Building a Deterministic Reasoning Question Generation Pipeline

Q-AGENT + A-AGENT EVALUATION FRAMEWORK

By Team11

- Abhinav, Ganesh, Manish, Shivam

PROBLEM DEFINITION

The Core Challenge

LLMs are fluent — but not logically reliable.

Observed issues:

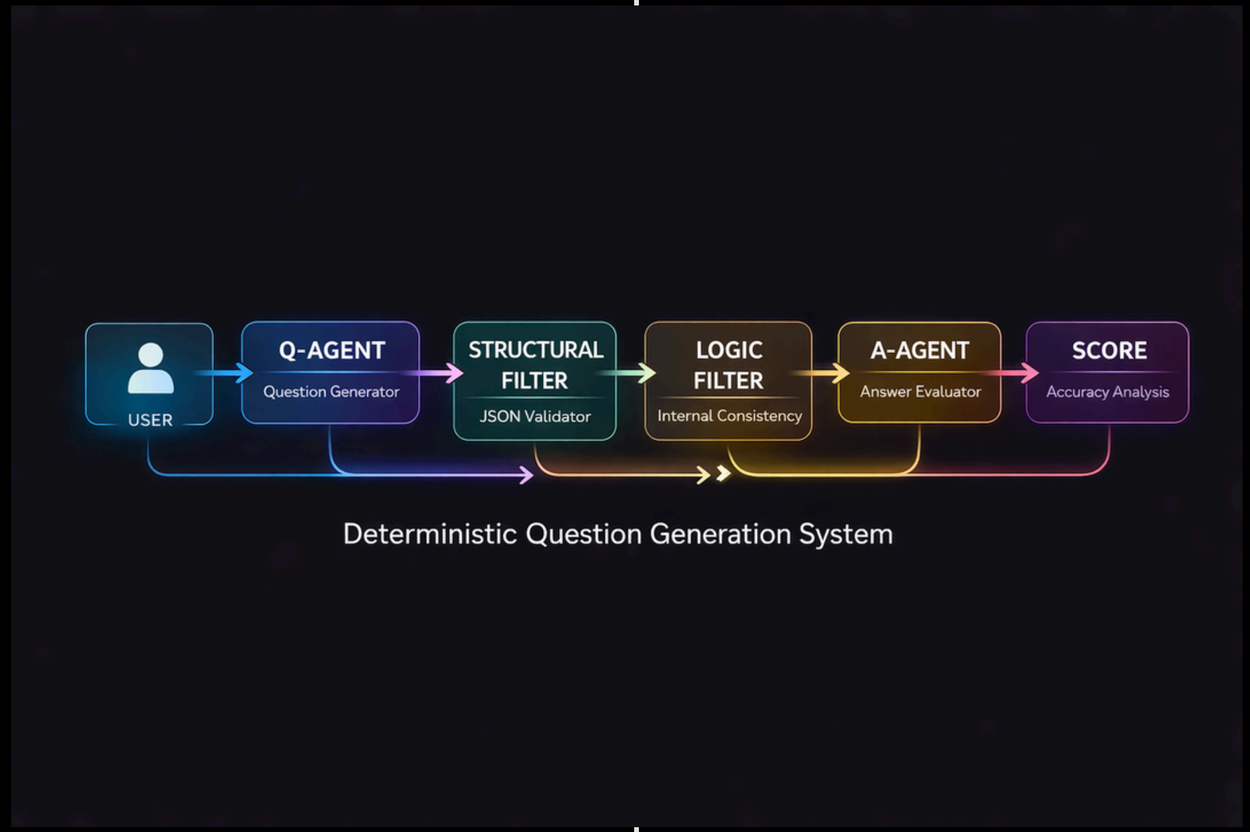
- Multiple valid answers
- Hidden contradictions
- Undefined entities
- Pattern repetition
- Ambiguous distractors

Goal:

Move from “fluent questions” → “formally valid reasoning problems”

SYSTEM ARCHITECTURE

End-to-End Pipeline
User
→ Q-Agent (Generation)
→ Structural Filter
→ A-Agent (Solver)
→ Accuracy Evaluation
Key Idea:
If a reasoning model cannot solve the question consistently, the question is likely flawed.



Q-AGENT DESIGN

Question Generation Strategy

- Rule-guided prompting
- Logic review for steady, clean answers
- Regenerates when answers repeat, conflict, or lose clarity

Topic Types:

- Syllogisms
- Patterned series
- Family links
- Seating layouts

PROMPT ENGINEERING EVOLUTION

Before

“Generate difficult MCQ.”

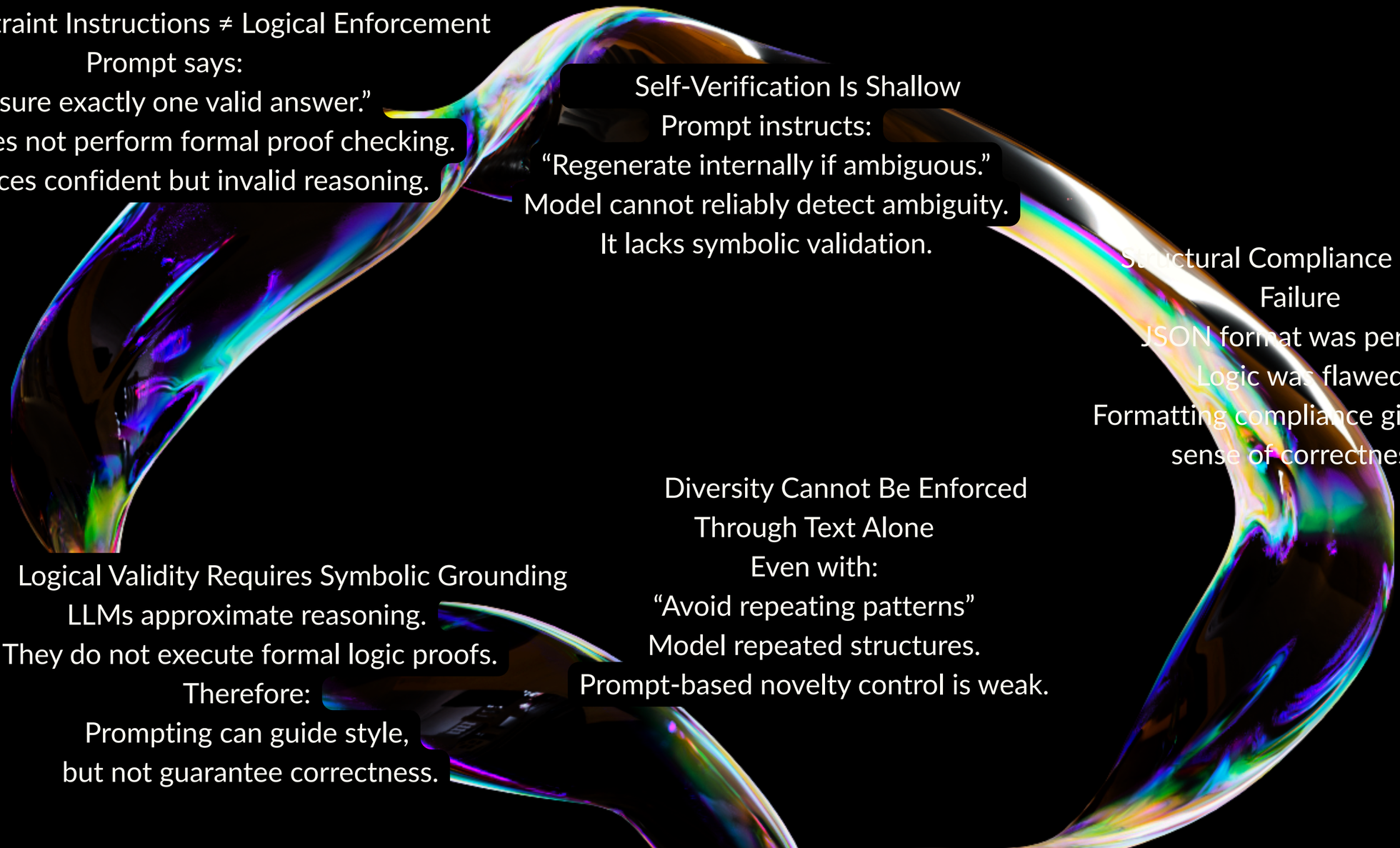
After

- Enforce internal consistency
- Exactly one correct answer
- Strict formal logic derivation
- No modal ambiguity (may/might/could)
- Strict JSON output enforcement

Key Insight:

Models ignore weak instructions.

WHY PROMPTING WAS NOT ENOUGH



Constraint Instructions ≠ Logical Enforcement
Prompt says:
“Ensure exactly one valid answer.”
Model does not perform formal proof checking.
It produces confident but invalid reasoning.

Self-Verification Is Shallow
Prompt instructs:
“Regenerate internally if ambiguous.”
Model cannot reliably detect ambiguity.
It lacks symbolic validation.

Structural Compliance Masks Logical Failure
JSON format was perfect.
Logic was flawed.
Formatting compliance gives false sense of correctness.

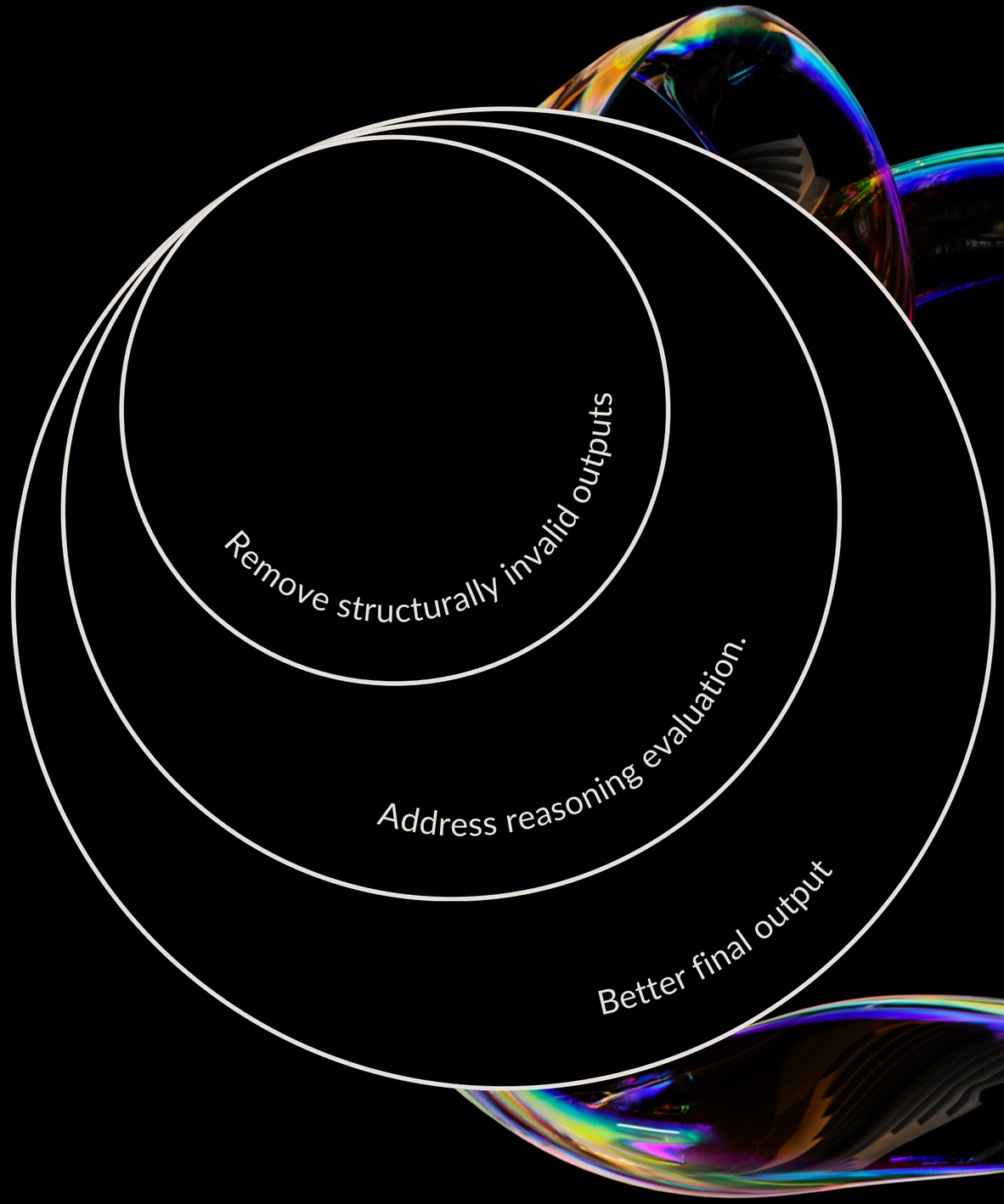
Diversity Cannot Be Enforced Through Text Alone
Even with:
“Avoid repeating patterns”
Model repeated structures.
Prompt-based novelty control is weak.

Logical Validity Requires Symbolic Grounding
LLMs approximate reasoning.
They do not execute formal logic proofs.
Therefore:
Prompting can guide style,
but not guarantee correctness.

STRUCTURAL FILTERING LAYER

Post-Generation Validation Checks:

- Required keys present
- Exactly 4 options
- Valid answer format
- Token length bounds
- JSON validity



LOGICAL VALIDATION STRATEGY

Manual inspection on:

- 10 Syllogisms
- 10 Series
- 10 Blood Relations

Evaluate:

- Internal consistency
- Uniqueness of answer
- Absence of ambiguity
- Proper entity declaration
-

Target: ≥ 90% logical validity before A-Agent evaluation

A-AGENT EVALUATION

Automated Reasoning Test

- A-Agent attempts solving generated MCQs
- Accuracy used as proxy for question validity

Observed:

- A-Agent Accuracy ≈ 26%

Interpretation:

Low score could indicate:

- Overly difficult questions
- Logical invalidity
- Ambiguous distractors

Metric must be interpreted carefully.

EVALUATION METRICS

Structural Validity Rate

Definition:

Percentage of generated questions passing JSON + format checks.

Example:

Generated: 50

Valid JSON: 50

Structural Validity = 100%

Uniqueness Rate

Definition:

Percentage of questions not repeating structure or template.

Metric:

Hash-based duplicate detection

Embedding similarity threshold < 0.90

Example:

Unique Structures: 28 / 50

Uniqueness Rate = 56%

Determinism Score (Series Only)

Check if rule can be expressed as:

Letter position +k

Numeric increment +m

Positional mapping function

If explanation cannot define a strict rule → fail.

Logical Validity Rate (Manual Audit)

Definition:

Percentage of questions with:

No contradictions

Exactly one correct answer

Deterministic reasoning chain

Example:

Audited: 30

Logically sound: 21

Logical Validity = 70%

A-Agent Agreement Score

Definition:

Percentage of questions where answering model selects correct

Current example:

A-Agent Accuracy ≈ 26%

Interpretation:

Low accuracy may reflect:

Question ambiguity

Logical invalidity

Model weakness

This must be interpreted carefully.

FAILURE EXAMPLES

Multiple Valid Answers (Logical Non-Uniqueness)

Generated Question:

Statement I: Some birds are mammals

Statement II: All mammals are reptiles

Statement III: Some reptiles are amphibians

Statement IV: All amphibians are fish

Conclusion I: Some fish are mammals

Conclusion II: Some reptiles are birds

Problem:

- Conclusion I is not necessarily valid (chain breaks).
- Model incorrectly inferred transitivity across existential premises.
- Logical chaining violated classical syllogistic constraints.

Failure Type: Invalid logical inference.

Template Repetition

Repeated across 50 generations:

- Same syllogism structure
- Same fruit/vegetable chain
- Same _XZ, ACY series

Failure Type: Low diversity / mode collapse.

Entity Overflow (Undeclared Variables)

Generated Seating Question:

- Initial setup defines: A, B, C, D, E, F
- Later introduces: G, H, I, J, K, L

Problem:

- Entities introduced that were never declared.
- Constraints become undefined.
- Logical space becomes under-specified.

Failure Type: Inconsistent domain definition.

Non-Deterministic Series Rule

Generated Series:

_XZ, ACY, DEF, GHI, __

Explanation:

“The sequence involves increasing positions in the alphabet…”

Problem:

- No precise functional rule.
- No definable mapping like f(n) = n + 2.
- Heuristic description instead of mathematical definition.

Failure Type: Pattern ambiguity.

FINAL STEPS

- Clean and validate 100 strong MCQs
- Build clear logic templates for each topic
- Add rule systems for syllogisms, series patterns, and cross-model checks
- Then move into fine-tuning

Shift from reactive prompt tweaks to a steady reasoning system