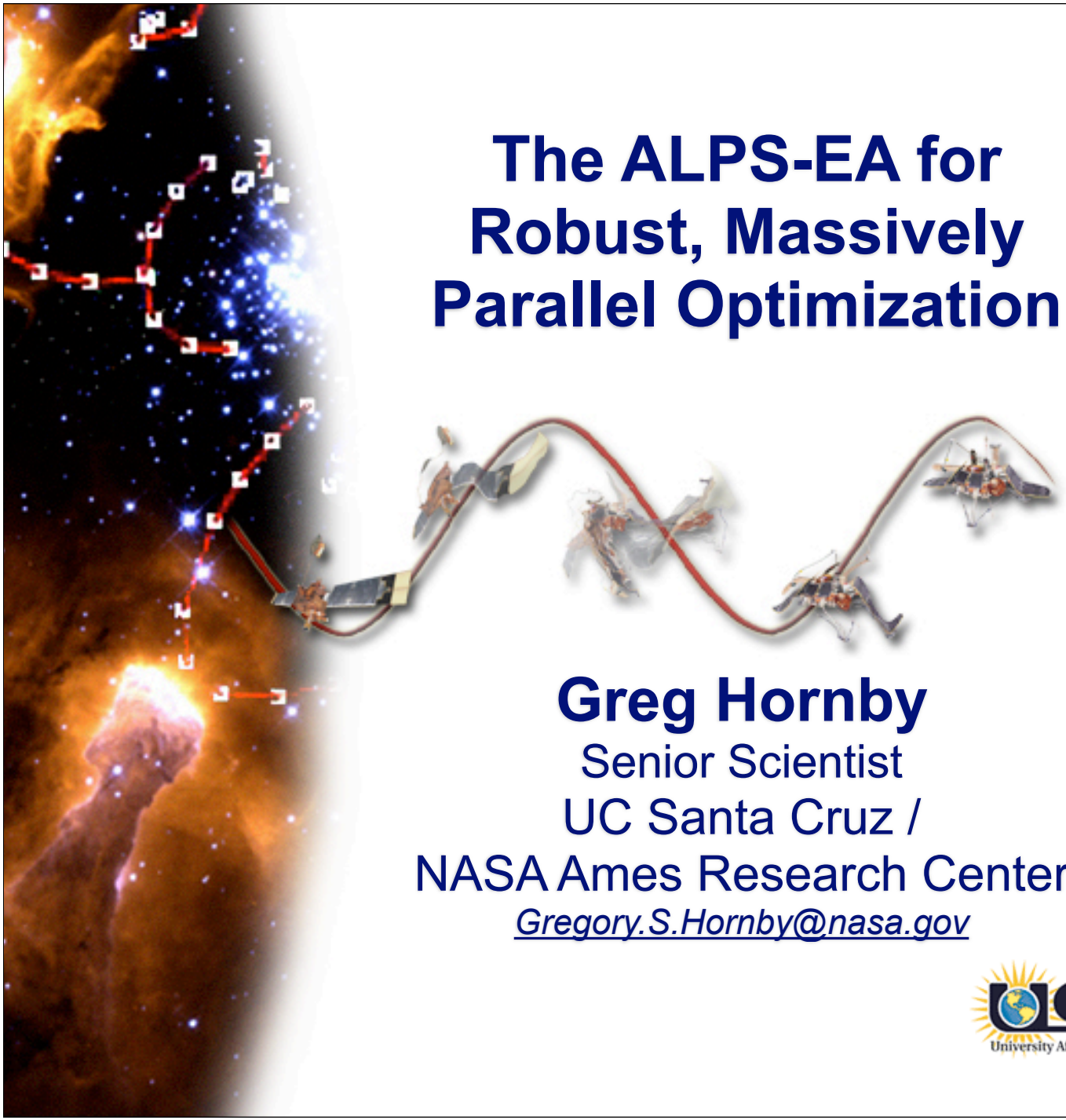


The ALPS-EA for Robust, Massively Parallel Optimization

Greg Hornby
Senior Scientist
UC Santa Cruz /
NASA Ames Research Center
Gregory.S.Hornby@nasa.gov

University Affiliated Research Center
UC SANTA CRUZ
NASA Ames Research Center

Introduction

Modern super-computers are clusters of thousands of processing cores:

- Pleiades has:
  - 12,800 processors.
  - 51,200 cores.

Challenge: How can adaptive algorithms fully utilize such computers?

- Need a parallel algorithm.
- Minimize inter-processor communication.

Part I:
Evolutionary Algorithms

Parallel Optimization

Parallel Computers:

- Lots of processors/cores.
- Have high inter-processor communication costs. (relative to copying in on-core memory).

Problem: many algorithms/applications are either:

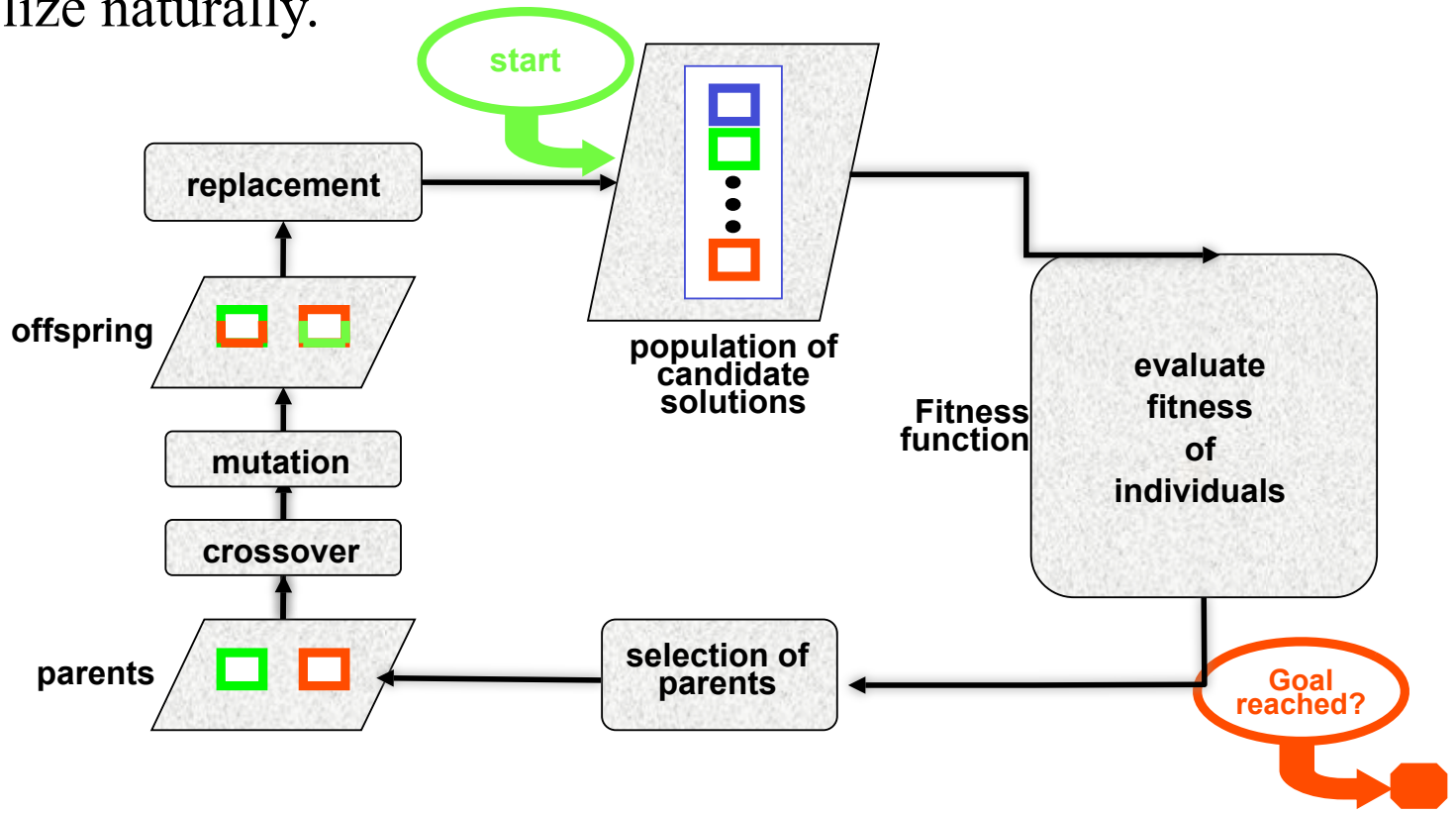
- Single-threaded and difficult to parallelize.
- Have lots of inter-processor communication with little processing.

=> Processors are frequently idle while waiting for data.

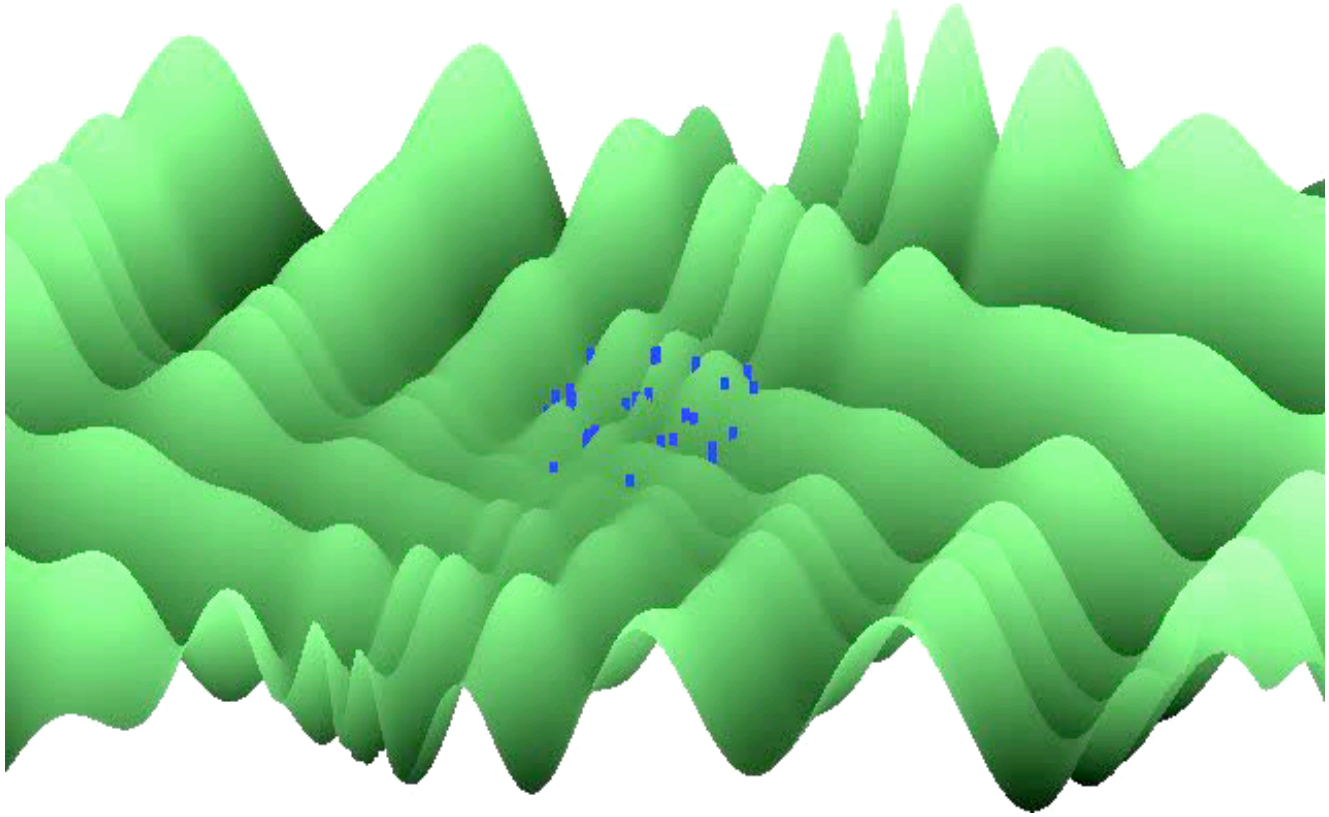
One approach: **Evolutionary Algorithms.**

Evolutionary Algorithms (EAs)

- A family of stochastic, population-based search algorithms.
- Inspired by neo-Darwinism.
- Parallelize naturally.



Example EA



- Maximize f(x, y):
  - f() is the rotated F101 function.
  - Using a standard Evolutionary Algorithm (EA).

Uses of EAs

Evolutionary Algorithms become more and more useful with the continuing increase in computer power.

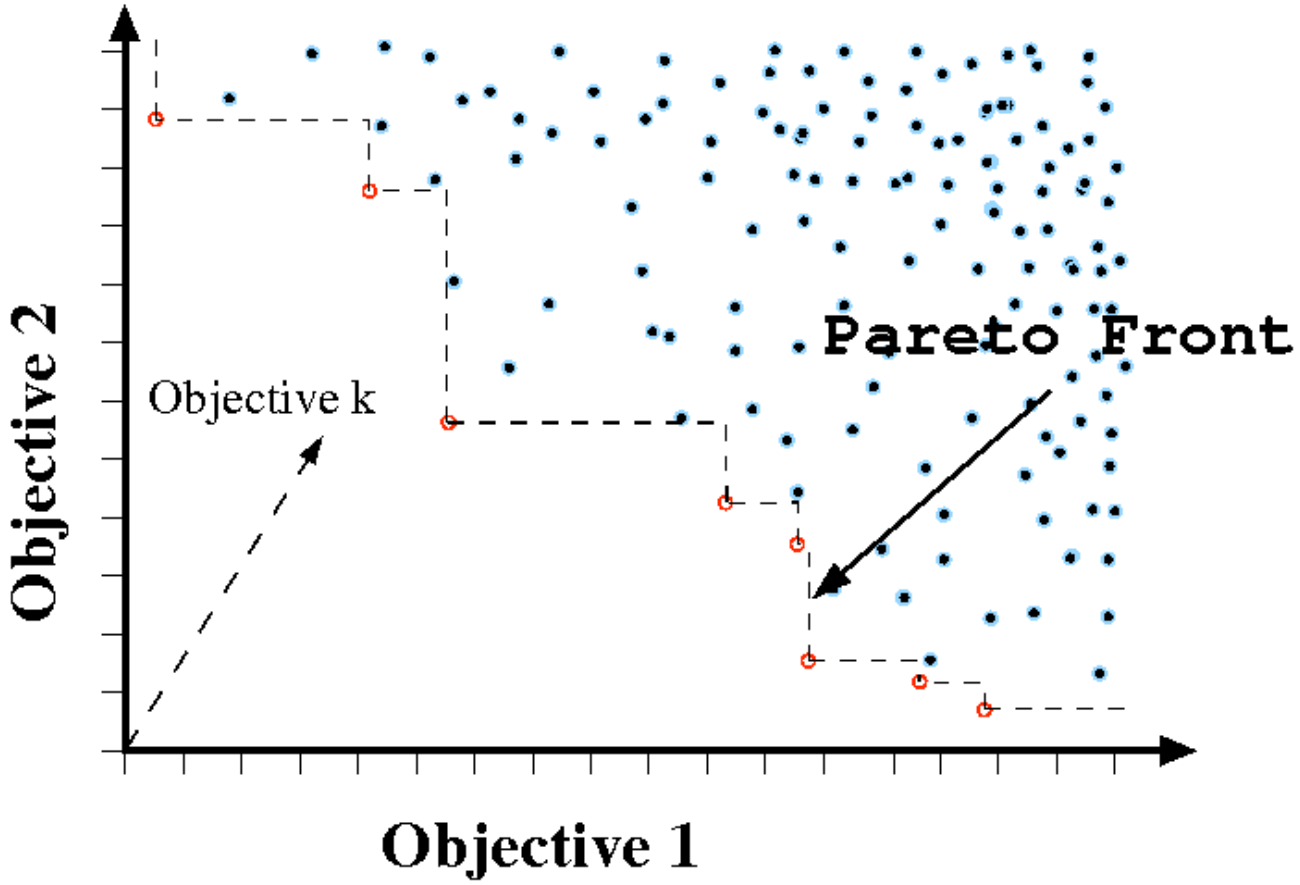
Many application areas:

- Analog (and digital) circuits.
- Antennas.
- Engineering design.
- MEMS devices.
- Neural Networks (optimize topology and/or weights).
- Parameter optimization (multi-modal, non-differentiable).
- Topological Design
- ...

Multi-Objective Optimization

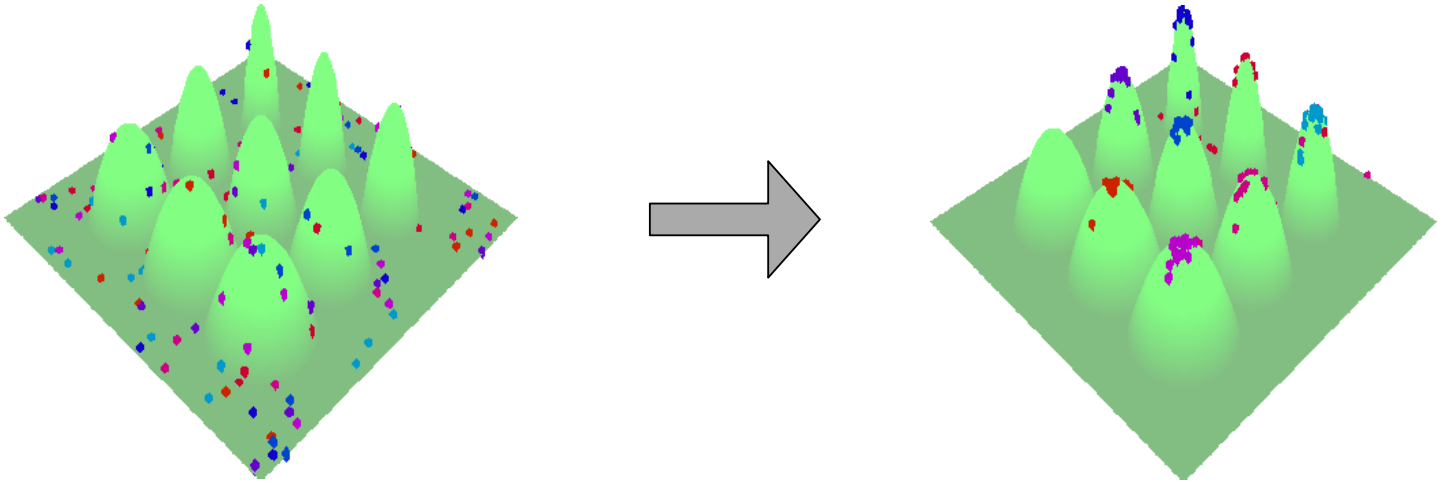
Well suited for multi-objective design:

- Aeronautical and other engineering design.



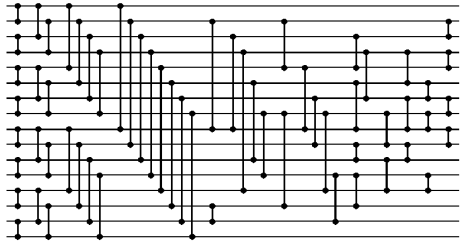
Parallelizing EAs

- First Parallel EA publication: [Tanese89].
- Three paradigms:
  - **Master-Slave:** One processor manages the population and sends the slave processors individuals to evaluate.
  - **Grid Model**: Each individual is located in some Euclidean Space and can only mate with neighbors (one individual on a core).
  - **Island Model**: Multiple sub-populations with migration between islands (multiple individuals on a processor).

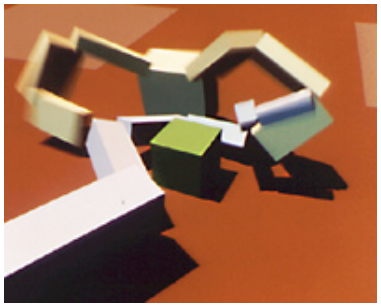


Early Parallel EAs

- W. Daniel Hillis. [90, 92]: evolved sorting network for 16 elements.
- Used a Connection Machine 2: 64,536 processors, population of 500 to 1000000, 'about 100 to 1000 generations per minute'.



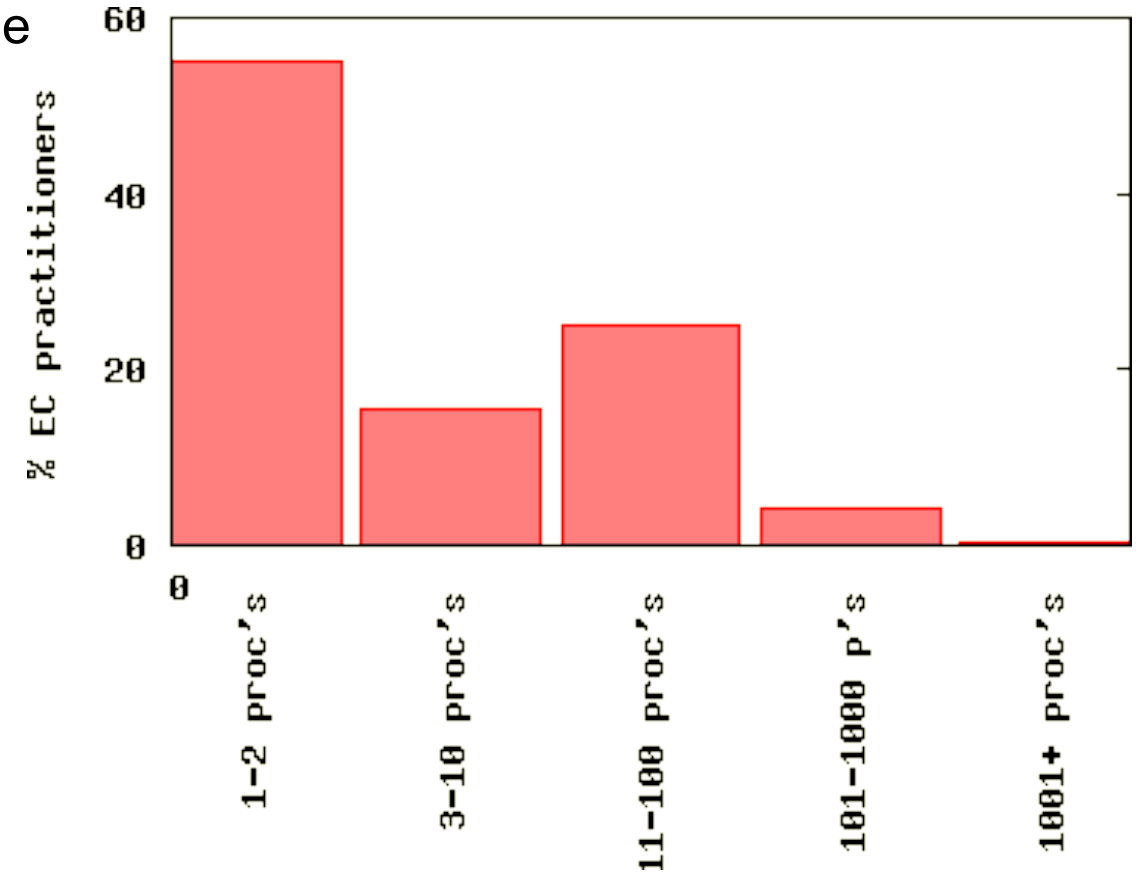
- Karl Sims. [94]: evolved virtual creatures on a CM-5.
- Total of 30,000 physics-based simulations in 3 hours.



Parallel EAs are Common

From a 2007 survey of EC practitioners:

- A sizable percentage use clusters.



Evolving Antennas

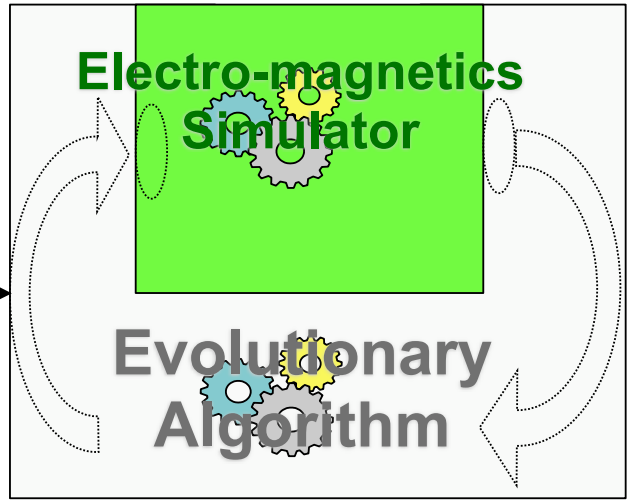
- Transmit Frequency: 8470 MHz
- Receive Frequency: 7209.125 MHz
- Antenna RF Input: 1.5W = 1.76 dBW = 31.76 dBm
- VSWR: < 1.2 : 1 at the antenna input port at Transmit Freq, < 1.5 : 1 at the antenna input port at Receive Freq
- Antenna Gain Pattern: Shall be 0 dBic or greater for angles 40 <= theta <=80; 0 <= phi <= 360
- Antenna pattern gain (this shall be obtained with the antenna mounted on the ST5 mock-up) shall be 0.0 dBic (relative to anisotropic circularly polarized reference) for angles 40 <= theta <=80; 0 <= phi <= 360, where theta and phi are the standard spherical coordinate angles as defined in the IEEE Standard Test

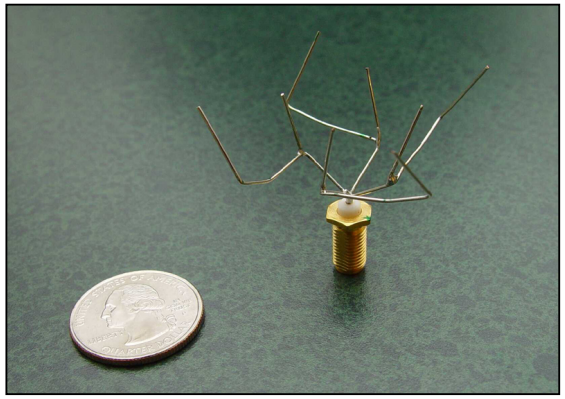
Requirements

- Antenna Input Impedance: 50 ohms at the antenna input port
- Magnetic dipole moment: < 60 mA-cm^2
- Grounding: Cable shields of all coaxial inputs and outputs shall be returned to RF ground at the transponder system chasis. The cases of all comm units will be electrically

Small Amount of E/M Expertise

Small Amount of EA Expertise



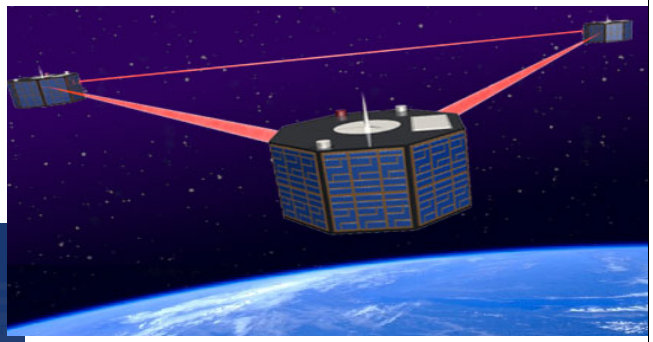


- Evolutionary Design Engine:
  - –Parallel, any Unix.
  - –Uses NEC4.
- Beowulf cluster of over 80 processors.

Result: Evolved Antenna

Evolved Antenna for ST5 Mission

- Evolved an X-band antenna for NASA's ST5 Mission.
  - 40% less design time; 50% less parts; twice as efficient.
- Launched on a Pegasus rocket: March 22, 2006.
- First Computer-Evolved Object in Space.



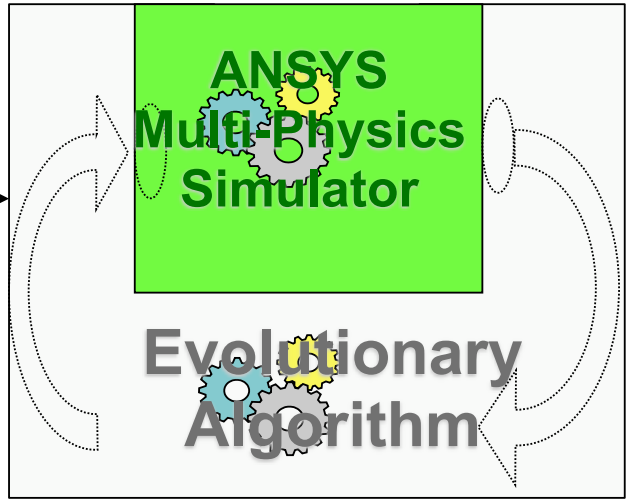
Evolving MEMS Designs

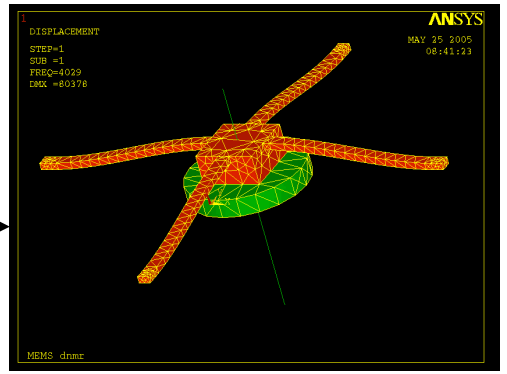
Evolutionary MEMS devices (or anything else) follows the same paradigm as for evolving antennas:

- Transmit Frequency: 8470 MHz
- Receive Frequency: 7209.125 MHz
- Antenna RF Input: 1.5W = 1.76 dBW = 31.76 dBm
- VSWR: < 1.2 : 1 at the antenna input port at Transmit Freq, < 1.5 : 1 at the antenna input port at Receive Freq
- Antenna Gain Pattern: Shall be 0 dBic or greater for angles 40 <= theta <=80; 0 <= phi <= 360
- Antenna pattern gain (this shall be obtained with the

Requirements
+
MEMS Expertise
+
EA Expertise

- Grounding: Cable shields of all coaxial inputs and outputs shall be returned to RF ground at the transponder system chasis. The cases of all comm units will be electrically





Result:
Evolved MEMS device

Part II:
Premature Convergence & ALPS

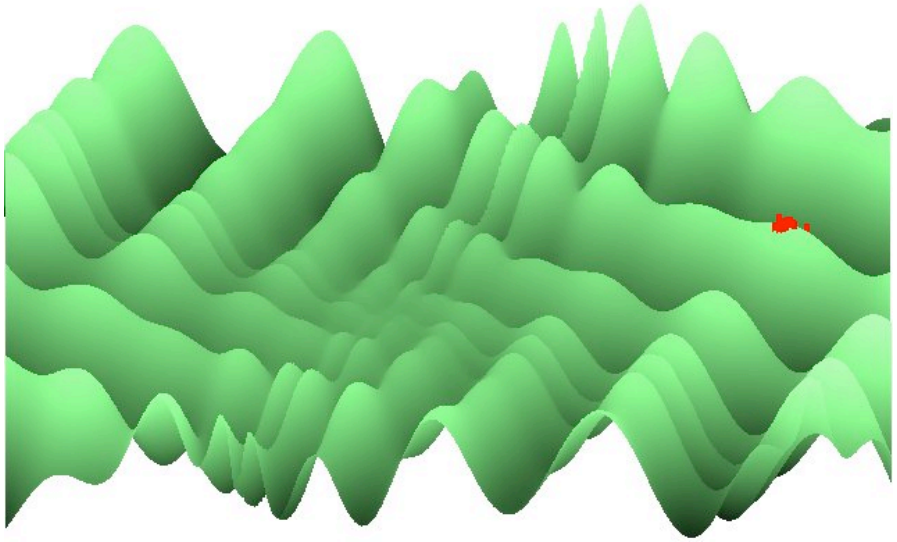
Motivation

- I use EAs on a small cluster (120 cores).
- Typically:
  - Start an evolutionary run.
  - After 30 minutes, there is no more improvement.
  - Optimization had **stopped**.
  - ★ *premature convergence*.
- How to fully utilize my cluster for an entire weekend?

Premature Convergence

Ideally, we want to run our EA and have it find the global optimum.

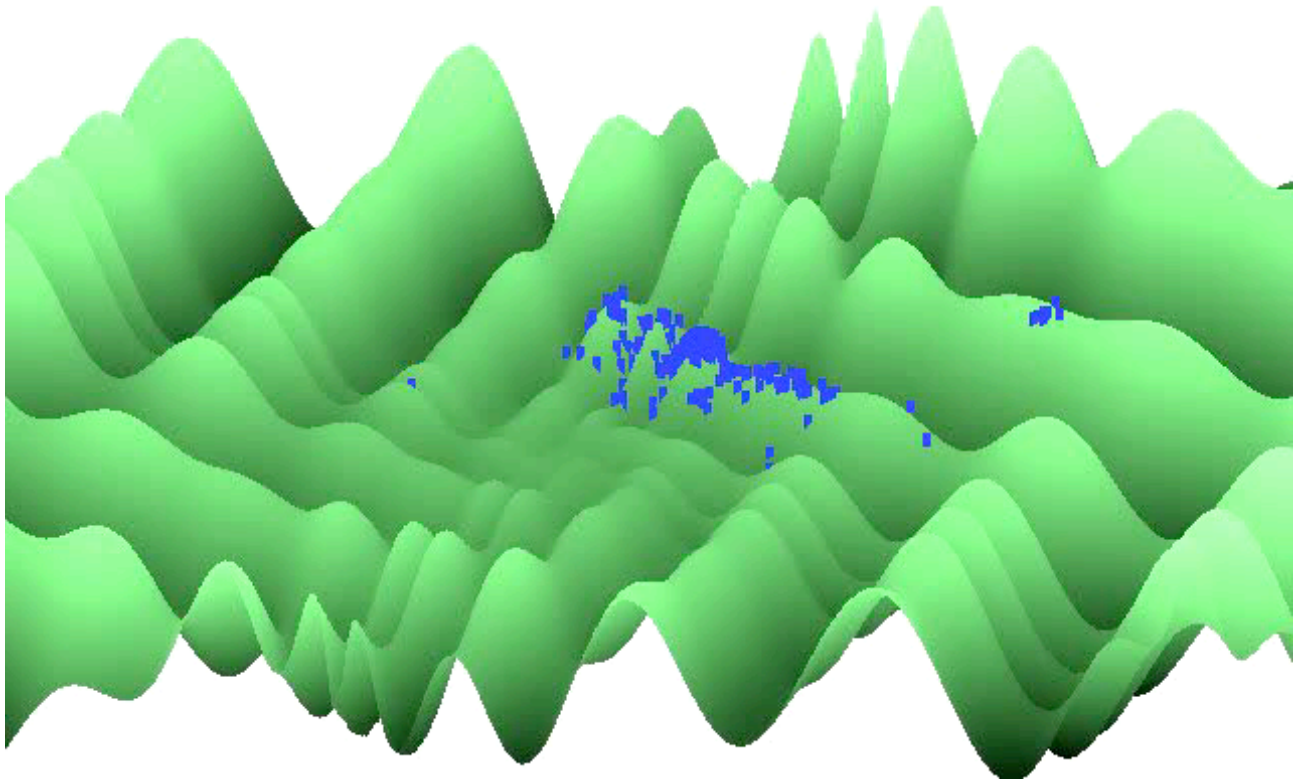
- A common problem is Premature Convergence:
  - The best fitness stops improving.
  - Population has converged to a local optima.
  - Mutation and recombination can't move the population to a new and better fitness peak.
  - Will **never** find the global optimum:
    - Worse than exhaustive search.
    - Worse than random search.



How to overcome Premature Convergence?

Work on Overcoming PC

- Larger population size, increase variation size, use diversity maintenance techniques, spatially distributed populations, probabilistic modeling ...

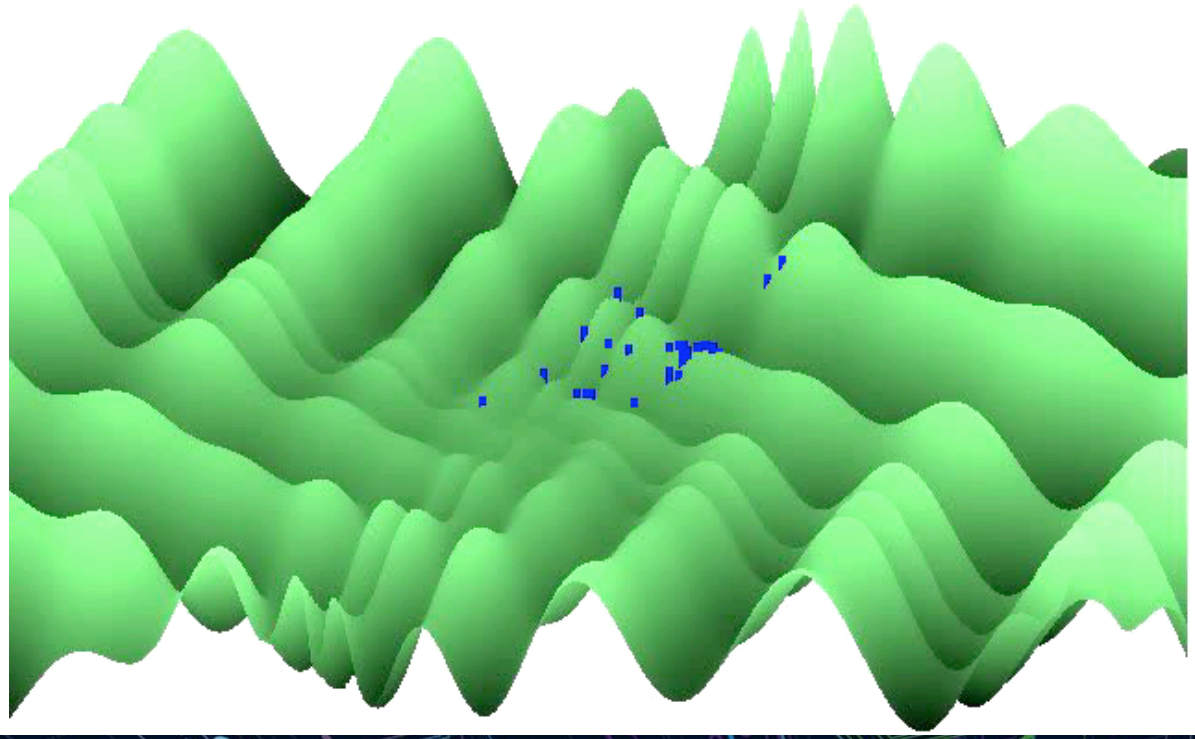


Multiple Independent Runs

- To overcome the initial population…

  …. use *multiple runs*: **Multi-Start EA**.

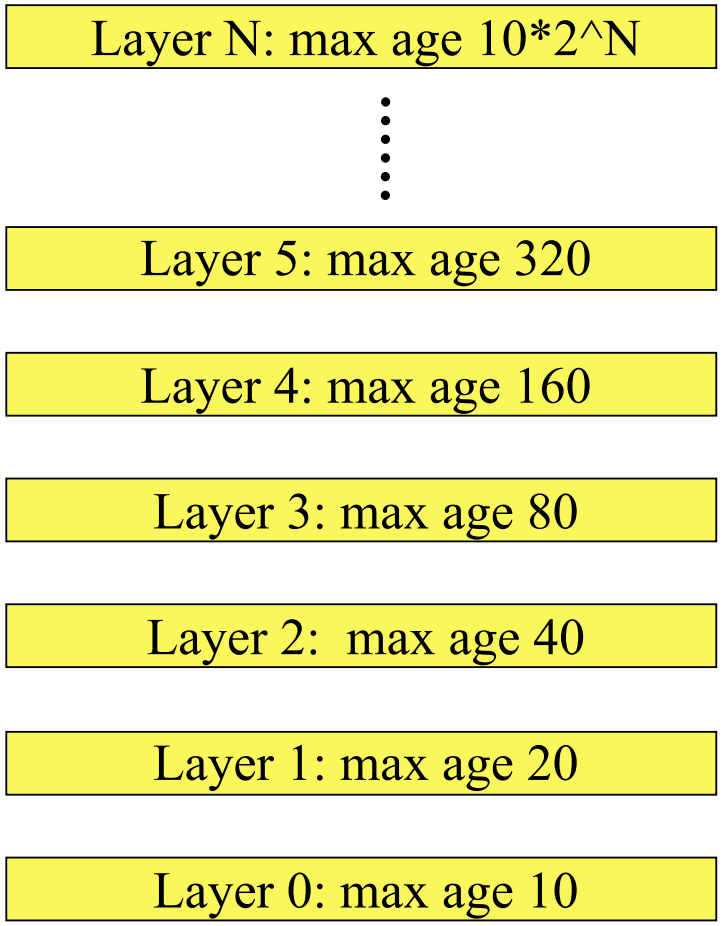
- We do this implicitly when we re-run with a new random number seed.



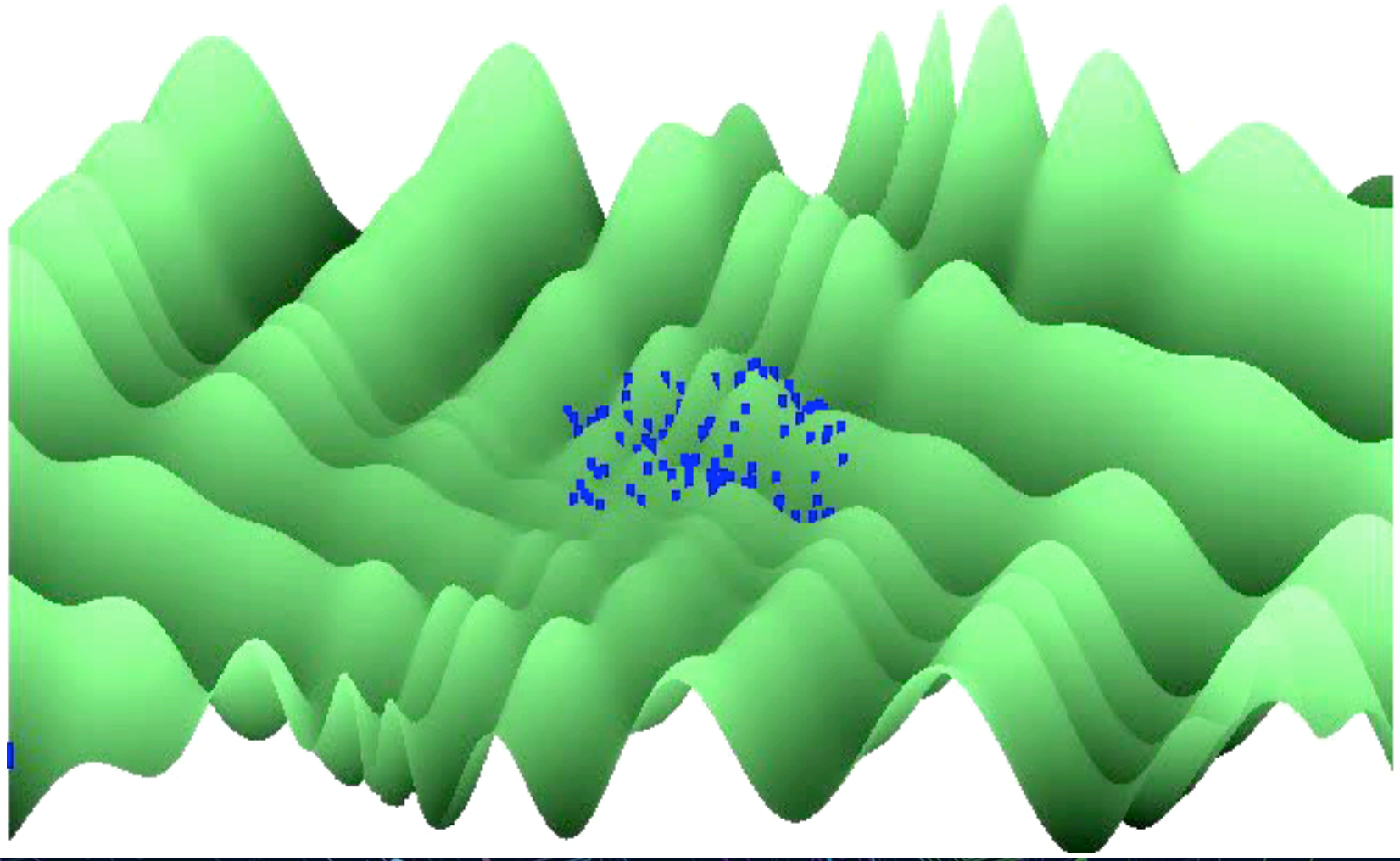
ALPS: Overview

The Age-Layered Population System (ALPS):

- Separate population into layers by age.
- Limited competition & breeding between adjacent layers.
- Regularly replace “youngest” layer with random individuals.



ALPS



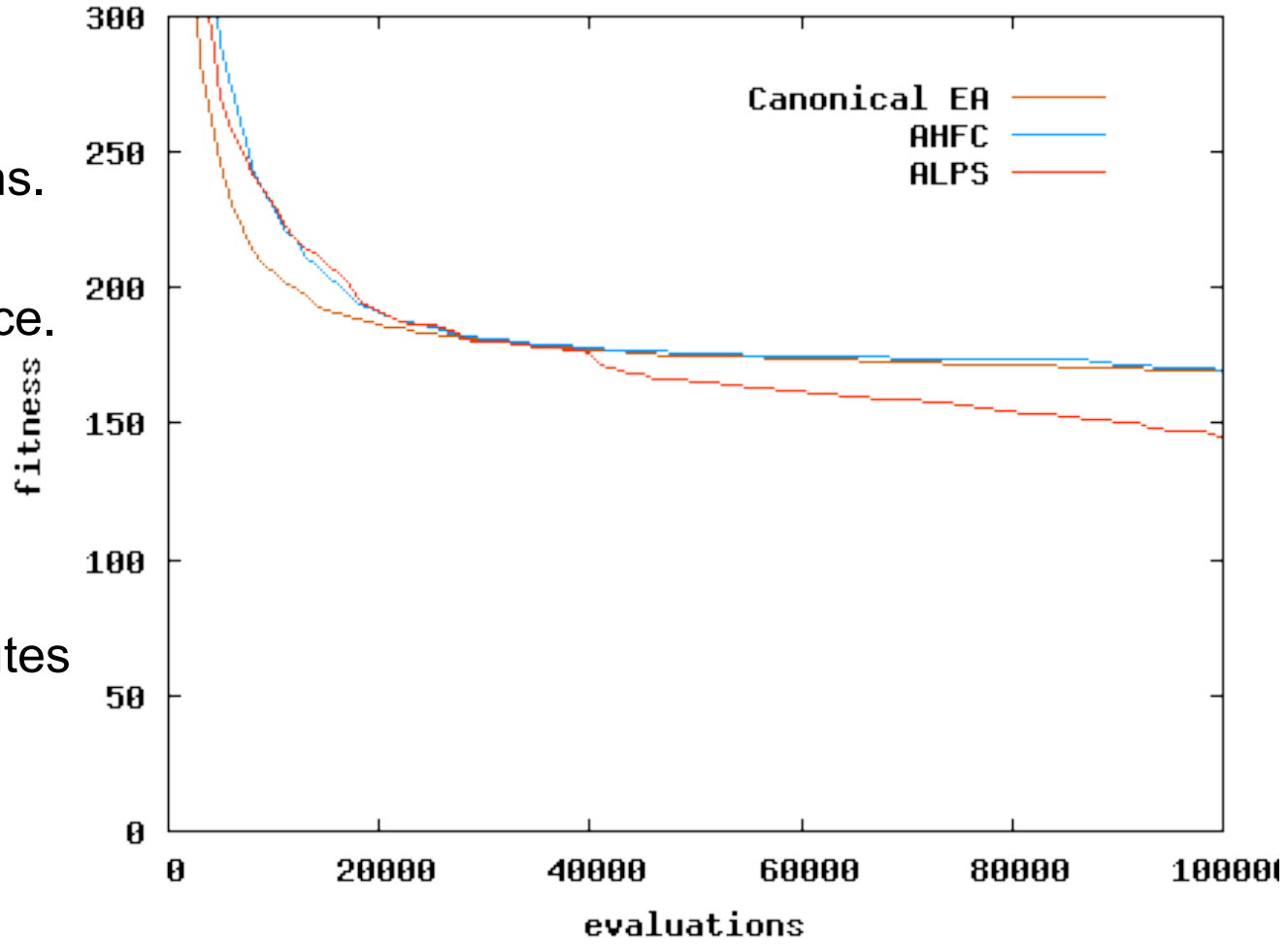
100,000 Evaluations

Typical EA Run:

- 1000 individuals
- 100 generations.

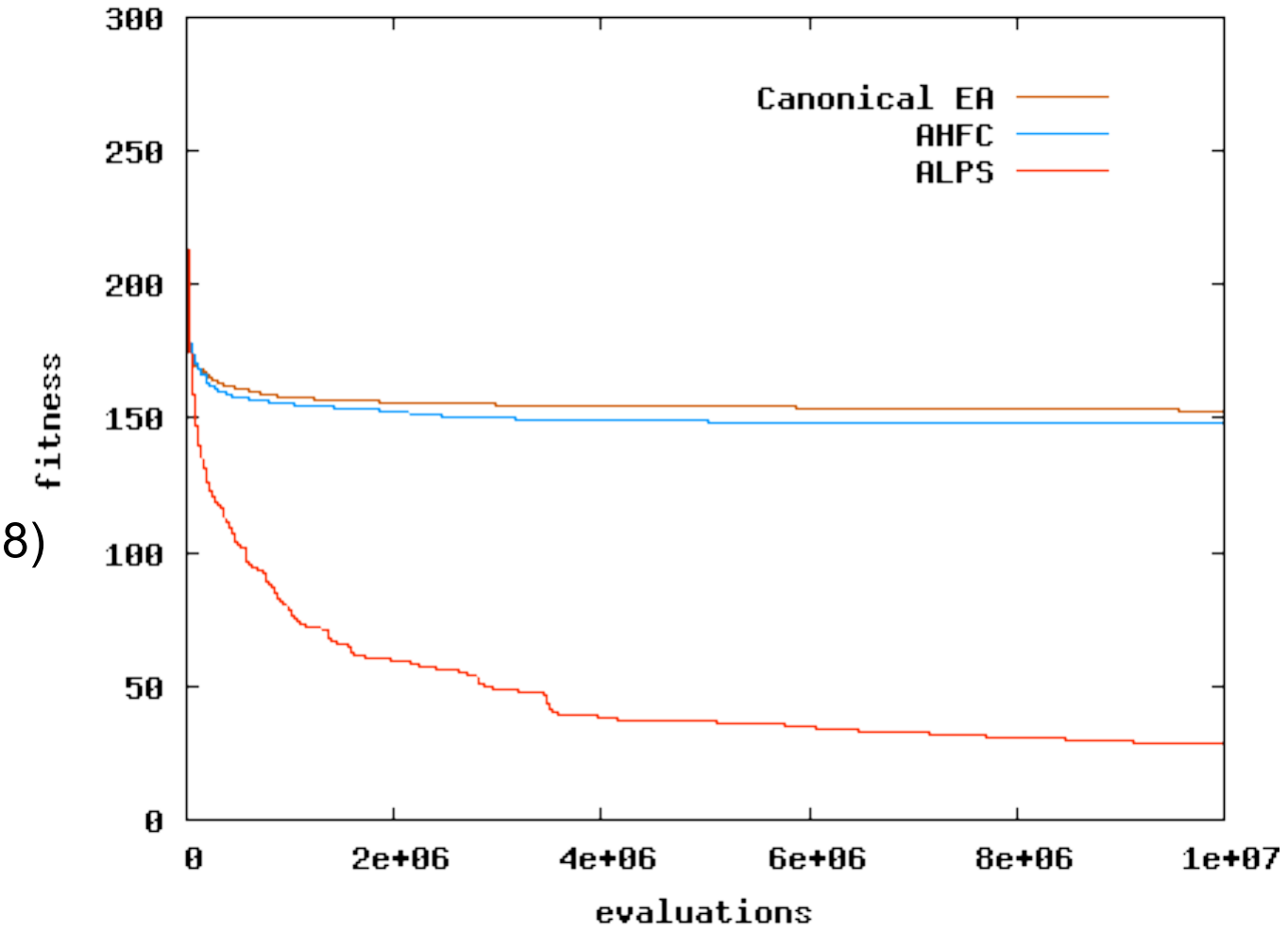
= 100000 Evaluations.

- Similar performance.
- Typical EAs have converged.
- Less than 30 minutes of computation….



10 Million Evaluations

- Canonical EA: (**153.1**+-16.4)
- AHFC: (**148.0**+-35.4)
- ALPS:(**28.7**+-10.8)



Conclusions

1. Many algorithms are poorly suited for massively parallel computers:
   - Either single-threaded or
   - Have high communication to computation ratio.

=> **Evolutionary Algorithms** parallelize naturally.

2. Algorithms converge prematurely.

=> Using the **Age-Layered Population Structure** avoids premature-convergence.