ATLAS-RTC: Stage-Aware Closed-Loop Token-Level Control for Structured LLM Output

Anonymous ACL submission

Abstract

LLM agents increasingly depend on structured outputs: tool-call payloads, JSON objects, schemas, and API request bodies. Existing constrained decoding systems such as XGrammar and Outlines provide strong syntactic guarantees by applying grammar-derived masks throughout generation, but uniform masking can be unnecessarily restrictive in value-generation zones where semantic content matters. We present ATLAS-RTC, a token-level runtime controller that tracks JSON generation phase with a stage-aware state machine and applies intervention only at structural decision points. On a 757-task benchmark spanning JSON schema generation, value-zone quality, agent tool calls, and reasoning-structured output, evaluated on Qwen2.5-7B-Instruct and Llama-3.1-8B-Instruct, ATLAS-RTC matches grammar-constrained decoding on first-attempt executable correctness for tool calls while reducing median tool-call latency on Qwen2.5-7B-Instruct from 1,576 ms with XGrammar to 533 ms. On value-rich schemas, ATLAS-RTC improves field-level value quality over XGrammar from 0.243 to 0.405 on Qwen2.5-7B-Instruct and from 0.168 to 0.378 on Llama-3.1-8B-Instruct. However, hard grammar methods remain stronger for pure schema validity, especially on nested and array-heavy schemas. We position ATLAS-RTC as a latency- and content-preserving runtime control layer for settings where structural reliability and value fidelity must be jointly optimized.

1 Introduction

LLM agents are deployed in pipelines where generated outputs are consumed by downstream systems. These outputs often must satisfy a formal interface: a tool-call schema, a JSON schema, an API request body, or a structured reasoning format. When outputs are malformed, downstream systems typically retry, repair, reject, or fail silently. Each option has a cost: retries increase latency and compute, repairs introduce ambiguity, rejections harm reliability, and silent failures can propagate incorrect state.

Structured output generation is commonly addressed through prompt engineering, post-hoc validation, fine-tuning, or constrained decoding. Prompt engineering has no mechanism to intervene when a bad token is about to be selected. Post-hoc validation detects failures only after they occur. Fine-tuning requires task-specific data and does not provide formal guarantees. Static constrained decoding systems, including XGrammar and Outlines, compile grammars or schemas into token masks that prevent syntactically invalid continuations. This approach is powerful for validity, but it applies the same enforcement mechanism across structural regions and value regions. In value regions, such as free-form strings or numeric fields, the model may need flexibility to express content naturally.

This paper studies a complementary approach: stage-aware runtime control. ATLAS-RTC treats structured generation as a closed-loop process. At each decoding step, a controller observes the partial output, estimates whether the generation trajectory is drifting toward a structural failure, and applies the weakest intervention likely to restore compliance. The central design choice is to separate structural decision points from value-generation zones.

We make the following contributions:

- A stage-aware controller that distinguishes key strings, value strings, structural junctions, and nested content.
- A graduated intervention ladder combining logit biasing, temperature reduction, token masking, and KV-cache rollback.
- A four-block, 757-task benchmark covering schema validity, value-zone quality, tool-call execution, and reasoning-structured output.

- An empirical tradeoff analysis: ATLAS-RTC improves value-zone field quality and tool-call latency, while hard grammar methods remain strongest for pure schema validity.

2 Related Work

Constrained decoding. PICARD (Scholak et al., 2021) incrementally parses SQL during generation. Grammar-constrained decoding for structured NLP tasks has been studied without fine-tuning (Geng et al., 2023). Outlines (Willard and Louf, 2023), DOMINO-style constrained generation (Beurer-Kellner et al., 2024), and XGrammar (Dong et al., 2025) generalize constrained generation to JSON-like and schema-constrained outputs. JSONSchemaBench (Geng et al., 2025) benchmarks structured output generation over real-world JSON schemas.

Distributional effects of constraints. Hard constraints can improve validity while changing the model's local token distribution (Park et al., 2024). ATLAS-RTC addresses this through stage awareness: intervene near structural decisions and back off in value zones.

Reasoning-aware and draft-conditioned decoding. CRANE (Banerjee et al., 2025) examines constrained LLM generation for reasoning. Draft-conditioned constrained decoding (Reddy et al., 2026) uses auxiliary drafts to condition structured generation. These approaches are complementary to ATLAS-RTC, which does not require a draft model or multi-stage generation process.

3 System Architecture

3.1 Problem Formulation

Let $x_{1:t}$ be the partial output sequence and $z_t \in \mathbb{R}^{|V|}$ the model logits at step t. Standard decoding samples $x_{t+1} \sim \mathrm{Softmax}(z_t)$. ATLAS-RTC applies a control signal u_t before sampling:

$$x_{t+1} \sim \mathrm{Softmax}(z_t + u_t). \tag{1}$$

The objective is to maximize contract satisfaction while minimizing intervention cost:

$$\max_{u_{1:T}} \mathbb{E}[V(x_{1:T})] - \lambda \cdot \mathrm{Cost}(u_{1:T}), \tag{2}$$

where V is a validation function and λ trades off reliability against intervention intensity.

3.2 Stage-Aware State Machine

The controller maintains a character-level parse state over the partially generated text. It tracks nesting depth, whether the generator is inside a string, whether the current string is a key or value, which required keys have been produced, and the current structural stage:

```
expect_open, expect_key, expect_colon,
expect_value,
in_value, expect_comma_or_close, closed.
```

The policy distinguishes structural junctions from value zones. It may intervene when the model is choosing an opening brace, key, colon, comma, or closing delimiter. It backs off while the model is producing a string value, number, boolean, null, or nested free-form content.

A practical detail is the anticipatory closing bias. When all required keys at the current tracked object have appeared, the controller boosts closing delimiters. This bias must be suppressed inside string values and during numeric generation; otherwise, the controller may inject closing braces inside values, producing artifacts such as `"devicePath": "/}}}"`.

3.3 Drift Detector and Calibration

At each step, ATLAS-RTC computes rolling-window features including entropy trend, invalid-token mass, steps since the last required key, output length, stage, and step fraction. A lightweight logistic classifier estimates failure probability from these features, combined with a heuristic structural risk score.

The detector must be calibrated per model family. Calibration collects 300 no-op trajectories, labels step records by eventual success or failure, trains a logistic classifier, and tunes the threshold for precision ≥ 0.80. On Qwen2.5-7B-Instruct: AUROC 0.708 at threshold 0.63. On Llama-3.1-8B-Instruct: AUROC 0.789, precision 0.858 at threshold 0.74.

During development, an early detector inverted the sign of the invalid-mass coefficient, causing the controller to intervene when generation was progressing well and back off when failing. This motivates validating any learned detector on held-out trajectories before deployment.

3.4 Graduated Intervention Ladder

Table 1 summarizes the action ladder. The most common intervention is structural biasing, especially anticipatory closing after required keys have been produced. Rollback truncates both the KV

Action	Trigger	Mechanism
No-op	Low risk	Do not alter logits
Bias	Structural junction	Additive logit bias
Temperature	High drift	Reduce temperature
Mask	Very high drift	Suppress invalid tokens
Rollback	Critical drift	Truncate KV cache, re-steer

Table 1: Graduated intervention ladder. The controller applies the weakest intervention available and backs off during value generation.

cache and generated token list; these must remain synchronized to avoid corrupted attention states.

4 Benchmark

4.1 Design

We evaluate on four blocks totaling 757 tasks:

Block A: Schema validity (217 tasks). Common-core schemas from JSONSchemaBench (Geng et al., 2025), stratified across 7 complexity levels. Metric: `schema_valid`.

Block B: Value-zone quality (60 tasks). Free-form value schemas requiring complex string content. Metrics: `schema_valid` and `field_accuracy`.

Block C: Tool-call execution (240 tasks). Tool-call generation across eight tool families evaluated with deterministic simulators. Metric: `exec_correct`.

Block D: Reasoning-structured output (240 tasks). Structured outputs for GSM8K, MATH500, GSM-Symbolic, and FOLIO-style reasoning. Metrics: `schema_valid` and `answer_correct`, reported separately because producing a well-formed output is not equivalent to solving the reasoning problem.

4.2 Models and Baselines

We evaluate Qwen2.5-7B-Instruct and Llama-3.1-8B-Instruct with per-model detector calibration. Baselines are XGrammar, Outlines, and prompt-only. All runs use greedy decoding across three seed-labeled runs for pipeline reproducibility. Zero variance under greedy decoding should not be interpreted as stochastic robustness.

5 Results

5.1 Main Results

Tables 2 and 3 show the main results. XGrammar remains best on Block A for both models. ATLAS-RTC's advantage appears in two regimes: tool-call latency at matched execution correctness, and value-zone quality in free-form schemas.

On Qwen2.5-7B-Instruct, prompt-only is already strong on several formatting tasks. ATLAS-RTC's primary advantage over prompt-only is value-zone field accuracy; its primary advantage over XGrammar is latency at matched tool execution. On Llama-3.1-8B-Instruct, prompt-only collapses across most blocks, and ATLAS-RTC provides a much larger reliability gain.

5.2 Tool-Call Execution and Latency

On Block C, ATLAS-RTC achieves 1.000 executable correctness on both model families, matching XGrammar and Outlines. Median tool-call latency on Qwen2.5-7B-Instruct is 533 ms for ATLAS-RTC and 1,576 ms for XGrammar (3× lower). When prompt-only is unreliable, stage-aware control can match hard constraints with substantially lower latency.

5.3 Value-Zone Quality

ATLAS-RTC obtains `field_accuracy` of 0.405 on Qwen2.5-7B-Instruct (vs. 0.243 for XGrammar, 0.208 for Outlines) and 0.378 on Llama-3.1-8B-Instruct (vs. 0.168, 0.149). Backing off during value generation preserves content quality better than uniform masking. This should be interpreted as field-level quality under the benchmark's deterministic scoring procedure, not as a guarantee of factual correctness.

5.4 Schema Validity and Stratum Analysis

XGrammar is strongest on Block A (0.912 on Qwen2.5-7B-Instruct, 0.943 on Llama-3.1-8B-Instruct). ATLAS-RTC reaches 0.737 and 0.765. Table 5 shows the stratum breakdown. Nested objects and arrays are the main weaknesses, consistent with the state machine tracking required keys at depth 1. ATLAS-RTC matches or exceeds all baselines on the free-form value stratum.

5.5 Reasoning-Structured Output

On Qwen2.5-7B-Instruct, XGrammar reaches perfect schema validity and slightly higher answer

Method	A: schema	B: schema	B: field	C: exec	D: schema	D: answer
ATLAS-RTC	0.737	**0.472**	**0.405**	**1.000**	0.908	0.658
XGrammar	**0.912**	0.400	0.243	**1.000**	**1.000**	**0.662**
Outlines	0.862	0.411	0.208	0.999	–	–
Prompt-only	0.871	0.467	0.368	**1.000**	0.883	0.633

Table 2: Results on Qwen2.5-7B-Instruct, averaged over three greedy seed-labeled runs.

Method	A: schema	B: schema	B: field	C: exec	D: schema	D: answer
ATLAS-RTC	0.765	**0.500**	**0.378**	**1.000**	0.600	**0.617**
XGrammar	**0.943**	0.428	0.168	**1.000**	**1.000**	0.583
Outlines	0.885	0.406	0.149	**1.000**	–	–
Prompt-only	0.009	0.017	0.017	0.013	0.042	0.054

Table 3: Results on Llama-3.1-8B-Instruct, averaged over three greedy seed-labeled runs.

Method	Block A (ms)	Block C (ms)
ATLAS-RTC	889	**533**
XGrammar	1610	1576
Outlines	1158	1232
Prompt-only	**860**	594

Table 4: Median per-task latency on Qwen2.5-7B-Instruct.

Stratum	ATLAS-RTC	XGrammar	Prompt
Arrays	0.628	**0.953**	0.907
Enum discriminator	0.714	**0.857**	0.800
Flat typed	0.909	**0.977**	0.955
Free-form value	**0.789**	0.737	**0.789**
Nested objects	0.444	**0.889**	0.667
Numeric constraints	0.711	**0.895**	0.816
Optional sparse	0.759	**0.966**	**0.966**

Table 5: Block A schema validity by stratum on Qwen2.5-7B-Instruct.

correctness (0.662 vs. 0.658). On Llama-3.1-8B-Instruct, XGrammar has perfect schema validity, but ATLAS-RTC has higher answer correctness (0.617 vs. 0.583) and far higher than prompt-only (0.617 vs. 0.054). Stage-aware control can substantially improve structured reasoning output over prompt-only on weaker instruction-following models.

6 Ablation Study

We ablate the graduated ladder by disabling temperature reduction, token masking, and KV-cache rollback, leaving only logit biasing active (bias-only). Table 6 compares full ladder against bias-only.

The full ladder helps most clearly on Qwen2.5-7B-Instruct Block C, where removing higher-intensity interventions lowers executable correctness from 1.000 to 0.988. However, on several

Model	Policy	A	B	C	D
Qwen2.5-7B	Full	0.737	0.472	**1.000**	**0.908**
Qwen2.5-7B	Bias	0.737	0.472	0.988	0.904
Llama-3.1-8B	Full	0.765	0.500	**1.000**	0.600
Llama-3.1-8B	Bias	**0.774**	**0.567**	1.000	**0.633**

Table 6: Full ladder vs. bias-only ablation. Reports `schema_valid` for A, B, D and `exec_correct` for C. Block D answer correctness is in Tables 2–3. Bold: better result per model-block pair.

Llama-3.1-8B-Instruct blocks, bias-only performs as well as or better than the full ladder, suggesting that high-intensity interventions can overcorrect when detector calibration is imperfect. This result does not isolate rollback alone, since the bias-only condition disables temperature reduction, masking, and rollback together. A schema-complexity-aware policy that enables higher-intensity interventions only when structural failure risk is clearly elevated is a direct camera-ready target.

7 Analysis

Where ATLAS-RTC wins. Block C tool-call execution under latency constraints is the clearest win. ATLAS-RTC matches hard-constrained methods on executable correctness and reduces median latency against XGrammar on Qwen2.5-7B-Instruct by $3\times$. Block B value-zone quality is the second win: backing off during free-form generation improves field-level content over uniform masking baselines on both models.

Where hard grammars remain better. For pure schema validity, XGrammar remains the strongest method, especially on nested and array-heavy schemas. If the deployment goal is only schema-

valid JSON and latency or content fidelity is secondary, hard grammar constraints are preferable.

Prompt-only is model-dependent. Prompt-only is strong on Qwen2.5-7B-Instruct for several blocks. It collapses on Llama-3.1-8B-Instruct. Runtime control should be evaluated as a reliability layer whose benefit depends on the model family and task distribution.

Detector calibration is a deployment requirement. Incorrect calibration can actively harm generation. Per-model calibration is a required deployment step.

8 Conclusion

We presented ATLAS-RTC, a stage-aware runtime controller for structured LLM output. The main result is a tradeoff, not a universal dominance claim. Hard grammar-constrained decoding remains best for pure schema validity, especially on nested and array-heavy schemas. ATLAS-RTC matches constrained methods on tool-call executable correctness with $3\times$ lower latency on Qwen2.5-7B-Instruct and improves value-zone field quality over grammar-masked baselines on both models. Block D results show why structured-output validity and reasoning answer correctness must be reported separately. Stage-aware token-level control is a useful complementary layer for LLM systems where structural reliability, latency, and value fidelity must be balanced.

9 Limitations

Nested schema support. The current implementation tracks required keys most reliably at depth 1. Required keys inside nested objects and arrays are not fully handled, explaining a substantial part of the Block A gap.

Metric limitations. `field_accuracy` is a deterministic benchmark metric. It rewards non-placeholder free-form content and field-level matches, but does not establish factuality. `answer_correct` captures exact or normalized final answers but does not measure reasoning trace quality.

Greedy decoding and seed labels. Seed-labeled runs under greedy decoding primarily verify pipeline reproducibility rather than stochastic robustness.

Comparison scope. CRANE and DCCD are discussed as related work but not directly evaluated. A complete version should compare against these systems on Block D.

References

Debangshu Banerjee, Tarun Suresh, Shubham Ugare, Sasa Misailovic, and Gagandeep Singh. 2025. CRANE: Reasoning with constrained LLM generation. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 2836–2857. PMLR.

Luca Beurer-Kellner, Marc Fischer, and Martin Vechev. 2024. Guiding LLMs the right way: Fast, non-invasive constrained generation. In *Proceedings of the 41st International Conference on Machine Learning*.

Yixin Dong, Charlie F. Ruan, Yaxing Cai, Ruihang Lai, Ziyi Xu, Yilong Zhao, and Tianqi Chen. 2025. XGrammar: Flexible and efficient structured generation engine for large language models. In *Proceedings of Machine Learning and Systems*, volume 7. ArXiv:2411.15100.

Saibo Geng, Hudson Cooper, Michał Moskal, Samuel Jenkins, Julian Berman, Nathan Ranchin, Robert West, Eric Horvitz, and Harsha Nori. 2025. Generating structured outputs from language models: Benchmark and studies. *arXiv preprint arXiv:2501.10868*.

Saibo Geng, Martin Josifoski, Maxime Peyrard, and Robert West. 2023. Grammar-constrained decoding for structured NLP tasks without finetuning. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 10932–10952. Association for Computational Linguistics.

Kanghee Park, Jiayu Wang, Taylor Berg-Kirkpatrick, Nadia Polikarpova, and Loris D'Antoni. 2024. Grammar-aligned decoding. In *Advances in Neural Information Processing Systems*.

Avinash Reddy, Thayne T. Walker, James S. Ide, and Amrit Singh Bedi. 2026. Draft-conditioned constrained decoding for structured generation in LLMs. *arXiv preprint arXiv:2603.03305*.

Torsten Scholak, Nathan Schucher, and Dzmitry Bahdanau. 2021. PICARD: Parsing incrementally for constrained auto-regressive decoding from language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9895–9901, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Brandon T. Willard and Rémi Louf. 2023. Efficient guided generation for large language models. *arXiv preprint arXiv:2307.09702*.