Historical roots:

- **Evolution Strategies (ESs):**
  - developed by Rechenberg, Schwefel, etc. in 1960s.
  - focus: real-valued parameter optimization
  - individual: vector of real-valued parameters
  - reproduction: Gaussian “mutation” of parameters
  - M parents, K>>M offspring

- **Evolutionary Programming (EP):**
  - Developed by Fogel in 1960s
  - Goal: evolve intelligent behavior
  - Individuals: finite state machines
  - Offspring via mutation of FSMs
  - M parents, M offspring

- **Genetic Algorithms (GAs):**
  - developed by Holland in 1960s
  - goal: robust, adaptive systems
  - used an internal “genetic” encoding of points
  - reproduction via mutation and recombination of the genetic code.
  - M parents, M offspring
Present Status:

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

Interesting dilemma:

• A bewildering variety of algorithms and approaches:
  – GAs, ESs, EP, GP, Genitor, CHC, messy GAs, …
• Hard to see relationships, assess strengths & weaknesses, make choices, …

A Personal Interest:

• Develop a general framework that:
  – Helps one compare and contrast approaches.
  – Encourages crossbreeding.
  – Facilitates intelligent design choices.

Viewpoint:
Starting point:

• Common features

• Basic definitions and terminology

Common Features:

• Use of Darwinian-like evolutionary processes to solve difficult computational problems.

• Hence, the name:

   Evolutionary Computation

Key Element:

An Evolutionary Algorithm

• 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”

An EA template:

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.
Instantiate by specifying:

- Population dynamics:
  - Population size
  - Parent selection
  - Reproduction and inheritance
  - Survival competition
- Representation:
  - Internal to external mapping
- Fitness

EA Population Dynamics:

- Population sizing:
  - Parent population size $M$:
    - degree of parallelism
  - Offspring population size $K$:
    - amount of activity w/o feedback

Examples:
- $M=1$, $K$ small: early ESs
- $M$ small, $K$ large: typical ESs
- $M$ moderate, $K=M$: traditional GAs and EP
- $M$ large, $K$ small: steady state GAs
- $M = K$ large: traditional GP
Selection pressure:

- Overlapping generations:
  - more pressure than non-overlapping

- Selection strategies (decreasing pressure):
  - truncation
  - tournament and ranking
  - fitness proportional
  - uniform

- Stochastic vs. deterministic

Reproduction:

- Preserve useful features
- Introduce variety and novelty

- Strategies:
  - single parent: cloning + mutation
  - multi-parent: recombination + mutation
  - ...

- Price’s theorem:
  - fitness covariance

Exploitation/Exploration Balance:

- Selection pressure: exploitation
  - reduce scope of search

- Reproduction: exploration
  - expand scope of search

- Key issue: appropriate balance
  - e.g., strong selection + high mutation rates
  - e.g., weak selection + low mutation rates

Representation:

- How to represent the space to be searched?
  - Genotypic representations:
    - universal encodings
    - portability
    - minimal domain knowledge
Representation:
• How to represent the space to be searched?
  – Phenotypic representations:
    • problem-specific encodings
    • leverage domain knowledge
    • lack of portability

Fitness landscapes:
• Continuous/discrete
• Number of local/global peaks
• Ruggedness
• Constraints
• Static/dynamic

The Art of EC:
• Choosing problems that make sense.
• Choosing an appropriate EA:
  – reuse an existing one
  – hand-craft a new one

EC: Using EAs to Solve Problems
• What kinds of problems?
• What kinds of EAs?
Intuitive view:

- parallel, adaptive search procedure.
- useful global search heuristic.
- a paradigm that can be instantiated in a variety of ways.
- can be very general or problem specific.
- strong sense of fitness “optimization”.

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 \textit{a priori} assumptions about convexity, continuity, differentiability, etc.
- relatively insensitive to noise
- easy to parallelize

Real-valued Param. Optimization:

- high dimensional problems
- highly multi-modal problems
- problems with non-linear constraints
Discrete Optimization:

- TSP problems
- Boolean satisfiability problems
- Frequency assignment problems
- Job shop scheduling problems

Multi-objective Optimization:

- Pareto optimality problems
- A variety of industrial problems

Properties of standard EAs:

- GAs:
  - universality encourages new applications
  - well-balanced for global search
  - requires mapping to internal representation

Properties of standard EAs:

- ESs:
  - well-suited for real-valued optimization.
  - built-in self-adaptation.
  - requires significant redesign for other application areas.
Properties of standard EAs:

• **EP:**
  – well-suited for phenotypic representations.
  – encourages domain-specific representation and operators.
  – requires significant design for each application area.

Other EAs:

• **GENITOR: (Whitley)**
  – “steady state” population dynamics
  – $K=1$ offspring
  – overlapping generations
  – parent selection: ranking
  – survival selection: ranking
  – large population sizes
  – high mutation rates

Other EAs:

• **GP: (Koza)**
  – standard GA population dynamics
  – individuals: parse trees of Lisp code
  – large population sizes
  – specialized crossover
  – minimal mutation

Other EAs:

• **Messy GAs: (Goldberg)**
  – Standard GA population dynamics
  – Adaptive binary representation
    • genes are position-independent
**Other EAs:**

- GENOCOP: (Michalewicz)
  - Standard GA population dynamics
  - Specialized representation & operators for real valued constrained optimization problems.

**Designing an EA:**

- Choose an appropriate representation
  - effective building blocks
  - semantically meaningful subassemblies

- Choose effective reproductive operators
  - fitness covariance

**Designing an EA:**

- Choose appropriate selection pressure
  - local vs. global search

- Choosing a useful fitness function
  - exploitable information

**Industrial Example: Evolving NLP Tagging Rules**

- Existing tagging engine
- Existing rule syntax
- Existing rule semantics
- Goal: improve
  - development time for new domains
  - tagging accuracy
Evolving NLP Tagging Rules

• Representation: (first thoughts)
  – variable length list of GP-like trees

• Difficulty: effective operators

Evolving NLP Tagging Rules

• Representation: (second thoughts)
  – variable length list of pointers to rules

• Operators:
  – mutation: permute, delete rules
  – recombination: exchange rule subsets
  – Lamarckian: add a new rule

Evolving NLP Tagging Rules

• Population dynamics:
  – multi-modal: $M > \text{small}$
    • typical: 30-50
  – high operator variance: $K/M > 1$
    • typical: 3-5 : 1
  – parent selection: uniform
  – survival selection: binary tournament

Evolving NLP Tagging Rules

• So, what is this thing?
  – A GA, ES, EP, …

• My answer:
  – a thoughtfully designed EA
Analysis tools:

- Schema analysis
- Convergence analysis
- Markov models
- Statistical Mechanics
- Visualization

New developments and directions:

- Exploiting parallelism:
  - coarsely grained network models
    - isolated islands with occasional migrations
  - finely grained diffusion models
    - continuous interaction in local neighborhoods

New developments and directions:

- Co-evolutionary models:
  - competitive co-evolution
    - improve performance via “arms race”
  - cooperative co-evolution
    - evolve subcomponents in parallel

New developments and directions:

- Exploiting Morphogenesis:
  - sophisticated genotype --> phenotype mappings
  - evolve plans for building complex objects rather than the objects themselves.
New developments and directions:

• Self-adaptive EAs:
  – dynamically adapt to problem characteristics:
    • varying population size
    • varying selection pressure
    • varying representation
    • varying reproductive operators
  – goal: robust “black box” optimizer

New developments and directions:

• Hybrid Systems:
  – combine EAs with other techniques:
    • EAs and gradient methods
    • EAs and TABU search
    • EAs and ANNs
    • EAs and symbolic machine learning

New developments and directions:

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

New developments and directions:

• Agent-oriented problems:
  – individuals more autonomous, active
  – fitness a function of other agents and environment-altering actions
  – standard optimization-oriented EAs not well-suited for this.
EA Generalizations:

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

EA Generalizations:

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

Conclusions:

- Powerful tool for your toolbox.
- Complements other techniques.
- Best viewed as a paradigm to be instantiated, guided by theory and practice.
- Success a function of particular instantiation.

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
- My book:
  - Evolutionary Computation: A Unified Approach
    - MIT Press, 2006