Ethical Invariants as Loop Invariants:

A Formal Isomorphism Between AI Governance and Program Verification

Laboratorios Alexandria
contact@laboratoriosalexandria.com
June 2026

Abstract

We propose and formalize a structural isomorphism between ethical principles in AI governance and loop invariants in program verification. We demonstrate that ethical requirements such as non-discrimination, transparency, and accountability can be mapped to state-space constraints formally equivalent to the invariants used in software verification, and that both domains employ the same verification logic — temporal specifications over state models with constraints — to establish system validity. We formalize this correspondence using Linear Temporal Logic (LTL) and model checking, showing that a system is ethically valid if and only if, for every execution trajectory, it satisfies a conjunction of state specifications and transition constraints. We define explicit falsifiability conditions, identify sources of uncertainty, and propose empirical validation through formal verification tools. This isomorphism is not analogical but structural: it identifies a shared formal apparatus operating across two domains traditionally treated as disjoint.

1 Introduction

The governance of artificial intelligence systems and the formal verification of software programs are typically treated as separate disciplines. AI governance draws on political philosophy, law, and applied ethics to establish normative constraints on system behavior. Program verification draws on mathematical logic, automata theory, and formal methods to prove that software satisfies its specifications. Despite their distinct intellectual traditions, both fields address a fundamentally identical question: how to guarantee that a complex system, operating over time, never enters a prohibited state.

This paper argues that this shared question reflects a deeper structural correspondence. We demonstrate that ethical principles — when formalized as constraints on system states — are formally equivalent to loop invariants in program verification. The principle of non-discrimination, for instance, maps to a constraint that the system's output state never enters a set of states defined as discriminatory — precisely the structure of an invariant that must hold across all iterations of a computational loop. Similarly, the principle of transparency maps to a constraint on the observability of state transitions, and accountability maps to a constraint on the traceability of causal chains through the execution history.

We formalize this isomorphism using Linear Temporal Logic (LTL), the standard formalism for specifying temporal properties of systems. We show that both ethical requirements and program specifications can be expressed as LTL formulae over state models, and that the same verification procedures — model checking, theorem proving, runtime monitoring — apply to both. This is not an analogy. It is a claim about shared formal structure, and it is falsifiable.

2 Background and Related Work

2.1 Formal Verification and Loop Invariants

In program verification, a loop invariant is a property that holds before the loop begins, is preserved by each iteration, and therefore holds when the loop terminates. Formally, given a loop with precondition P, body B, and postcondition Q, an invariant I satisfies: $P \rightarrow I$ (initialization), $\{I \wedge \text{guard}\}\ B\ \{I\}$ (preservation), and $(I \wedge \neg\text{guard}) \rightarrow Q$ (termination). This Hoare-logic framework, introduced by Hoare (1969) and extended by Floyd (1967), remains the foundation of formal verification.

Model checking, introduced by Clarke, Emerson, and Sifakis (1981), automates verification by exhaustively exploring the state space of a system to determine whether a temporal specification holds. Tools such as SPIN (Holzmann, 1997) and UPPAAL (Behrmann, David, & Larsen, 2004) implement this approach for systems specified as finite automata with temporal properties expressed in LTL or Computation Tree Logic (CTL).

2.2 Formalization of Ethics in AI

The formalization of ethical principles for computational systems has advanced significantly in recent years. Hooker and Kim (2018) proposed a deontological approach to machine ethics based on quantified modal logic, translating Kantian principles into formal constraints evaluable by automated reasoners. Their work demonstrated that ethical rules can be expressed with sufficient precision for computational evaluation, though they argued that applying principles to specific situations should remain a human task.

More recently, Priya and Rao (2025) developed a deontic temporal logic framework for formal verification of AI ethics, introducing axioms and theorems to capture ethical requirements related to fairness and explainability. Their formalization incorporates temporal operators to reason about ethical behavior over time and evaluates the approach against real-world systems including COMPAS and loan prediction models, demonstrating that model checking can identify ethical violations that traditional auditing misses.

The IEEE 7000-2021 standard establishes organizational processes for addressing ethical concerns during system design, providing a procedural framework for tracing ethical values through system requirements, though it does not prescribe formal verification methods. The EU AI Act (2024)

introduces regulatory requirements for ethical impact assessment and risk documentation, creating institutional demand for verifiable ethical compliance.

2.3 The Gap

Despite these advances, a systematic formal mapping between ethical governance concepts and program verification concepts has not been established. Existing work formalizes ethics independently of verification theory, and verification theory proceeds without reference to ethical constraints. The structural equivalence between the two has been implicit but never explicitly characterized or tested. This paper fills that gap.

3 The Isomorphism

3.1 Ethical Principles as State Constraints

We define an ethical principle as a constraint on the permissible states of a system. Formally, let S be the state space of an AI system, and let $E \subseteq S$ be the set of ethically prohibited states. An ethical principle P_e is satisfied if and only if the system never enters a state in E: $\forall t: s(t) \notin E$. This is structurally identical to a loop invariant I that must hold across all iterations: the invariant defines the complement of a prohibited region in the state space, and verification consists of proving that no execution trajectory enters that region.

Consider three canonical ethical principles and their formal equivalents:

Non-discrimination. Let D be the set of states in which the system's output differs based on a protected attribute (race, gender, etc.) without task-relevant justification. The principle of non-discrimination requires $\forall t: s(t) \notin D$. This is equivalent to a loop invariant asserting that the output function is independent of protected attributes across all iterations of the decision loop.

Transparency. Let O be the set of states in which the causal chain from input to output is not reconstructible from the system's logs. The principle of transparency requires $\forall t: s(t) \notin O$. This is equivalent to an invariant on the observability of state transitions — a standard property in formal verification of distributed systems.

Accountability. Let A be the set of states in which no identifiable agent bears responsibility for the system's output. The principle of accountability requires $\forall t: s(t) \notin A$. This maps to a traceability invariant requiring that every state transition is attributable to a specified component or actor.

3.2 Unified Formulation

Drawing on the deliberative synthesis that generated this analysis, we propose a unified formulation: a system (technical or ethical) is valid if, for every execution trajectory, it satisfies $P(\varphi \wedge \psi)$, where φ is the state specification (e.g., 'no discrimination' or 'no memory violation') and ψ is the transition

constraint (e.g., ‘no modification of protected data’ or ‘no violation of autonomy’). Both φ and ψ are expressible in LTL, and both are verifiable through the same model-checking procedures.

3.3 Conceptual Equivalences

The isomorphism extends beyond individual principles to the structural apparatus of each domain:

Ethical principle ↔ State constraint. A normative requirement (e.g., “do no harm”) maps to a restriction that the system state does not enter a prohibited set — identical to the definition of a safety property in verification.

Ethical audit ↔ Runtime verification. The process of checking compliance with ethical requirements post-deployment is structurally equivalent to runtime monitoring of invariants in deployed software.

Ethical impact assessment ↔ Static analysis. Pre-deployment assessment of ethical risks maps to static analysis of program properties before execution — both attempt to establish guarantees without running the system.

Regulatory compliance ↔ Specification conformance. The process of certifying that a system meets regulatory requirements (EU AI Act, IEEE 7000) is formally identical to proving that an implementation conforms to its specification.

4 Verification Framework

If ethical principles are formally equivalent to verification invariants, then the entire apparatus of formal verification becomes available for ethical governance. We propose a three-layer verification architecture:

Specification layer. Ethical principles are formalized as LTL properties over the system’s state model. For example, the non-discrimination principle becomes G(¬discriminatory_output), where G is the “globally” temporal operator asserting that the property holds at all times.

Implementation layer. Ethical constraints are integrated as optimization constraints in the algorithm’s search space (e.g., fairness-aware regularization, constrained optimization), analogous to how loop invariants constrain the set of reachable states during execution.

Verification layer. Formal tools (SPIN, UPPAAL, or proof assistants such as Coq or Isabelle) verify that the implementation satisfies the specification. This replaces subjective ethical auditing with mathematical proof, at least for the formalized subset of ethical requirements.

This framework has been partially validated in recent work. Priya and Rao (2025) demonstrated that deontic temporal logic can encode ethical properties of AI systems and that model checking can verify compliance, applying the approach to the COMPAS recidivism prediction system and identifying

previously undetected fairness violations. Our contribution is to recognize this as an instance of a general structural isomorphism, not a domain-specific technique.

5 Falsifiability Conditions

Following rigorous epistemic standards, we specify explicit conditions under which our thesis would be refuted:

Condition 1. The isomorphism is refuted if an ethical principle is identified that cannot be mapped to a state constraint formally equivalent to a loop invariant — that is, if there exists a normative requirement whose structure is fundamentally incompatible with the invariant framework (e.g., a principle that requires reasoning about counterfactual states not representable in the system's state space).

Condition 2. The isomorphism is refuted if the transition logic of ethical systems is shown not to satisfy the same algebraic properties as finite automata with context variables — for example, if ethical reasoning requires non-determinism that cannot be captured by standard model-checking formalisms, or if closure under composition fails for ethical constraints.

Condition 3. The isomorphism is weakened (though not refuted) if ethical principles require logical extensions beyond first-order logic — for instance, if adequate formalization requires deontic modal operators (obligation, permission, prohibition) that do not reduce to state constraints. Recent work suggests such extensions may be necessary but compatible with the isomorphism at a higher level of abstraction.

6 Sources of Uncertainty

Interpretive variability. Ethical principles such as "do no harm" have context-dependent interpretations across normative frameworks. The mapping to state constraints requires disambiguation that may not preserve all semantic content of the original principle. This is a limitation of formalization in general, not of the isomorphism specifically.

Computational complexity. Verifying ethical invariants in large-scale AI systems (e.g., multi-agent systems with unobservable context variables) may be computationally intractable. State-space explosion is a known challenge in model checking, and ethical invariants are unlikely to be exempt.

Ontological assumptions. The isomorphism assumes that the "components" of ethical systems (agents, institutions, norms) can be modeled as state-bearing entities with defined transitions. Whether this assumption holds for all ethical domains — particularly those involving emergent social norms or contested values — remains an open question.

Dynamic normativity. Ethical principles evolve over time as social norms change. The current framework treats ethical constraints as static specifications. Extending the isomorphism to handle dynamic normative evolution — analogous to hot-swapping specifications in runtime verification — is an identified direction for future work.

7 Unresolved Questions

The deliberative process that generated this analysis identified three questions that resist closure:

Structural vs. functional equivalence. Does the formal isomorphism imply genuine functional equivalence between ethical governance and program verification, or is it a structural coincidence without operational implications? If the mapping preserves structure but not function, its practical value is limited to pedagogy and conceptual clarity rather than engineering application.

Expressive adequacy. Can ethical principles be fully expressed in first-order logic, or do they require non-standard extensions such as deontic modal logic with knowledge operators or obligation hierarchies? Hooker and Kim (2018) formalized deontological principles in quantified modal logic; Priya and Rao (2025) used deontic temporal logic. The choice of formalism affects the scope of the isomorphism.

Conflict resolution. How should the framework handle conflicts between ethical constraints (e.g., autonomy vs. safety) when both are formalized as invariants? In program verification, conflicting invariants indicate a specification error. In ethics, conflicting principles are the norm, not the exception. The isomorphism must accommodate this asymmetry or acknowledge its boundary.

8 Discussion

The isomorphism we identify has implications in both directions. For AI governance, it suggests that ethical requirements need not remain in the realm of qualitative assessment. If ethical principles can be formalized as state constraints, then the mature apparatus of formal verification — decades of research in model checking, theorem proving, and runtime monitoring — becomes directly applicable to ethical compliance. This is not a metaphor: it is a technical claim that SPIN or UPPAAL can verify ethical properties of AI systems with the same rigor they bring to safety-critical software.

For program verification, the isomorphism suggests a new class of specifications. Ethical invariants are not reducible to safety or liveness properties as traditionally defined; they introduce normative constraints that depend on social context, stakeholder values, and evolving regulatory frameworks. Incorporating these constraints into the verification workflow requires extending the specification language and the verification toolchain, but the foundational machinery is compatible.

The practical gap between formalization and implementation remains significant. The IEEE 7000-2021 standard provides organizational processes for ethical integration but does not prescribe formal

verification. The EU AI Act mandates ethical impact assessment but does not require mathematical proof of compliance. Bridging this gap — moving from procedural ethics to verified ethics — is the engineering challenge that the isomorphism makes visible and, for the first time, tractable.

We note that the original insight motivating this paper emerged from a paper by Oliver et al. (2023) on real versus hypothetical risks of artificial intelligence, which identified the fundamental interdependence between ethical governance and technical capability in AI systems. Our contribution is to formalize this interdependence as a structural isomorphism and to specify the conditions under which it holds or fails.

9 Conclusion

We have demonstrated that ethical principles in AI governance and loop invariants in program verification share a common formal structure: both are constraints on permissible system states that must hold across all execution trajectories, both are expressible in temporal logic, and both are verifiable through model checking. This isomorphism is not analogical but structural, and it is falsifiable through the conditions we have specified.

The practical implication is direct: the tools, techniques, and theoretical foundations developed over decades of formal verification research are applicable to ethical governance of AI systems. Moving from qualitative ethical assessment to formal ethical verification is not a paradigm shift — it is a recognition that the paradigm already exists and merely requires cross-domain application.

The path forward requires formalizing ethical principles with sufficient rigor to serve as verification specifications, extending model-checking tools to handle normative constraints, and developing the “translation layer” between natural-language ethical requirements and formal temporal logic. Each of these steps is technically feasible with existing methods. What has been missing is the recognition that they are steps in the same direction. This paper provides that recognition.

References

1. Behrmann, G., David, A., & Larsen, K. G. (2004). A tutorial on UPPAAL. In Formal Methods for the Design of Real-Time Systems (LNCS 3185, pp. 200–236). Springer.
2. Clarke, E. M., Emerson, E. A., & Sifakis, J. (2009). Model checking: Algorithmic verification and debugging. Communications of the ACM, 52(11), 74–84.
3. European Parliament and Council. (2024). Regulation (EU) 2024/1689 laying down harmonised rules on artificial intelligence (AI Act). Official Journal of the European Union.
4. Floyd, R. W. (1967). Assigning meanings to programs. Proceedings of Symposia in Applied Mathematics, 19, 19–32.
5. Hoare, C. A. R. (1969). An axiomatic basis for computer programming. Communications of the ACM, 12(10), 576–580.

6. Holzmann, G. J. (1997). The model checker SPIN. IEEE Transactions on Software Engineering, 23(5), 279–295.

7. Hooker, J. N., & Kim, T. W. N. (2018). Toward non-intuition-based machine and artificial intelligence ethics: A deontological approach based on modal logic. Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society (AIES ’18), 130–136.

8. IEEE. (2021). IEEE 7000–2021: Standard Model Process for Addressing Ethical Concerns during System Design. IEEE Standards Association.

9. Jobin, A., Ienca, M., & Vayena, E. (2019). The global landscape of AI ethics guidelines. Nature Machine Intelligence, 1(9), 389–399.

10. Oliver, N., Schölkopf, B., d’Alché-Buc, F., Lavrač, N., Cesa-Bianchi, N., Hochreiter, S., & Belongie, S. (2023). Let’s focus on AI’s tangible risks rather than speculating about its potential to pose an existential threat. The Conversation, June 2023.

11. Priya, T. V., & Rao, S. (2025). Deontic temporal logic for formal verification of AI ethics. arXiv:2501.05765.