Adversarial AI for Solving Complex Security Problems in Engineered Systems

Daniel Tauritz, PhD

AI4Sec:FND Course
Auburn University

January 18, 2022

Appointments

Los Alamos National Laboratory (LANL)
Guest Scientist, A-4: Advanced Research in Cyber Systems

Appointments

Los Alamos National Laboratory (LANL)
Guest Scientist, A-4: Advanced Research in Cyber Systems

Sandia National Laboratories (SNL)
Cyber Security Consultant, Department 05623 – Special Cyber Initiatives

Appointments

Los Alamos National Laboratory (LANL)
Guest Scientist, A-4: Advanced Research in Cyber Systems

Sandia National Laboratories (SNL)
Cyber Security Consultant, Department 05623 – Special Cyber Initiatives

---

Auburn University (AU)
Interim Director and Chief Cyber AI Strategist, Auburn Cyber Research Center
Head, Biomimetic Artificial Intelligence Research Group (BioAI Group)
Director, Biomimetic National Security Artificial Intelligence (BONSAI) Lab
AU Director, LANL/AU Cyber Security Sciences Institute (CSSI)
Associate Professor, Department of Computer Science & Software Engineering
Affiliated Faculty, McCrary Institute for Cyber and Critical Infrastructure Security

Part I: Engineered Systems & Security

Why You Should Care

- You rely in your daily life on a myriad of engineered systems

Why You Should Care

- You rely in your daily life on a myriad of engineered systems
- Modern engineered systems tend to be cyber-physical in nature

Why You Should Care

- You rely in your daily life on a myriad of engineered systems
- Modern engineered systems tend to be cyber-physical in nature
- Cyber-physical engineered systems are extremely vulnerable to attack

Why You Should Care

- You rely in your daily life on a myriad of engineered systems
- Modern engineered systems tend to be cyber-physical in nature
- Cyber-physical engineered systems are extremely vulnerable to attack
- Cyber-physical engineered system attack surfaces tend to be astronomically large and infeasible to fully secure

Why You Should Care

- You rely in your daily life on a myriad of engineered systems
- Modern engineered systems tend to be cyber-physical in nature
- Cyber-physical engineered systems are extremely vulnerable to attack
- Cyber-physical engineered system attack surfaces tend to be astronomically large and infeasible to fully secure
- Only AI is capable of examining the combinatorially large number of unique attacks and defenses on modern engineered systems

What is an Engineered System?

NSF's Engineering Research Center website defines engineered systems as:

> "a combination of components that work in synergy to collectively perform a useful function. The engineered system could, for example, wholly or in part constitute a new technology for a new product line a new manufacturing process, a technology to improve the delivery of a service, or an infrastructure system."

What is an Engineered System?

NSF's Engineering Research Center website defines engineered systems as:

"a combination of components that work in synergy to collectively perform a useful function. The engineered system could, for example, wholly or in part constitute a new technology for a new product line a new manufacturing process, a technology to improve the delivery of a service, or an infrastructure system."

Examples:

- Modern Automobiles, Planes, and Trains
- Industry 4.0: Chemical Plant, Biotechnology, Agriculture
- Advanced Manufacturing Facility
- Smart Electric Grid
- Internet, Enterprise Computer Networks, Cloud Computing

Critical Infrastructure Sectors

DHS' Cybersecurity and Infrastructure Security Agency (CISA) lists 16 critical infrastructure sectors:

- Chemical
- Commercial Facilities
- Communications
- Critical Manufacturing
- Dams
- Defense Industrial Base
- Emergency Services
- Energy
- Financial Services
- Food and Agriculture
- Government Facilities
- Healthcare and Public Health
- Information Technology Sector
- Nuclear Reactors, Materials, and Waste
- Transportation Systems
- Water and Wastewater Systems

Engineered System Security as a Game

- Two or more adversaries: one defender and one or more attackers

Engineered System Security as a Game

- Two or more adversaries: one defender and one or more attackers
- Attackers range from so-called "script-kiddies" to organized crime, terrorist organizations, and adversarial nation states

Engineered System Security as a Game

- Two or more adversaries: one defender and one or more attackers
- Attackers range from so-called "script-kiddies" to organized crime, terrorist organizations, and adversarial nation states
- Attacker goals are diverse

Engineered System Security as a Game

- Two or more adversaries: one defender and one or more attackers
- Attackers range from so-called "script-kiddies" to organized crime, terrorist organizations, and adversarial nation states
- Attacker goals are diverse
- Defender needs to simulatenously defend against this wide variety of attackers

Engineered System Security as a Game

- Two or more adversaries: one defender and one or more attackers
- Attackers range from so-called “script-kiddies” to organized crime, terrorist organizations, and adversarial nation states
- Attacker goals are diverse
- Defender needs to simulatenously defend against this wide variety of attackers
- Asymetric non-zero sum game

Engineered System Security as a Game

- Two or more adversaries: one defender and one or more attackers
- Attackers range from so-called "script-kiddies" to organized crime, terrorist organizations, and adversarial nation states
- Attacker goals are diverse
- Defender needs to simulatenously defend against this wide variety of attackers
- Asymetric non-zero sum game
- Game theory allows for mathematical analysis of adversarial models

Engineered System Security as a Game

- Two or more adversaries: one defender and one or more attackers
- Attackers range from so-called “script-kiddies” to organized crime, terrorist organizations, and adversarial nation states
- Attacker goals are diverse
- Defender needs to simulatenously defend against this wide variety of attackers
- Asymetric non-zero sum game
- Game theory allows for mathematical analysis of adversarial models
- Classic game theory does not scale to complex, real-world systems

Engineered System Security as a Game

- Two or more adversaries: one defender and one or more attackers
- Attackers range from so-called "script-kiddies" to organized crime, terrorist organizations, and adversarial nation states
- Attacker goals are diverse
- Defender needs to simulatenously defend against this wide variety of attackers
- Asymetric non-zero sum game
- Game theory allows for mathematical analysis of adversarial models
- Classic game theory does not scale to complex, real-world systems
- Computational game theory achieves scalability by approximating Nash equilibria

Part II: Adversarial AI

AI for security versus security of AI

- Security of AI (e.g., Adversarial Machine Learning)

AI for security versus security of AI

- Security of AI (e.g., Adversarial Machine Learning)
- AI for security

AI for security versus security of AI

- Security of AI (e.g., Adversarial Machine Learning)
- AI for security
- When applying AI to solve security problems, the AI can be vulnerable itself

Terminology

- Many computational problems can be formulated as **generate-and-test** search problems

Terminology

- Many computational problems can be formulated as **generate-and-test** search problems
- A **search space** contains the set of all possible solutions

Terminology

- Many computational problems can be formulated as **generate-and-test** search problems
- A **search space** contains the set of all possible solutions
- A **search space generator** is *complete* if it can generate the entire search space

Terminology

- Many computational problems can be formulated as **generate-and-test** search problems
- A **search space** contains the set of all possible solutions
- A **search space generator** is *complete* if it can generate the entire search space
- An **objective function** tests the quality of a solution

Terminology

- Many computational problems can be formulated as **generate-and-test** search problems
- A **search space** contains the set of all possible solutions
- A **search space generator** is *complete* if it can generate the entire search space
- An **objective function** tests the quality of a solution
- A **heuristic** is a problem-dependent rule-of-thumb

Terminology

- Many computational problems can be formulated as **generate-and-test** search problems
- A **search space** contains the set of all possible solutions
- A **search space generator** is *complete* if it can generate the entire search space
- An **objective function** tests the quality of a solution
- A **heuristic** is a problem-dependent rule-of-thumb
- A **meta-heuristic** determines the sampling order over a search space with the goal to find a near-optimal solution (or set of solutions)

Terminology

- Many computational problems can be formulated as **generate-and-test** search problems
- A **search space** contains the set of all possible solutions
- A **search space generator** is *complete* if it can generate the entire search space
- An **objective function** tests the quality of a solution
- A **heuristic** is a problem-dependent rule-of-thumb
- A **meta-heuristic** determines the sampling order over a search space with the goal to find a near-optimal solution (or set of solutions)
- A **hyper-heuristic** is a meta-heuristic for a space of programs

Algorithmic Toolbox

- A **Black-Box Search Algorithm (BBSA)** is a meta-heuristic which iteratively generates trial solutions employing solely the information gained from previous trial solutions, but no explicit problem knowledge

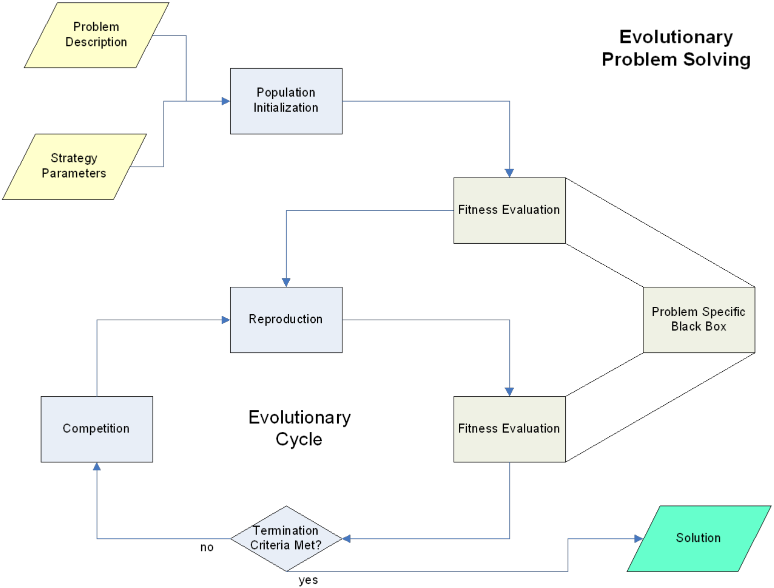
Algorithmic Toolbox

- A **Black-Box Search Algorithm (BBSA)** is a meta-heuristic which iteratively generates trial solutions employing solely the information gained from previous trial solutions, but no explicit problem knowledge
- **Evolutionary Algorithms (EAs)** can be described as a class of *stochastic*, *population-based* BBSAs inspired by *Evolution Theory*, *Genetics*, and *Population Dynamics*

Algorithmic Toolbox

- A **Black-Box Search Algorithm (BBSA)** is a meta-heuristic which iteratively generates trial solutions employing solely the information gained from previous trial solutions, but no explicit problem knowledge
- **Evolutionary Algorithms (EAs)** can be described as a class of *stochastic*, *population-based* BBSAs inspired by *Evolution Theory*, *Genetics*, and *Population Dynamics*
- **Genetic Programming (GP)** is a type of EA for searching a space of programs

Evolutionary Cycle



Genetic Programming

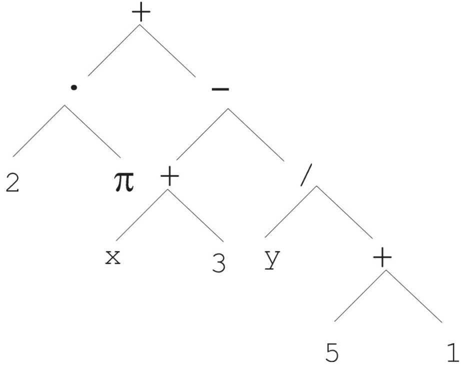
- EA with Hierarchical Representation for Model Identification

Genetic Programming

- EA with Hierarchical Representation for Model Identification
- Koza style Tree GP is the most prevalent

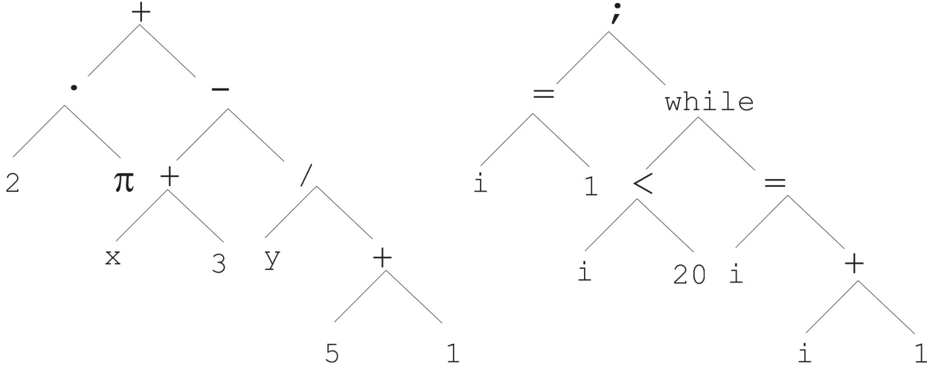
Genetic Programming

- EA with Hierarchical Representation for Model Identification
- Koza style Tree GP is the most prevalent

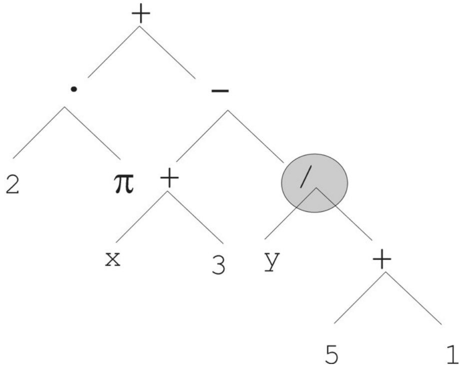


Genetic Programming

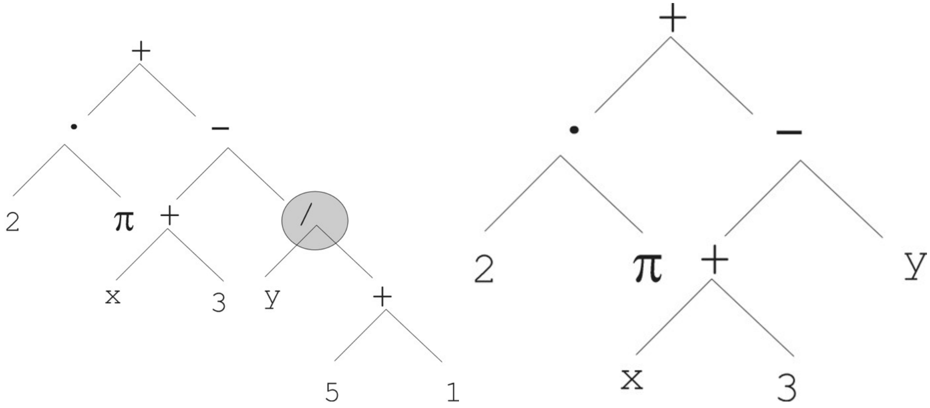
- EA with Hierarchical Representation for Model Identification
- Koza style Tree GP is the most prevalent



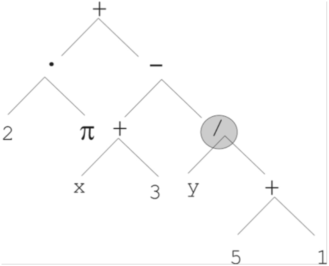
Genetic Programming - Mutation



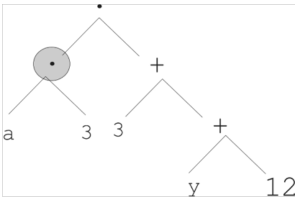
Genetic Programming - Mutation



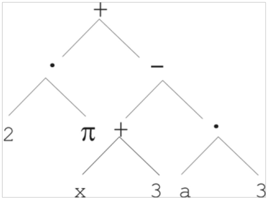
Genetic Programming - Recombination



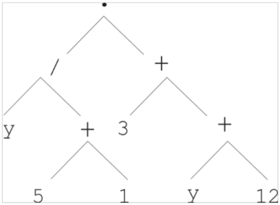
Parent 1



Parent 2



Child 1



Child 2

Real-World Game-Theoretic Problems

- Game Theory: multi-agent problem with conflicting utility functions

Real-World Game-Theoretic Problems

- Game Theory: multi-agent problem with conflicting utility functions
- Real-world examples:
  - economic & military strategy

Real-World Game-Theoretic Problems

- Game Theory: multi-agent problem with conflicting utility functions
- Real-world examples:
  - economic & military strategy
  - arms control

Real-World Game-Theoretic Problems

- Game Theory: multi-agent problem with conflicting utility functions
- Real-world examples:
  - economic & military strategy
  - arms control
  - auctions

Real-World Game-Theoretic Problems

- Game Theory: multi-agent problem with conflicting utility functions
- Real-world examples:
  - economic & military strategy
  - arms control
  - auctions
  - cyber security

Real-World Game-Theoretic Problems

- Game Theory: multi-agent problem with conflicting utility functions
- Real-world examples:
  - economic & military strategy
  - arms control
  - auctions
  - cyber security
- Common problem: real-world games are typically incomputable

Real-World Game-Theoretic Problems

- Game Theory: multi-agent problem with conflicting utility functions
- Real-world examples:
  - economic & military strategy
  - arms control
  - auctions
  - cyber security
- Common problem: real-world games are typically incomputable
- Solution: Computational Game Theory

Approximating Incomputable Games

- Consider the space of each user's actions

Approximating Incomputable Games

- Consider the space of each user's actions
- Perform local search in these spaces

Approximating Incomputable Games

- Consider the space of each user's actions
- Perform local search in these spaces
- Solution quality in one space is dependent on the search in the other spaces

Approximating Incomputable Games

- Consider the space of each user's actions
- Perform local search in these spaces
- Solution quality in one space is dependent on the search in the other spaces
- The simultaneous search of co-dependent spaces is naturally modeled as an armsrace

Classical Computational Solver Limitations

Complex real-world problems can be (practically) unsolvable with classic approaches

- Black box

Classical Computational Solver Limitations

Complex real-world problems can be (practically) unsolvable with classic approaches

- Black box
- “Ill-behaved” search space

Classical Computational Solver Limitations

Complex real-world problems can be (practically) unsolvable with classic approaches

- Black box
- "Ill-behaved" search space
- Intractable

Classical Computational Solver Limitations

Complex real-world problems can be (practically) unsolvable with classic approaches

- Black box
- “Ill-behaved” search space
- Intractable
- Evolution has a demonstrated ability to solve very complex problems

Coevolutionary Algorithm (CoEA)

CoEAs are a special type of EAs where the fitness of an individual is dependent on other individuals (i.e., individuals are explicitly part of the environment)

Coevolutionary Algorithm (CoEA)

CoEAs are a special type of EAs where the fitness of an individual is dependent on other individuals (i.e., individuals are explicitly part of the environment)

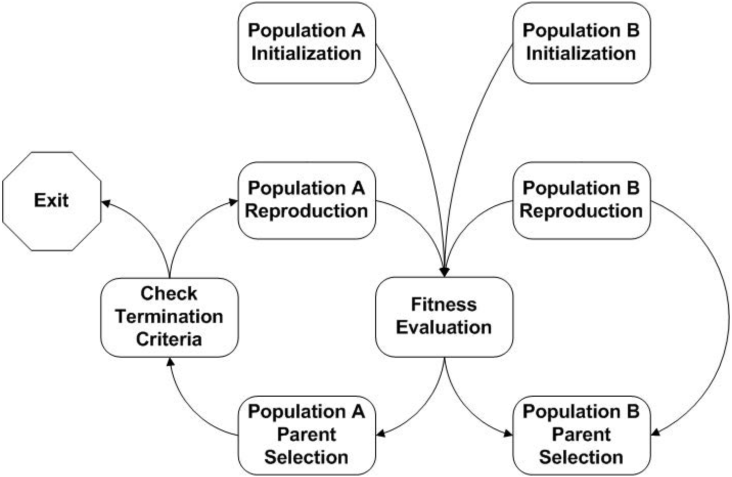
- Single species vs. multiple species

Coevolutionary Algorithm (CoEA)

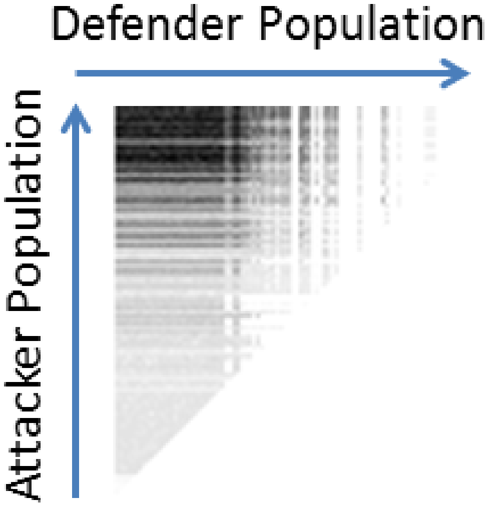
CoEAs are a special type of EAs where the fitness of an individual is dependent on other individuals (i.e., individuals are explicitly part of the environment)

- Single species vs. multiple species
- Cooperative vs. competitive coevolution

Two-Population Competitive Coevolutionary Cycle

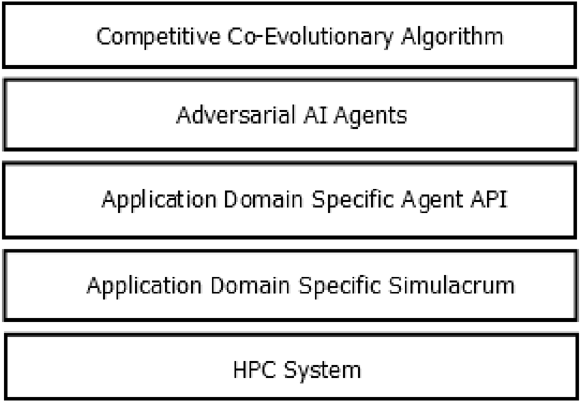


CIAO Plot Example

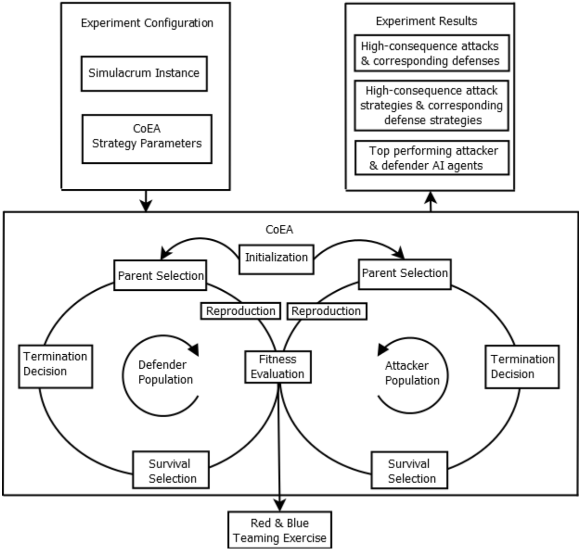


Part III: Engineered System Security through AI Armsraces

CEADS system diagram



CEADS CompCoEA operation



Outcomes

Coevolving attacke and defender AI agents can produce three distinct capabilities:

Outcomes

Coevolving attacke and defender AI agents can produce three distinct capabilities:

Attacks & Defenses Automated identification of vulnerabilities and candidate mitigations that are already tested against a large set of attacks.

Outcomes

Coevolving attacke and defender AI agents can produce three distinct capabilities:

Attacks & Defenses Automated identification of vulnerabilities and candidate mitigations that are already tested against a large set of attacks.

Attack & Defense Strategies Automated wargaming in order to identify high-consequence attack strategies and corresponding defense strategies.

Outcomes

Coevolving attacke and defender AI agents can produce three distinct capabilities:

Attacks & Defenses Automated identification of vulnerabilities and candidate mitigations that are already tested against a large set of attacks.

Attack & Defense Strategies Automated wargaming in order to identify high-consequence attack strategies and corresponding defense strategies.

Attacker & Defender AI Agents Automated generation of highly-trained AI agents that can be deployed in live systems to augment human operators, or even autonomously engage in real-time with adversaries, both human and AI.

How to Apply CEADS to an Engineered System

- Create simulacrum

How to Apply CEADS to an Engineered System

- Create simulacrum
- Design representation for AI agent actions

How to Apply CEADS to an Engineered System

- Create simulacrum
- Design representation for AI agent actions
- Create AI controller logic including sensory inputs

How to Apply CEADS to an Engineered System

- Create simulacrum
- Design representation for AI agent actions
- Create AI controller logic including sensory inputs
- Define attacker & defender fitness functions

How to Apply CEADS to an Engineered System

- Create simulacrum
- Design representation for AI agent actions
- Create AI controller logic including sensory inputs
- Define attacker & defender fitness functions
- Tune Competitive CoEA

Challenges

Operationalization In order to achieve operationalization, the attack & defense actions must be comprehensive and reflective of current and future threats, and must model targeted real-world systems with very high-fidelity requiring extensive knowledge of the targeted systems and a process for testing and eventually validating on said systems.

Challenges

Operationalization In order to achieve operationalization, the attack & defense actions must be comprehensive and reflective of current and future threats, and must model targeted real-world systems with very high-fidelity requiring extensive knowledge of the targeted systems and a process for testing and eventually validating on said systems.

Scalability Running a single automated red & blue teaming exercise employing adversarial AI agents can be quite computationally expensive depending on the size of the modeled engineered system and the fidelity to which it is being modeled. CoEAs typically require on the order of thousands of red & blue teaming exercises for a single experiment.

Addressing Scalabillity

The following approaches can be pursued to address scalability:

- Parallelization: CoEAs are embarrassingly parallel, so as long as the hardware resources are available to execute multiple instances simultaneously, a near linear speedup can be achieved.

Addressing Scalabillity

The following approaches can be pursued to address scalability:

- Parallelization: CoEAs are embarrassingly parallel, so as long as the hardware resources are available to execute multiple instances simultaneously, a near linear speedup can be achieved.
- Mixed Fidelity Execution: As long as low-fidelity models are within reasonable bounds of high-fidelity models, very significant speedups can be achieved by defaulting execution to low-fidelity models and only re-executing the most promising AI agents at computationally expensive high-fidelity. This is also known as surrogate modeling.

Addressing Scalabillity

The following approaches can be pursued to address scalability:

- Parallelization: CoEAs are embarrassingly parallel, so as long as the hardware resources are available to execute multiple instances simultaneously, a near linear speedup can be achieved.
- Mixed Fidelity Execution: As long as low-fidelity models are within reasonable bounds of high-fidelity models, very significant speedups can be achieved by defaulting execution to low-fidelity models and only re-executing the most promising AI agents at computationally expensive high-fidelity. This is also known as surrogate modeling.
- Partial Evaluations: By adding hooks into the simulacrum to query and control it during execution, agents performing sufficiently poorly can have their fitness estimated based on a partial evaluation, thus saving precious computational time for evaluating more promising agents. Additionally, poorly performing agents can be sampled at lower rates than promising ones for additional time savings.

Take Home Message

- Modern engineered systems are extremely vulnerable to attacks

Take Home Message

- Modern engineered systems are extremely vulnerable to attacks
- Adversarial AI applied to a high-fidelity simulacrum of the engineered system may be able to automatically identify vulnerabilities and corresponding mitigations

Take Home Message

- Modern engineered systems are extremely vulnerable to attacks
- Adversarial AI applied to a high-fidelity simulacrum of the engineered system may be able to automatically identify vulnerabilities and corresponding mitigations
- Appropriate configuring adversarial AI and creating high-fidelity sumulacra requires human engineers

Take Home Message

- Modern engineered systems are extremely vulnerable to attacks
- Adversarial AI applied to a high-fidelity simulacrum of the engineered system may be able to automatically identify vulnerabilities and corresponding mitigations
- Appropriate configuring adversarial AI and creating high-fidelity sumulacra requires human engineers
- AI is not a panacea, but can augment human experts to significantly improve results