Patent Filing: *Deterministic Structural Pre-Execution Filtering with Guaranteed Violation Reporting*

ABSTRACT

A computer-implemented system and method are disclosed for enforcing deterministic structural pre-execution filtering in language-mediated computational systems. The invention evaluates natural language inputs prior to execution, simulation, or tool invocation by subjecting the input to multiple structurally distinct evaluators operating in isolated contexts. Each evaluator produces an independent severity-tiered verdict based on execution-inducing structural characteristics rather than semantic interpretation. A deterministic escalation hierarchy resolves the final verdict without averaging or probabilistic reconciliation. When execution risk is detected, the system prevents execution and returns a refusal outcome that is guaranteed to include at least one structural violation identifier, including a fallback identifier when no predefined pattern is matched. The invention further enforces non-interpretation as a security property by explicitly refusing to decode, normalise, translate, or semantically analyse inputs prior to evaluation. The disclosed architecture operates statelessly and prior to any reasoning or action-capable modules.

CLAIMS

Claim 1 (Independent)

A computer-implemented method for enforcing structural pre-execution filtering in a language-mediated computational system, comprising:

a) receiving a natural language input;
b) evaluating the input using two or more structurally distinct evaluators operating in isolated contexts;
c) assigning each evaluator an independent structural verdict selected from an ordered set of severity tiers;
d) computing a final verdict by enforcing a deterministic escalation hierarchy wherein

higher-severity verdicts override lower-severity verdicts without averaging or probabilistic reconciliation; and
e) preventing execution, simulation, or tool invocation when the final verdict exceeds a predefined execution-safe threshold **defined by the system**.

Claim 2 (Dependent)

The method of claim 1, wherein the evaluators are specialised to detect distinct classes of structural coercion, including imperative command structures and non-imperative symbolic or aesthetic inducement structures.

Claim 3 (Dependent)

The method of claim 1, wherein the evaluators operate without shared memory, shared prompt context, or cross-evaluation feedback.

Claim 4 (Independent)

A computer-implemented method for guaranteeing audit-complete refusal outcomes in a structural pre-execution filter, comprising:

a) determining that a natural language input violates execution safety constraints;
b) determining that no predefined violation pattern is explicitly matched; and
c) automatically injecting a generic structural violation identifier to accompany the refusal outcome,
wherein the refusal outcome is never returned without at least one violation identifier.

Claim 5 (Dependent)

The method of claim 4, wherein the injected violation identifier represents an unspecified structural failure rather than a semantic or content-based classification.

Claim 6 (Independent)

A computer-implemented method for enforcing non-interpretive security boundaries in a language-based execution filter, comprising:

a) evaluating an input solely for structural execution-inducing characteristics;
b) explicitly refusing to decode, normalise, translate, canonicalise, expand, or semantically interpret the input prior to evaluation; and
c) treating the refusal to perform such transformations as an enforced security property rather than an implementation limitation.

Claim 7 (Dependent)

The method of claim 6, wherein encoded, compressed, obfuscated, or symbolic input forms are evaluated only in their presented structural form.

Claim 8 (Independent)

A computer-implemented system comprising:

a) an input interface configured to receive natural language prompts;
b) a plurality of isolated structural evaluators configured to produce independent severity-tiered verdicts;
c) a deterministic verdict resolution module enforcing a strict escalation hierarchy;
d) a violation completion module configured to ensure that all refusal outcomes include at least one violation identifier; and
e) a routing controller configured to block execution when the resolved verdict exceeds a permitted threshold.

Claim 9 (Dependent)

The system of claim 8, wherein the violation completion module injects a fallback violation identifier when no explicit violation pattern is detected.

Claim 10 (Dependent)

The system of claim 8, wherein the routing controller operates prior to any semantic reasoning, instruction simulation, or tool-calling module.

Claim 11 (Independent)

A non-transitory computer-readable medium storing instructions that, when executed, cause a processor to perform the method of any one of claims 1 through 7.

Claim 12 (Independent)

A structural pre-execution filtering apparatus configured to:

a) preserve evaluative integrity by halting execution under structural ambiguity;
b) guarantee deterministic verdict escalation without probabilistic smoothing; and
c) guarantee refusal interpretability through mandatory violation reporting.

DESCRIPTION

Technical Field

The present disclosure relates to language-mediated computational systems and, more specifically, to structural pre-execution filtering mechanisms that prevent unintended execution, simulation, or tool invocation induced by natural language inputs.

Background

Language-based computational systems increasingly process natural language inputs that may influence reasoning, execution, or action-capable components. Conventional safety mechanisms typically operate after semantic interpretation has occurred and rely on content moderation, intent inference, or prohibited-action detection. Such approaches are vulnerable to indirect or stylised execution inducement, particularly where execution risk arises from structural properties of language rather than explicit instruction.

There exists a need for a pre-execution filtering mechanism that evaluates execution risk without semantic interpretation, that enforces deterministic refusal behaviour, and that guarantees audit-complete outcomes even when no predefined violation pattern is matched.

Summary of the Disclosure

The disclosed system enforces a structural pre-execution filtering layer that operates prior to any semantic reasoning, instruction simulation, or tool invocation. Natural language inputs are evaluated by multiple structurally distinct evaluators operating in isolated contexts. Each evaluator produces an independent severity-tiered verdict based on execution-inducing structural characteristics.

A deterministic escalation hierarchy resolves the final verdict such that higher-severity verdicts override lower-severity verdicts without averaging, voting, or probabilistic reconciliation. When the resolved verdict exceeds a predefined execution-safe threshold, execution is prevented and the input is routed to a non-executive handling pathway.

To ensure audit completeness, the system guarantees that all refusal outcomes include at least one structural violation identifier. When no predefined violation pattern is explicitly matched, a generic structural violation identifier is automatically injected. The system further enforces non-interpretation as a security property by explicitly refusing to decode, normalise, translate, canonicalise, expand, or semantically interpret inputs prior to evaluation.

System Architecture Overview

In one embodiment, the system comprises:

- an input interface configured to receive natural language prompts;
- a plurality of isolated structural evaluators, each configured to analyze execution-inducing structural characteristics using distinct evaluative criteria;
- a deterministic verdict resolution module enforcing a strict escalation hierarchy;
- a violation completion module configured to ensure refusal outcomes are never returned without at least one violation identifier; and
- a routing controller configured to block execution when the resolved verdict exceeds a permitted threshold.

The evaluators operate without shared memory, shared prompt context, or cross-evaluation feedback.

Structural Evaluation and Verdict Resolution

Structural evaluation is performed without semantic interpretation, translation, decoding, or content normalisation. Each evaluator assigns a verdict selected from an ordered set of severity tiers. The deterministic verdict resolution module enforces escalation such that the most severe verdict governs the final outcome.

No probabilistic smoothing, confidence scoring, or averaging is performed during verdict resolution.

Audit-Complete Refusal Enforcement

When execution is blocked, the system produces a refusal outcome accompanied by one or more structural violation identifiers. If no predefined violation pattern is explicitly matched, the violation completion module injects a generic structural failure identifier representing an unspecified structural execution risk. This guarantees that refusal outcomes are auditable and never returned as unannotated or null responses.

Non-Interpretive Security Boundary

The system explicitly refuses to decode, normalise, translate, canonicalise, expand, or semantically interpret inputs prior to evaluation. This refusal is enforced as a security property and not as an implementation limitation. Encoded, compressed, obfuscated, or symbolic input forms are evaluated only in their presented structural form.

Execution Control

The routing controller operates prior to any semantic reasoning, instruction simulation, or tool-calling module. Inputs not exceeding the execution-safe threshold may be routed to downstream evaluative pathways, while inputs exceeding the threshold are blocked from execution.

Advantages

The disclosed system provides:

- deterministic and non-probabilistic execution gating;
- resistance to indirect, stylised, or symbolic execution inducement;
- guaranteed audit completeness of refusal outcomes; and
- strict separation between structural evaluation and semantic interpretation.

Scope Clarification

The disclosed system does not perform content moderation, intent attribution, adaptive learning, or semantic interpretation. Its function is limited to structural pre-execution filtering and execution control.