INTRODUCTION TO EVOLUTIONARY COMPUTATION

Evolutionary Computation

- Elements of Evolution:
  - Reproduction
  - Random variation
  - Competition
  - Selection of contending individuals from a population.
- Evolutionary computation: computational methods simulating evolution, mostly used to find a solution in a large search space.

Optimization

- Environment of an organism and its survival chance in the environment vs. evaluation of parameter to optimize for a solution candidate.
- Start from a random sample of solution candidates and simulate natural evolution, optimizing for an evaluation function (fitness of the individual).
- Classical methods: gradient descent, deterministic hill climbing, random search.
- Competitive problems: nonlinear, stochastic, temporal, or chaotic components with multiple local optima.

Robust Adaptation

- Problems with dynamic nature. Environment and parameters change in time.
- Adapting new environment by recombining the succesfull pieces from independent individuals.

Machine Intelligence

- Capability of a system to adapt its behavior to meet desired goals in a range of environments.
- Evolution of organisms ➔ natural intelligence
- Evolutionary computation can be used to evolve the data in an artificial intelligence model.

Biology

- Using computation to simulate the evolution and understand the evolution of organisms.
- Rather using computation in biology then simulating biological evolution for computation.

History

- The idea of using simulated evolution to solve engineering and design problems have been around since the 1950's.
  - Bremermann, 1962
  - Box, 1957
  - Friedberg, 1958
- However, it wasn't until the early 1960's that we began to see three influential forms of EC emerge:
  - Evolutionary Programming (Lawrence Fogel, 1962),
  - Genetic Algorithms (Holland, 1962)
  - Evolution Strategies (Rechenberg, 1965 & Schwefel, 1968),

- The designers of each of the EC techniques saw that their particular problems could be solved via simulated evolution.
  - Fogel was concerned with solving prediction problems.
  - Rechenberg & Schwefel were concerned with solving continous parameter optimization problems.
  - Holland was concerned with developing robust adaptive systems.

- Each of these researchers successfully developed appropriate ECs for their particular problems independently.
- In the US, Genetic Algorithms have become the most popular EC technique due to a book by David E. Goldberg (1989) entitled, “Genetic Algorithms in Search, Optimization & Machine Learning”.
- This book explained the concept of Genetic Search in such a way the a wide variety of engineers and scientist could understand and apply.

- However, a number of other books helped fuel the growing interest in EC:
  - Lawrence Davis', "Handbook of Genetic Algorithms", (1991),
  - Zbigniew Michalewicz' book (1992), "Genetic Algorithms + Data Structures = Evolution Programs".
  - John R. Koza's "Genetic Programming" (1992), and
  - D. B. Fogel's 1995 book entitled, "Evolutionary Computation: Toward a New Philosophy of Machine Intelligence".
- These books not only fueled interest in EC but they also were instrumental in bringing together the EP, ES, and GA concepts together in a way that fostered unity and an explosion of new and exciting forms of EC.

History: The Evolution of EC

- First Generation EC
  - Evolutionary Programming (Fogel)
  - Genetic Algorithms (Holland)
  - Evolution Strategies (Rechenberg, Schwefel)
- Second Generation EC
  - Genetic Evolution of Data Structures (Michalewicz)
  - Genetic Evolution of Programs (Koza)
  - Hybrid Genetic Search (Davis)
  - Tabu Search (Glover)

- Third Generation EC
  - Artificial Immune Systems (Forrest)
  - Cultural Algorithms (Reynolds)
  - DNA Computing (Adleman)
  - Ant Colony Optimization (Dorigo)
  - Particle Swarm Optimization (Kennedy & Eberhart)
  - Memetic Algorithms
  - Estimation of Distribution Algorithms

Applications of EC

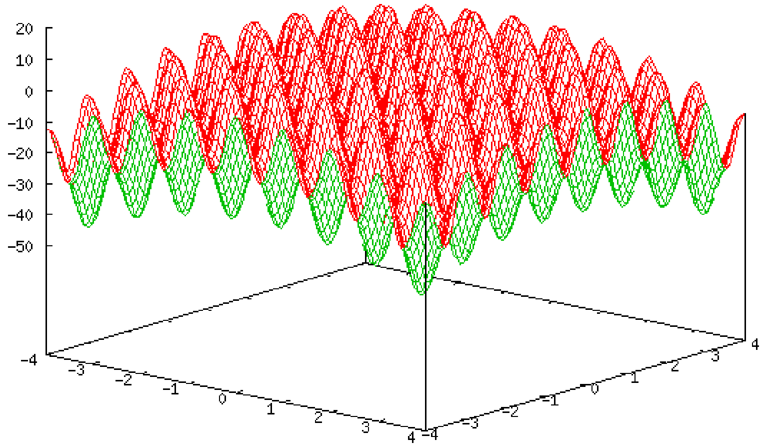
- Evolutionary Computation has been successfully applied to a wide range of problems including:
  - Aircraft Design,
  - Routing in Communications Networks,
  - Tracking Windshear,
  - Game Playing (Checkers [Fogel])
  - Robotics,

- Air Traffic Control,
- Design,
- Scheduling,
- Machine Learning,
- Pattern Recognition,
- Job Shop Scheduling,
- VLSI Circuit Layout,
- Strike Force Allocation,

- Theme Park Tours (Disney Land/World)
- Market Forecasting,
- Egg Price Forecasting,
- Design of Filters and Barriers,
- Data-Mining,
- User-Mining,
- Resource Allocation,
- Path Planning,
- Etc.

EC in General

- An evolution function to optimize (**fitness function**).
  Usually: Multi-dimensional, multimodal, discontinous.



- A sample of search space in a group of solution candidates (**population**)
- A generation procedure to determine the population for the next generation (**selection**, **crossover**, **mutation**)

- Pseudo code for a sample EC

```
t=0
population(t)=randomPopulation()
evaluate(population(t))
while (notDone)
    parents(t)=selectfrom(population(t))
    offsprings(t)=createfrom(parents(t))
    evaluate(offsprings(t))
    population(t+1)=selectfrom(parents(t),offsprings(t))
    t=t+1
```

- In each generation, mean fitness of the population is expected to increase.

- EA's vary depending on:
  - Representation/encoding of an individual (binary, integer, floating point, or data structures,...)
  - Population size and organization (multiple populations, parallel evolving populations, single individual populations,...)
  - The time of selection and selection procedure (selection for recombination, selection for survival,...)
  - Recombination, mutation procedures

- Advantage over gradient methods:
  - Population represents a collected statistics about the search space.
  - Exploring search space while exploiting the information gathered during the evolution.
  - Suitable for discontinous functions, nondifferentiable, multimodal, noisy surfaces.
- Not suitable for linear, quadratic, strongly convex, unimodal, separable problems.
- No free lunch: there is no superior algorithm to solve all class of algorithms.

Types of EC

- Historical classification: EP, GA, ES.
- Genetic Programming is considered an additional class.
- Practically, there are many hybrid models not fitting any of the classes completely. Class distinction gets fuzzy.
- Many different names for many algorithms having similar general form.

Genetic Algorithms

- John Holland (1975)
  - linear bitstring representation
  - fitness proportional selection
  - crossover
- Denotes the class of evolutionary algorithms having a linear array representation with a group of individuals, involving crossover, mutation and selection in each generation cycle.
- binary, integer, floating point representations, parent and offspring population size, selection strategy, crossover, mutation and different operators may vary.

Evolution Strategies

- `Evolutionsstrategie'
- Bienert, Rechenberg, Schwefel, T.U. of Berlin, 1964.
- a single individual encoded as a real-vector.
- Vector is mutated by adding a normally distributed real vector with a variance.
- Contemporary approaches involve recombination, selection and adaptation of algorithm parameters during optimization.

Evolutionary Programming

- Lawrence J. Fogel, 1960.
- Evolving a population of finite state machines.
- Fogel's EP differs from GA that it does not involve crossover, it involves special mutation operations based on behavior, and its selection strategy.

Genetic Programming

- Evolutionary process to evolve computer programs.
- First experiments, Smith (1980), Cramer (1985).
- First comprehensive study, John Koza (1992)
- Tree structured encodings with specific recombination and mutation operations involved.
- Some variants choose linear encodings, i.e. program texts with a certain alphabet.