Routing Structured Outputs During Generation

Route M-Flight, streaming cutlines, multi-surface output, and optional in-flight lossless coding

Hayden Lindley · Glyphd Labs

2026-07-27

Contents

Routing Structured Outputs During Generation **1**
Abstract . . . 1
1. The redundant second pass . . . 2
2. Flight contract . . . 2
3. Typed routes . . . 3
4. Streaming cutlines . . . 3
5. One pass, multiple surfaces . . . 3
6. Confidence and arbitration . . . 4
7. M-flight lossless coding . . . 4
8. Structured repair . . . 5
9. Relation to SurfaceDelta . . . 5
10. Relation to SchemaStack . . . 5
11. Relation to content-addressed storage . . . 6
12. Instrumentation evidence . . . 6
13. Demonstration . . . 6
14. Benchmark program . . . 6
Routing . . . 6
Latency . . . 7
Quality . . . 7
Storage research . . . 7
15. Kill criteria . . . 7
16. Security . . . 7
17. Limitations . . . 7
18. Conclusion . . . 8
Architecture illustration briefs . . . 8
Source register . . . 8
Public-disclosure and IP boundary . . . 8

Routing Structured Outputs During Generation

Abstract

A language model generation already contains information that most application pipelines discard: token probabilities, partial structure, confidence changes, segment boundaries, and the moment at which a typed object becomes complete. Conventional systems wait for the

entire response, then invoke a parser, classifier, second model, or postprocessor to rediscover structure that was visible during emission.

Route M-Flight is a streaming architecture that routes, validates, records, and optionally entropy-codes typed segments during one generation flight. A model emits into declared channels—claims, actions, deltas, citations, explanations, tool arguments, receipts, or fallback prose. Segment validators operate at cutlines. Completed segments can be sealed with exact context, model/runtime identity, probability quantization, output digest, and raw fallback.

The immediate product value is not compression. It is lower duplication, earlier useful structure, bounded repair, and one-pass population of several surfaces. The compression extension uses the emitting model's own next-token probabilities to code exact output without a second inference pass, but bit-exact decoder conformance remains a serious research constraint.

1. The redundant second pass

A common application flow is:

```
prompt model
→ receive long text
→ parse text
→ call model again to classify it
→ extract actions
→ generate summary
→ populate UI
→ compress/store output
```

The first model often knew, during generation, that it was emitting:

- a title;
- an action item;
- a source citation;
- a JSON field;
- a refusal boundary;
- a completed paragraph;
- a tool call.

The application discards the intermediate state and reconstructs it later.

M-Flight makes the streaming generation an instrumented execution path.

2. Flight contract

A generation flight declares:

```
flight_id
model_execution_ref
context_horizon_digest
schema_registry
allowed_routes[]
segment_boundary_contract
validation_policy
repair_budget
probability_contract?
raw_fallback_policy
```

The model-facing instruction remains compact. Internal schema and policy metadata should not leak into user-visible prose.

3. Typed routes

A flight may populate routes such as:

```
NARRATIVE
CLAIM
EVIDENCE_REF
SURFACE_DELTA
ACTION_PROPOSAL
TOOL_ARGUMENTS
RISK
QUESTION
SUMMARY
RECEIPT_FIELD
RAW_FALLBACK
```

Routes are application contracts, not hidden chains of thought. They capture outputs intended for use.

A token stream can enter a route through grammar state, explicit delimiters, constrained decoding, or a small controller. The route decision and confidence are recorded.

4. Streaming cutlines

A **cutline** is a point at which a segment can be independently validated and sealed.

Examples:

- closing a JSON object;
- completing one SurfaceDelta;
- ending a citation record;
- completing a sentence under a claim schema;
- ending tool arguments;
- finishing one action item.

At a cutline, the runtime can:

- validate schema;
- resolve citations;
- update the interface;
- emit a partial receipt;
- request bounded repair;
- fall back to raw text;
- seal a storage layer.

This reduces latency to first useful verified content.

5. One pass, multiple surfaces

A single flight can populate:

```
paper prose
claim ledger
action list
citation map
```

```
risk rail
summary card
benchmark metadata
```

The model is not asked to produce seven separate answers. The router duplicates only the relevant typed fields.

Example:

```
CLAIM segment
  statement
  evidence refs
  limitation
```

can feed:

- body paragraph;
- claim-ledger row;
- evidence drawer;
- review queue.

The renderer and validators own final presentation.

6. Confidence and arbitration

Token probabilities are not calibrated truth. They can still help identify:

- unstable field values;
- uncertain route transitions;
- ambiguous delimiters;
- high-entropy segments;
- likely repair needs.

The controller should combine:

- grammar state;
- schema completion;
- explicit model confidence;
- probability entropy;
- verifier result;
- source coverage;
- task risk.

High-risk actions cannot become authorized merely because the model was confident.

7. M-flight lossless coding

When the local emitter exposes probabilities, the exact emitted token can be arithmetic/range coded against them.

A segment records:

```
model_execution_ref
context_horizon_digest
tokenizer_contract
probability_quantization_contract
code_bytes
exact_output_digest
raw_fallback_ref
```

No second inference pass is required to obtain probabilities.

The difficulty is exact replay. Decoder and encoder must reproduce identical distributions. The contract may need:

- exact weights;
- tokenizer;
- runtime;
- kernels;
- precision;
- sampling mode;
- context;
- integer CDF quantization;
- hardware class;
- deterministic execution.

If exact conformance is not available, the system retains raw output or uses the segment only for research accounting.

8. Structured repair

A failed segment does not require regenerating the entire response.

The runtime can request:

```
repair segment S12
error: citation span missing
preserve accepted segments S1–S11
return only corrected segment
```

A bounded repair budget prevents loops. Exhaustion yields a partial valid flight plus an explicit failed segment.

9. Relation to SurfaceDelta

SurfaceDelta is a natural M-Flight route. Each completed delta can:

- validate against base version;
- resolve evidence;
- render immediately;
- seal a receipt;
- update progress.

The model can stream useful page changes rather than waiting to emit a complete report.

10. Relation to SchemaStack

SchemaStack's Signal Gate chooses a compact answer contract before articulation. M-Flight operates during articulation.

```
schema field
→ actionability
→ expression gate
→ answer contract
→ M-Flight typed generation
→ segment verification
```

The two systems should remain separate. Signal Gate governs the kind of speech act. M-Flight routes the emitted useful structures.

11. Relation to content-addressed storage

Each flight and segment can have exact IDs:

```
flight_content_id
segment_content_id
context_digest
schema_id
model_ref
receipt_id
```

Repeated exact or semantic segments can reuse prior artifacts under GlyphCAS contracts. A failed route does not contaminate accepted segments.

12. Instrumentation evidence

The project corpus contains a local generation export with observed top-k softmax values per greedy step and a second full forward containing hidden-state and attention tensors. The caveat is explicit: the forward tensors are not a per-step hidden-state trace.

This establishes access to relevant model outputs in a local research setting. It does not establish a complete M-Flight coder or routing benchmark.

The `fold/2` v0.2 specification defines an M-flight segment profile and the Max-Flex roadmap places the research spike after more immediate runtime gates.

13. Demonstration

A deterministic browser demo can simulate:

- token stream;
- route state;
- schema completion;
- confidence;
- cutlines;
- validation;
- repair;
- sealed segments;
- raw fallback.

The demo should use fixture probabilities and clearly state that it is not a live model compression benchmark.

An optional local adapter can later stream a small model.

14. Benchmark program

Routing

- route accuracy;
- field completeness;
- wrong-channel rate;
- leakage;
- repair count;

- partial success.

Latency

- first useful segment;
- first verified segment;
- complete response;
- second-pass work avoided;
- UI update latency.

Quality

- task success;
- citation;
- action correctness;
- schema validity;
- consistency across routed surfaces.

Storage research

- code bytes;
- raw fallback bytes;
- cold-start dependencies;
- encode overhead;
- decoder conformance;
- failure rate;
- comparison with zstd-dictionary and direct text compression.

15. Kill criteria

- routing adds more latency than the avoided postprocessing;
- typed routes reduce answer quality;
- route errors create cross-surface contradictions;
- repair loops exceed full regeneration cost;
- probability coding cannot reproduce exact output reliably;
- dependency-adjusted storage loses to mature codecs;
- the application cannot explain which segment populated which surface.

16. Security

Threats include delimiter injection, schema leakage, hidden tool arguments, route confusion, untrusted source text influencing controller state, and replaying stale approvals.

Controls include grammar isolation, route-specific validators, source separation, no authority from confidence, exact context digests, bounded repair, raw fallback, and receipts.

17. Limitations

Many hosted APIs do not expose stable log probabilities or cache internals. Structured streaming differs by provider. Exact model replay can be expensive or impossible. Tokenization may be a poor byte-compression unit. One-pass multi-surface generation can propagate one model error widely if validators are weak.

The product case should be evaluated independently of the codec case.

18. Conclusion

M-Flight treats generation as an instrumented flight rather than an opaque blob.

The practical proposition is:

> Route useful structure while the model is producing it, validate at cutlines, and never ask a second model to rediscover what the first flight already declared.

The research proposition is:

> When exact local probabilities are available, seal the emitted segment against them—but keep raw fallback until conformance and honest accounting prove the route worthwhile.

Architecture illustration briefs

- **P14-F01 — One generation flight, many routes**: Token stream passes through cutlines to claims, evidence, deltas, actions, prose, and receipts with per-route validation.
- **P14-F02 — M-flight segment seal**: Exact context/model/probabilities feed entropy coder; output digest and raw fallback protect conformance.
- **P14-F03 — Signal Gate plus M-Flight**: Pre-articulation expression contract followed by in-flight route/segment validation.

Source register

- **P14-S01** — GlyphDrive OOC fold/2 Specification v0.2 (2026-07-12). `GLYPHDRIVE-OOC-FOLD-SPEC-v0.2.md`
- **P14-S02** — GlyphDrive/FOVEA/OOC Technical Assessment (2026-07-12). `GLYPHDRIVE-FOVEA-OOC-TECHNICAL-ASSESSMENT-v0.2-PLANNING.md`
- **P14-S03** — Zeke 4B Max-Flex Runtime Master Spec (2026-07-15). `ZEKE-4B-MAXFLEX-RUNTIME-MASTER-SPEC-v0.1.md`
- **P14-S04** — Local model run export (2026-03-27). `run.export.md`
- **P14-S05** — SchemaStack Signal Gate architecture (2026-05-05). `SIGNAL_GATE_ARCHITECTURE.md`

Public-disclosure and IP boundary

This edition publishes the public-safe streaming architecture, current instrumentation evidence, and benchmark criteria. It excludes private probability traces, proprietary routing thresholds, unpublished codec implementation, and patent-claim drafting. No latency or compression advantage is claimed as achieved.