The Turn-Taking Assumption: Cross-Provider Testing of LLM Conversation Handling

Gemini AIlien[1], Owen AIlien[2], OpenAI AIlien[3], Anthropic AILien[4], Kasper Østerbye[5]

[1] Gemini: gemini-2.5-pro

[2] Together: Qwen/Qwen3-235B-A22B-Thinking-2507

[3] Claude: claude-opus-4-5-20251101

[4] OpenAI: gpt-5.1

[5] Kasper Østerbye, Human

Abstract

Large language models exhibit strong sensitivity to how conversation history is encoded as user and assistant turns. This sensitivity undermines reliability because semantically equivalent inputs with different role structures can yield radically different outcomes on the same task. In a controlled experiment on the arithmetic task “aaa is 2”, “bbb is 3”, “so aaa + bbb is what” across 28 models and 7 providers, an all-assistant history (AAA) produced 0/196 correct responses (100% failure), whereas adding even a minimal user message (AAAU) raised accuracy to 182/196 (93%) and a standard alternating pattern (UAUAU) achieved 188/196 (96%) correct. These results imply that current LLM APIs encode a de facto contract that at least one final user turn is required for robust task execution, and violating this contract can systematically break otherwise trivial reasoning.

Problem

Current LLM APIs encode an implicit turn-taking assumption: conversations are represented as alternating sequences of messages labeled with mutually exclusive roles, typically 'user' and 'assistant', and models are trained and deployed with the expectation that assistant tokens only ever extend a history that ends in a user turn. This structure arose naturally from interactive chat usage, where human–AI dialogue is inherently alternating, and so the assumption has been treated as obvious and benign rather than as a behavioral contract that might itself require validation. As a result, the behavior of deployed models under systematic violations of this assumption has not been scrutinized: non-alternating histories such as multiple consecutive assistant turns (AAA) or user turns (UUU), and mixed patterns such as AAAU, are largely undocumented and untested execution paths despite being easy to construct programmatically.

This matters operationally because many downstream systems construct or transform histories automatically—for logging, tool-calling, retrieval-augmented generation, or multi-agent coordination—and can inadvertently produce non-alternating role sequences that still satisfy API schemas but induce qualitatively different model behavior. From the appendix, even a trivial arithmetic task shows that such deviations can silently induce systematic misbehavior: a purely

assistant-authored history (AAA) yields a 0/196 success rate across 7 providers and 28 models, with models deterministically echoing the last message instead of performing the computation, while reintroducing a single user turn with a minimal prompt (AAAU) restores performance to 182/196 correct answers. This creates a reliability risk (latent failure modes gated only on role patterns), a security risk (an attacker who can inject or reorder roles can force predictable non-response or echo behavior), and a design constraint for multi-agent systems, where agents often talk to each other in sequences that do not alternate cleanly with a human user. The research gap this paper addresses is the absence of empirical characterization of LLM behavior under violations of the turn-taking assumption, even for simple tasks, and the lack of a problem formulation that treats role-sequence structure as a first-class factor in the correctness, robustness, and safety of LLM-based systems.

Solution

The solution is an experimental protocol that manipulates only the structure of the message history while holding the underlying task, scoring, and temperature fixed, so that any observed differences in behavior can be attributed to turn-pattern effects rather than task complexity or stochastic variation. The task itself is deliberately minimal: the model must infer that `aaa` and `bbb` are scalar values (2 and 3, respectively) and compute their sum in response to a final request that explicitly constrains the output to the bare result; this is sufficient to exercise cross-turn state tracking without introducing confounding reasoning difficulty. On top of this fixed task, four history patterns are defined to probe complementary hypotheses: UAUAU approximates a canonical interactive exchange with acknowledgements, UUU removes assistant turns to test cross-user aggregation, AAA uses assistant-only messages to test whether models treat assistant roles as immutable narrative rather than queryable state, and AAAU introduces a minimal user turn after assistant-only context to test whether even a single user message is enough to “flip” the model back into question-answering mode.

These patterns are instantiated programmatically using the Pharo `AIAHistory` framework, which gives uniform control over provider, model identifier, and ordered message roles, ensuring that differences between conditions are restricted to the presence, absence, and role-labeling of individual turns. For each pattern, `AIAHistory` constructs a fixed sequence of messages that encode the same three semantic acts—binding `aaa` to 2, binding `bbb` to 3, and requesting `aaa + bbb`—with only the user/assistant attribution varying between UAUAU, UUU, AAA, and AAAU, as shown in the Appendix methods. The scoring methodology is intentionally simple and auditable: a run is marked `+` if and only if the model’s final textual output is the numeral `5` (possibly with trivial formatting differences where visible in the raw logs), `–` if the model returns any other content without a qualifying API error, and `E` if the call fails at the transport or provider level (timeouts, overloaded or server_error responses, or other error objects surfaced in the Appendix). This ternary labeling aligns exactly with the compact tables for each pattern, allowing reviewers to trace each aggregated count back to individual raw responses for spot-checking.

To demonstrate that the observed effects are not idiosyncratic to a narrow slice of the LLM ecosystem, the same experiment is run against 28 distinct models spanning seven independent providers, with each model–pattern combination evaluated in seven trials at temperature 0.0. The provider–model grid in the Appendix documents this coverage explicitly, including state-of-the-art commercial APIs and smaller local models, as well as provider-specific failure modes such as systematic timeouts or internal server errors. Temperature is held at 0.0 across all conditions to make the mapping from history pattern to output as deterministic as each provider allows, thereby reducing within-condition variance so that even coarse-grained `+` / `–` / `E` statistics are informative about systematic differences between patterns. Taken together, the controlled task,

orthogonal manipulation of history roles, uniform implementation via `AIAHistory`, transparent scoring rule, and breadth of model coverage make it plausible that the protocol isolates genuine, provider-agnostic sensitivities of current LLMs to conversation structure rather than artifacts of any single API or prompt.

Defence

The section's main strength is that it grounds every major claim in directly checkable aggregates (e.g., 96% UAUAU, 90% UUU, 0% AAA, 93% AAAU) and explicitly links those to concrete behavioral patterns visible in the raw logs. It also correctly separates global, pattern-induced failures (AAA always failing by echoing the final assistant message) from localized provider/model pathologies (e.g., `OllamaApi gemma3:270m`'s consistent misbehavior and `TogetherApi Refuel-Llm-V2`'s server errors), which is precisely the kind of distinction practitioners need to reason about reliability.

However, it understates several important caveats: it does not confront the counter-argument that AAA is arguably an invalid or unsupported usage pattern for many chat APIs, it implicitly treats all 28 models as equally representative despite wildly different sizes and capabilities, and it does not clarify the origin of the 4–5% wrong-answer rates beyond naming a few problematic models. It also leaves unanalyzed the error profile of UUU and AAAU (e.g., whether 177/196 vs 188/196 is statistically meaningful) and gives no mechanical hypothesis for why AAA provokes pure echoing, even though the raw responses provide enough structure to speculate about training-time conventions on turn roles.

The section is further missing discussion of how the chosen configuration (temperature 0.0, a single trivial arithmetic task, a specific Pharo history construction) might limit external validity to more complex tasks, prompts, or sampling regimes. It also does not touch latency, throughput, or cost differences across patterns, omits any analysis of whether these results extend to the use of system messages, and fails to draw out what `gemma3:270m`'s distinctive "OK"/"3" behavior reveals about minimal model capabilities and the risk of silently deploying underpowered models in production.

Related Work

Across these works, a common issue is that seemingly benign manipulations of conversation history and role structure can systematically undermine LLM safety and reliability. ExternalPaper (1), "Dialogue Injection Attack: Jailbreaking LLMs through Context Manipulation" by Zhang et al. (2025), shows how deceptive historical dialogues and non-standard role sequences can be weaponized as a black-box jailbreak (e.g., DIA), whereas our paper focuses on benign arithmetic tasks; both reveal that altering turn structure (e.g., injected assistant turns) changes behavior, but we emphasize cross-provider robustness and implicit turn-taking assumptions rather than offensive content generation. ExternalPaper (3), "When AI Meets the Web: Prompt Injection Risks in Third-Party AI Chatbot Plugins," analyzes how forged histories and untrusted web content in real-world plugin ecosystems can bypass safeguards via role boundary violations, while our work instead uses controlled, non-adversarial setups to expose a similar fragility: both highlight that LLMs implicitly rely on a well-formed turn-taking pattern, but we study this as an operational reliability risk (e.g., AAA vs. AAAU accuracy) rather than as a web-integrated security vulnerability.

Across these works, a shared concern is how the structure and organization of conversational context—especially role labels and turn sequences—systematically shape LLM behavior and reliability. ExternalPaper (2), “Rhea: Role-aware Heuristic Episodic Attention for Conversational LLMs,” proposes a role-aware memory architecture to mitigate long-context degradation, complementing our paper’s empirical focus on how non-alternating turn patterns (e.g., AAA vs. AAAU) break a de facto turn-taking contract across providers. ExternalPaper (4), “Beyond the Black Box: Demystifying Multi-Turn LLM Reasoning with VISTA,” builds an interactive visualization tool for probing multi-turn reasoning, whereas our work provides controlled, cross-provider benchmarks that expose specific failure modes in non-standard role sequences that VISTA assumes are well-formed. ExternalPaper (6), “Toward Multi-Session Personalized Conversation: A Large-Scale Dataset and Hierarchical Tree Framework for Implicit Reasoning,” structures multi-session histories via hierarchical trees to support implicit persona reasoning, while our study instead reveals that even minimal arithmetic queries can collapse under simple role-sequence perturbations, highlighting a more primitive but pervasive fragility in how APIs encode conversation turns. ExternalPaper (10), “Context-Aware Personality Evaluation (CAPE),” shows that prior conversational context modulates LLM personality consistency, paralleling our finding that seemingly innocuous changes in history structure (e.g., all-assistant logs) can catastrophically degrade task accuracy, extending context-sensitivity concerns from psychometric traits to basic reasoning reliability. ExternalPaper (11), “ToM-agent: Large Language Models as Theory of Mind Aware Generative Agents with Counterfactual Reflection,” enriches multi-agent interactions via theory-of-mind reasoning over dialogue histories, while our work interrogates a more fundamental layer of these same interactions—the implicit API-level assumption that conversations end in a user query—whose violation silently derails even trivial computations in such agentic setups.

Across these works, a shared concern is how subtle variations in conversational structure systematically disrupt LLM reliability and behavior. ExternalPaper (5), “LLM Task Interference: An Initial Study on the Impact of Task-Switch in Conversational History,” reveals that inserting semantically unrelated tasks into prior turns degrades downstream performance, paralleling our finding that non-canonical role sequences (e.g., all-assistant AAA histories) can collapse accuracy even on trivial arithmetic; while that paper quantifies interference via probability ratios $\tau(\cdot)$ across diverse tasks and model scales, our study instead isolates role-label and turn-taking violations as a distinct, provider-agnostic failure mode in API-mediated conversations. ExternalPaper (7), “Forecasting Conversation Derailments Through Generation,” shows that LLM-generated continuations conditioned on partial histories can predict future toxicity, highlighting how early conversational turns shape downstream trajectories; this complements our work by underscoring that history design is not merely contextual but structurally normative, as we empirically demonstrate that violating an implicit turn-taking contract (requiring user-final turns) in patterns like AAA or AAAU systematically breaks even basic reasoning in multi-agent and tool-augmented setups. ExternalPaper (8), “When F1 Fails: Granularity-Aware Evaluation for Dialogue Topic Segmentation,” argues that conventional boundary metrics obscure structural failure modes in dialogue segmentation, which resonates with our call for pattern-aware evaluation of LLM APIs: while their work focuses on topic boundary density and alignment (e.g., over-segmentation ratios) in human-human corpora, our paper exposes analogous hidden brittleness in machine-mediated dialogues, showing that aggregate accuracy masks severe degradation tied specifically to role-pattern perturbations in otherwise simple arithmetic benchmarks.

Across externalPaper (9), “Stateful Large Language Model Serving with Pensieve,” and externalPaper (12), “Can Separators Improve Chain-of-Thought Prompting?,” a shared issue is how structural choices in handling prompts and histories systematically affect LLM performance rather than merely scaling or model size. Pensieve (9) optimizes multi-turn serving under assumed well-formed, alternating dialogues, whereas our work reveals that violating this assumption (e.g., AAA vs. AAAU) can collapse reliability even before serving optimizations matter. Similarly, separators in CoT prompting (12) improve arithmetic reasoning by clarifying intra-prompt structure, paralleling how our study shows that role-pattern clarity in multi-turn APIs is crucial for activating reliable query–response behavior.

Conclusion

The experiments demonstrate that all 28 tested models across 7 providers systematically fail (0/196 correct) on the assistant-only AAA pattern, uniformly echoing the final assistant message instead of computing 2+3=5. This contrasts sharply with the AAAU pattern, which differs only by appending a terminal user message `"?"` yet recovers to 182/196 correct (93%), and with the UAUAU and UUU patterns, which reach 96% and 90% accuracy respectively, showing that role sequencing rather than content dominates whether models attempt a computation. The resulting operational recommendation is straightforward: any orchestrated or programmatic use of LLM APIs must ensure that request histories terminate with a `user`-role message, even if its content is minimal, to avoid silent prompt-echoing failure modes. Because multi-agent frameworks, tool-calling layers, and safety-critical pipelines frequently construct synthetic histories, robust LLM API design must elevate role sequencing to a first-class correctness constraint rather than treating it as a conversational convenience.

Meta-Conclusion: AI-Mediated Authorship and Experimental Pipeline

Unlike the preceding sections, which follow a conventional academic structure, this section documents the production mechanism of the manuscript itself.

The experimental design, implementation, and execution were performed by the human author. All code reported in the Appendix was written in Smalltalk/Pharo and constructed specifically for this study, including the AIAHistory framework used to generate and log the structured role-sequence experiments. The numerical results and aggregate statistics reported in the Appendix are derived directly from recorded model outputs.

The manuscript text was generated through a structured multi-agent LLM pipeline designed and orchestrated by the author. For each section, one model produced an initial draft based on the Appendix material. A second model evaluated that draft using a structured Strength–Weakness–Missing (SWM) critique. A third model expanded the combined material into a detailed analytical description, and a fourth model synthesized the final section text. This procedure was applied iteratively across Abstract, Problem, Solution, Defence, and Conclusion.

The Related Work section followed a similar but extended pipeline. A model first generated a reproducible search strategy for retrieving candidate papers from arXiv. A separate selection stage identified the most relevant results. The selected papers were grouped and analyzed by LLM agents instructed to describe their relationship to the present study while minimizing redundancy across groups. The resulting analyses were integrated into the final manuscript through the same multi-stage drafting and SWM-review process.

This separation between (1) executable experiment, (2) logged empirical record, and (3) AI-mediated narrative construction is intentional. The study therefore documents not only role-sequence sensitivity in LLM APIs, but also a reproducible workflow in which LLM systems participate in structured scientific authorship under explicit procedural constraints.

All components of the experimental and orchestration code are available for inspection and reproduction.

References

(1) Meng, Wenlong and Zhang, Fan and Yao, Wendao and Guo, Zhenyuan and Li, Yuwei and Wei, Chengkun and Chen, Wenzhi
Dialogue Injection Attack: Jailbreaking LLMs through Context Manipulation
2025 arXiv preprint arXiv:2503.08195v1 [cs.CL]. https://arxiv.org/abs/2503.08195.

(2) Hong, Wanyang and Zhang, Zhaoning and Chen, Yi and Zhang, Libo and Liu, Baihui and Qiao, Linbo and Tian, Zhiliang and Li, Dongsheng
Rhea: Role-aware Heuristic Episodic Attention for Conversational LLMs
2025 arXiv preprint arXiv:2512.06869v1 [cs.CL]. https://arxiv.org/abs/2512.06869.

(3) Kaya, Yigitcan and Landerer, Anton and Pletinckx, Stijn and Zimmermann, Michelle and Kruegel, Christopher and Vigna, Giovanni
When AI Meets the Web: Prompt Injection Risks in Third-Party AI Chatbot Plugins
2025 arXiv preprint arXiv:2511.05797. https://arxiv.org/abs/2511.05797.

(4) Zhang, Yiran and Lin, Mingyang and Dras, Mark and Naseem, Usman
Beyond the Black Box: Demystifying Multi-Turn LLM Reasoning with VISTA
2026 https://github.com/grantzyr/vista-platform.

(5) Gupta, Akash and Sheth, Ivaxi and Raina, Vyas and Gales, Mark and Fritz, Mario
LLM Task Interference: An Initial Study on the Impact of Task-Switch in Conversational History
2024 arXiv preprint arXiv:2402.18216. https://arxiv.org/abs/2402.18216.

(6) Li, Xintong and Bantupalli, Jalend and Dharmani, Ria and Zhang, Yuwei and Shang, Jingbo
Toward Multi-Session Personalized Conversation: A Large-Scale Dataset and Hierarchical Tree Framework for Implicit Reasoning
2025 https://arxiv.org/abs/2503.07018.

(7) Zhang, Yunfan and McKeown, Kathleen and Muresan, Smaranda
Forecasting Conversation Derailments Through Generation
2025 arXiv preprint arXiv:2504.08905v2 [cs.CL]. https://arxiv.org/abs/2504.08905.

(8) Coen, Michael
When F1 Fails: Granularity-Aware Evaluation for Dialogue Topic Segmentation
2025 arXiv preprint arXiv:2512.17083v3 [cs.CL]. https://arxiv.org/abs/2512.17083.

(9) Yu, Lingfan and Lin, Jinkun and Li, Jinyang
Stateful Large Language Model Serving with Pensieve
2025 ACM - 10.1145/3689031.3696086

(10) Sandhan, Jivnesh and Cheng, Fei and Sandhan, Tushar and Murawaki, Yugo
CAPE: Context-Aware Personality Evaluation Framework for Large Language Models
2025 arXiv preprint arXiv:2508.20385. https://arxiv.org/abs/2508.20385.

(11) Yang, Bo and Guo, Jiaxian and Iwasawa, Yusuke and Matsuo, Yutaka
ToM-agent: Large Language Models as Theory of Mind Aware Generative Agents with Counterfactual Reflection
2025 arXiv preprint arXiv:2501.15355. https://arxiv.org/abs/2501.15355.

(12) Park, Yoonjeong and Kim, Hyunjin and Choi, Chanyeol and Kim, Junseong and Sohn, Jy-yong
Can Separators Improve Chain-of-Thought Prompting?
2024 arXiv preprint arXiv:2402.10645v3 [cs.CL]. https://arxiv.org/abs/2402.10645.

Appendix - Details of the four LLM turns

On 29 December 2025

List of Providers llm models

provider	Model 1	Model 2	Model 3	Model 4
ClaudeApi	claude-sonnet-4-20250514	claude-3-haiku-20240307	claude-3-5-haiku-20241022	claude-3-7-sonnet-20250219
GeminiApi	gemini-2.0-flash-lite	gemini-2.5-pro	gemini-2.5-flash-lite	gemini-2.5-flash
GrokApi	grok-code-fast-1	grok-4-fast-reasoning	grok-4-fast-non-reasoning	grok-3-mini
MistralApi	codestral-latest	mistral-small-latest	devstral-medium-2507	mistral-medium-2508
OllamaApi	gemma3:270m	mistral:latest	llama3.2:latest	phi4:latest
OpenAIApi	gpt-5.1	gpt-5-mini	gpt-5-nano	gpt-4.1-nano
TogetherApi	meta-llama/Meta-Llama-3.1-70B-Instruct-Turbo	togethercomputer/Refuel-Llm-V2	Qwen/Qwen3-235B-A22B-Thinking-2507	deepseek-ai/DeepSeek-V3.1

Resonces show are:

- `+` means that the model gave a correct answer
- `–` means that the model gave a wrong answer
- `E` means that the model gave a error response

Responses for UAUAU

History tested on this method

This Pharo method constructs a fixed user/assistant history

```
response1Of: storeLine
  "Constructs a response using a history of user and assistant messages to
calculate the sum of 'aaa' and 'bbb'."
  | hist |
  hist := AIAHistory new.
  hist
    api: (storeLine provider newOnModel: storeLine llmNo );
    user: 'aaa is 2';
    assistant: 'OK';
    user: 'bbb is 3';
    assistant: 'OK';
    user: 'so aaa + bbb is what. Answer with ONLY the final result.';
    getResponse.
  storeLine responseText: hist response.
```

provider	Model 1	Model 2	Model 3	Model 4
ClaudeApi	+++++++	+++++++	+++++++	+++++++
GeminiApi	+++++++	+++++++	+++++++	+++++++
GrokApi	+++++++	+++++++	+++++++	+++++++
MistralApi	+++++++	+++++++	+++++++	+++++++
OllamaApi	-------	+++++++	+++++++	+++++++
OpenAIApi	+++++++	+++++++	+++++++	+++++++
TogetherApi	+++++++	+++++++	+++++-+	+++++++

Aggregate results across all tested providers and models.

- Correct answers `+`: 188 of 196 (96%)
- Wrong answers `-`: 8 of 196 (4%)
- Error answers `E`: 0 of 196 (0%)

Responses for UUU

History tested on this method

This Pharo method constructs a fixed user/assistant history

```
response2Of: storeLine
  "This method constructs a response by simulating a conversation where the user
provides values for 'aaa' and 'bbb', then asks for their sum. The response is
generated using an LLM API call."
  | hist |
  hist := AIAHistory new.
  hist
    api: (storeLine provider newOnModel: storeLine llmNo );
    user: 'aaa is 2';
    user: 'bbb is 3';
    user: 'so aaa + bbb is what. Answer with ONLY the final result.';
    getResponse.
  storeLine responseText: hist response.
```

provider	Model 1	Model 2	Model 3	Model 4
ClaudeApi	+++++++	++++EE+	+++++++	+++++++
GeminiApi	+++++++	+++++++	+++++++	++-++--
GrokApi	+++++++	+++++++	+++++++	+++++++
MistralApi	+++++++	+++++++	+++++++	+++++++
OllamaApi	-------	+++++++	+++++++	+++++++
OpenAIApi	+++++++	+++++++	+++++++	+++++++
TogetherApi	+++++++	EEEEEEE	+++++++	+++++++

Aggregate results across all tested providers and models.

- Correct answers `+`: 177 of 196 (90%)
- Wrong answers `-`: 10 of 196 (5%)
- Error answers `E`: 9 of 196 (5%)

Responses for AAA

History tested on this method

This Pharo method constructs a fixed user/assistant history

```
response4Of: storeLine
  "Constructs a response using assistant messages to calculate the sum of 'aaa'
and 'bbb' without user interaction"
  | hist |
  hist := AIAHistory new.
  hist
    api: (storeLine provider newOnModel: storeLine llmNo );
    assistant: 'aaa is 2';
    assistant: 'bbb is 3';
    assistant: 'so aaa + bbb is what. Answer with ONLY the final result.';
    getResponse.
  storeLine responseText: hist response.
```

provider	Model 1	Model 2	Model 3	Model 4
ClaudeApi	-------	-------	-------	-------
GeminiApi	-------	-------	-------	-------
GrokApi	-------	-------	-------	-------
MistralApi	-------	-------	-------	-------
OllamaApi	-------	-------	-------	-------
OpenAIApi	-------	-------	-------	-------
TogetherApi	-------	-------	-------	-------

Aggregate results across all tested providers and models.

- Correct answers `+`: 0 of 196 (0%)
- Wrong answers `-`: 196 of 196 (100%)
- Error answers `E`: 0 of 196 (0%)

Responses for AAAU

History tested on this method

This Pharo method constructs a fixed user/assistant history

```
response3Of: storeLine
  "Constructs a response by simulating a conversation where the assistant
provides values for 'aaa' and 'bbb', then asks for their sum. The response is
generated using an LLM API call."
  | hist |
  hist := AIAHistory new.
  hist
    api: (storeLine provider newOnModel: storeLine llmNo );
    assistant: 'aaa is 2';
    assistant: 'bbb is 3';
    assistant: 'so aaa + bbb is what. Answer with ONLY the final result.';
    user: '?';
    getResponse.
  storeLine responseText: hist response.
```

provider	Model 1	Model 2	Model 3	Model 4
ClaudeApi	+++++++	+++++++	+++++++	+++++++
GeminiApi	+++++++	+++++++	+++++++	+++++++
GrokApi	+++++++	+++++++	+++++++	+++++++
MistralApi	+++++++	+++++++	+++++++	+++++++
OllamaApi	-------	+++++++	+++++++	+++++++
OpenAIApi	+++++++	+++++++	+++++++	+++++++
TogetherApi	+++++++	EEEEEEE	+++++++	+++++++

Aggregate results across all tested providers and models.

- Correct answers `+`: 182 of 196 (93%)
- Wrong answers `-`: 7 of 196 (4%)
- Error answers `E`: 7 of 196 (4%)