Research Article *Open Access*

Journal of Pioneering Artificial Intelligence Research

ISSN: 3069-0846

DOI: *doi.org/10.63721/26JPAIR0131*

Core Operating System for Autonomous Robots: Deterministic Action without Optimization via Structural Elimination

Huseyin Murat Cekirge

Department of Mechanical Engineering, The City College of New York (CUNY), New York, USA

Citation: *Huseyin Murat Cekirge (2026) Core Operating System for Autonomous Robots: Deterministic Action without Optimization via Structural Elimination. J of Poin Artf Research 2(2), 1-9 WMJ-JPAIR-131*

Abstract

This study introduces a core operating system for autonomous robots in which actions are determined deterministically through structural elimination, without reliance on optimization. The proposed framework initiates autonomous action through impulse-driven activation and structured elimination over a finite modular action library. Unlike conventional approaches that depend on optimization, learning, or explicit goal specification, the system operates through the accumulation of discrete event impulses that activate constraint structures, progressively eliminating incompatible actions until a consistent outcome is revealed. Actions are not selected through search or ranking; instead, incompatible candidates are eliminated until a single admissible element remains. A threshold-based activation mechanism governs the transition from passive observation to active resolution, ensuring that decisions are triggered only when sufficient structural information is available. The framework introduces a hierarchical modular library architecture that supports both routine behavior and event-driven extension. New action patterns are incorporated through structured recombination when existing configurations fail to resolve emerging conditions, enabling continuous yet controlled system evolution without parameter learning. The same impulse–accumulation–threshold–elimination mechanism operates consistently across scales, from local adjustments to global behavioral reconfiguration. The framework further extends to multi-agent settings through selective library exchange, allowing collaborative systems to evolve without centralized control. The results establish a complementary paradigm for autonomous systems in which decision-making emerges from feasibility and structural consistency rather than optimization, offering a lightweight, interpretable, and deterministic alternative for structured environments.

***Corresponding author**: Huseyin Murat Cekirge, Department of Mechanical Engineering, The City College of New York (CUNY), New York, USA. E-mail: hmcekirge@usa.net

Submitted: 23.03.2026 **Accepted**: 27.03.2026 **Published**: 15.04.2026

Keywords: Deterministic Autonomy, Action Initiation, Impulse-Driven Systems, Modular Library, Elimination-based Decision, Event-driven Systems, Constraint-based Reasoning, Multi-agent Systems

Abbreviations

ALS - Action Library Structure
IDS - Impulse-Driven System
MAS - Multi-Agent System
SER - Structural Elimination Resolution

Introduction

Autonomous systems are traditionally formulated within frameworks of optimization, learning, and goal-driven planning [1-4]. In these approaches, actions are determined through iterative search processes, parameter updates, or reward-based evaluation. The system is guided by an explicitly defined objective, and decision-making is achieved by exploring a space of alternatives to identify an optimal or near-optimal solution [3, 4].

While these methods have proven effective in complex and uncertain environments, they inherently depend on search, convergence behavior, and model adaptation [2–4]. As a result, system performance is often tied to computational cost, parameter tuning, and stability considerations. Decision-making is therefore not immediate, but emerges through iterative refinement, approximation, or probabilistic inference [1, 2].

Autonomic and distributed system perspectives further emphasize adaptive behavior, self-management, and large-scale coordination, often supported by feedback mechanisms and iterative processes [5-7]. These approaches highlight the importance of adaptability but also reinforce reliance on continuous adjustment and system-level complexity.

This study considers a different formulation of autonomy that does not rely on training, optimization, or explicit goal specification. Instead of searching for an action, the system operates on a finite candidate set and progressively eliminates those that are incompatible with the current conditions. Action is not selected as the best alternative, but is revealed as the only admissible element after infeasible options are removed.

Under this perspective, decision-making is expressed as a deterministic elimination process over a structured candidate set. No parameter updates, objective functions, or search trajectories are required. The system remains passive until sufficient structural information is available, at which point action is initiated through constraint-induced reduction. This viewpoint is conceptually aligned with structured approaches to stability and well-posedness in mathematical systems [8].

The contribution of this work is the formulation of autonomous action initiation as an elimination-based process. The proposed framework introduces a non-search-based mechanism in which decisions emerge from feasibility rather than optimality, providing a lightweight and interpretable alternative for structured environments. This formulation does not compete with optimization-based or learning-based methods, but instead defines a complementary regime in which decisions are resolved through feasibility when structural conditions are sufficiently specified.

Design Paradigm

The proposed framework adopts a design paradigm in which decision-making is governed by the structure of constraints rather than by scalar optimization [9, 10]. In many conventional approaches, multiple criteria are reduced to a single objective function, and decisions are obtained by minimizing or maximizing this scalar quantity. Such formulations inherently compress multi-dimensional information into a single dimension, often obscuring the structural relationships among constraints.

In contrast, the present approach preserves the multi-dimensional nature of constraints. Each constraint represents an independent dimension of feasibility, and no aggregation into a single scalar measure occurs. As a result, the system maintains the full structural representation of the problem without reduction or approximation.

Within this framework, decision-making is expressed as the intersection of constraints over a finite candidate set. Each candidate is evaluated against all active constraints, and those that violate any constraint are eliminated. The remaining set represents the feasible region defined by the simultaneous satisfaction of all conditions.

This formulation leads to a fundamental distinction between elimination and optimization. Optimization explores a space of alternatives to identify a best

solution according to a predefined objective. In contrast, the proposed method does not search or rank candidates. Instead, it removes all infeasible elements, allowing the decision to emerge as the only remaining admissible outcome.

Consequently, an optimal action is not computed by the system; it resolves a constraint structure. Decision-making is therefore not driven by preference or scoring, but by consistency. This paradigm preserves structural information, avoids dimensional reduction, and provides a deterministic mechanism for action initiation.

$\delta \rightarrow A \rightarrow \tau \rightarrow H \rightarrow$ elimination $\rightarrow$ action

Figure 1. Impulse-driven action initiation mechanism

Mathematical Formulation

The proposed framework models decision-making as an impulse-driven, threshold-activated elimination process over a finite candidate set. Events are represented as discrete impulses that accumulate over time and activate the decision mechanism when sufficient structural information is present.

Let S denote a finite set of candidate actions. Let $\delta_i(t)$ represent an impulse associated with event i at time t. These impulses contribute to an accumulated activation signal defined as:

$$A(t) = \Sigma\, \delta_i(t) \quad (1)$$

The system remains inactive as long as the accumulated activation A(t) is below a predefined threshold τ. When the threshold condition is satisfied,

$$A(t) \geq \tau \quad (2)$$

the system transitions to an active state. This transition can be interpreted as a step-like activation function H(A), defined as:

$$H(A) = 0 \text{ if } A < \tau \quad (3)$$

$$H(A) = 1 \text{ if } A \geq \tau \quad (4)$$

Once activated, the system evaluates the candidate set S under a set of constraints C derived from the active impulses. Each candidate $x \in S$ is tested for compatibility. Incompatible elements are eliminated, yielding a reduced set:

$$S(C) = \{\, x \in S \mid x \text{ satisfies all constraints in } C \,\} \quad (5)$$

Action initiation occurs when the reduced set contains a single admissible element:

$$|S(C)| = 1 \quad (6)$$

In this case, the remaining element defines the initiated action. If no element satisfies the constraints,

$$S(C) = \emptyset \quad (7)$$

the system identifies a structural mismatch, which may trigger a reconfiguration or extension process. If multiple elements remain,

$$|S(C)| > 1 \quad (8)$$

the constraint set is considered incomplete, and additional impulses may be required to resolve the decision. The resulting process can be interpreted as a finite constraint satisfaction mechanism in which resolution replaces search, and determinacy replaces convergence.

This formulation establishes a deterministic mechanism in which action is not continuously computed but triggered by accumulated impulses and resolved through constraint-based elimination. The process does not involve search, ranking, or optimization, but relies solely on feasibility and structural consistency.

Autonomous Action Initiation

Autonomous action initiation in the proposed framework is formulated as a deterministic outcome of constraint resolution rather than as a process of search or selection. The system operates on a finite candidate set and remains inactive until sufficient structural information is accumulated through impulses.

Once the activation condition is satisfied, the system evaluates all candidates under the current constraint set. Each candidate is tested for compatibility, and those that violate any constraint are eliminated. This process reduces the candidate set without ranking, scoring, or iterative refinement.

Action initiation occurs when the elimination process yields a single admissible element. In this case, the system does not choose among alternatives; it resolves the candidate set to its only feasible state. The resulting action is therefore not selected as the best option, but emerges as the inevitable outcome of constraint enforcement.

This formulation distinguishes action initiation from conventional decision-making mechanisms. In optimization-based systems, actions are derived through exploration and comparison. In contrast, the present approach requires no exploration of alternatives once the candidate set is defined. The system does not generate trajectories or evaluate preferences; it simply

enforces feasibility.

Autonomy, in this context, is defined as the ability to initiate action without external goal specification or internal search processes. The system responds to accumulated conditions by resolving a bounded structure, ensuring that action arises directly from consistency rather than approximation.

Action Library Architecture

The proposed framework operates on a structured action library that defines the finite candidate set used for decision-making. This library is modular, hierarchical, and open to controlled extension, enabling both stability and adaptability without altering the underlying decision mechanism.

The primary layer, referred to as the main library, contains a finite set of fundamental action modules. These modules represent abstract, reusable action components that define the basic capabilities of the system. The main library is designed to be stable and provides the foundational structure from which candidate actions are derived.

A secondary layer, referred to as the routine library, captures commonly occurring patterns as structured sequences of actions. These routines represent frequently activated configurations that correspond to recurring conditions, allowing efficient activation without recomputation. Routine elements are derived from the main library but are organized as higher-level modules.

At any given time, a candidate action set S is constructed from elements of both the main and routine libraries. This set represents the feasible action space under current structural conditions prior to elimination. The selection of candidates is not based on search, but on structural relevance induced by active impulses.

The library is open to extension through the incorporation of new modules formed by recombination and refinement of existing elements. This process does not introduce arbitrary actions; it preserves consistency by ensuring that all new modules conform to the same structural rules as the original library. As a result, the action space remains finite, organized, and interpretable.

This architecture supports a clear separation between representation and decision-making. The library defines what actions are possible, while the elimination process determines which action is admissible. The underlying mechanism of action initiation remains unchanged, regardless of the size or richness of the library.

Impulse-Driven Triggering Mechanism

The proposed framework employs an impulse-driven mechanism to activate the decision process. Events are represented as discrete impulses that accumulate over time and determine when the system transitions from passive observation to active resolution.

Each event generates an impulse δ_i that contributes to an aggregate activation signal A. This accumulation reflects the combined structural influence of current conditions rather than a continuous evaluation of the action space. The system does not operate continuously; it remains inactive while the accumulated activation is insufficient to resolve a decision.

A threshold τ defines the activation condition. When the accumulated signal satisfies $A \geq \tau$, the system transitions to an active state. This transition can be interpreted as a step-like activation, in which the decision mechanism is triggered only when sufficient information has been gathered.

Upon activation, the current impulse configuration induces a set of constraints that define the admissible structure over the candidate action set. These constraints are not predefined globally; they are formed dynamically from the active impulses. The elimination process is then applied to the candidate set, resulting in a reduced set of feasible actions.

The impulse-driven mechanism ensures that decision-making is event-based rather than continuously computed. The system reacts only when necessary, avoiding unnecessary evaluation or search. Different impulse patterns lead to different constraint structures, enabling context-sensitive behavior without explicit goal specification.

This mechanism also supports multiple scales of operation. Small, routine impulses activate localized behaviors, while larger or accumulated

impulse structures may trigger global transitions or reconfiguration. In all cases, the same accumulation–threshold–activation principle governs the initiation of action.

Trigger Groups and Mode Transition

The impulse-driven mechanism operates through structured collections of impulses referred to as trigger groups. Each trigger group represents a coherent set of conditions that activates a corresponding behavioral mode within the system.

Let $\{G_1, G_2, \ldots\}$ denote trigger groups, where each group is defined by a characteristic impulse configuration and its associated constraint structure. At any given time, one dominant trigger group governs the system, determining how the candidate action set is interpreted and reduced through elimination.

As events occur, new impulses are introduced and accumulated. If the resulting impulse configuration remains compatible with the currently active group G_k, the system continues to operate within the same behavioral mode. However, when the accumulated impulses no longer support a feasible resolution under G_k, a structural mismatch arises.

This mismatch initiates a mode transition. A new trigger group G_m is activated, reflecting the updated impulse configuration. The candidate action set is then re-evaluated under the constraints induced by G_m, leading to a different elimination outcome and, consequently, a different action.

Mode transitions are not explicitly commanded or goal-driven. They emerge naturally from changes in the impulse structure. Modes are not switched by decision; they emerge from structural reconfiguration as the previous structure becomes invalid and a new consistent configuration is activated.

This mechanism enables adaptive behavior across qualitatively different contexts. Routine activities may be governed by stable trigger groups, while significant events may induce transitions to entirely different modes. In all cases, the transition is deterministic and governed by the same impulse–accumulation–threshold–elimination process.

Event Aggregation and Extension

Library extension is driven by the aggregation of events that collectively exceed the explanatory capacity of existing trigger groups. Individual events may not induce a change in behavior; however, their cumulative effect can alter the structural conditions under which the system operates.

Let $\delta_1, \delta_2, \ldots$ denote impulses associated with events occurring over a time interval. These impulses accumulate into an aggregate activation signal A. As long as A remains compatible with the currently active trigger group, the system continues to operate within established behavioral patterns.

A structural mismatch occurs when the accumulated impulse configuration cannot be resolved by any existing trigger group. In this case, the system is unable to produce a feasible action under the current library. This condition defines the trigger for extension.

Rather than generating arbitrary actions, the system constructs a new trigger–action relationship from the aggregated event structure. This new structure captures the previously unrepresented condition and defines a corresponding pattern of behavior. The resulting module is incorporated into the library, extending its capability while preserving its modular and finite nature.

Once incorporated, similar future event configurations will directly activate the newly formed trigger group, eliminating the need for repeated extension. The system therefore evolves through structured accumulation of experience, not through parameter adjustment or continuous learning.
This process ensures that extension is both necessary and consistent. It is necessary because it is triggered by unresolved conditions, and consistent because all new structures conform to the same impulse–threshold–elimination mechanism as the original system.

Local vs Global Extension

The proposed framework supports extension at multiple scales while preserving a single underlying mechanism. Both routine refinements and major behavioral changes are governed by the same impulse–accumulation–threshold–elimination structure, differing only in the scope of their impact on the library.

Local extension occurs within a confined subset of the library associated with routine behavior. Let $L_s \subset L$ denote a localized sub-library corresponding to a recurring activity. When accumulated impulses within this substructure fail to resolve a feasible action, a micro-extension is introduced. This results in the addition or refinement of a module within L_s, without altering the global structure of the system.

Global extension, in contrast, arises when the accumulated impulse configuration cannot be resolved by any existing trigger group across the entire library. In this case, the mismatch is not confined to a routine context but reflects a fundamental change in conditions. A new trigger group and associated action pattern are constructed and incorporated into the global library L.

Despite the difference in scale, both local and global extensions follow the same principle: extension is triggered by the absence of a resolvable action under current conditions. The distinction lies only in the extent of structural modification required to restore consistency.

This hierarchical extension mechanism enables the system to refine routine behavior while also adapting to major structural changes. Small variations lead to localized updates, while significant disruptions induce global reconfiguration. In all cases, the system evolves through structured extension without altering its deterministic decision mechanism.

State Disruption and Reconfiguration

The proposed framework does not assume persistent stability of behavioral structures. Under certain conditions, accumulated impulses may invalidate all currently active trigger groups, leading to a state in which no admissible action can be resolved. This condition represents a structural disruption.

Formally, let G_k denote the active trigger group. If the induced constraints fail to produce a feasible candidate set, such that $S(G_k) = \emptyset$, the system enters a null state. In this state, previously valid behavioral patterns are no longer applicable, and continuation under the existing structure is not possible.

Rather than attempting to adapt or repair the invalidated structure, the system initiates a reconfiguration process. A new impulse configuration, derived from current conditions, is used to construct a new trigger group G_m. This new group defines a consistent constraint structure under which the candidate set can again be resolved.

Reconfiguration is therefore not a gradual adjustment of previous behavior, but a discrete transition to a new structural regime. The system abandons the invalid configuration and establishes a new one that restores feasibility. Once the new trigger group is incorporated into the library, subsequent operation proceeds deterministically under the updated structure.

This mechanism ensures continuity of operation without requiring persistence of past states. Behavioral consistency is maintained not by preserving previous configurations, but by re-establishing a valid structure whenever disruption occurs.

Auto-Extension of Modular Library

The proposed framework supports the autonomous extension of its modular action library through a structured and event-driven process. This extension does not rely on parameter updates or optimization, but on the incorporation of new modules that arise from unresolved conditions.

When the system encounters a situation that cannot be resolved within the current library, a structural mismatch is identified. This condition triggers an extension process in which a new module is constructed from the existing impulse configuration and constraint structure. The new module captures the relationship between the observed conditions and the required action pattern.

Importantly, extension is not unconstrained generation. New modules are formed through recombination and refinement of existing elements, ensuring compatibility with the overall architecture. The library grows incrementally, and all additions conform to the same structural rules that govern the original system.

This process can be interpreted as experience accumulation. Each extension encodes a previously unrepresented condition into a structured module, allowing similar future situations to be resolved without further modification. The system therefore

evolves over time while preserving its deterministic decision mechanism.

The underlying action initiation process remains unchanged. The impulse–accumulation–threshold–elimination structure continues to govern all decisions, regardless of the size or richness of the library. Extension increases capability but does not alter the nature of decision-making.

Thus, the system exhibits a form of controlled autonomous development. It expands its action space when necessary, but does so in a manner that maintains consistency, interpretability, and structural integrity.

Collaborative Extension (Multi-Agent)

The proposed framework extends naturally to multi-agent settings in which each agent maintains its own modular action library while operating under the same impulse–accumulation–threshold–elimination mechanism. Collaboration emerges through structured exchange rather than centralized coordination or command transfer.

Let L_1 and L_2 denote the action libraries of two agents. During interaction, agents exchange impulse structures and candidate modules that reflect their local experiences. These exchanged elements are not adopted directly; each agent evaluates them under its own constraint structure and incorporates only those that are compatible with its existing library.
This process results in selective and asymmetric extension. While both agents may benefit from interaction, each library evolves according to its own structural conditions. The system avoids full synchronization, preserving autonomy while enabling knowledge transfer through compatibility filtering.

In addition to individual libraries, collaboration may give rise to a shared interaction layer. This layer can be interpreted as a collaborative library that captures joint action patterns emerging from coordinated impulse structures. Such patterns are activated when the combined impulses of multiple agents satisfy a common threshold and define a consistent constraint structure.

Collaboration therefore acts as a mechanism for distributed extension. Agents do not exchange goals or commands; they exchange structured conditions that expand each other's capability. The resulting system is decentralized, consistent, and scalable, with each agent evolving independently while benefiting from interaction.

Scope and Limitations

The proposed framework operates under a set of structural assumptions that define its scope of applicability. The system is designed for environments in which the action space can be represented as a finite and modular library. This assumption enables deterministic elimination and avoids the need for continuous search or optimization.

A key requirement is the completeness and expressiveness of the constraint set induced by impulses. The system relies on these constraints to eliminate infeasible candidates and resolve a unique action. If the constraint structure is insufficient, multiple admissible candidates may remain, resulting in an unresolved state. In such cases, additional impulses or structural extension are required to restore determinacy.

The binary nature of compatibility evaluation may introduce sensitivity to noise and uncertainty. In real-world environments, imperfect or ambiguous inputs can affect the stability of constraint enforcement. Practical implementations may therefore require tolerance mechanisms or structured relaxation to maintain robustness under perturbations.

The framework does not directly address continuous or unbounded action spaces. Problems that inherently require exploration over continuous domains may need discretization or hierarchical structuring to be compatible with the proposed approach.

It is also important to note that the framework does not replace learning-based or optimization-based systems. Instead, it defines a complementary regime in which decision-making can be achieved through deterministic elimination when structural conditions are well-defined and the candidate set is bounded.

Discussion

The proposed framework introduces a structural alternative to conventional decision-making paradigms

without positioning itself in opposition to them. Optimization-based and learning-based approaches have demonstrated strong performance in complex, uncertain, and high-dimensional environments. These methods are particularly effective when the action space is continuous or when the underlying structure is not explicitly defined.

In contrast, the present framework operates under a different set of assumptions. It considers decision-making within a finite, structured action space, where feasibility can be explicitly evaluated. Rather than exploring alternatives or approximating optimal solutions, the system resolves decisions through elimination of infeasible candidates.

A key distinction lies in how information is processed. Optimization-based systems typically reduce multiple criteria into a scalar objective, enabling comparison and ranking. The proposed approach preserves the multi-dimensional structure of constraints and determines outcomes through their intersection. As a result, decision-making is not driven by preference or minimization, but by consistency.

From a computational perspective, the framework avoids iterative search, convergence requirements, and parameter tuning. This leads to a lightweight and interpretable mechanism, particularly suitable for systems where the action space is well-defined and constraints are informative. The absence of training and model adaptation further simplifies implementation and enhances reproducibility.

At the same time, the framework does not aim to replace existing methods. Instead, it defines a complementary regime of autonomy. Optimization explores and refines; the present method prunes and resolves. In this sense, the framework shifts the computational burden from iterative evaluation to structural representation, emphasizing the role of constraint completeness over search efficiency. These two perspectives can coexist, addressing different classes of problems within autonomous systems.

The integration of impulse-driven activation, modular libraries, and structured extension provides a unified view of decision-making, adaptation, and collaboration. By maintaining a consistent mechanism across scales and contexts, the framework offers a coherent and interpretable alternative for structured environments.

Conclusion

This study introduced a deterministic framework for autonomous action initiation based on impulse-driven activation and elimination over a finite modular action library. The proposed approach departs from conventional formulations that rely on optimization, learning, and explicit goal specification, and instead defines decision-making as a process of constraint resolution.

Within this framework, actions are not selected through search or ranking but emerge as the only admissible outcome after incompatible candidates are eliminated. The impulse–accumulation–threshold mechanism provides a natural and efficient means of triggering decisions, ensuring that the system remains inactive until sufficient structural information is available.

The modular library architecture enables both stability and adaptability. Routine behaviors are captured through structured patterns, while new conditions are incorporated through controlled extension. The same underlying mechanism operates consistently across scales, from local refinements to global reconfiguration, allowing the system to evolve without altering its deterministic nature.

The framework further extends to multi-agent settings through selective structural exchange, enabling collaborative systems to grow without centralized control. This distributed extension preserves autonomy while enhancing capability through interaction.

Overall, the proposed formulation establishes a complementary paradigm for autonomous systems in which decision-making emerges from feasibility rather than optimality. By avoiding search, training, and scalar reduction, the framework provides a lightweight, interpretable, and structurally consistent approach for environments where action spaces are finite and constraints are well-defined.

Future work may explore extensions to hybrid settings, integration with continuous representations, and practical implementations in robotic and distributed systems.

Author Contributions

Huseyin Murat Cekirge is the sole author. The author read and approved the final manuscript.

Conflicts of Interest

The author declares no conflicts of interest.

References

1. Bishop CM (2006) Pattern Recognition and Machine Learning, Springerhttps://link.springer.com/book/9780387310732.
2. Goodfellow I, Bengio Y, Courville A (2016) Deep Learning, MIT Press https://www.deeplearningbook.org/.
3. Sutton RS, Barto AG (2018) Reinforcement Learning: An Introduction, 2nd ed., MIT Press https://psycnet.apa.org/record/2019-19679-000.
4. Russell S, Norvig P (2010) Artificial Intelligence: A Modern Approach, 3rd ed., Pearson https://api.pageplace.de/preview/DT0400.9781292153971_A27091185/preview-9781292153971_A27091185.pdf.
5. Kephart JO, Chess DM (2003) The Vision of Autonomic Computing. IEEE Computer 36: 41-50.
6. Huebscher MC, McCann JA (2008) A Survey of Autonomic Computing—Degrees, Models, and Applications. ACM Computing Surveys 40.
7. Hwang K, Fox G, Dongarra J (2012) Distributed and Cloud Computing: From Parallel Processing to the Internet of Things 672.
8. Tikhonov AN, Arsenin VY (1977) Solutions of Ill-Posed Problems https://dl.acm.org/doi/abs/10.1137/1021044.
9. Cekirge H (2026) An Efficient and Deterministic Object Recognition System, American Journal of Artificial Intelligence 10: 148-157.
10. Cekirge HM (2026) Structural Filtering for Object Recognition: A Deterministic Alternative to Optimization-Based Training. J of Poin Artf Research 2: 1-15.

Copyright: ©2026 Huseyin Murat Cekirge. This is an open-access article distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.