Evolutionary Computation: Where We Are and Where We're Headed

Kenneth De Jong

Computer Science Department
George Mason University
kdejong@gmu.edu
www.cs.gmu.edu/~eclab

Historical roots

- Several parallel independent developments in the 1960s.
- Focus: Nature-inspired problem solving
  - i.e., computer science and engineering
- Not modeling of natural systems
  - Interesting area, but not the initial focus

Common Inspiration

- Problem solving based on a Darwinian notion of an evolutionary system.
- Basic elements:
  - a population of “individuals”
  - a notion of “fitness”
  - a birth/death cycle biased by fitness
  - a notion of “inheritance”

Historical roots:

- **Evolution Strategies (ESs)**:
  - developed by Rechenberg, Schwefel, etc. in 1960s.
  - focus: real-valued parameter optimization
  - individual: vector of real-valued parameters

Historical roots:

- **Evolutionary Programming (EP):**
  - Developed by Fogel in 1960s
  - Goal: evolve intelligent behavior
  - Individuals: finite state machines

Historical roots:

- **Genetic Algorithms (GAs)**:
  - developed by Holland in 1960s
  - goal: robust, adaptive systems
  - used an internal "genetic" encoding

Resulted in:

- wide variety of evolutionary algorithms (EAs)
- wide variety of applications
  - optimization
  - search
  - learning, adaptation
- well-developed analysis
  - theoretical
  - experimental

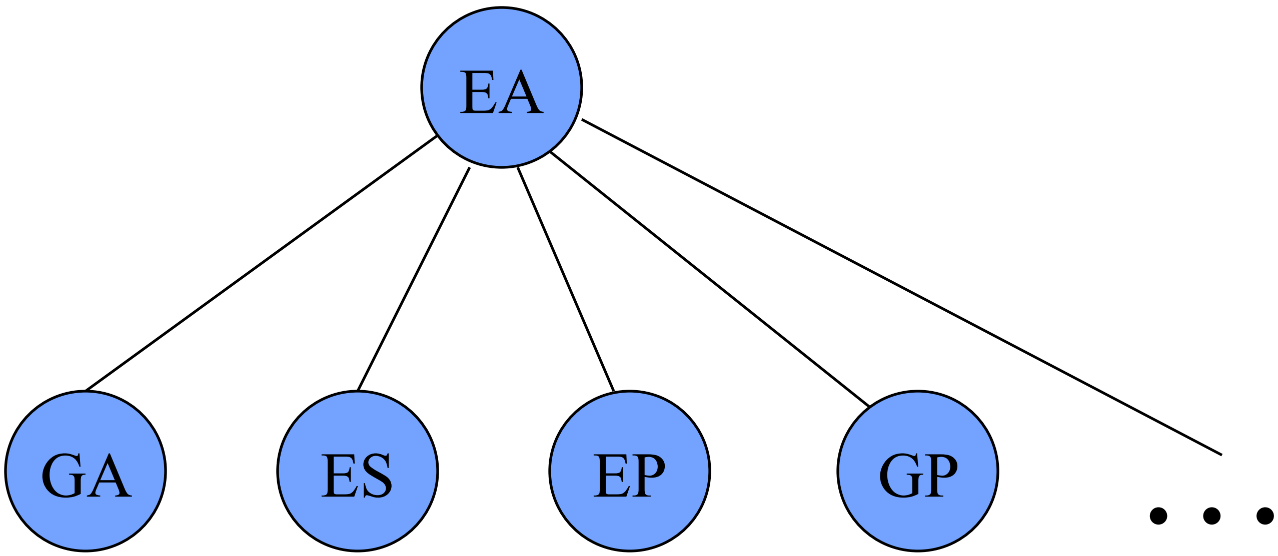
Also resulted in:

- A bewildering variety of algorithms.
- Historically inconsistent terminology.
- Hard to see relationships, assess strengths & weaknesses, make choices, ...

A Personal Interest:

- Develop a unifying framework that:
  - Helps one compare and contrast approaches.
  - Encourages crossbreeding.
  - Facilitates intelligent design choices.

Viewpoint:



A Simple Evolutionary Algorithm:

1. Randomly generate an initial population.

2. Do until some stopping criteria is met:

    Select individuals to be parents (biased by fitness).
    Produce offspring.
    Select individuals to die (biased by fitness).

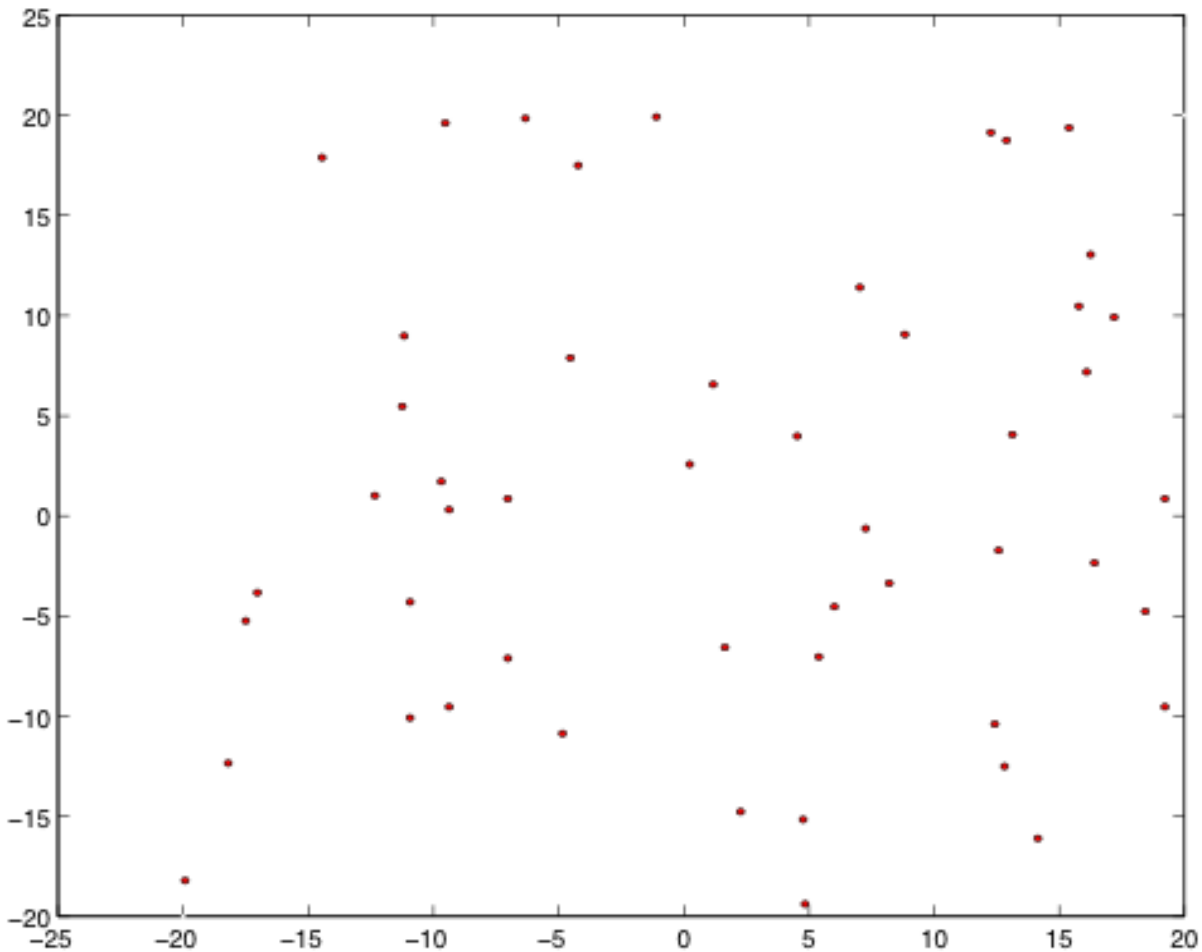
    End Do.

3. Return a result.

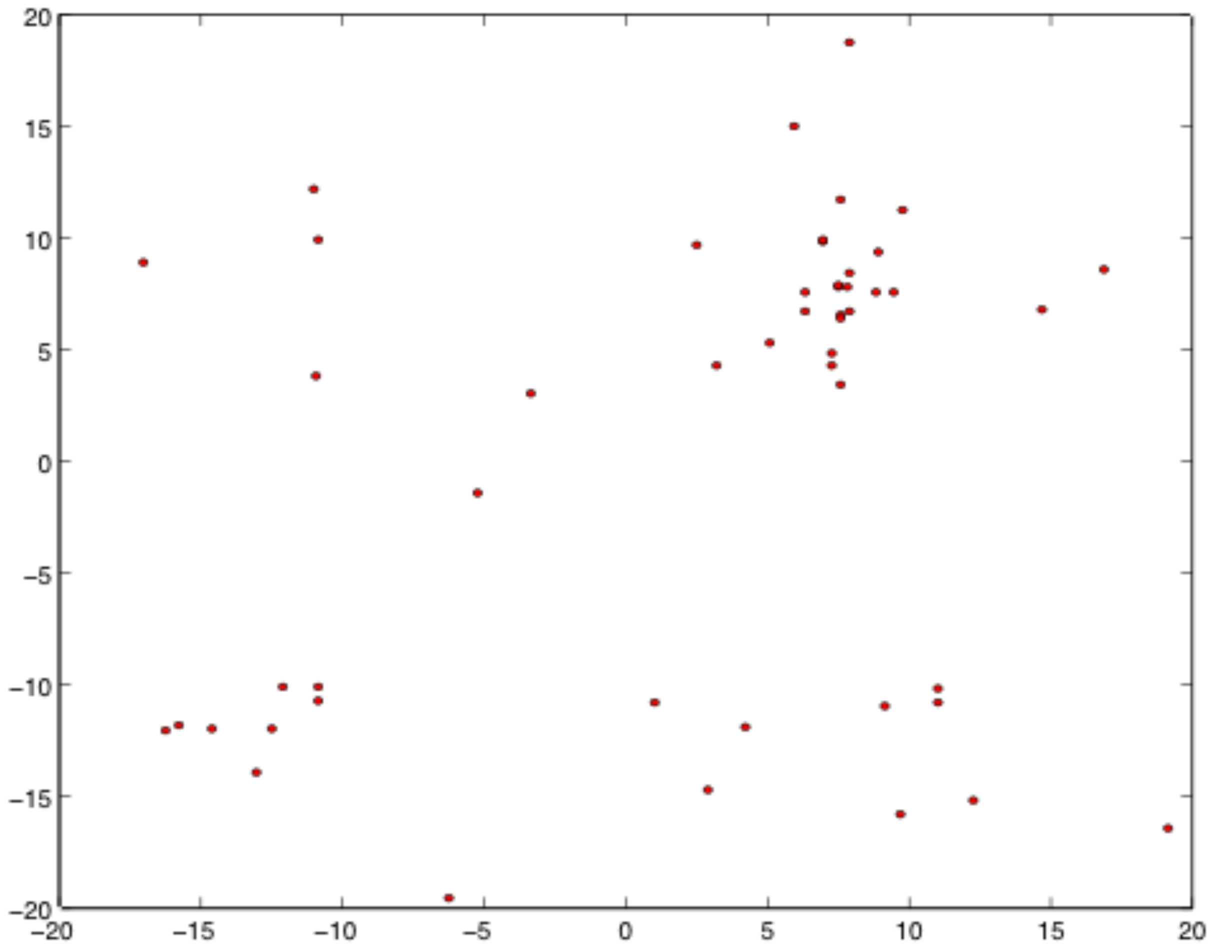
Simple Example

- Individuals have 2 traits: $<t_1, t_2>$
- Each trait can vary from -25 to 20
- Unknown fitness surface defined over this trait space.
- Goal: find highly fit individuals quickly

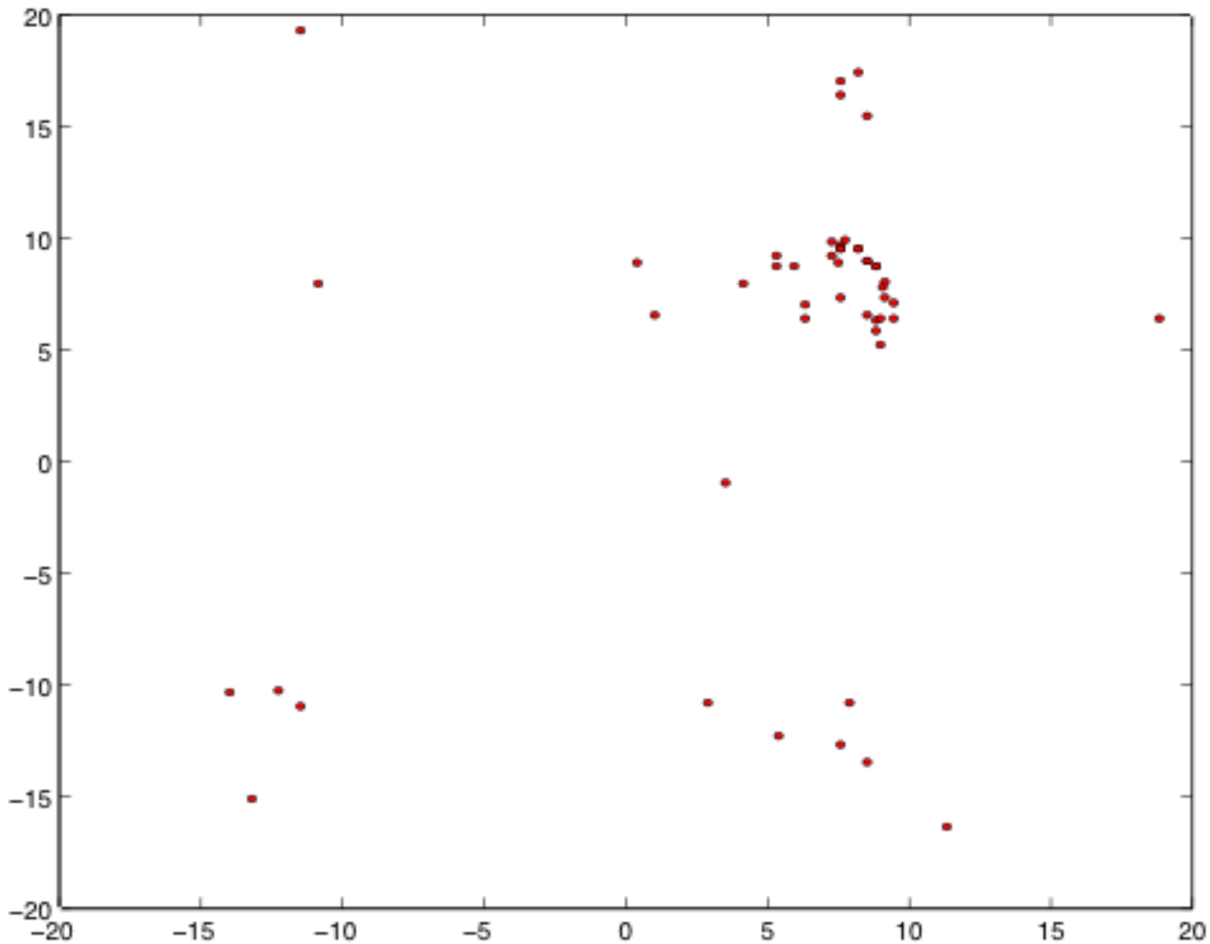
Generation 0 (random)



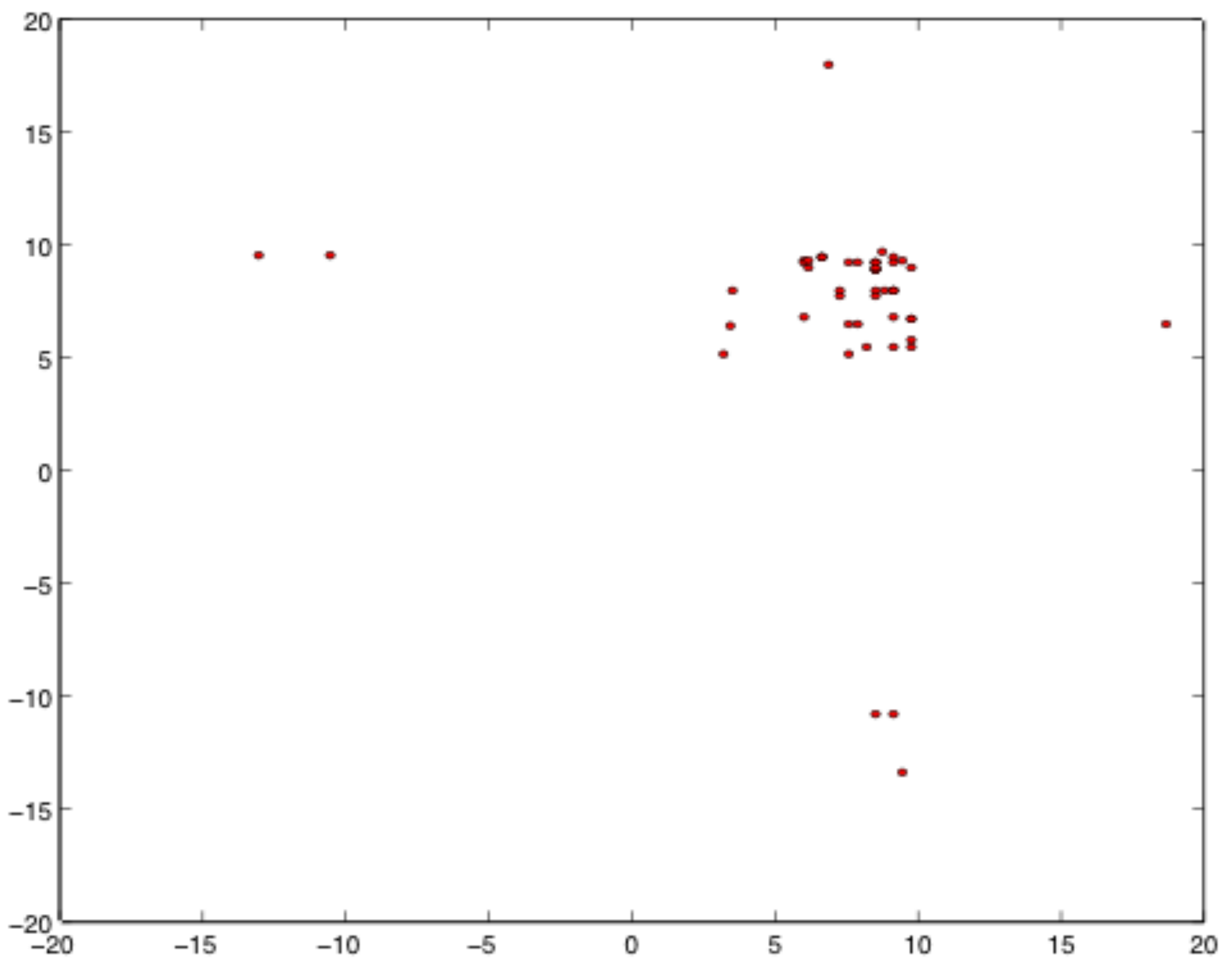
Generation 5



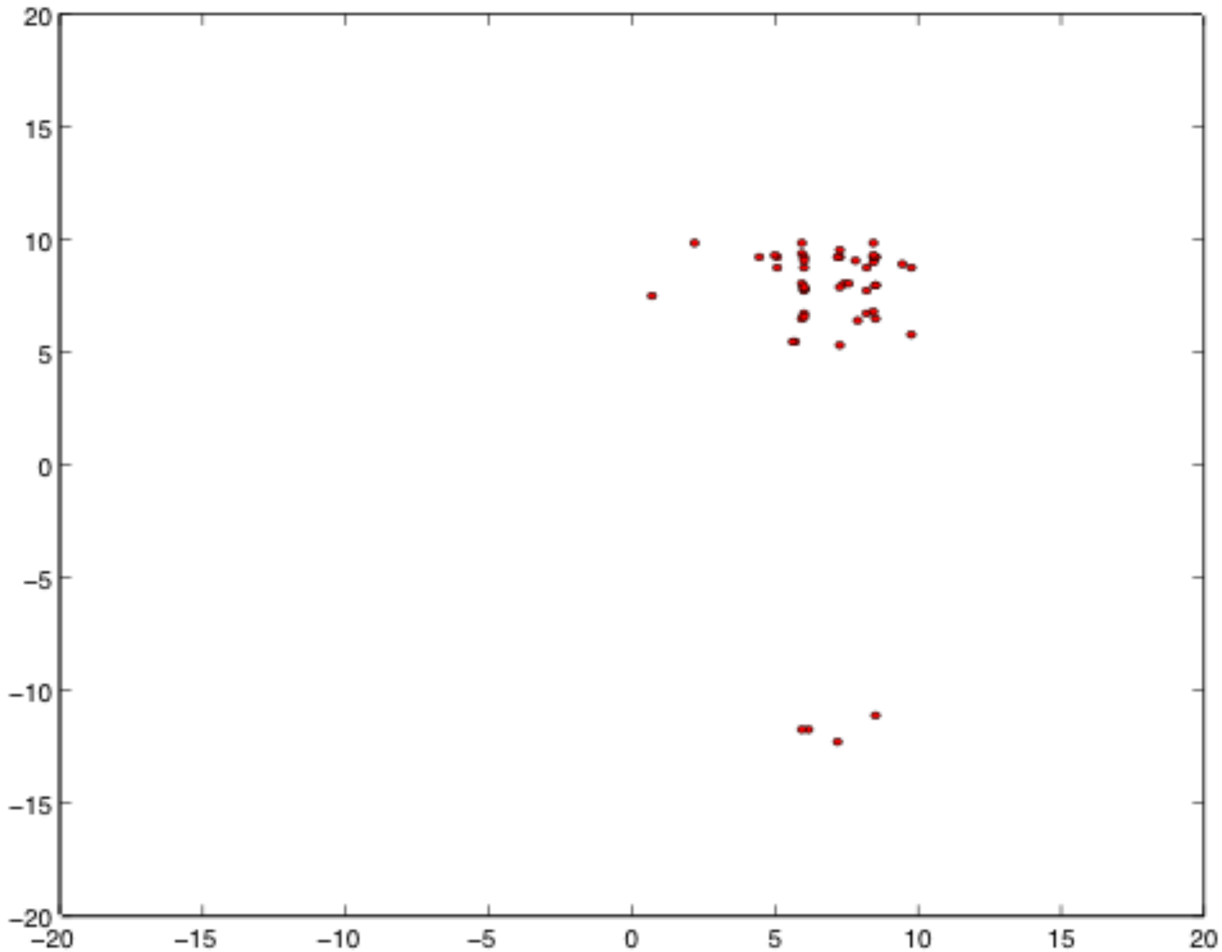
Generation 10



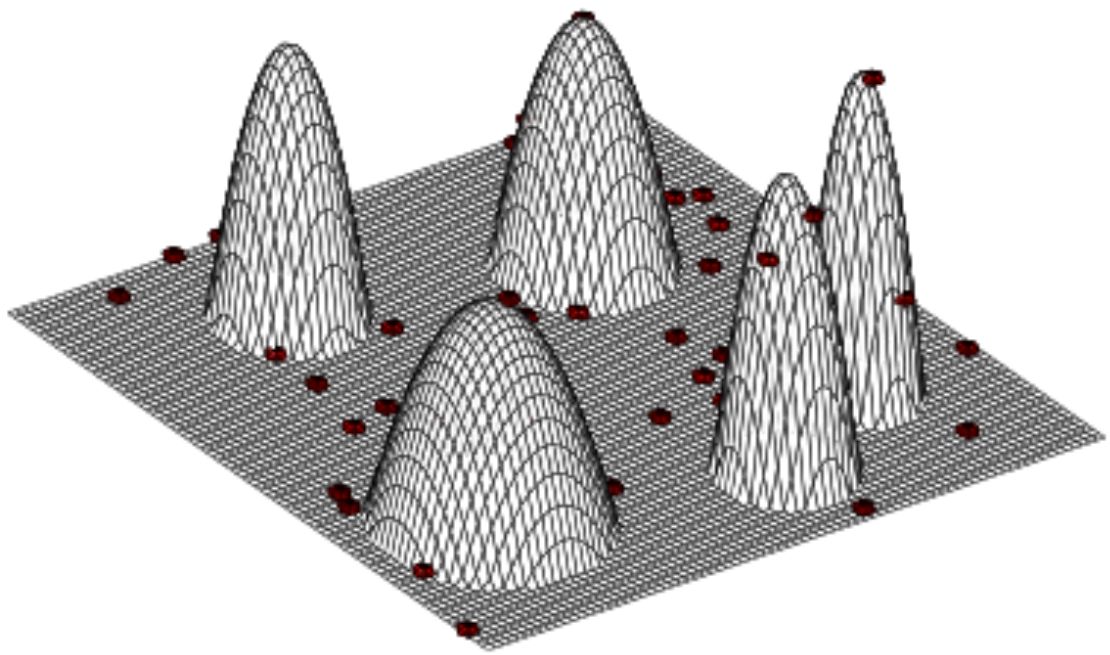
Generation 20



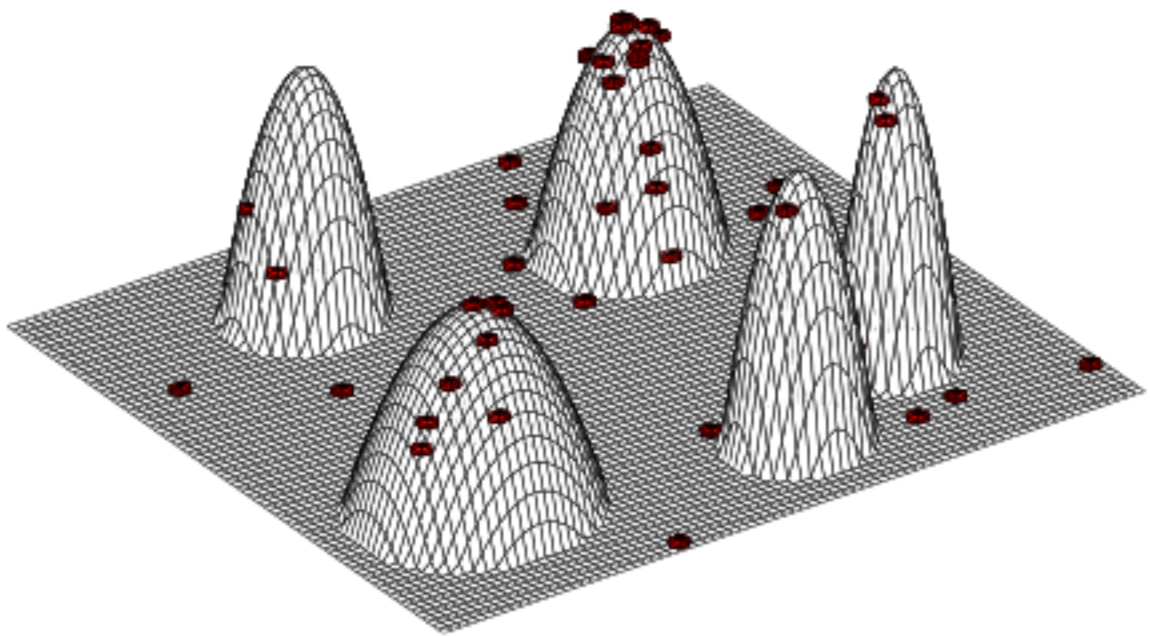
Generation 30



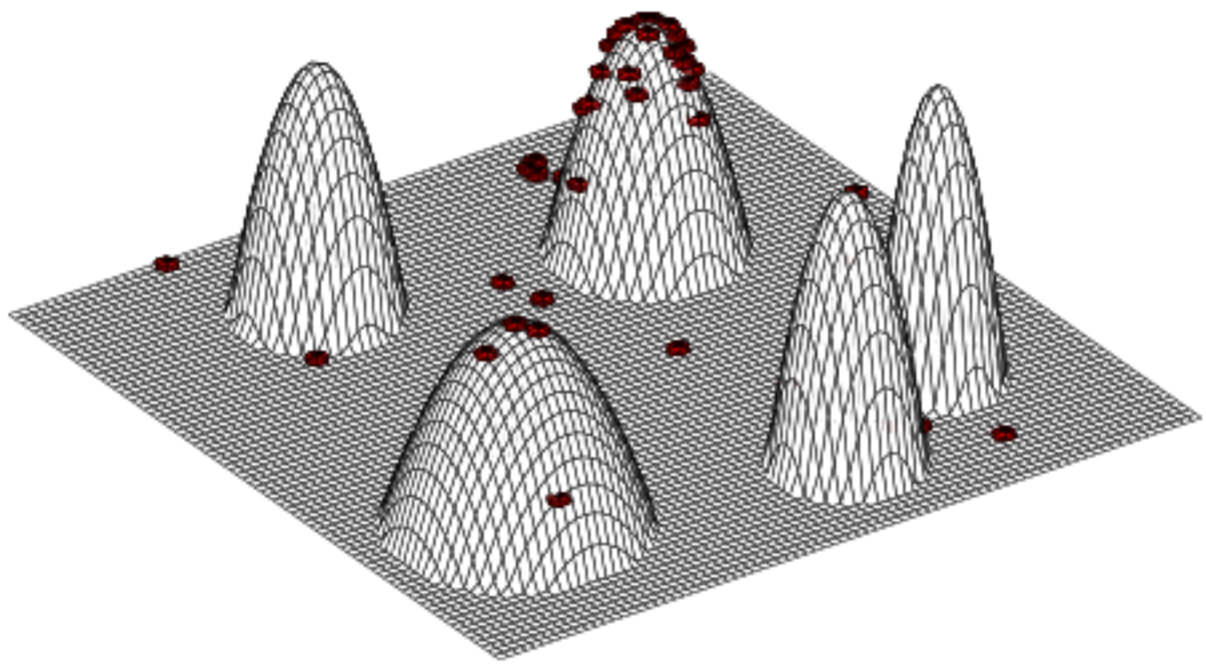
Generation 0



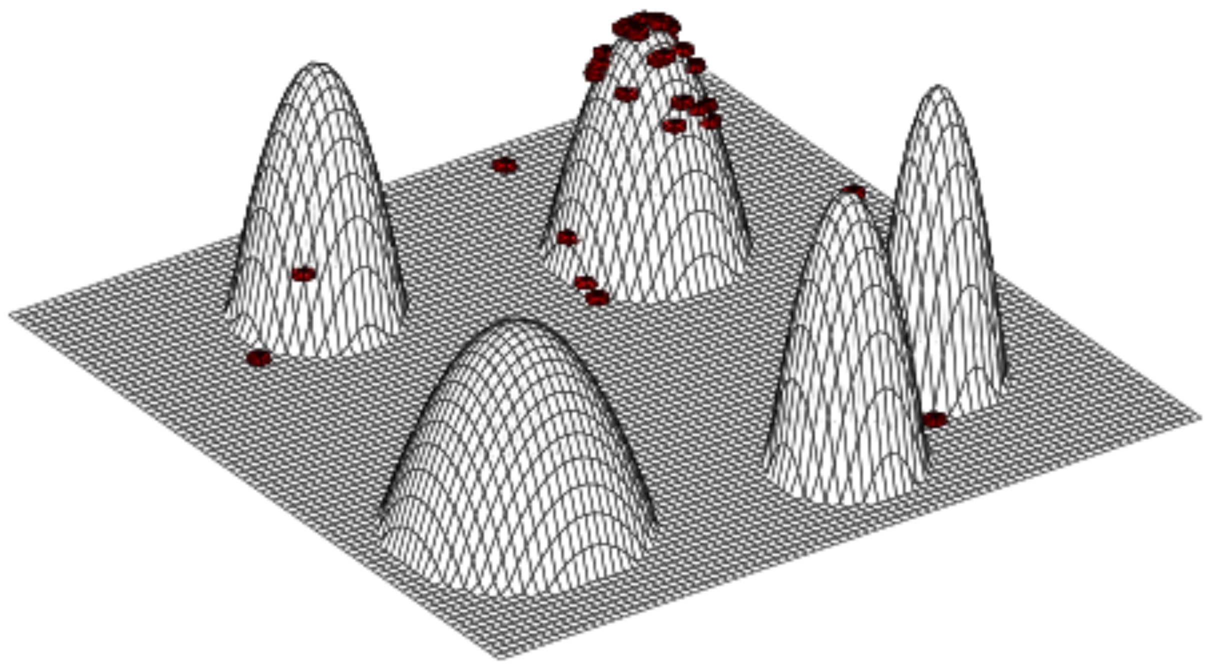
Generation 5



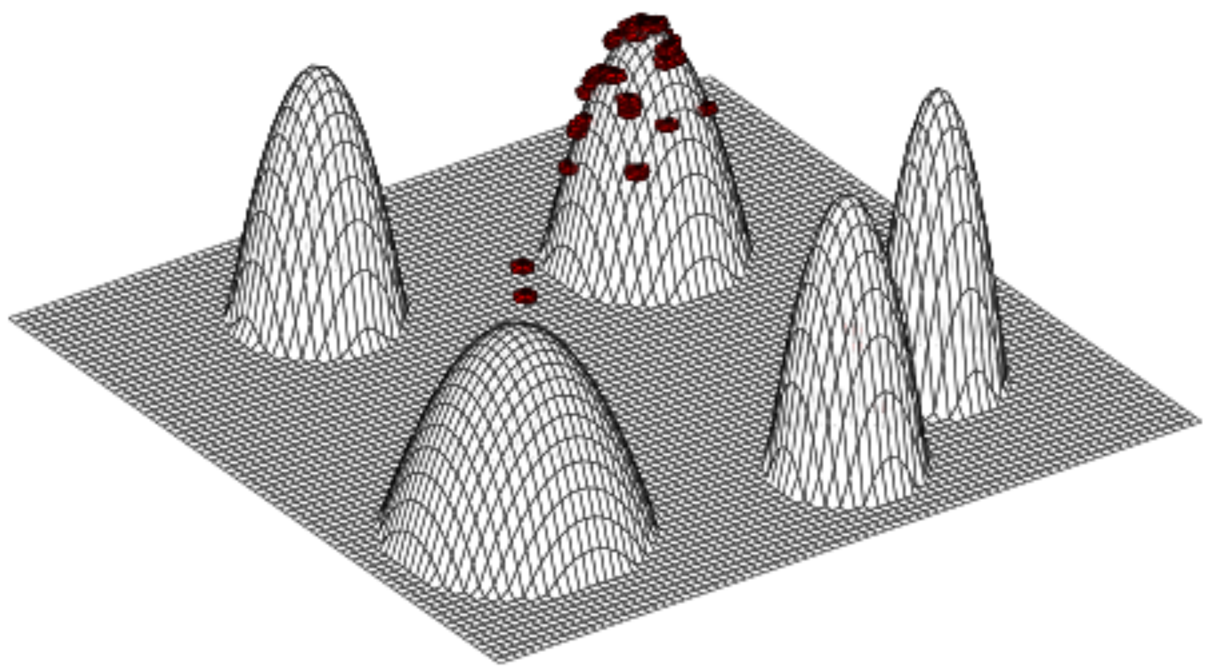
Generation 10



Generation 20



Generation 30



Important Properties:

- Parallel, adaptive search
- Competition for resources:
  - survival (space)
  - reproduce (cpu)
- Stochastic
- Strong sense of fitness “optimization”

Unified Perspective

- EA as a meta-heuristic to be instantiated:
  - Core: a parallel adaptive search heuristic.
  - Instantiation decisions to match problem characteristics.
    - Adopt favorable biases

Key Instantiation Decisions:

- How to represent the space to be searched.
- How to generate "interesting" new samples from existing ones.
- How to choose population sizes.
- How elitist (greedy) should selection be.
- Answers: my tutorial!

Application Areas

- Historically dominant area:

  Parameter-oriented optimization

- Other interesting areas:
  - Searching structure spaces
  - Machine learning
  - Adaptation

Evolutionary Optimization:

- fitness: function to be optimized
- individuals: points in the space
- reproduction: generating new sample points from existing ones.

Useful Optimization Properties:

- applicable to continuous, discrete, mixed optimization problems.
- no *a priori* assumptions about convexity, continuity, differentiability, etc.
- relatively insensitive to noise
- easy to parallelize

Example Areas

- Parameter optimization
  – Mixed parameter data types
  – High dimensional, highly multimodal
- Discrete optimization
  – TSP, SAT, JSS, ...
- Multi-objective optimization
  – Pareto optimality

Beyond Parameter Optimization

- Search spaces that are not easily parameterized:
  - Data structures
  - Program spaces
  - Conceptual design spaces
  - …

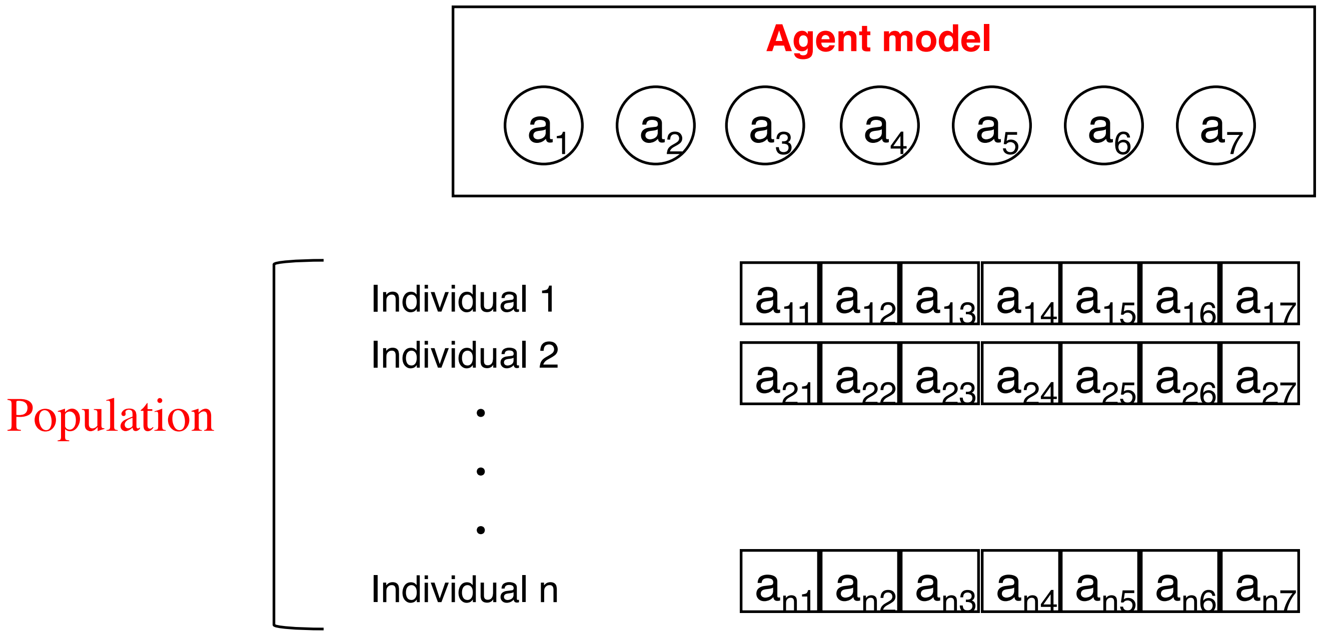
Example:

Evolving agent behaviors

- What internal elements are evolving?
    - Parameters
    - Structures
    - Code
- What are the mechanisms of change?
    - Mutation
    - Recombination
    - ?

1. Evolving agent parameters:

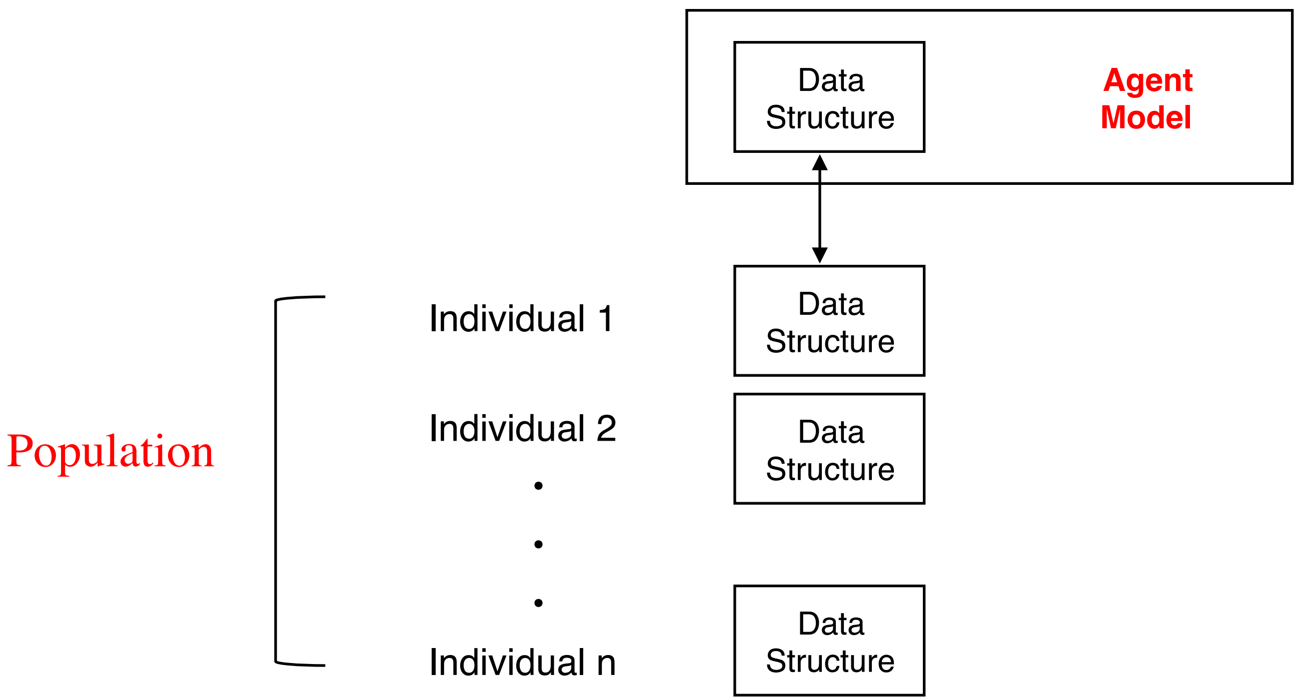
- Identify key parameters that control agent's behavior



Genome: parameter set
Mutation: alter individual parameters
Recombination: combine parameters from different parents

2. Evolving agent data structures:

- Modify internal structures that constrain behavior



- Genome represents a graph, tree, ...
- Requires specialized recombination/mutation operators

3. Evolving agent programs:

- Modify behavioral procedures

Population

Individual 1	IF a and b and c THEN A IF d and e and c THEN B . . .
Individual 2	IF q and r THEN C IF s and e and c THEN B . . .
. . .	
Individual n	IF a and b and c THEN A IF e and c THEN B . . .

- Genome: entire program
- Recombination: combine rules from different parents
- Mutation: alter individual rules

Evolving Executable Objects

- Central issue:

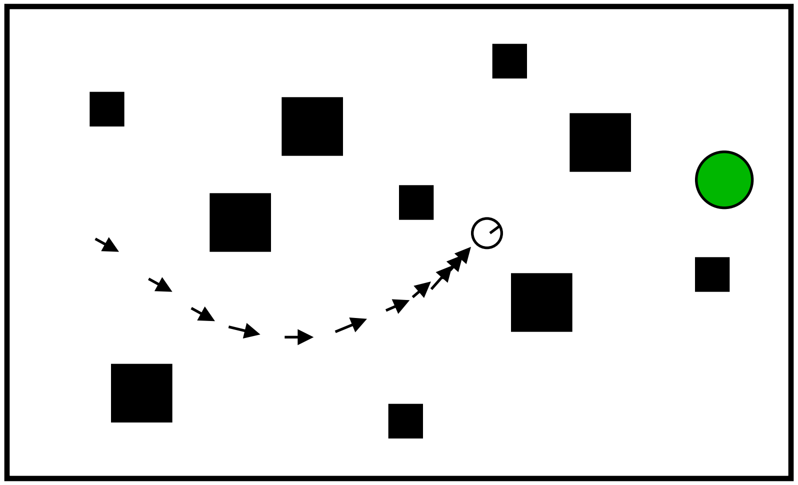
  Evolution-friendly programming languages?

- Syntax, semantics amenable to evolutionary change?
  - Modularity, composability, …
- Candidates:
  – Lisp, Rules, ANNs, CAs, FSMs, …

Example: Evolutionary Robotics

- Collaboration with Naval Research Lab
- Goal: evolve autonomous agent behaviors
- Approach:
  - represent behaviors as rule-based programs
- Behaviors evolved off-line via simulation
- High-fitness behaviors downloaded to robots

Collision Avoidance and Navigation

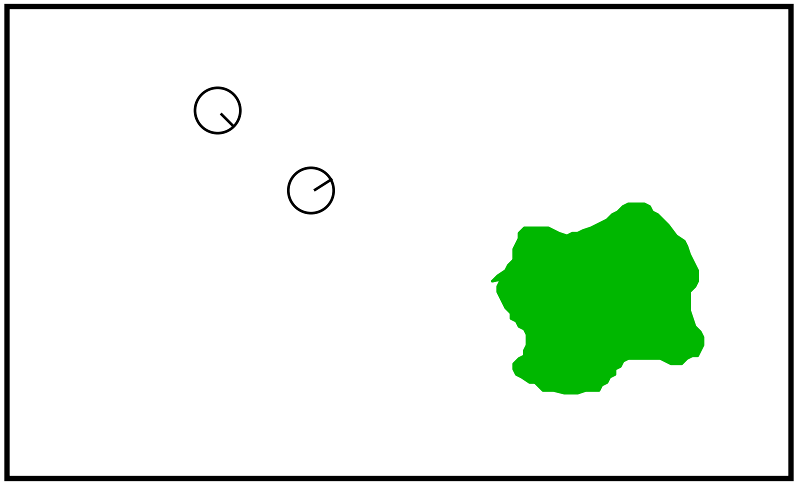


- Goal:
  - get a single agent to reliably perform complex navigation tasks.
- Approach:
  - Evolve behaviors offline via simulation
  - Download & test on real robot

Navigation and
Collision Avoidance

Evolved Behaviors

Evolving Herding Behavior:



- Goal:
  - Evolve sheep dog herding skills
- Approach:
  - Evolve behaviors offline via simulation
  - Download & test on real robot

Herding

Evolved Behavior

Adapting to Partial System Failures

Monitor detects change in internal systems.

Robot performs simulation in virtual world to learn new behaviors.

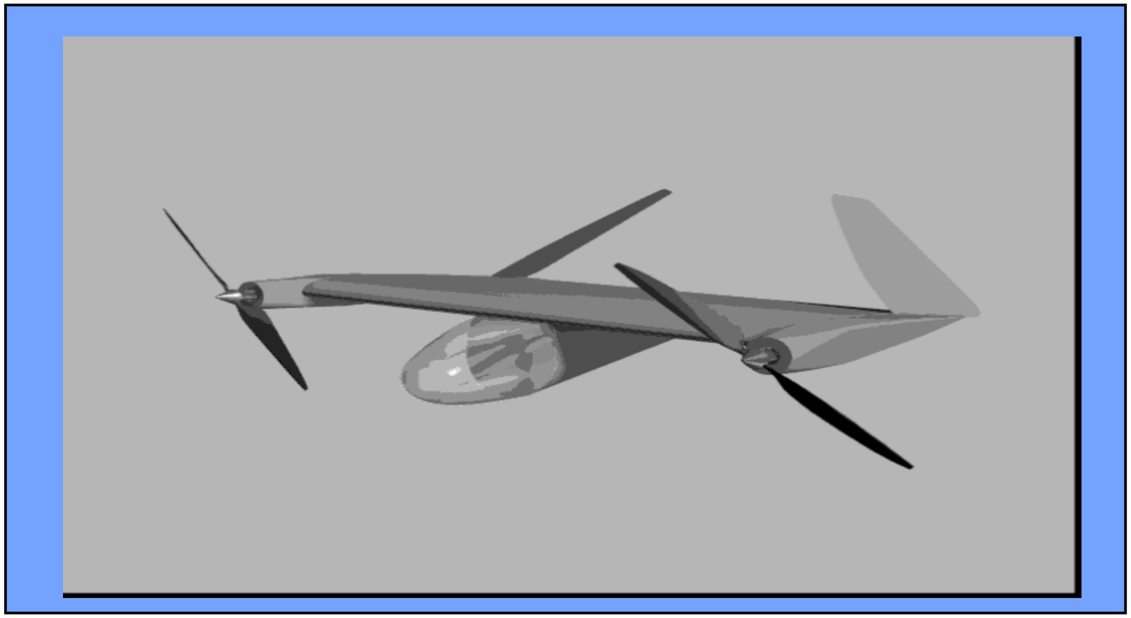
Successful behaviors performed on-line.

[redacted]

Evolving Multiple Cooperating Agents

- Example task domains:
  - micro airplane surveillance tasks
  - robo soccer

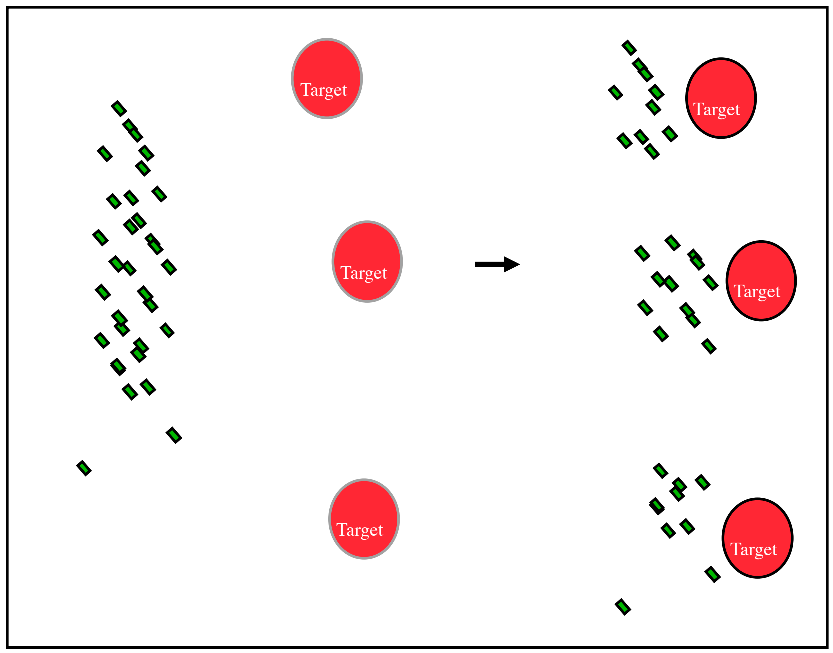
Example: Micro Air Vehicles



MAV specs:

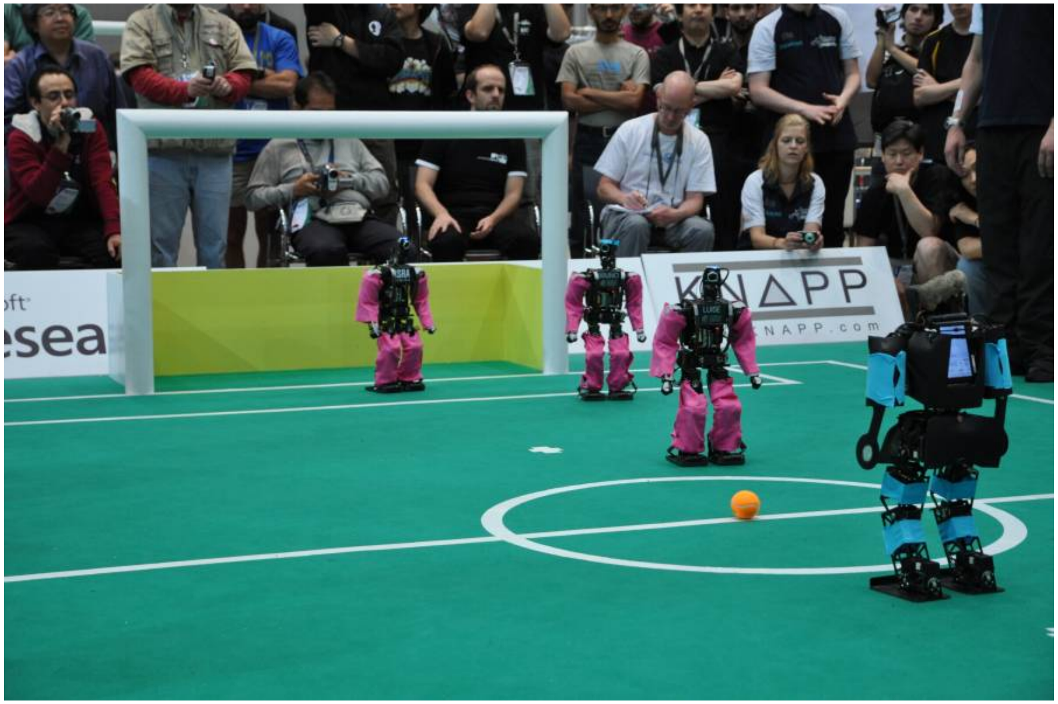
- 6-12 inch wingspan
- 50 - 100 grams gross mass

Goal: evolve collective behaviors for teams of MAVs



Multi-Agent Evolutionary Design

- RoboCup soccer:
  - www.cs.gmu.edu/~robotics



Skeptical?

Concerned?

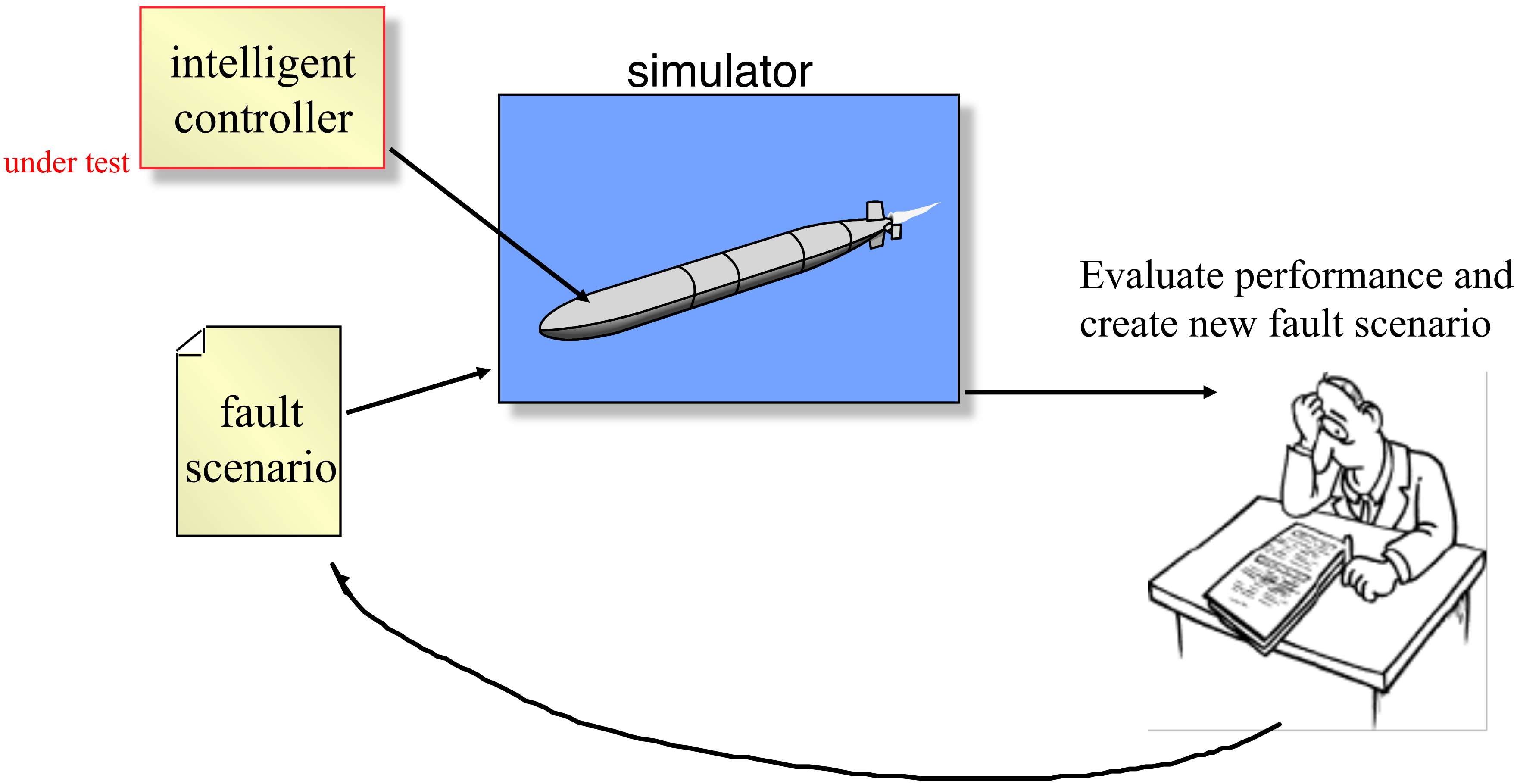
Embarrassing moments at gene parties

Adaptive Testing EAs:

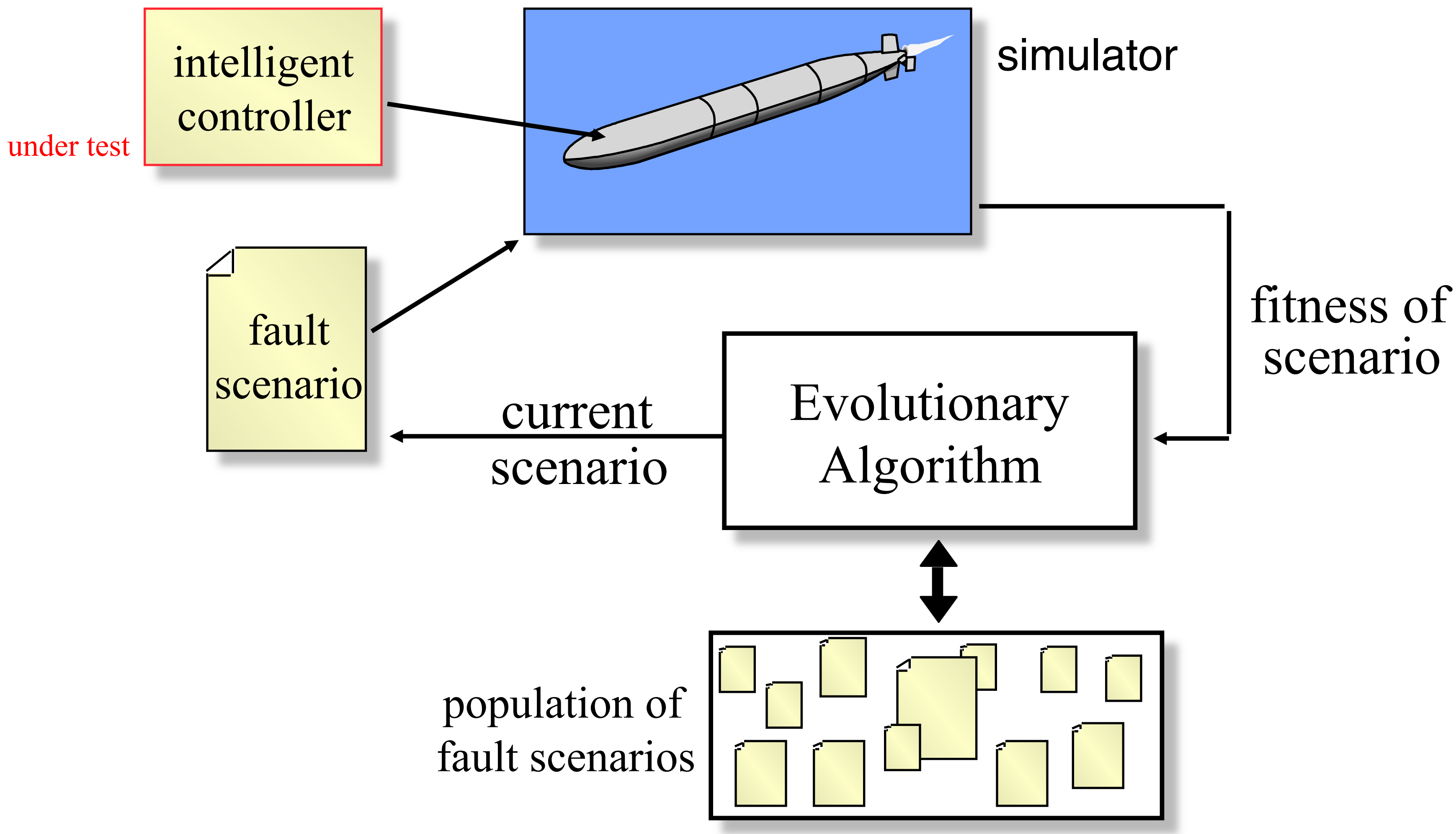
- How to validate autonomous systems?
  - Prove theorems?
  - Hire test engineers?

- Interesting alternative:
  - Use EAs to evolve new scenarios.
  - Scenario's fitness related to the difficulties it creates for autonomous agents.

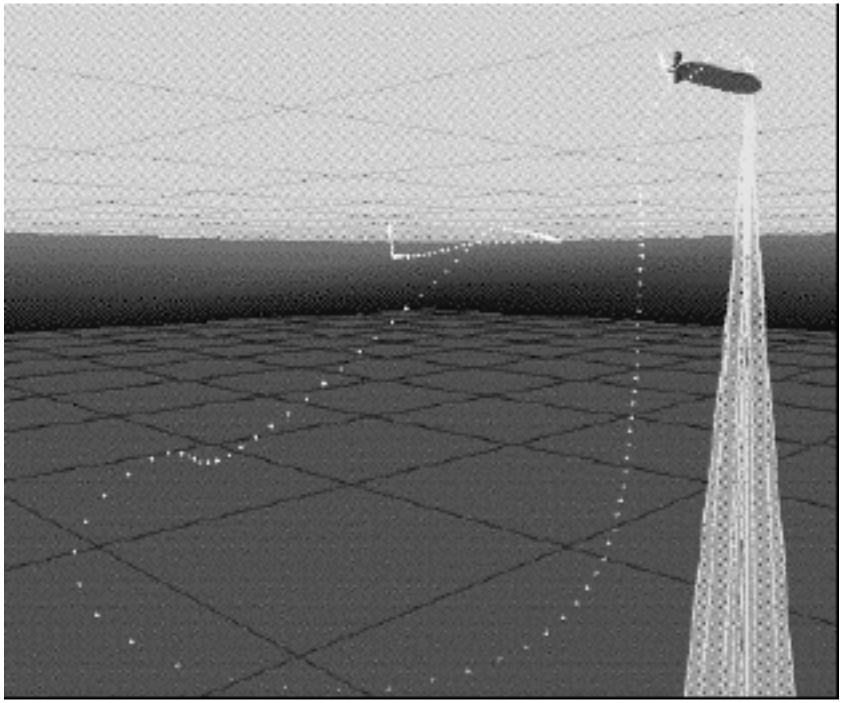
The Traditional Test Cycle



Adaptive Testing



Evolutionary Testing of Draper Labs AUV



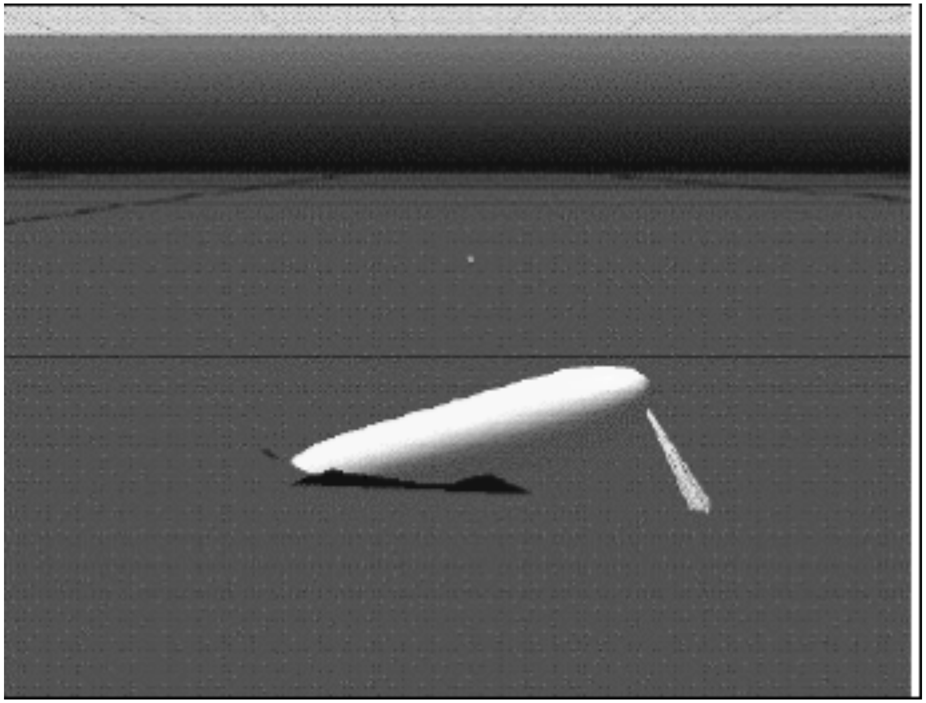
- Very high fidelity simulation of autonomous underwater vehicle.
- Modeled real vehicle (Draper/Darpa).

Evolutionary Testing of AUV:

- Controller failures found included:
  - Vehicle exceeds maximum roll rate, oscillates, etc.
  - Vehicle crashes into bottom
  - Vehicle unable to complete mission in time

• Some missions demonstrated failures that had not been observed before.

• Fault scenarios judged very interesting by controller and vehicle design teams.



Broader context:

- How to understand/test/evaluate complex systems?
- Interesting approach: ABM+EC
  - Build an agent-based model of a complex system
  - Use EAs to evaluate, test, etc.
- Developed tools:
  - ECJ: EC toolkit
  - MASON: multi-agent simulation toolkit

Examples:

- Network security (evolve hackers)
- Homeland security (evolve terrorist scenarios)
  - Water supply
  - Power grids
  - …

Terrorist Threats to Existing Water Systems

- Biological or chemical contamination is placed in a water system.
- Large number of system's nodes are affected.
- Goal: to maximize the negative impact measured by the number of nodes affected.

Evolutionary Testing

- ABM: Combo of MASON model and EPANet models of existing water systems.
- ECJ: Evolved terrorist scenarios that maximized simulated damage.

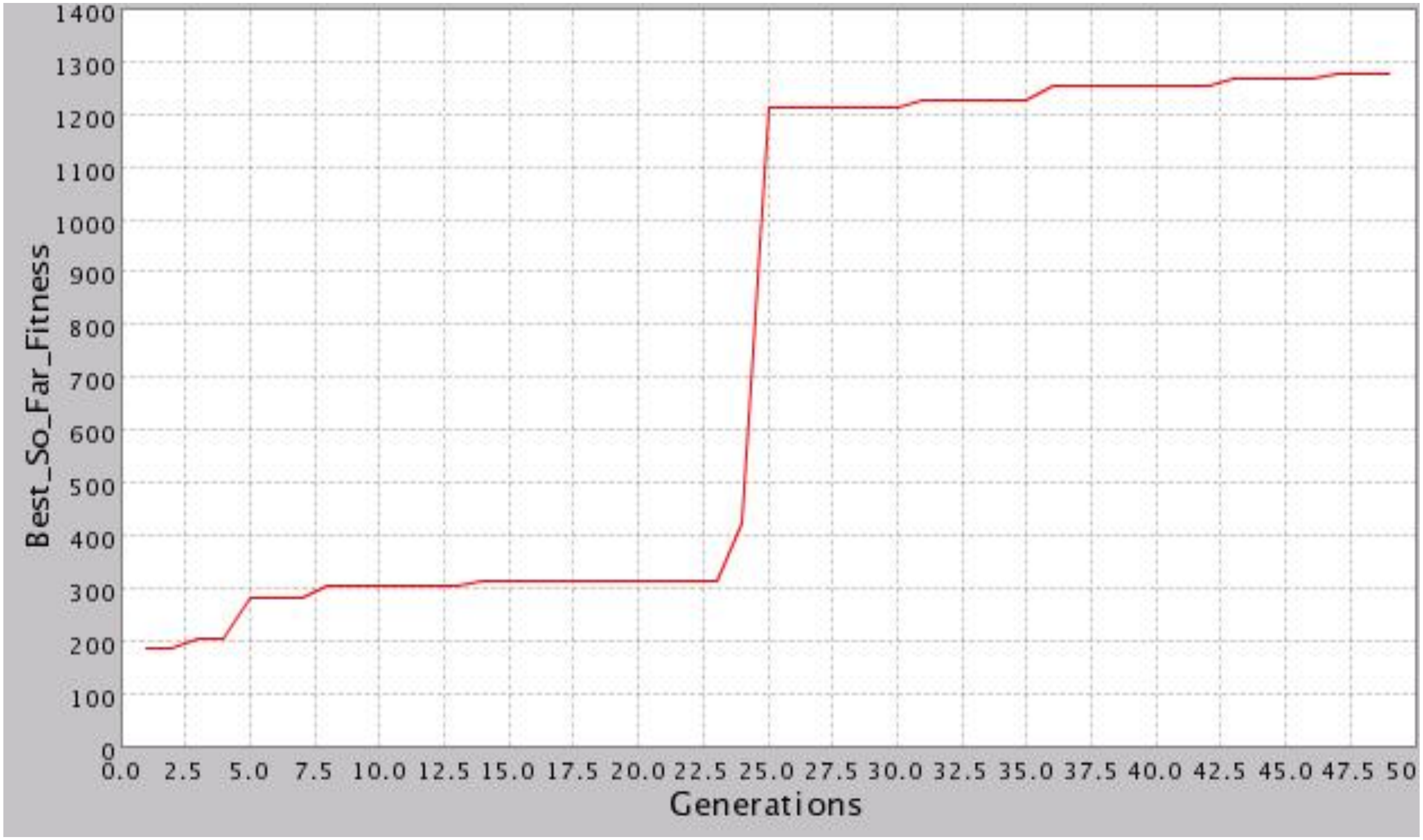
Initial Findings

- Identified several serious weaknesses
- Gave focus for limited budget remedies
- Useful for what-if scenarios

Computer Network Security

- Starting point: a AB network intrusion model
  - 2500 potentially vulnerable systems
  - Modeled Blue force (security) responses
  - Modeled Red force (hacker) tools
- Evolved Red force tactics over time

Evolving Hacker Performance



“In Silico” Science

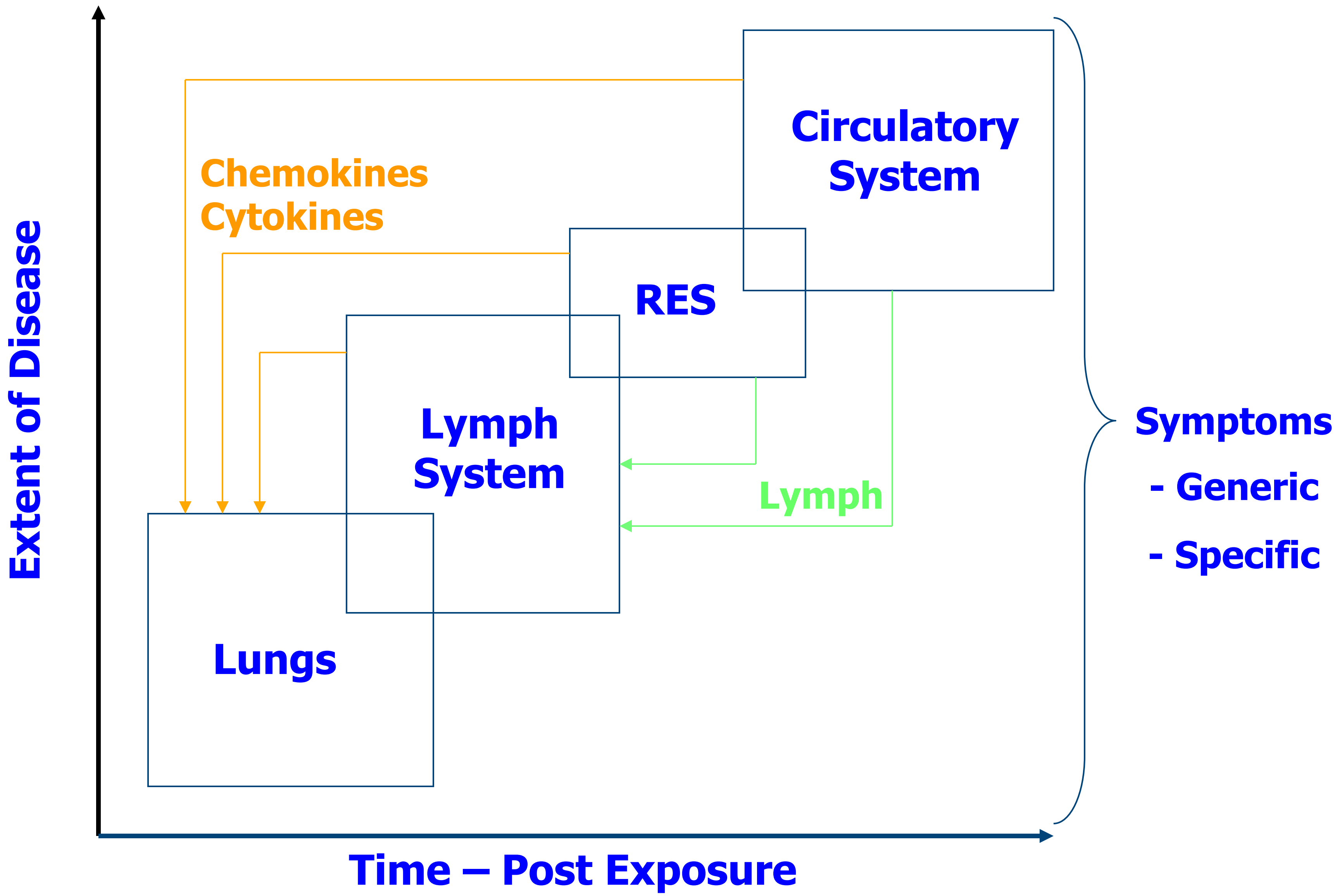
- 3rd way of doing science:
  – *In vitro, in vivo*, …
- Serious alternative to:
  – Experiments with animals
  – Experiments with humans

Example:

Understanding Inhalation Anthrax

- Very real concerns: 1990 exposures
- Clear lack of understanding
- Clear opportunity for *in silico* approaches

Developed an AB Anthrax Model



Applying EAs

- Explore space of treatment scenarios
  - Suggest new interventions
- Hypothesis generation:
  - Suggest new lab experiments

Summary: Unified framework

- Meta-heuristic perspective:
  - Consider properties of the objects to be evolved
  - Select internal representation
  - Choose reproductive operators
  - Choose population sizes
  - Choose selection pressure
- Result: a well-designed EA

However ...

- New applications pressing state of the art.
- Unified view of "simple EAs" is not sufficient.
- Principled extensions are required.

Additional inspiration from nature?

- Relationship of simple EAs to Biology
  - Fairly remote!
- E.g.,
  - No notion age, growth, development
  - No distinction between genotype and phenotype
  - No notion of gender, multiple species
  - …

Relationship of simple EAs to Biology

- Clear tension between:

  Biological fidelity and computational efficiency

- New directions:
  - Exploring other biological aspects that improve computational efficiency.

EC direction: Adaptive EAs

Goal: reduce tuning efforts

- EAs have their own parameters
- How to tune them?
  - Manually?
  - Meta-EAs?
- Far better to have:
  - Self-adapting mechanisms

EC direction: Adaptive EAs

Adaptive strategies:

- Adapt across multiple runs/restarts
- Adapt during a single run:
  - Via feedback control mechanisms
    - E.g., adaptive Gaussian mutation
  - Via evolutionary mechanisms
    - E.g., parameters part of the genome

EA direction: Exploiting Parallelism

Low-hanging fruit: parallel evaluation

- E.g., master-slave architectures

- Tougher challenges:
  - Coarsely-grained distributed systems
    - E.g., across beowulf clusters
  - Finely-grained multi-threaded systems
    - E.g., exploit gpu clusters

Coarsely-grained parallelism

- EA Island models:
  - Multiple EAs running on separate machines
  - Heterogeneous EAs?
  - Occasional migrations to other machines

- Challenges:
  - Clearly different than single population models
  - When to use them?
  - How to configure them?

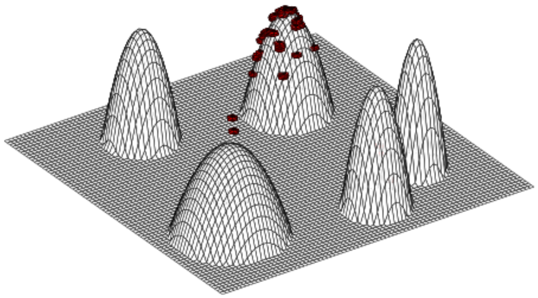
Fine-grained parallelism

- EA cellular models:
  - Introduce a spatial topology in the population
  - Micro EA in each cell
  - Only local interactions with neighbors
  - All micro EAs run in parallel
- Challenges:
  - Clearly different than single population models
  - When to use them?
  - How to configure them?

EC direction: Multi-objective optim.

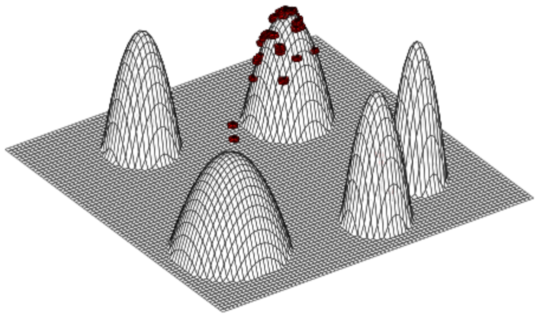
- Impressive initial results
- Need to scale up to >3-4 objectives
- Role of more complex EAs
- Need deeper theoretical understanding

EC direction: Non-stationarity



- Time-varying environments:
  - fitness landscape changes during evolution
  - goal: adaptation, tracking
  - standard optimization-oriented EAs not well-suited for this.

EC direction: Multiple Species



- Speciation: Non-random mating
  - Helps maintain diversity
  - Simultaneously explore multiple peaks
  - Issues:
    - How, when, how many?
    - Emergent?

EC direction: Co-evolution

- Multiple populations/species
- Fitness is a function of other populations/ species
- Examples:
  - Competitive co-evolution
  - Cooperative co-evolution

Competitive co-evolution

- Improve performance via "arms race"
  - E.g., Hillis' sorting networks
  - E.g., competitive games
- Challenges:
  - When to use?
  - How to configure?

Cooperative co-evolution

- Improve performance via "teamwork"
  - Decompose problem into subcomponents
  - Evolve subcomponents in parallel
  - Fitness is a function of other components
- Challenges:
  - When to use?
  - How to configure?

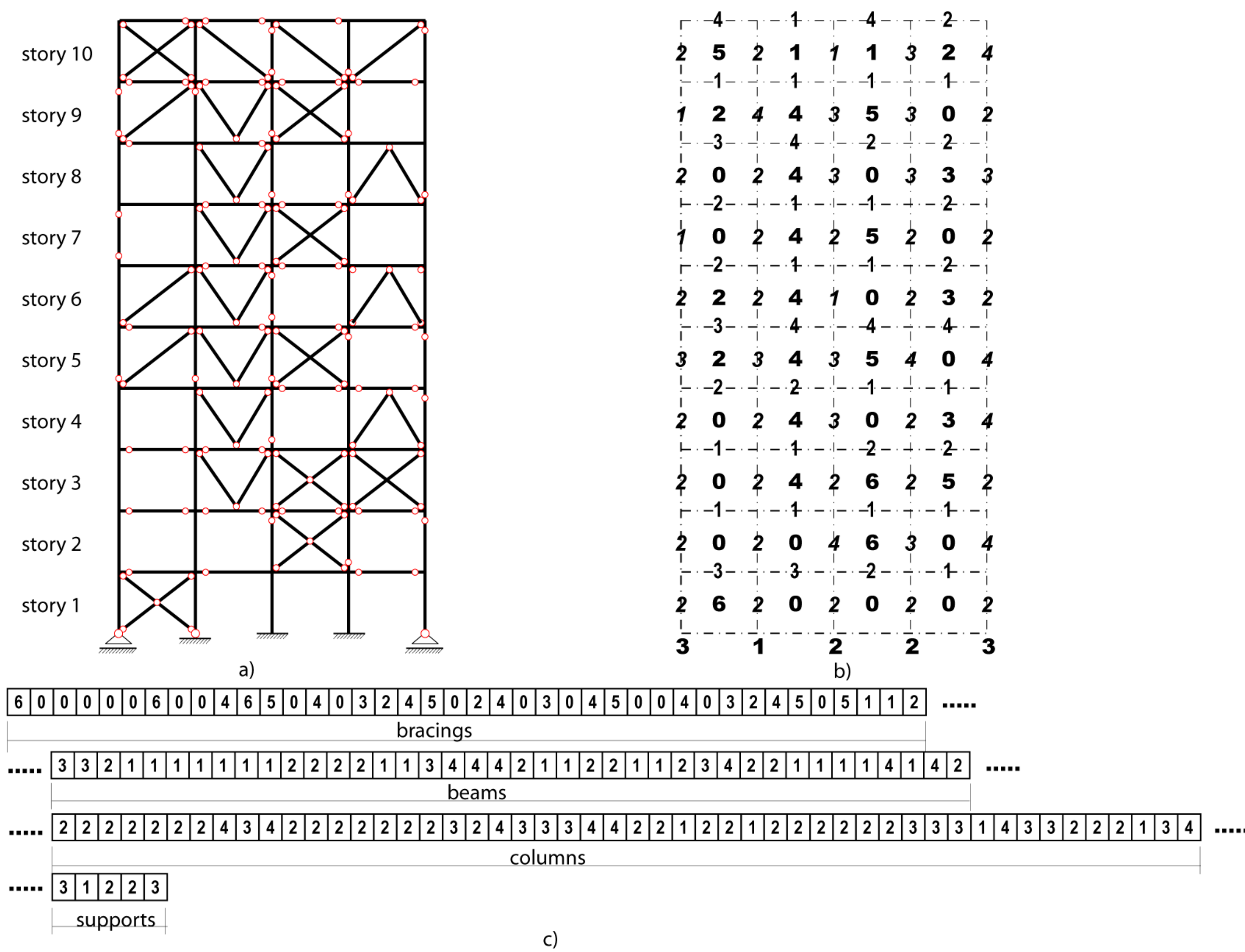
EC direction: Morphogenesis

- Inspiration from biology:
  - Strong distinction between:
    - Genotype (plans)
    - Phenotype (objects)
- Evolve plans, not objects
- Morphogenesis: Generate objects from plans

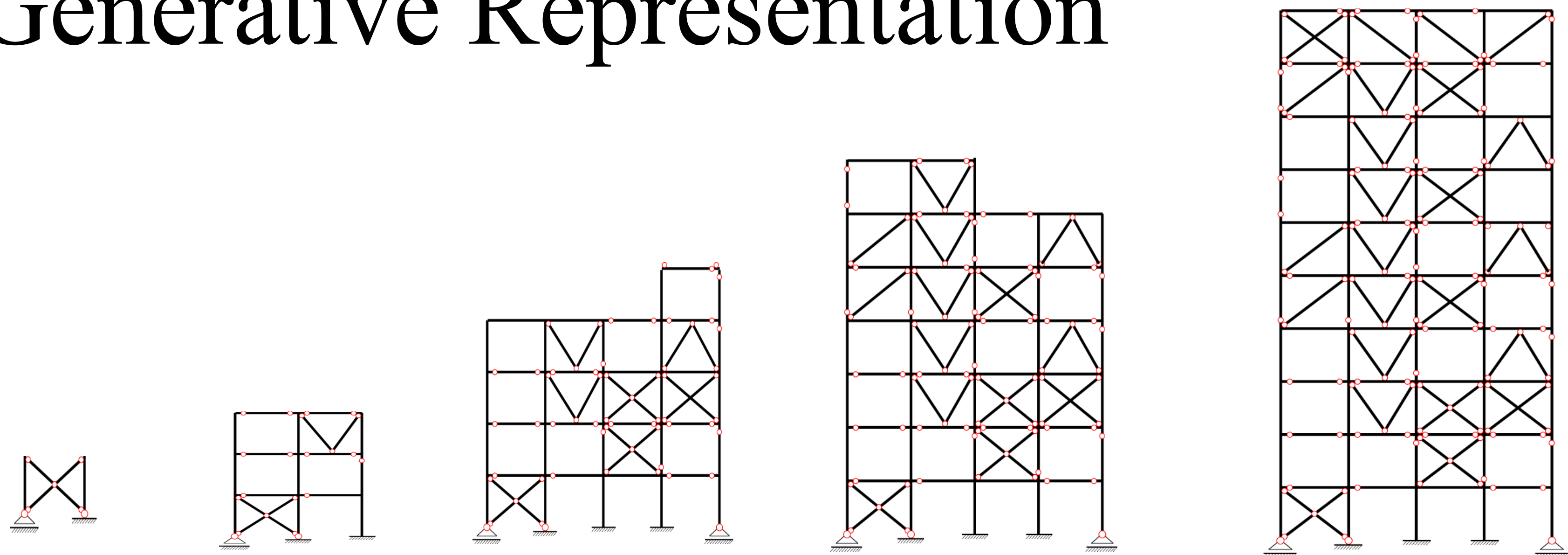
Example: Evolutionary Design

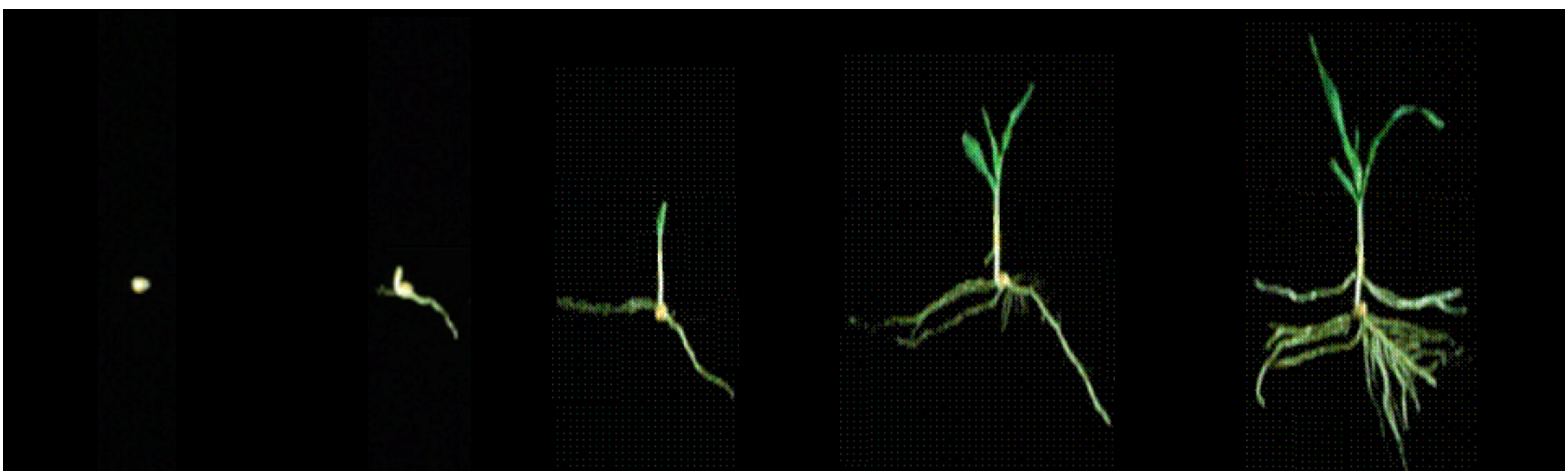
- Standard EAs are good for parameter optimization of designs.
- Different EAs are needed for conceptual design.

Phenotypic Representation



Generative Representation





Generative Representations

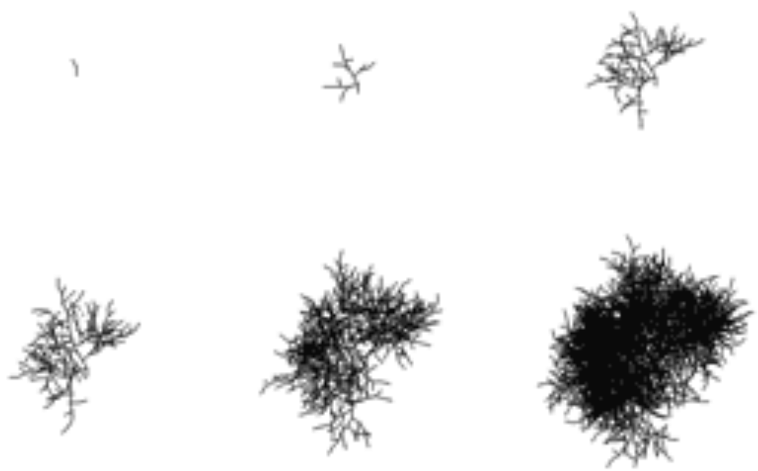
- “Inspiration from nature” strategy has yielded a variety of generative models:
  - L-systems
  - Cellular automata
  - Genetic regulatory networks
  - …

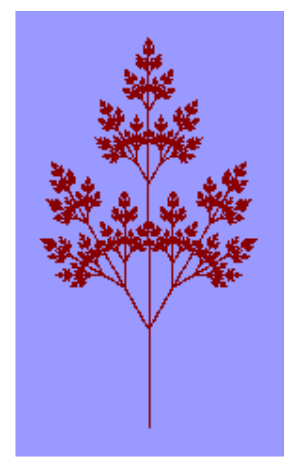
L-systems

- Short for Lindenmayer systems
  - Developed ~ 1968 by A. Lindenmayer
  - Goal: model plant morphology
  - Approach: formal “rewrite” grammars:

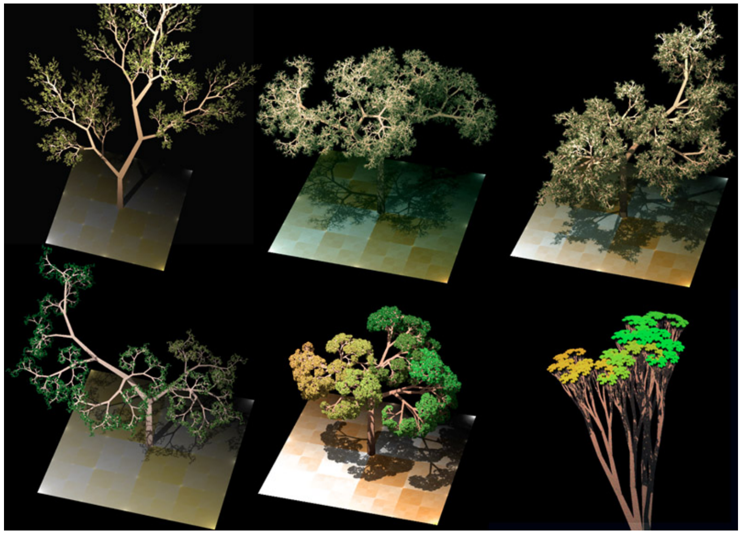
L-systems

- Examples:





L-systems



- Evolutionary design:
  - Evolve set of rewrite rules to achieve a goal.
- Examples:
  - Artificial vegetation (Ochoa, Jacob, Moch, …)
  - Tables (Popovici)
  - Robot morphology (Hornby, Pollack, Bentley, …)
  - Neural networks (Kitano, …)

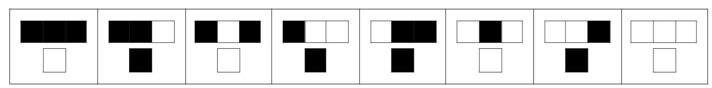
Cellular Automata

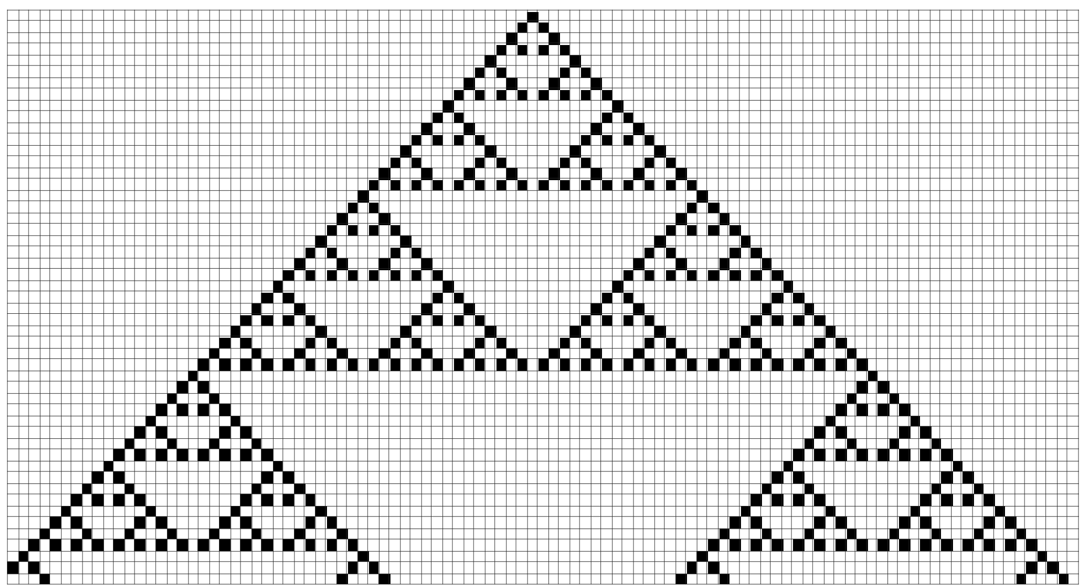
- Similar in spirit to L-systems:
  - Symbols, axioms, rewrite rules
  - Explicit bounded spatial topology
- Classical examples:
  - Conway's Game of Life
  - Wolfram's pattern generators

Cellular Automata

- Elements:
  - A topology of cells
  - Initial state of each cell
  - State transition rewrite rules
    - Based on neighboring states
- Simple rules generate emergent complexity

Cellular Automata





Cellular Automata

- Evolutionary design:

  Evolve transition rules to achieve a goal

- Examples:
  - Artificial societies (Axtell, Ilichinski, …)
  - Programs (Mitchell, Crutchfield, Sipper, …)
  - Chemical structures (Gerhardt, Schuster, …)
  - Building designs (Kicinger, Arciszewski, …)

Genetic Regulatory Networks
(GRNs)

- Similar in spirit to L-systems, CAs:
  - Symbols, axioms, production rules
  - Explicit gene/cell model
    - Genes: independent rewrite rules
    - Cells: collections of genes + shared global memory
      - Similar to AI "blackboard" models

GRNs

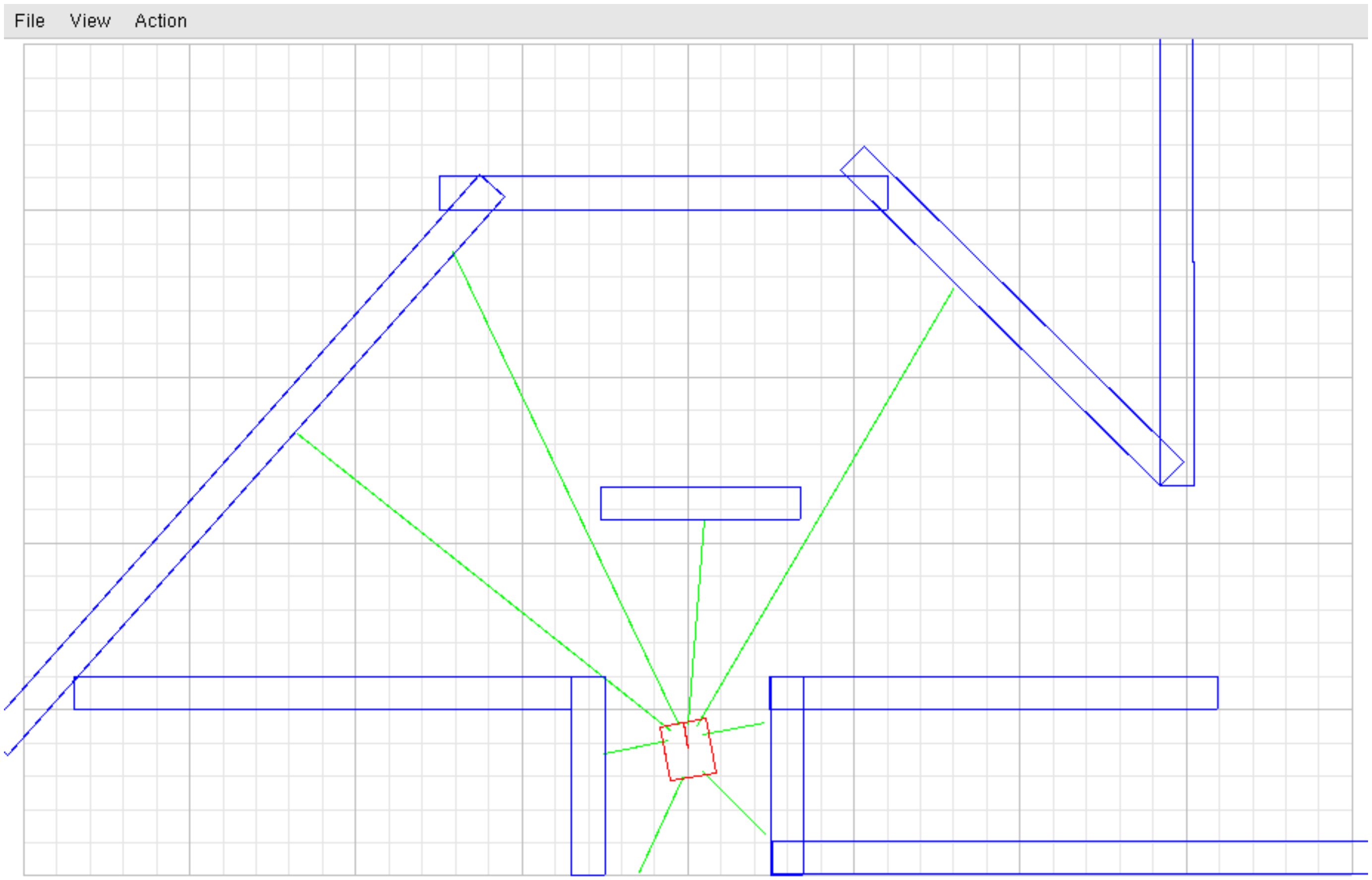
- Evolutionary design:

  Evolve production rules to achieve a goal

- Examples:
  – Robot control (Kumar, Grajdeanu, …)
  – Multi-cellular objects (Miller, Federici, Gordon, …)

GRNs

- Example: obstacle avoidance



GRNs

- Example: artificial embryogeny
  - Seed: single cell
  - Morphogenesis (via embryogeny):
    - Model cell division, growth, maturation
  - Grow complex, multidimensional objects

Generative Representations

- Interesting open EC question:
  - Which generative representations are more evolution-friendly?
    - L-systems
    - Cellular automata
    - Genetic regulatory networks
    - …

EC direction: Agent orientation

- Individuals more autonomous, active
- Fitness is a function of other agents and environment-altering actions
- E.g.,
  - Evolutionary robotics
  - Evolution of cooperation

EC direction: Analysis

- Need stronger analysis tools:
  - Markov models
  - Statistical mechanics
  - Evolutionary game theory
  - Test problem generators
  - Visualization

EC direction: Hybrid systems

- Continuing to explore:
  - Memetic algorithms: EAs and local search
  - COGANNs: EAs and ANNs
  - EAs and symbolic machine learning
  - EAs and agent-based models
  - …

EA Generalizations:

- Meta-heuristics:
  - Heuristics for designing heuristics
    - E.g., hill climbing, greedy, …
  - Adopt no-free lunch view
  - Instantiate templates in a problem-specific manner

EA Generalizations:

- Nature-Inspired Computation:
  - Early example: simulated annealing
  - Today: evolutionary algorithms
  - Others:
    - particle swarm optimization
    - ant colony optimization
    - ...

Conclusions:

- We've come a long way.
- Many new challenges ahead.
- A strategy for continued success:
  - An expanded unified framework that leads to:
    - Principled design
    - Principled extensions

More information:

- Journals:
  - Evolutionary Computation (MIT Press)
  - Trans. on Evolutionary Computation (IEEE)
  - Genetic Programming & Evolvable Hardware
- Conferences:
  - GECCO, CEC, PPSN, FOGA, …
- Internet:
  - www.cs.gmu.edu/~eclab
- Lots of books including:
  - Evolutionary Computation: A Unified Approach
    - Kenneth De Jong, MIT Press, 2006