Evolutionary Programming

- One glance overview
- Purpose
- Historical background: FSM evolution
- Selection, mutation
- Self-adaptation in modern EP
- Empirical findings, experimental comparison

Evolutionary programming

- Developed: USA in the 1960's
- Early names: D. Fogel
- Typically applied to:
  - traditional EP: machine learning tasks by finite state machines
  - contemporary EP: (numerical) optimization
- Attributed features:
  - very open framework: any representation and mutation op's OK
  - crossbred with ES (contemporary EP)
  - consequently: hard to say what "standard" EP is
- Special:
  - no recombination
  - self-adaptation of parameters standard (contemporary EP)

Purpose

Simulate Evolution as a Learning Process to Generate Artificial Intelligence

- Intelligence defined as the capability of a system to adapt its behaviour to meet its goals in a range of environments (D. Fogel 1995)
- Intelligence viewed as adaptive behaviour
- Prediction of the environment is a prerequisite to intelligent behaviour (prediction and response in the light of a given goal)
- Adaptation is not possible without a capability to predict

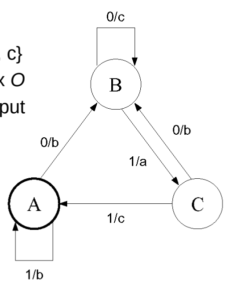
Historical EP

Developed by L. Fogel (1962):

- Evolve a population of finite state machines (FSMs)
- FSMs provide successively better predictions of an environmental sequence
- Predictions in light of a given goal

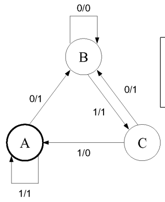
Example of a Finite State Machine

- States S = {A, B, C}
- Inputs I = {0, 1}, output O = {a, b, c}
- Transition function $\delta : S \times I \rightarrow S \times O$
- Transforms input stream into output stream



Finite State Machine as predictors

Performance measured on the basis of the machine's prediction capability, e.g. by $\text{output}_i = \text{input}_{i+1}$



present state	C	B	C	A	A	B
input symbol	0	1	1	1	0	1
next state	B	C	A	A	B	C
output symbol	1	1	0	1	1	1

Initial state: C
Input string: 011101
Output string: 110111
Good predictions: 60%

Search operators

Mutation: Representation “naturally” determines the mutation operators

- Change an output symbol
- Change a state-transition
- Add a state
- Delete a state
- Change the start state

Crossover: None
“no crossover between species”

Selection

P(t): μ parents,
P’(t): μ offspring
Selection by conducting pairwise competitions in round-robin format:

- Each solution $\bar{a} \in P(t) \cup P'(t)$ is evaluated against q other randomly chosen solutions from the population
- For each comparison, a "win" is assigned if $\bar{a}$ is better than its opponent
- The μ solutions with the greatest number of wins are retained to be parents of the next generation

Typically: $q = 10$

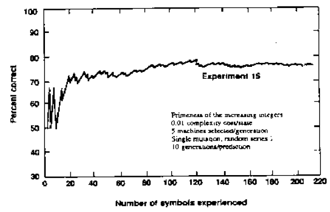
This stochastic variant of $(\mu + \mu)$-selection allows for less fit solutions to propagate

Experiments on predicting primes (1)

Fogel et. al. (1966)

- $X_{prime} : \mathbb{N} \to \{0, 1\}$
- $X_{prime}(n) = \begin{cases} 1 & \text{if } n \text{ is prime} \\ 0 & \text{otherwise} \end{cases}$
- Input alphabet = output alphabet = {0, 1}
- Input string is $X_{prime}(1), X_{prime}(2), \ldots$
- Payoff (fitness) function:
  - 1 point for correct prediction of next input
  - 0 for incorrect prediction
  - penalty for `too much’ states

Experiments on predicting Primes (2)



Percentage of correctly predicted primes in the first 200 prime numbers

After 202 input symbols the best FSM was overfitted:

- had *one* state
- set both output symbols to 0

Philosophical view of EP

- Evolution as a process of adaptive behaviour
- Benefits of optimised behavior accumulate to the reproductive population
- Solution is an abstraction of a reproductive population
- Representation follows from the task at hand
- Mutation operations follow from the representation (requirement: behavioural link between parent and offspring)
- Selection should be made probabilistic

Modern EP

Applied for continuous parameter optimisation

Similar to evolution strategy, with:

- Self-adaptation of n standard deviations (meta-EP)
- Self-adaptation of covariances (Rmeta-EP)
- $\mu = \lambda$ (i.e., parent and offspring population size are identical)
- No analogue of recombination
- Probabilistic $(\mu + \mu)$-selection

Mutation operator (1)

Self-adaptation in general:

- adds strategy parameters to candidates
- evolves strategy parameters with object variables i.e., they both undergo variation and selection

Self-adaptation in EP:

- $\langle x_1,\dots,x_n \rangle$ becomes $\langle x_1,\dots,x_n, \sigma_1,\dots, \sigma_n \rangle$
- mutation: Gaussian random variable · step size σ

  $\sigma'_i = \sigma_i \cdot (1 + \alpha \cdot N_i(0, 1))$

  $x'_i = x_i + \sigma'_i \cdot N_i(0, 1)$
- the order (mutate σ 1st, then x) is important

Mutation operator (2)

Results by Beyer (1995):

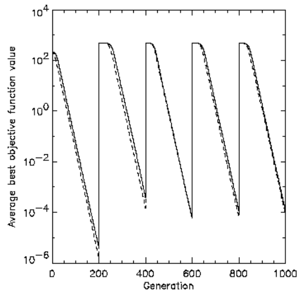
- For $\tau_0 = \alpha$ (small), $n_\sigma = 1$, the evolution strategy and the evolutionary programming method behave identically

  (ES works by so-called lognormal σ mutation mechanism, EP uses a normal distribution)
- Self-adaptation works for a variety of different pdf's for the modification of step sizes

  (e.g. Cauchy instead of Gaussian)

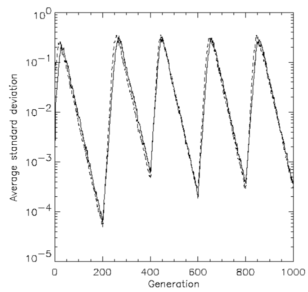
Experimental Test (1)

Best objective function value, sphere model, ES / EP:



Experimental Test (2)

Average mutation rate, sphere model, ES / EP:



Empirical findings on self-adaptation

- Often, lognormal modifications outperform normal modifications.
  $\Rightarrow$ EP typically uses the ES method.
  (Saravanan 1994, Saravanan, Fogel 1994, Saravanan, Fogel, Nelson 1995)
- On noisy objective functions, this behaviour inverts.
  (Angeline 1996)
- It is important to modify σ_i first and use σ'_i to modify the object variables. (Gehlhaar, Fogel 1996)

? Self-adaptation works also with $(\mu + \lambda)$-selection.

? Self-adaptation works also with $\mu = \lambda$.

? Self-adaptation works also without recombination.

The last three results from (Gehlhaar, Fogel 1996).

$\Rightarrow$ Careful check of the last three statements required!