

Evolutionary Computing

An intro to

Genetic Algorithms,
Evolution Strategies, &
Genetic Programming

Evolutionary Computation

- ◆ Human intelligence involves (at some level), the ability to adapt to an ever-changing environment
- ◆ Humans are products of evolution
- ◆ Hence, by modeling evolution we might be able to generate intelligent behavior
- ◆ Evolutionary Computation simulates evolution, on a computer
- ◆ Result is a series of optimization algorithms, based on a [simple] set of rules
- ◆ Optimization iteratively improves quality of soln, until optimal or feasible soln is found.

Simulating Evolution

- ◆ 1858 – Charles Darwin’s theory of evolution
 - *Neo-Darwinism* paradigm (Keeton, 1980; Mayr, 1988)
 - » Weismann’s theory of natural selection
 - » Mendel’s concept of genetics
 - *ND* based on processes of **reproduction, mutation, competition, and selection**
- ◆ For computing:
 - Evolution – ‘process of leading to the maintenance or increase of a population’s ability to survive and reproduce in a specific environment (Hartl and Clark, 1989) => *evolutionary fitness*
 - » Optimize fitness with a quantitative *fitness function*
(ability to predict environmental changes and respond adequately)

Genetic Algorithms

A class of stochastic search algorithms, based loosely on biological evolution

Basic Steps:

- 1- Choose chromosome length, population N , crossover probability P_c mutation probability P_m
- 2 - Define the fitness function $f(x)$ to measure performance
- 3 - Randomly generate an initial population N of chromosomes
- 4 - Calculate the fitness of each individual chromosome
- 5 - Select a pair of chromosomes for mating from current population (w/highly fit chroms. more likely for mating)

Genetic Algorithms

- ◆ Most evolutionary algorithms are variations of Gas
- ◆ 1975 – John Holland (1970s founder of EC)
 - » A computer scientist
 - » Algorithms that manipulate strings of binary digits
- ◆ Represented by a sequence of procedural steps for moving from one population of artificial ‘chromosomes’ to a new population
- ◆ Uses *natural selection, crossover & mutation* (genetically inspired)
- ◆ Each ‘chromosome’ consists of many ‘genes’

Genetic Algorithms

Basic Steps (cont.)

- 6 - Create a pair of offspring chromosomes, by applying crossover and mutation operators
- 7 - Place created offspring in new population
- 8 - Repeat from step 5 (mating), until $\text{size}(\text{new population}) = \text{size}(\text{old})$
- 9 - Replace the old (parent) population with the new (offspring) population
- 10 - Repeat from step 4 (calculate fitness), until termination criteria satisfied (population fitness value OR # generations)

Evolution Strategies

- ◆ 1960s – Germany
 - » Students: Rechenberg, 1965 & Schwefel, 1981
- ◆ Proposed to solve technical optimization problems
 - » (eg – optimal shapes of bodies in a flow)
- ◆ Alternative to engineer's intuition
 - » No objective function exists
- ◆ Unlike Gas:
 - » they use only a **mutation** operator
 - » Not represented in coded form – purely numerical optim.
- ◆ Which is best? Application dependent
- ◆ Trade-offs? Generality –vs- coding complexity

Genetic Programming

- ◆ Inspired by John Koza, 1992
- ◆ Takes away the explicit programming requirements of trad'l programs
- ◆ Uses **natural selection**
- ◆ Evolves the **computer code**, rather than just a bit string