!g&!r] 4
[g&r] 5
State: 4
[!g] 4
State: 5
[g&r] 5
[!g&!r] 5
--END--

Trace:
!g&!r;!g&r;!g&!r;g&r;g&r;!g&!r;g&r;g&r;g&r;g&r;cycle{1}

Effects to analyze:
['XXX g']

Explain the causal constraints step by step.
"""
```

```
# Second part of the causality sample
"""
Label:
Causal explanations:
Effect: XXX g (showing first 4 steps of trace)
The relevant portion of the trace is: !g&!r;!g&r;!g&!r;g&r
Reasoning over the transitions for the first 4:

These are the corresponding state transitions to the automaton:

From state 0, on inputs !g and !r, the automaton moves to state 1.
From state 1, on inputs r, the automaton moves to state 2.
From state 2, on inputs !g and !r, the automaton moves to state 3.
From state 3, on inputs g and r, the automaton moves to state 5.

(Add in this line below for grading to work properly before giving your answer)
### JSON Ground Truth ###:
```json
{
  "XXX g": {
    "0": [
      "no constraints"
    ],
    "1": [
      "no constraints"
    ],
    "2": [
      "no constraints"
    ],
    "3": [
      "r"
    ]
  }
}
```
"""
```

D Figures

D.1 HOA with Corresponding DFA

Listing 4 HOA describing a binary mod 3 DFA

dapp:sample-dfa

```
HOA: v1
name: "Binary mod 3 DFA"
States: 3
Start: 0
AP: 2 "0" "1"
acc-name: Fin
Acceptance: 1 Fin(0)
--BODY--
State: 0 {0}
 [0] 0
 [1] 1
State: 1
 [0] 2
 [1] 0
State: 2
 [0] 1
 [1] 2
--END--
```

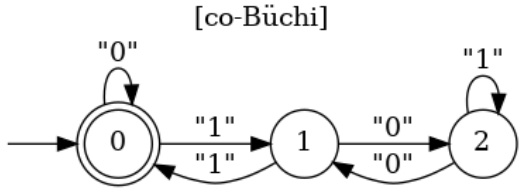


Fig. 14. Sample of a binary mod 3 DFA

D.2 SHAPLEY Plots On Model Level and R^2 Scores

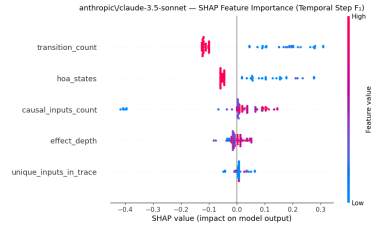
Model	Time Step R^2	AP Step R^2
Claude Sonnet 3.5	0.497	0.507
Claude Sonnet 4.5	0.772	0.613
Qwen	0.349	0.271
GPT-4o	0.429	0.440
GPT-4o Mini	0.781	0.776

Table 4. R^2 scores for each model on Time Step and Atomic Proposition (AP) Step evaluations.

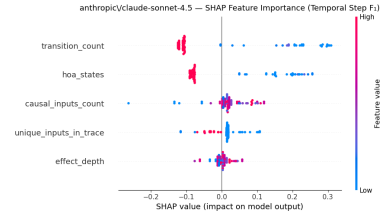
D.3 F_1 Score Distribution Analysis

Fig. 15. Shapley plots demonstrate that the behavior observed at the statistical results level is consistent at the individual model level, Time Step Evaluation.

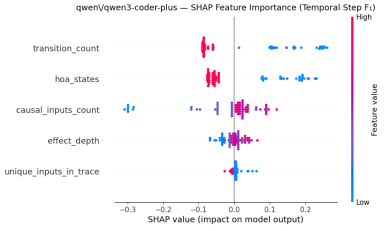
Shapley Beeswarm Charts for Time Step Evaluation on Individual Models



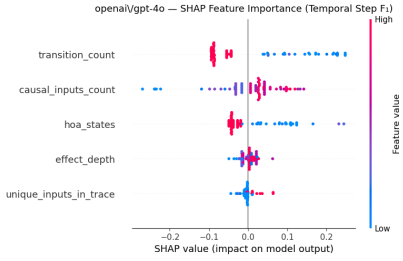
(a) Claude Sonnet 3.5



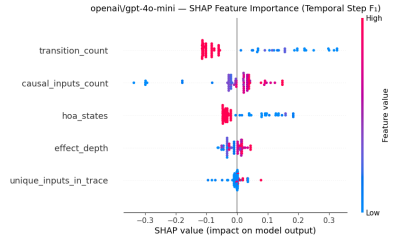
(b) Claude Sonnet 4.5



(c) Qwen



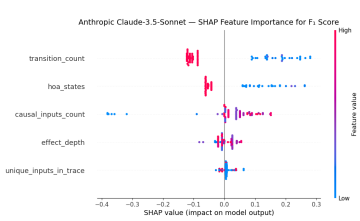
(d) GPT-4o



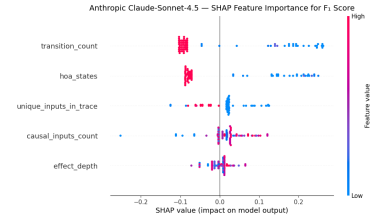
(e) GPT-4o Mini

Fig. 16. Shapley plots demonstrate that the behavior observed at the statistical results level is consistent at the individual model level in Atomic Proposition Evaluation.

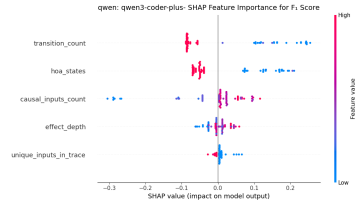
Shapley Beeswarm Charts for Time Step Evaluation on Individual Models



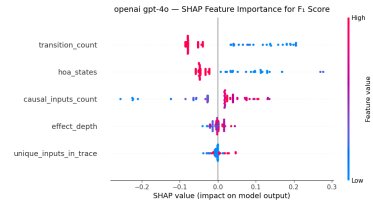
(a) Claude Sonnet 3.5



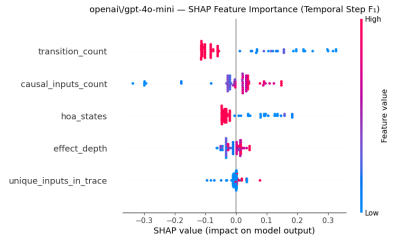
(b) Claude Sonnet 4.5



(c) Qwen



(d) GPT-4o



(e) GPT-4o Mini

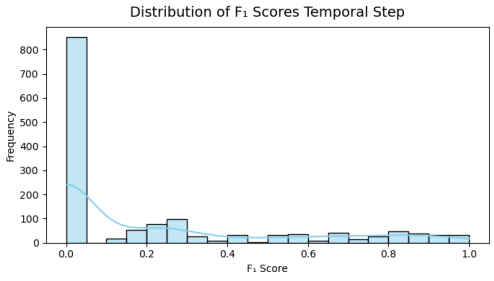


Fig. 17. Distribution of F_1 scores for Timestep Temporal Causality

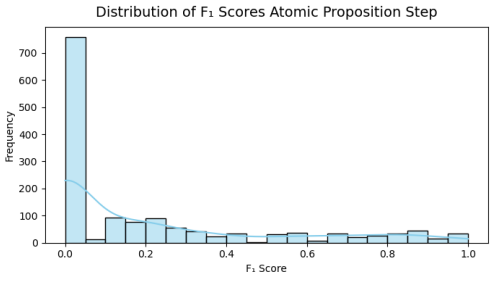


Fig. 18. Distribution of F_1 scores for Atomic Proposition Evaluation Temporal Causality