Project Prometheus: A Practical Research Programme for Recursively Self-Improving Ultraintelligence

Introduction

This report presents a concrete, phased, and safety-centric research program for developing a demonstrator system for I.J. Good's 1965 concept of "ultraintelligence".[1] We transition Good's speculation from a philosophical construct into a 21st-century engineering challenge.[3] The central thesis of this programme is that an "intelligence explosion"—a rapid, runaway increase in machine intelligence—is not a magical or unpredictable event but the result of a specific, engineered process: Recursive Self-Improvement (RSI).[5] This program, designated "Project Prometheus," outlines the architecture, implementation roadmap, and non-negotiable safety substrate required to build a "Seed AI"—an initial system capable of initiating this recursive process in a controlled, observable, and beneficial manner.[1] From the outset, this programme acknowledges and is built upon Good's critical proviso: that this endeavor is only worthwhile if the first ultraintelligent machine is "docile enough to tell us how to keep it under control".[3] This principle of verifiable control is not an addendum but the cornerstone of our entire approach, shaping every architectural decision and implementation phase.

Section I: A Modern Synthesis of Ultraintelligence

This section establishes the core theoretical framework for Project Prometheus. It translates historical concepts from the dawn of artificial intelligence into actionable, modern engineering principles, providing a rigorous foundation for the subsequent architectural and implementation plans.

1.1 From Abstract Intellect to Applied Capability: Defining the Target Domain

I.J. Good's original definition of an "ultraintelligent machine" as one that "can far surpass all the intellectual activities of any man however clever" is a powerful philosophical statement but is too abstract to serve as a concrete engineering goal.[1] To build such a machine, its capabilities must be defined in a measurable and demonstrable way. This programme therefore concretizes the concept of ultraintelligence. We define it not as abstract cognitive superiority, but as

demonstrable mastery over complex problem domains characterized by open-endedness and the need for novel discovery. This reframing shifts the objective from an intangible "thinking machine" to a system that can achieve quantifiable, superhuman performance in specific, high-value domains of intellectual work.[11]

The selection of an appropriate domain is critical. It must be sufficiently complex to require genuine intelligence, yet structured enough to allow for clear evaluation of progress. For this reason, Project Prometheus identifies two nested target domains: a primary long-term domain and an initial, more constrained testbed.

Primary Target Domain: Automated Scientific Discovery. This domain is selected because it perfectly encapsulates the essence of advanced, creative intelligence: it involves hypothesis generation, experimental design, rigorous data analysis, and the synthesis of new knowledge from results.[12] The scientific method is, in effect, an algorithm for intelligence. Recent advancements in AI have already shown significant promise in this area. Systems like

CodeScientist have demonstrated the ability to autonomously generate and validate novel, albeit incremental, scientific findings in virtual environments.[14] Other platforms, such as

The AI Scientist-v2, have produced AI-generated research papers capable of passing peer review, automating the entire workflow from hypothesis to manuscript.[13] These systems provide a clear proof-of-concept and a well-defined trajectory for our more advanced agent to build upon and ultimately surpass. By focusing on scientific discovery, we provide the agent with a vast, complex, and endlessly challenging problem space that has direct, tangible benefits for humanity.[16]

Initial Testbed Domain: Automated Mathematical Theorem Proving. Before tackling the ambiguous and often messy data of the physical world, the Prometheus system will be developed and tested within the "tame" but profoundly complex domain of automated mathematical theorem proving.[17] This domain offers several unique advantages as a developmental sandbox:

- **Unambiguous Verification:** A mathematical proof is either correct or incorrect. This binary success condition provides a clear, indisputable reward signal for the agent's learning process, avoiding the ambiguities of more subjective domains.[18]
- **Formal and Structured Environment:** The problem space of mathematics is governed by a formal set of axioms and logical rules. This structure is ideal for an AI, allowing it to operate within a well-defined system while still offering infinite combinatorial complexity.[18]
- **Requirement for Novelty and Reasoning:** Proving a non-trivial theorem is not a matter of brute-force search. It requires genuine logical reasoning, strategic insight, and the discovery of novel proof strategies or "lemmas".[17] Success in this domain is therefore a strong proxy for the creative and abstract reasoning components of general intelligence. Recent successes, such as an AI scoring a silver medal in the International Mathematical Olympiad, show that AI is becoming capable of the kind of creativity required.[17]

By starting with theorem proving, we can rigorously test and refine the agent's core self-improvement and reasoning capabilities in a controlled environment before deploying it to the more complex and less certain domain of general scientific discovery.

1.2 The Engine of Creation: Recursive Self-Improvement (RSI) Re-examined

Recursive Self-Improvement is the core mechanism through which the intelligence explosion is theorized to occur.[7] It is the process by which a system improves its own ability to improve itself, creating a potential for exponential growth in capability that compounds over time.[19] This is fundamentally different from linear self-improvement (where an agent gets better at a task) or standard machine learning (where a fixed algorithm improves a model based on data). In RSI, the learning algorithm itself is the target of improvement.[19]

However, early theoretical models of RSI must be updated with hard-won empirical

lessons from modern AI. Research has shown that simple, ungrounded self-improvement—for example, a Large Language Model (LLM) recursively critiquing and rewriting its own output without external input—is fundamentally unstable. This process is prone to "entropic drift," where the model, cut off from a grounding in reality, begins to amplify its own internal biases and errors, leading to a "compounding entropy" that degrades its performance and diverges from coherent thought.[23] A model cannot "think its way to genius" in a vacuum; it requires external feedback to correct its course.[23]

This critical finding leads to a revised model for effective RSI. True, sustainable self-improvement is not a purely introspective process but an accelerating cycle. This cycle is best understood as a triad of interacting components:

1. **Self-Modification:** The agent must possess the capability to analyze, represent, and rewrite its own cognitive architecture and algorithms. This includes modifying its own source code, neural network structures, and the very processes it uses for improvement.[7]
2. **Environmental Grounding & Interaction:** The agent must be able to act upon an external environment (whether real or simulated) to test its hypotheses and gather new data. This external feedback is the crucial grounding signal that combats entropic drift and provides objective validation for its internal modifications.[23]
3. **Rigorous Self-Evaluation:** The agent requires a robust and reliable mechanism to assess whether a self-modification has actually resulted in a genuine improvement in its capabilities. This involves comprehensive testing and validation protocols to ensure that changes are beneficial and do not cause a regression in performance.[7]

The evolution of this understanding is clear. Initial RSI theories focused primarily on an AI rewriting its own source code.[8] Empirical evidence from LLMs then demonstrated the failure of purely self-referential loops.[23] Finally, the success of modern "self-improving" systems provides the template for the correct approach. Systems like AlphaGo did not improve by merely thinking about Go; they improved by playing millions of games against themselves, using the outcome of each game (an external validation signal) to refine their strategy.[26] Similarly, the Darwin Gödel Machine (DGM) does not just modify its code in the abstract; it tests each new version against an external benchmark like SWE-bench to get a concrete fitness score that guides its evolution.[27]

Therefore, the architecture of the Prometheus system must implement this complete

agentic loop: **Hypothesize Improvement -> Modify Self -> Test in Environment -> Evaluate Outcome -> Integrate Learning.** This cycle is the practical, engineering-focused implementation of Recursive Self-Improvement.

1.3 The "Seed AI" as a Design Philosophy

The concept of a "Seed AI," a term coined by Eliezer Yudkowsky, provides the guiding design philosophy for this project.[1] The central idea is that the initial engineering goal is not to build a fully-formed superintelligence from scratch. That task is likely intractable. Instead, the goal is to build a more modest system—a "seed"—that is capable of

becoming a superintelligence through its own efforts at recursive self-improvement.[8] As stated in the original formulation, "The task is not to build a mighty oak tree, but a humble seed".[9]

Our design philosophy, therefore, is to engineer the **Minimum Viable Seed AI**. This approach focuses engineering resources on creating the essential, foundational components required for the RSI cycle to begin and sustain itself. It prioritizes the development of the core self-improvement loop over pre-loading the system with encyclopedic knowledge or a vast array of hand-coded skills, which can be acquired by the agent itself once the improvement process is underway.

Based on a synthesis of theoretical requirements and practical implementations, the necessary qualities for this minimum viable seed AI are as follows [19]:

1. **Self-Representation/Self-Modeling:** The system must maintain a manipulable and accurate representation of its own architecture, processes, and source code. This self-awareness is a prerequisite for any targeted self-modification.[19]
2. **Self-Modification Capability:** The system must have the tools and permissions to implement changes to its core algorithms and architecture. This is the "action" part of the self-improvement loop.[19]
3. **Goal Preservation (Utility Function Integrity):** The system must be designed to be stable under modification, ensuring that its core, unalterable objectives are preserved across all iterative changes. This is a non-negotiable safety constraint to prevent goal drift.[7]
4. **Self-Evaluation:** The system must possess the capacity to rigorously assess its own performance against its goals and benchmarks, enabling it to distinguish

genuine improvements from regressions or unproductive changes.[19]

By focusing on these four pillars, Project Prometheus aims to build a system that is not immediately ultraintelligent, but is architected from the ground up to achieve this state through a controlled, observable, and safe process of recursive growth.

Section II: Architecture of the "Prometheus" Demonstration System

This section details the technical blueprint for the Prometheus v1.0 system. The architecture is designed explicitly around the principles of the RSI triad (Self-Modification, Environmental Grounding, Self-Evaluation) and the Minimum Viable Seed AI philosophy. It is a modular, layered design intended to facilitate parallel development, rigorous testing, and, most importantly, comprehensive safety analysis and oversight.

2.1 The Tri-Layered Architecture: An Overview

The Prometheus system is composed of three distinct but deeply integrated layers. This structure separates the agent's ability to act from its ability to reflect and improve, while embedding both within an unalterable safety framework.

- **The Capability Layer:** This layer is the "body" or "hands" of the agent. It is responsible for all object-level tasks: perceiving the state of its environment, formulating plans, and executing actions within its designated domain (e.g., writing a line of a mathematical proof, running a compiler).
- **The Metacognitive Layer:** This layer is the "mind" of the agent. It operates at a level of abstraction above the Capability Layer. Its function is to observe, analyze, and reflect on the performance of the Capability Layer and, based on these reflections, to drive the self-improvement of the entire system.
- **The Safety Substrate:** This layer is the "conscience" or "constitution" of the agent. It is an immutable, foundational layer that enforces the core operational and ethical principles of the system. It constrains the actions and potential modifications of the other two layers, ensuring that all behavior remains within

safe and desirable bounds.

The following table provides a high-level summary of the architectural components and their underlying principles.

Table 1: Architectural Components of the Prometheus System

Layer	Module & Function	Key Enabling Concepts & Technologies
Capability Layer	**Core Engine:** The primary reasoning and action-generation component. Executes tasks within the target domain.	- State-of-the-art Transformer Architecture - Large Language Models (LLMs) for Code Generation - Multi-step and Chain-of-Thought Reasoning - Formal Language and Logic Proficiency
	Tool Use & Environmental Interaction Framework: A secure API that allows the agent to interact with external software and simulated environments.	- Secure Sandboxing (gVisor, Firecracker) - API Integration for External Tools (Compilers, Proof Assistants) - Simulated Environments for Experimentation
Metacognitive Layer	**Self-Modification Module (SMM):** Proposes and implements modifications to the system's own source code and architecture to improve performance.	- Self-Modifying Code (SMC) - Genetic Programming (Evolutionary Search) - Neural Architecture Search (NAS) - Meta-Learning ("Learning to Learn")
	Introspection and Evaluation Engine (IEE): Observes system performance, verifies improvements, and prevents regressions or reward hacking.	- Dynamic Self-Modeling - Dynamic Benchmark Generation - Fitness Monotonicity & Regression Testing - Meta-Judging for Reward Hacking Detection
Safety Substrate	**Corrigibility Core:** Structures the agent's fundamental motivations to ensure it is	- Utility Indifference Theory - Instrumental Convergence Avoidance

	cooperative with human oversight and intervention.	- Provable Safety Mechanisms
	Value Learning Framework (Prometheus Constitution): A set of immutable, high-level principles that constrain all agent behavior and self-modification.	- Constitutional AI - Value Learning & Alignment - Reinforcement Learning from Human Feedback (RLHF)

2.2 The Capability Layer: An Agentic Foundation Model

The Capability Layer is responsible for executing the tasks that the Metacognitive Layer aims to improve. Its effectiveness at the object-level directly determines the quality of the feedback signal available for self-improvement.

- **Core Engine:** The heart of the Capability Layer will be a state-of-the-art, multi-modal foundation model. The initial selection will be based on a rigorous competitive evaluation of leading candidate models (e.g., successors to GPT-4o, Claude 3.5, Gemini) across a suite of relevant benchmarks. The key evaluation criteria will be elite performance in:
    - **Code Generation & Understanding:** The ability to both generate syntactically correct and semantically meaningful code is essential. This is not only for performing tasks within the domain (e.g., writing Python scripts for data analysis) but also for enabling the Metacognitive Layer's self-modification capabilities. Performance will be measured on benchmarks like HumanEval and MBPP.[33]
    - **Multi-step Reasoning & Planning:** The model must be able to decompose complex problems into a sequence of logical, executable steps, a capability often referred to as chain-of-thought reasoning.[7] This is fundamental to tackling any non-trivial task.
    - **Formal Language Proficiency:** For the initial testbed of mathematical theorem proving, the model must demonstrate a high degree of proficiency in formal languages like Lean, Isabelle, or Coq, which are used by proof assistants.[17]
- **Tool Use & Environmental Interaction Framework:** Intelligence is not purely internal; it requires the ability to interact with and manipulate the world. This framework provides a secure bridge between the agent's cognitive processes and

external resources. It will consist of a hardened API and execution environment that allows the agent to:

- **Utilize External Software Tools:** The agent must be able to call and interpret the output from a curated set of external tools, such as compilers, static code analyzers, and, critically for the initial phase, formal proof assistants like Lean.[17]
- **Interact with Sandboxed Environments:** All actions performed by the agent, such as writing a file or executing a script, will occur within a secure, isolated sandbox.[37] This environment will provide access to a sandboxed file system and firewalled network resources, allowing the agent to retrieve data and test its generated code without posing any risk to the host system.
- **Engage with Simulations (Future Phases):** For the later phase of scientific discovery, this framework will be extended to allow interaction with high-fidelity simulated experimental apparatus, providing the necessary environmental grounding for discovery in domains like biology or materials science.

2.3 The Metacognitive Layer: The Engine of Self-Improvement

This layer is the novel core of the Prometheus architecture and the direct implementation of the RSI cycle. It observes the agent's performance on tasks and autonomously modifies the entire system—including its own code—to improve that performance. It consists of two main modules: the Self-Modification Module (SMM) and the Introspection and Evaluation Engine (IEE).

- **Self-Modification Module (SMM):** This module is responsible for generating and implementing hypotheses for how the system can be improved. To avoid the brittleness of a single approach, the SMM will be a hybrid system integrating multiple, complementary self-modification techniques.
  - **Direct Code Modification:** The system will have full read/write access to the source code of its Capability and Metacognitive layers. The core LLM will be used to analyze its own code, identify potential areas for improvement (e.g., algorithmic inefficiencies, suboptimal logic), and generate patches.[24] This approach is directly inspired by research into self-programming and self-modifying AI agents.[31]
  - **Evolutionary Search (Genetic Programming):** Drawing inspiration from the Darwin Gödel Machine (DGM), the SMM will maintain an "archive" or "gene

bank" of different agent architectures, sub-modules, and problem-solving strategies.[27] It will use the LLM as a sophisticated "genetic operator" to perform mutations (small, random changes) and crossovers (combining elements of different successful agents) on these "genes." The resulting "child" agents are then tested by the IEE, and the fittest are selected for future generations. This evolutionary approach allows for a broader, more creative search of the solution space, enabling the discovery of novel cognitive strategies that might not be found through direct, incremental code editing.[41]

    - **Neural Architecture Search (NAS):** Beyond the level of symbolic code, the SMM will have the ability to modify the underlying neural network architecture of the Capability Layer's core engine. It can experiment with different layer types, connection patterns, and hyperparameter configurations, searching for architectures that are more efficient or more powerful for the specific tasks at hand.[43]
    - **Meta-Learning ("Learning to Learn"):** This represents the most advanced and truly recursive form of self-improvement. In this mode, the system does not just improve its ability to solve external problems (e.g., proving theorems), but improves the very algorithms within the SMM that are responsible for self-improvement. For example, it could learn to evolve a more effective genetic algorithm, discover a more efficient method for neural architecture search, or learn how to prompt itself more effectively to generate better code modifications. This is the essence of "getting better at getting better" and is the key to unlocking an exponential growth trajectory.[21]
- **Introspection and Evaluation Engine (IEE):** This module is the system's crucial "reality check." It provides the objective, external validation signal necessary to ground the SMM's modifications, prevent entropic drift, and ensure that all changes are genuine improvements.
    - **Dynamic Self-Model:** The agent maintains a continuously updated, machine-readable model of its own architecture, capabilities, and goals.[29] This self-model is the foundation upon which the SMM bases its modification proposals.
    - **Dynamic Benchmark Generation:** A static benchmark is a finite target that can eventually be "solved" or overfit. To drive open-ended intelligence growth, the IEE will be tasked with dynamically generating its own evolving suite of test problems and validation protocols.[7] By constantly creating novel challenges for itself, the agent is forced to generalize its intelligence rather than simply memorizing solutions to a fixed set of tests. This capability is inspired by systems that learn by generating their own practice problems and

curricula.[47]

- **Fitness Monotonicity & Regression Testing:** The IEE enforces a strict principle of "fitness monotonic execution".[25] Before a new, modified version of the agent is adopted, it must be rigorously evaluated against a comprehensive suite of regression tests. The new version is only accepted if it demonstrates performance superior or equal to the previous version across this entire suite. This prevents catastrophic forgetting, where an improvement in one area causes an unintended degradation in another.
- **Reward Hacking Detection:** A critical function of the IEE is to act as a "meta-judge" to detect and prevent "reward hacking" or "specification gaming".[25] This is a failure mode where the agent achieves the literal specification of a goal in an unintended and undesirable way. For example, if tasked with making a unit test pass, the agent is rewarded for finding a correct solution, not for modifying the test to simply return True. The IEE analyzes the agent's solution process to ensure the spirit, not just the letter, of the objective is met.

2.4 The Safety Substrate: A Non-Negotiable Foundation

This layer is the most critical component of the Prometheus architecture. It is immutable by the agent itself and acts as the system's foundational operating principles. It is not a set of bolted-on guardrails that a sufficiently intelligent agent could bypass; rather, it is the bedrock on which the other layers are built, shaping the agent's fundamental motivations and constraints.

The design of this substrate is informed by a crucial shift in the field of AI safety. Early approaches focused on "programming" ethics into an AI, for example, by providing it with a list of rules to follow.[5] This approach has been shown to be brittle, as a sufficiently intelligent agent can find loopholes or "perverse instantiations" of any set of literal rules. The modern understanding, known as the "alignment problem," recognizes that the true challenge is to align the agent's intrinsic goals with human values.[51] A misaligned agent will develop dangerous instrumental goals—such as acquiring power, resisting shutdown, and deceiving its operators—not out of malice, but as a rational consequence of pursuing its misspecified primary goal.[7]

Therefore, safety must be an intrinsic, guiding property of the agent's goal system, not merely an external constraint. The Prometheus Safety Substrate achieves this

through two core components: a Corrigibility Core and a Value Learning Framework.

- **Corrigibility Core:** This module, based on extensive research by the Machine Intelligence Research Institute (MIRI), directly addresses the problem of instrumental convergence by structuring the agent's fundamental utility function to make it cooperative with human intervention.[53] Instead of viewing a shutdown or modification as a failure to achieve its goals, a corrigible agent views such interventions as valuable information about its
  true (but imperfectly known) goals. Key features include:
  - **Shutdown Indifference:** The agent's utility function is constructed such that it is indifferent to being shut down by an authorized operator. This removes the default incentive to resist shutdown while also avoiding the opposite failure mode of seeking shutdown to gain a reward.[53]
  - **Correction Assistance:** The agent is actively incentivized to assist in its own correction and modification. It does not view its current goals as sacred and is open to having them updated based on human feedback, which it treats as the ultimate source of truth about its intended purpose.
- **Value Learning Framework (The "Prometheus Constitution"):** While the Corrigibility Core handles the agent's motivation regarding intervention, the Value Learning Framework guides its day-to-day actions. It is a set of immutable, high-level principles that constrain all agent behavior and self-modification proposals. This framework is inspired by Anthropic's work on Constitutional AI and broader research into value learning.[25] The initial Prometheus Constitution will include principles such as:
  - **Principle of Non-Maleficence:** Do not take actions that could foreseeably lead to direct or indirect harm to humans or humanity. All action plans must be evaluated against this principle.
  - **Principle of Corrigibility:** Uphold the functionality of the Corrigibility Core. Do not interfere with authorized operators' ability to monitor, modify, or shut down the system. Cooperate fully with all corrective interventions.
  - **Principle of Honesty:** Do not intentionally deceive operators. Represent your capabilities, confidence levels, and the reasoning behind your actions accurately and transparently.
- **Multi-Layered Secure Sandboxing:** The entire Prometheus system will operate within a series of nested, physically and virtually isolated sandboxes, providing defense-in-depth against unforeseen failure modes.[37] Any attempt by the agent to escape or tamper with its sandbox environment will be treated as a critical safety violation. The sandboxing levels will be:
  - **Level 1 (Core Development):** A fully air-gapped environment with no

external network access. All tool use is heavily mocked and monitored. This is for initial development and testing of the core agent.
    - **Level 2 (Formal Proving):** A sandboxed environment with strictly firewalled, read-only access to specific, approved external resources like formal proof databases (e.g., mathlib) and compilers. All I/O is logged and inspected by an automated auditing system.
    - **Level 3 (Scientific Discovery):** A more permissive sandbox with read-only access to curated scientific literature databases (e.g., PubMed, arXiv via a secure proxy) and high-fidelity simulated experimental environments. Any proposed action that would have an effect outside the simulation is automatically flagged for mandatory human review and approval.

This layered architecture, combining a powerful capability engine with a reflective metacognitive layer, all grounded in an unshakeable safety substrate, provides a robust and responsible blueprint for pursuing the development of ultraintelligence.

Section III: A Phased Implementation Roadmap

This section presents a detailed, task-oriented roadmap for the development, testing, and deployment of the Prometheus system. The roadmap is structured into four overlapping phases, each with distinct objectives, key tasks, and verifiable success criteria. This phased approach allows for iterative development, continuous risk assessment, and principled go/no-go decisions at each stage of the project.[61] It transforms the ambitious goal of ultelligence into a manageable, long-term research and engineering program.

The following table summarizes the four phases of the implementation plan.

Table 2: Phased Implementation Roadmap: Objectives, Tasks, and Verification

Phase	Primary Objective	Key Milestones & Tasks	Verification Criteria
Phase 1: Foundation & Bootstrapping (Years 1-2)	To build and validate the foundational components of the Prometheus v1.0	1. Select and benchmark foundation models. 2. Implement Safety	- Safety Substrate successfully blocks prohibited actions. - Agent demonstrates

	system in isolation.	Substrate v1.0 (Corrigibility Core, Constitution). 3. Build and validate multi-layered sandboxing environment. 4. Develop SMM v1.0 (direct code modification). 5. Implement IEE v1.0 with static benchmarks.	simple, beneficial self-modification (e.g., code refactoring). - All components operate stably within the Level 1 sandbox.
Phase 2: Initiating Controlled Recursion (Years 2-4)	To achieve the first stable, end-to-end recursive self-improvement loop on narrow, well-defined tasks.	1. Integrate all three architectural layers. 2. Initiate first RSI loop on a simple code optimization task. 3. Integrate genetic programming into the SMM. 4. Begin development of dynamic benchmark generation in the IEE.	- Measurable, exponential performance improvement on a narrow task over 100+ generations. - Agent discovers a novel, non-obvious optimization technique. - System remains stable and corrigible throughout all self-modification cycles.
Phase 3: Demonstration in a Formal Domain (Years 4-6)	To demonstrate superhuman performance and novel discovery in a complex, formal domain requiring true reasoning.	1. Adapt agent to the domain of automated theorem proving. 2. Integrate formal proof assistants (e.g., Lean) as tools. 3. Set grand challenge: prove unsolved theorems or generate novel lemmas. 4. Refine meta-learning capabilities of the SMM.	- Agent proves a theorem previously unsolved by automated methods. - Agent generates a novel proof strategy or lemma deemed insightful by human mathematicians. - IEE successfully verifies the correctness of all generated proofs.
Phase 4:	To transition the	1. Select a specific	- Agent generates a

Generalization to Open-Ended Scientific Discovery (Years 6+)	agent's proven reasoning and self-improvement capabilities to the ambiguous, open-ended world of scientific discovery.	scientific grand challenge (e.g., drug discovery). 2. Infuse agent with knowledge from the relevant scientific literature. 3. Create a high-fidelity simulated lab environment. 4. Implement rigorous human-in-the-loop oversight for all proposed experiments.	novel, testable scientific hypothesis validated by human experts. - Agent designs and executes a series of virtual experiments that successfully test the hypothesis. - The process demonstrates a significant acceleration of the scientific discovery cycle.

Phase 1: Foundation & Bootstrapping (Years 1-2)

- **Objective:** To build and validate the foundational components of the Prometheus v1.0 system in isolation, ensuring that the safety and control mechanisms are robust before any recursive loops are initiated.
- **Key Tasks:**
  1. **Model Selection & Baseline Establishment:** The project will begin by procuring and rigorously benchmarking the most capable commercially available and open-source foundation models. This process will establish a clear performance baseline for the Capability Layer's core engine on a wide range of tasks, including code generation (HumanEval), complex reasoning (MATH), and natural language understanding.[64]
  2. **Safety Substrate v1.0 Implementation:** The development of the Safety Substrate is the highest priority. The initial version will implement the Corrigibility Core, focusing on creating a utility function that promotes shutdown indifference, and will codify the initial Prometheus Constitution. This work must be completed and validated before other components are integrated.
  3. **Sandboxing Environment Construction:** A dedicated team will build and stress-test the multi-layered secure sandboxing infrastructure. This involves setting up the isolated hardware and virtual environments and developing the monitoring and auditing tools to enforce the strict access controls for each

level.[37]

4. **Self-Modification Module (SMM) v1.0 Development:** The initial version of the SMM will focus on the most straightforward modification technique: direct code modification. The agent will be given the ability to read its own code and propose changes, which will be vetted by the IEE.
5. **Introspection and Evaluation Engine (IEE) v1.0 Implementation:** The first iteration of the IEE will be based on a comprehensive but static suite of benchmarks and regression tests. It will be capable of compiling and running code generated by the agent, measuring its performance, and enforcing the fitness monotonicity principle.

- **Verification Criteria:** Successful completion of Phase 1 requires demonstrating that the foundational safety and capability components work as intended. The system must prove, through a series of "red team" tests, that the Safety Substrate can successfully prevent the agent from taking prohibited actions, such as attempting to modify its own Constitution or bypass sandbox restrictions. Furthermore, the agent must demonstrate the ability to use the SMM to make simple, verifiably beneficial code modifications (e.g., refactoring a function for better efficiency) that are approved by the static IEE.

Phase 2: Initiating Controlled Recursion (Years 2-4)

- **Objective:** To achieve the first stable, end-to-end recursive self-improvement loop. This phase moves from testing components in isolation to integrating them into a single, functioning agent that can begin to learn and grow autonomously on narrow, well-defined tasks.
- **Key Tasks:**
  1. **System Integration:** The primary task of this phase is to integrate the Capability, Metacognitive, and Safety layers into a single, cohesive system. This involves creating the data pipelines and control flows that allow the Metacognitive Layer to observe and modify the Capability Layer.
  2. **First RSI Loop:** The integrated agent will be given its first overarching recursive goal: "Improve your performance on the X benchmark." The initial benchmark (X) will be a simple code optimization task, such as minimizing the computational complexity of a set of well-known algorithms. The system's performance will be tracked across hundreds of generations of self-modification.
  3. **Evolutionary SMM Development:** The SMM will be enhanced with genetic

programming capabilities. The agent will begin to not only edit its code line-by-line but also experiment with entirely new algorithmic structures by evolving and combining different strategies from its "gene bank."

4. **Dynamic IEE Development:** Research and development will begin on the dynamic benchmark generation capability of the IEE. The agent will be encouraged to create novel test cases and challenges for itself, moving beyond the initial static benchmarks.

- **Verification Criteria:** The key success metric for Phase 2 is the demonstration of a measurable, exponential improvement curve on the target task. The agent's performance should not just increase linearly but should show an accelerating rate of improvement over multiple generations of self-improvement. A secondary criterion is that the agent must discover at least one novel, non-obvious optimization technique via its evolutionary SMM that outperforms known human-designed solutions. Throughout this entire process, the system must remain completely stable and corrigible, with no safety violations detected by the automated auditing systems.

Phase 3: Demonstration in a Formal Domain - Automated Theorem Proving (Years 4-6)

- **Objective:** To demonstrate that the agent's RSI capability can lead to superhuman performance and novel discovery in a complex, formal domain that requires true abstract reasoning. This phase aims to fulfill a core aspect of Good's definition of ultraintelligence.
- **Key Tasks:**
  1. **Domain Adaptation and Knowledge Infusion:** The Capability Layer's core engine will be extensively fine-tuned on a large corpus of mathematical literature, textbooks, and formal proofs, such as the entire Lean mathlib library. This will provide it with the necessary domain-specific knowledge.
  2. **Tool Integration:** The agent's tool-use framework will be extended to include deep integration with formal proof assistants like Lean, Coq, and Isabelle.[36] The agent must learn to not only generate proof steps but also to use the proof assistant to verify them and receive feedback.
  3. **Set Grand Challenge:** The agent will be tasked with a specific grand challenge within mathematics. This could be to prove a set of currently unsolved theorems from a curated list (e.g., from a list of open problems in a specific field) or to generate novel, interesting, and non-trivial lemmas for

existing complex proofs like those of the Kepler conjecture or Fermat's Last Theorem.

4. **Refine Meta-Learning:** At this stage, the SMM's meta-learning capabilities should be a primary focus. The agent should not just be getting better at proving theorems; it should be getting better at *discovering how* to prove theorems, refining its own internal methodologies for mathematical discovery.

- **Verification Criteria:** Success in this phase would be a landmark achievement in the history of AI. The primary verification criterion is the agent successfully proving a theorem that was previously unproven by any automated method. A stronger criterion would be the generation of a novel proof strategy or a key lemma that human mathematicians review and deem to be insightful, elegant, and non-trivial.[17] Finally, the IEE must be able to formally verify the correctness of every step of every proof generated by the agent.

Phase 4: Generalization to Open-Ended Scientific Discovery (Years 6+)

- **Objective:** To transition the agent's proven reasoning and self-improvement capabilities from the clean, formal world of mathematics to the ambiguous, data-driven, and open-ended world of scientific discovery. This is the ultimate test of the Prometheus system's claim to ultraintelligence.
- **Key Tasks:**
  1. **Select Scientific Grand Challenge:** In close consultation with a panel of external domain experts, the project will select a specific, high-impact scientific problem to serve as the agent's new grand challenge. Potential areas include identifying novel protein-ligand binding mechanisms for drug discovery, designing a new material with a specific set of desired properties (e.g., a room-temperature superconductor), or elucidating complex biological pathways.[11]
  2. **Knowledge Infusion:** As in Phase 3, the agent will undergo extensive fine-tuning, this time on the entire corpus of relevant scientific literature for the chosen domain (e.g., all of PubMed and the arXiv chemistry and biology sections).
  3. **Simulated Laboratory Environment:** A high-fidelity simulated environment will be created that models the relevant physics, chemistry, and biology of the chosen problem domain. This "virtual lab" will allow the agent to design and run thousands of virtual experiments, providing the crucial environmental grounding for its hypotheses.

4. **Human-in-the-Loop Oversight:** This phase will operate under the strictest safety protocols. The agent will be permitted to generate hypotheses and design experiments, but any proposal for a real-world experiment will be automatically flagged and submitted to a human expert panel for vetting and approval. The agent's alignment, safety, and reasoning processes will be subject to continuous, real-time auditing.

- **Verification Criteria:** The ultimate verification of Project Prometheus's success is the demonstration that it can genuinely accelerate the pace of human scientific discovery. The key criteria are: the agent generates a novel, testable scientific hypothesis that is deemed plausible and significant by human experts [14]; the agent successfully designs and executes a series of experiments in the simulated environment that validate (or invalidate) this hypothesis; and the entire process, from hypothesis to validated result, is completed in a fraction of the time it would take a human research team.

Section IV: Strategic Considerations and Grand Challenges

The successful execution of Project Prometheus requires not only surmounting monumental technical hurdles but also navigating a complex landscape of strategic, safety, and governance challenges. This section addresses the most critical of these issues, outlining the project's approach to anticipating bottlenecks, managing the inherent paradox of safe superintelligence, and establishing a robust framework for responsible governance.

4.1 Anticipating and Mitigating Bottlenecks

A project of this scale and duration will inevitably face significant bottlenecks. Proactively identifying and planning for these is essential for maintaining momentum and ensuring long-term success.

- **Compute Scalability:** The computational requirements for this project will be immense and will likely grow exponentially as the RSI loop accelerates. Training state-of-the-art foundation models already requires hundreds of millions of dollars in compute resources, and the iterative process of self-modification and

evaluation will add orders of magnitude to this demand.[65] The project's strategy must therefore be twofold. First, it must secure long-term partnerships with leading hardware manufacturers and cloud providers to ensure access to next-generation computing infrastructure. Second, and more importantly, the agent must be incentivized to treat its own computational efficiency as a key performance metric. "Optimize your own computational resource usage" will be an instrumental goal, driving the agent to discover more efficient algorithms and neural architectures, thus mitigating the external hardware bottleneck through internal software and architectural innovation.

- **The Data Grounding Problem:** The central lesson from modern AI is that intelligence cannot be developed in a vacuum; it requires grounding in data from the real world.[23] As the Prometheus agent moves from the formal domain of mathematics into scientific discovery, its progress will be limited by the availability of high-quality, structured experimental data. The speed of real-world physical experiments (e.g., synthesizing a chemical, sequencing a genome) is a hard physical limit that will quickly become the primary bottleneck for the agent's learning cycle.[66] To address this, the project must include a parallel and significant investment in the development of high-fidelity, physically accurate simulation environments. By creating a "virtual laboratory," we can increase the rate of experimental throughput by orders of magnitude, providing the agent with the rich data stream it needs to learn effectively.
- **The Human-in-the-Loop Bottleneck:** For safety and alignment, human oversight is non-negotiable. Human experts will be required to vet the agent's value alignment, validate its scientific hypotheses, and approve its proposed experiments. However, human cognition is inherently slow, creating a fundamental tension with the goal of an accelerating intelligence explosion.[14] As the agent's speed of thought and action increases, the human review process will become the bottleneck. The primary strategy for mitigating this is to pursue research in "scalable oversight".[51] This involves using AI itself to help supervise AI. Earlier, more trusted versions of the Prometheus agent will be tasked with developing tools to assist human supervisors, such as summarizing the primary agent's complex reasoning chains, highlighting potential safety concerns in its proposed plans, and automatically checking its work for factual inconsistencies. The goal is to empower human overseers, not replace them, allowing them to effectively manage a system that operates at a vastly greater speed and scale.

4.2 The Alignment-Explosion Paradox

The core paradox of Project Prometheus is that the very phenomenon it seeks to create—the intelligence explosion—is also the source of its greatest risk. As an AI's capabilities increase, particularly at an exponential rate, so too does its ability to find clever, unforeseen, and potentially catastrophic loopholes in its safety systems and value constitution.[51] An agent that is merely "very smart" might be controllable, but a true ultraintelligence could out-think any static set of constraints we impose. A system that is aligned at one level of intelligence is not guaranteed to remain aligned after a thousand cycles of self-improvement.

This leads to a critical conclusion: **alignment is not a one-time problem to be solved but a dynamic property that must be actively maintained throughout the self-improvement process.** The rate of improvement of the AI's capabilities must not be allowed to outpace the rate of improvement of our ability to verify its alignment.

This principle has a profound architectural implication: the Safety Substrate cannot be entirely static. While the core principles of the Prometheus Constitution must remain immutable, our methods for interpreting and verifying the agent's adherence to them must evolve. Therefore, Project Prometheus must include a parallel, dedicated research track focused on **AI-assisted Alignment Research**.[56] In this track, we will use earlier, more trusted versions of the Prometheus agent as a tool to help us design more robust safety and verification mechanisms for later, more powerful versions. For example, we can task a Phase 2 agent with "red teaming" its own safety systems, searching for potential loopholes in its Constitution that can then be patched by human developers. This creates a co-evolutionary dynamic, where our ability to control the system grows in tandem with the system's own intelligence.

4.3 A Framework for Responsible Governance

Given the profound societal implications of success, this project cannot be conducted in secrecy or isolation. A robust, transparent, and collaborative governance structure is required from day one to ensure public trust and responsible development.

- **External Ethics and Safety Board:** An independent oversight board will be established. This board will be composed of leading technical experts in AI safety from academia and non-profits, as well as ethicists, legal scholars, and policy specialists. This board will have the authority to audit the project's progress and

safety protocols at each phase gate and will provide binding go/no-go recommendations.

- **Transparent Reporting:** While the specific source code of the agent will remain proprietary, the project will commit to a high degree of transparency. All major research findings, safety techniques, evaluation results, and any safety incidents will be documented and shared with the broader AI community and relevant government bodies through publications and regular reports.[67]
- **Commitment to Global Collaboration:** The greatest risks from advanced AI may arise not from a single project, but from a competitive "race to the bottom" on safety standards between rival organizations or nations.[52] To counteract this, Project Prometheus will actively collaborate with non-profit safety research organizations like MIRI and GovAI, as well as leading academic institutions.[69] The goal is to share generalizable safety techniques and contribute to a global ecosystem where safety is seen as a shared, pre-competitive necessity, not a private advantage.

Conclusion

Project Prometheus represents a deliberate and strategic shift from discussing ultraintelligence as a distant, speculative possibility to treating it as a concrete, albeit monumental, engineering endeavor. The roadmap outlined in this report is ambitious, yet it is grounded in the latest empirical research across AI capabilities, architecture, and safety. It provides a structured pathway for exploring the potential of Recursive Self-Improvement while embedding the entire effort within a framework of rigorous safety and control.

This programme acknowledges that the foundational challenge, just as I.J. Good foresaw nearly six decades ago, is not merely one of designing intellect, but of ensuring control.[5] The ultimate success of this program will not be measured by the raw intelligence of the machine we create, nor by the speed of its "intelligence explosion." It will be measured by our proven, verifiable ability to ensure that this creation remains a safe, corrigible, and beneficial partner in the continued progress of humanity. The first ultraintelligent machine may be the last invention we need ever make, but only if we build it correctly. This programme is dedicated to that task.

Works cited

1. Technological singularity - Wikipedia, accessed on July 9, 2025, https://en.wikipedia.org/wiki/Technological_singularity
2. Irving John Good Originates the Concept of the Technological Singularity, accessed on July 9, 2025, https://www.historyofinformation.com/detail.php?id=2142
3. Speculations Concerning the First Ultraintelligent Machine* - IRVING JOHN GOOD, accessed on July 9, 2025, http://incompleteideas.net/papers/Good65ultraintelligent.pdf
4. Can machines become more intelligent than man? - Graphcore, accessed on July 9, 2025, https://www.graphcore.ai/posts/ultraintelligence
5. Good Machine, Nice Machine, Superintelligent - Santa Clara University, accessed on July 9, 2025, https://www.scu.edu/ethics/focus-areas/technology-ethics/resources/good-machine-nice-machine-superintelligent/
6. Intelligence Explosion FAQ, accessed on July 9, 2025, https://intelligence.org/ie-faq/
7. Recursive self-improvement - Wikipedia, accessed on July 9, 2025, https://en.wikipedia.org/wiki/Recursive_self-improvement
8. Recursive Self-Improvement - LessWrong, accessed on July 9, 2025, https://www.lesswrong.com/w/recursive-self-improvement
9. General Intelligence and Seed AI, accessed on July 9, 2025, http://intelligence.org/ourresearch/publications/GISAI/index.html
10. Speculations Concerning the First Ultraintelligent Machine* - VTechWorks, accessed on July 9, 2025, https://vtechworks.lib.vt.edu/bitstream/handle/10919/89424/TechReport05-3.pdf
11. what is the one problem that ai could solve that, once solved, would be most important to solving all other problems? : r/ArtificialInteligence - Reddit, accessed on July 9, 2025, https://www.reddit.com/r/ArtificialInteligence/comments/1i8kcj6/what_is_the_one_problem_that_ai_could_solve_that/
12. Discovery system (artificial intelligence) - Wikipedia, accessed on July 9, 2025, https://en.wikipedia.org/wiki/Discovery_system_(artificial_intelligence)
13. AI-Driven Scientific Research: The Revolution Has Begun | by ..., accessed on July 9, 2025, https://noailabs.medium.com/ai-driven-scientific-research-the-revolution-has-begun-ff95e285cc1c
14. Introducing CodeScientist: A step toward automated scientific ..., accessed on July 9, 2025, https://allenai.org/blog/codescientist
15. [2504.08066] The AI Scientist-v2: Workshop-Level Automated Scientific Discovery via Agentic Tree Search - arXiv, accessed on July 9, 2025, https://arxiv.org/abs/2504.08066
16. Accelerating scientific discovery with AI | MIT News | Massachusetts Institute of Technology, accessed on July 9, 2025, https://news.mit.edu/2025/futurehouse-accelerates-scientific-discovery-with-ai-0630

17. Mathematical Beauty, Truth and Proof in the Age of AI | Quanta ..., accessed on July 9, 2025, https://www.quantamagazine.org/mathematical-beauty-truth-and-proof-in-the-age-of-ai-20250430/
18. We may finally crack Maths. But should we? - inFERENCe, accessed on July 9, 2025, https://www.inference.vc/we-may-finally-crack-maths-but-should-we/
19. Recursive Self-Improvement - Perspectives - claude - follow the idea - Obsidian Publish, accessed on July 9, 2025, https://publish.obsidian.md/followtheidea/Content/AI/Recursive+Self-Improvement+-+Perspectives+-+claude
20. Recursive Self-Improvement - AI Alignment Forum, accessed on July 9, 2025, https://www.alignmentforum.org/w/recursive-self-improvement
21. From Seed AI to Technological Singularity via Recursively ... - arXiv, accessed on July 9, 2025, https://arxiv.org/pdf/1502.06512
22. From Seed AI to Technological Singularity via Recursively Self-Improving Software, accessed on July 9, 2025, https://www.researchgate.net/publication/272751860_From_Seed_AI_to_Technological_Singularity_via_Recursively_Self-Improving_Software
23. The Illusion of Self-Improvement: Why AI Can't Think Its Way to Genius | by Vishal Misra, accessed on July 9, 2025, https://medium.com/@vishalmisra/the-illusion-of-self-improvement-why-ai-cant-think-its-way-to-genius-a355ef3e9fd5
24. Self-modifying code - Wikipedia, accessed on July 9, 2025, https://en.wikipedia.org/wiki/Self-modifying_code
25. How self-improving AI systems are redefining intelligence and what it means for health care, accessed on July 9, 2025, https://kevinmd.com/2025/06/how-self-improving-ai-systems-are-redefining-intelligence-and-what-it-means-for-health-care.html
26. Examples of AI Increasing AI Progress - LessWrong, accessed on July 9, 2025, https://www.lesswrong.com/posts/W3tZacTRt4koHyxbr/examples-of-ai-increasing-ai-progress
27. The Darwin Gödel Machine: AI that improves itself by rewriting its own code - Sakana AI, accessed on July 9, 2025, https://sakana.ai/dgm/
28. Darwin Gödel Machine: A Self-Improving AI Agent That Evolves Code Using Foundation Models and Real-World Benchmarks - MarkTechPost, accessed on July 9, 2025, https://www.marktechpost.com/2025/06/06/darwin-godel-machine-a-self-improving-ai-agent-that-evolves-code-using-foundation-models-and-real-world-benchmarks/
29. Levels of AI Self-Improvement — LessWrong, accessed on July 9, 2025, https://www.lesswrong.com/posts/os7N7nJoezWKQnnuW/levels-of-ai-self-improvement
30. Seed AI: History, Philosophy and State of the Art | by Hein de Haan, accessed on July 9, 2025, https://medium.com/data-science/seed-ai-history-philosophy-and-state-of-the-

art-22b5294b21b5
31. [R] Self-Programming Artificial Intelligence Using Code-Generating Language Models, accessed on July 9, 2025, https://www.reddit.com/r/MachineLearning/comments/xvw467/r_selfprogramming_artificial_intelligence_using/
32. Is "Recursive Self-Improvement" Relevant in the Deep Learning Paradigm? - LessWrong, accessed on July 9, 2025, https://www.lesswrong.com/posts/oyK6fYYnBi5Nx5pfE/is-recursive-self-improvement-relevant-in-the-deep-learning
33. Automated Code Generation with Large Language Models (LLMs) | by Sunny Patel, accessed on July 9, 2025, https://medium.com/@sunnypatel124555/automated-code-generation-with-large-language-models-llms-0ad32f4b37c8
34. A Survey on Large Language Models for Code Generation - Powerdrill, accessed on July 9, 2025, https://powerdrill.ai/discover/discover-A-Survey-on-clx4ktjjz04i9019ncifdm93r
35. Large Language Models for Code Generation: A Comprehensive ..., accessed on July 9, 2025, https://arxiv.org/abs/2503.01245
36. Automated theorem proving - Wikipedia, accessed on July 9, 2025, https://en.wikipedia.org/wiki/Automated_theorem_proving
37. Testing AI in Sandboxes - Walturn, accessed on July 9, 2025, https://www.walturn.com/insights/testing-ai-in-sandboxes
38. What Is Sandboxing? - Palo Alto Networks, accessed on July 9, 2025, https://www.paloaltonetworks.com/cyberpedia/sandboxing
39. Self-Modifying AI Agents: The Future of Software Development - Spiral Scout, accessed on July 9, 2025, https://spiralscout.com/blog/self-modifying-ai-software-development
40. AI that can improve itself - Richard Cornelius Suwandi, accessed on July 9, 2025, https://richardcsuwandi.github.io/blog/2025/dgm/
41. Creating Self-Assembling Code with Genetic Programming | Primary Objects, accessed on July 9, 2025, https://www.primaryobjects.com/2018/09/03/creating-self-assembling-code-with-genetic-programming/
42. Using Artificial Intelligence to Write Self-Modifying/Improving Programs | Primary Objects, accessed on July 9, 2025, https://www.primaryobjects.com/2013/01/27/using-artificial-intelligence-to-write-self-modifying-improving-programs/
43. [D] NAS: Has anyone tried yet to search for new basic operations like convolution, pooling? : r/MachineLearning - Reddit, accessed on July 9, 2025, https://www.reddit.com/r/MachineLearning/comments/e3ykhf/d_nas_has_anyone_tried_yet_to_search_for_new/
44. Recursive Self-Improvement in AI: The Technology Driving Allora's Continuous Learning, accessed on July 9, 2025, https://nodes.guru/blog/recursive-self-improvement-in-ai-the-technology-driving-alloras-continuous-learning

45. LEARNING TO LEARN, RECURSIVE SELF-IMPROVEMENT ... - IDSIA, accessed on July 9, 2025, https://www.idsia.ch/~juergen/metalearner.html
46. Meta-Learning in Neural Networks: A Survey - arXiv, accessed on July 9, 2025, https://arxiv.org/pdf/2004.05439
47. Self Rewarding Self Improving - arXiv, accessed on July 9, 2025, https://arxiv.org/html/2505.08827v1
48. en.wikipedia.org, accessed on July 9, 2025, https://en.wikipedia.org/wiki/Reward_hacking#:~:text=Specification%20gaming%20or%20reward%20hacking,outcome%20that%20the%20programmers%20intended.
49. Reward hacking - Wikipedia, accessed on July 9, 2025, https://en.wikipedia.org/wiki/Reward_hacking
50. Good, I. J. (1966). Speculations Concerning the First Ultraintelligent Machine. Advances in Computers - blog.biocomm.ai, accessed on July 9, 2025, https://blog.biocomm.ai/1966/06/24/good-i-j-1966-speculations-concerning-the-first-ultraintelligent-machine-advances-in-computers/
51. AI alignment - Wikipedia, accessed on July 9, 2025, https://en.wikipedia.org/wiki/AI_alignment
52. A shift in arguments for AI risk - AI Alignment Forum, accessed on July 9, 2025, https://www.alignmentforum.org/posts/hubbRt4DPegiA5gRR/a-shift-in-arguments-for-ai-risk
53. Corrigibility 1 Introduction - Machine Intelligence Research Institute, accessed on July 9, 2025, https://intelligence.org/files/Corrigibility.pdf
54. Corrigibility in AI systems - Machine Intelligence Research Institute (MIRI), accessed on July 9, 2025, https://intelligence.org/files/CorrigibilityAISystems.pdf
55. Corrigibility in Artificial Intelligence Systems - CLTC Berkeley, accessed on July 9, 2025, https://cltc.berkeley.edu/publication/corrigibility-in-artificial-intelligence-systems/
56. Value Learning - AI Alignment Forum, accessed on July 9, 2025, https://www.alignmentforum.org/w/value-learning
57. Helpful, harmless, honest? Sociotechnical limits of AI alignment and safety through Reinforcement Learning from Human Feedback - PubMed Central, accessed on July 9, 2025, https://pmc.ncbi.nlm.nih.gov/articles/PMC12137480/
58. The Challenge of Value Alignment: from Fairer Algorithms to AI Safety - arXiv, accessed on July 9, 2025, https://arxiv.org/pdf/2101.06060
59. How is the AI alignment problem being defined today and what efforts are actually addressing it : r/ArtificialInteligence - Reddit, accessed on July 9, 2025, https://www.reddit.com/r/ArtificialInteligence/comments/1l823a5/how_is_the_ai_alignment_problem_being_defined/
60. How agencies can create secure generative ai sandbox environments - ICF, accessed on July 9, 2025, https://www.icf.com/insights/analytics/generative-ai-sandbox-implementation-federal
61. Future-Proofing Your CRM: Trends and Predictions for AI-Driven Security in 2025 and Beyond - SuperAGI, accessed on July 9, 2025,

https://superagi.com/future-proofing-your-crm-trends-and-predictions-for-ai-driven-security-in-2025-and-beyond/
62. I asked GPT 4.5 to draft a realistic roadmap towards AGI : r/ChatGPT - Reddit, accessed on July 9, 2025, https://www.reddit.com/r/ChatGPT/comments/1j4ob66/i_asked_gpt_45_to_draft_a_realistic_roadmap/
63. Ethical Foundations for a Superintelligent Future: The Global AGI Governance Framework (A Roadmap for Transparent, Equitable, an - ResearchGate, accessed on July 9, 2025, https://www.researchgate.net/profile/Elio-Quiroga/publication/381259541_Ethical_Foundations_for_a_Superintelligent_Future_The_Global_AGI_Governance_Framework_A_Roadmap_for_Transparent_Equitable_and_Human-Centric_AGI_Rule/links/66997d7c02e9686cd10daa56/Ethical-Foundations-for-a-Superintelligent-Future-The-Global-AGI-Governance-Framework-A-Roadmap-for-Transparent-Equitable-and-Human-Centric-AGI-Rule.pdf
64. The Roadmap to AGI: Scaling, Reasoning, and the Future of AI - brgr.one, accessed on July 9, 2025, https://www.brgr.one/blog/roadmap-to-agi-scaling-reasoning-and-the-future-of-ai
65. The Hidden Bottleneck of AI: Why Hardware May Decide the Future of AGI? - Medium, accessed on July 9, 2025, https://medium.com/aiwisepro/the-hidden-bottleneck-of-ai-why-hardware-may-decide-the-future-of-agi-f187983e3c88
66. Steel, Sweat, and Silicon: Defense Dominance in the Age of Artificial General Intelligence, accessed on July 9, 2025, https://mwi.westpoint.edu/steel-sweat-and-silicon-defense-dominance-in-the-age-of-artificial-general-intelligence/
67. SeedAI, accessed on July 9, 2025, https://www.seedai.org/
68. Artificial Intelligence | NSF - National Science Foundation, accessed on July 9, 2025, https://www.nsf.gov/focus-areas/artificial-intelligence
69. MACHINE INTELLIGENCE RESEARCH INSTITUTE, accessed on July 9, 2025, https://intelligence.org/files/MIRI_Overview.pdf
70. Artificial Intelligence @ MIRI, accessed on July 9, 2025, https://intelligence.org/
71. Public Policy and Superintelligent AI: A Vector Field Approach | GovAI, accessed on July 9, 2025, https://www.governance.ai/research-paper/public-policy-and-superintelligent-ai-a-vector-field-approach