Fall 2010 Graduate Course on
Dynamic Learning

Chapter 5: Evolutionary Monte Carlo

October 10, 2010

Byoung-Tak Zhang

School of Computer Science and Engineering &
Cognitive Science and Brain Science Programs
Seoul National University

http://bi.snu.ac.kr/~btzhang/

Overview

- Evolutionary Computation (EC)
  - Biological Inspiration
  - Basic Elements
- Probabilistic Evolutionary Algorithms
  - EDA
  - BEA
  - eMCMC: Evolutionary Monte Carlo (EMC)
- EC and PF
  - Similarities
  - Differences

Charles Darwin (1859)

"Owing to this struggle for life, any variation, however slight and from whatever cause proceeding, if it be in any degree profitable to an individual of any species, in its infinitely complex relations to other organic beings and to external nature, will tend to the preservation of that individual, and will generally be inherited by its offspring."

Evolutionary Algorithms

- A Computational Model Inspired by Natural Evolution and Genetics
- Proved Useful for Search, Machine Learning and Optimization
- Population-Based Search (vs. Point-Based Search)
- Probabilistic Search (vs. Deterministic Search)
- Collective Learning (vs. Individual Learning)
- Balance of Exploration (Global Search) and Exploitation (Local Search)

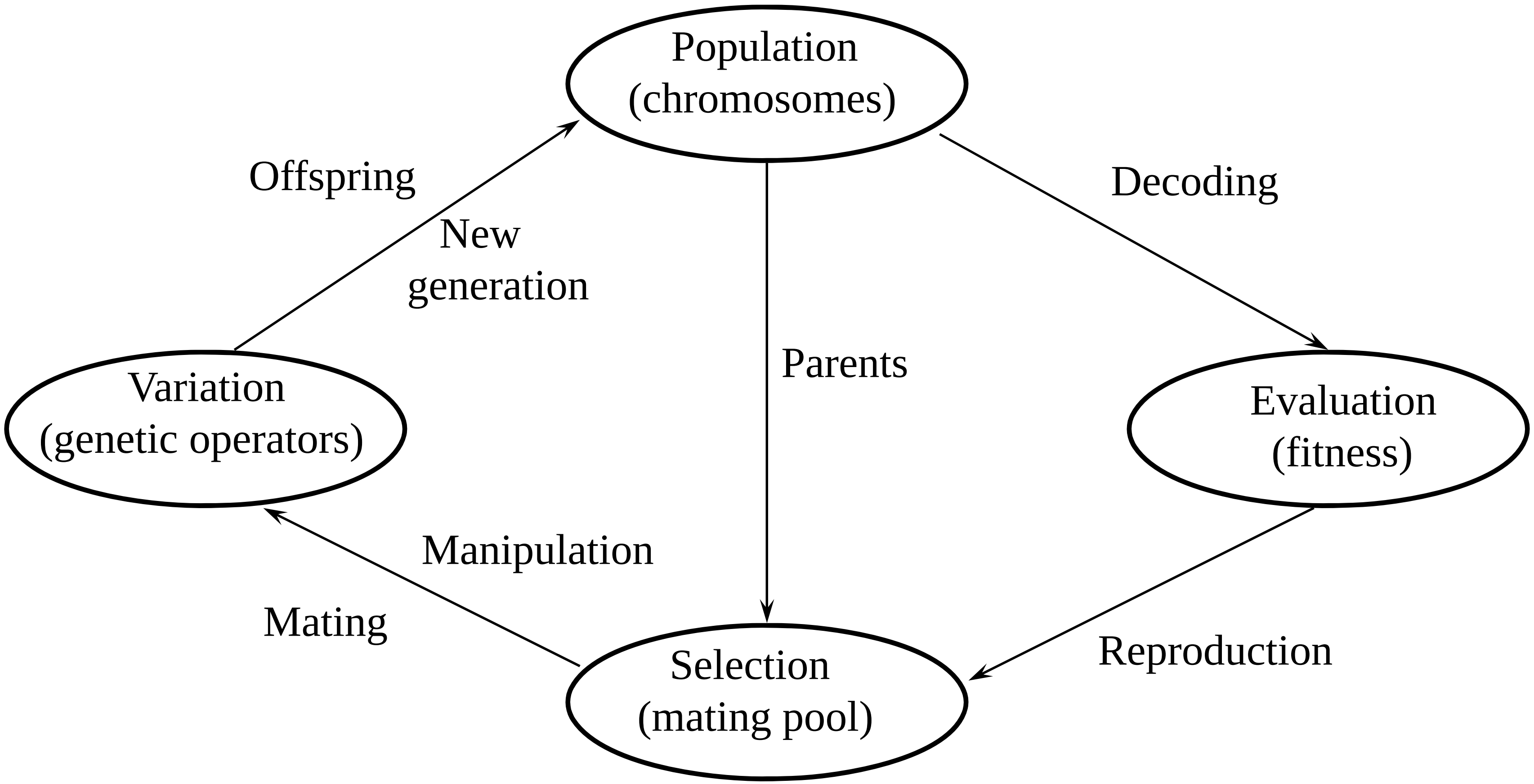
Biological Terminology

- Gene
  - Functional entity that codes for a specific feature e.g. eye color
  - Set of possible alleles
- Allele
  - Value of a gene e.g. blue, green, brown
  - Codes for a specific variation of the gene/feature
- Locus
  - Position of a gene on the chromosome
- Genome
  - Set of all genes that define a species
  - The genome of a specific individual is called genotype
  - The genome of a living organism is composed of several
  - Chromosomes
- Population
  - Set of competing genomes/individuals

Analogy to Evolutionary Biology

- Individual (Chromosome) = Possible Solution
- Population = A Collection of Possible Solutions
- Fitness = Goodness of Solutions
- Selection (Reproduction) = Survival of the Fittest
- Crossover = Recombination of Partial Solutions
- Mutation = Alteration of an Existing Solution

Simulated Evolution



Procedure

```
begin
    t <- 0;
    initialize P(t);
    evaluate P(t);
    while (not termination condition) do
        recombine P(t) to yield C(t);
        evaluate C(t);
        select P(t+1) from P(t) and C(t);
        t <- t+1;
    end
end
```

Evolutionary Algorithm

1. Initialize $t \leftarrow 0$.
   - For $i = 1, \ldots, N$: generate $x_t^{(i)}$.
   - For $i = 1, \ldots, N$: evaluate fitness $f_t^{(i)} = f(x_t^{(i)})$, $t \leftarrow 1$.
2. Variation
   - Generate offspring $P(t) = \{x_t^{(i)}\}$ from parents $P(t\text{-}1) = \{x_{t-1}^{(i)}\}$ by genetic operators.
   - For $i = 1, \ldots, N$: evaluate fitness $f_t^{(i)} = f(x_t^{(i)})$.
3. Selection
   - Select population $\tilde{P}(t) = \{\tilde{x}_t^{(i)}\}$ from offspring $P(t) = \{x_t^{(i)}\}$ and parents $P(t\text{-}1) = \{x_{t-1}^{(i)}\}$ according to fitness $\tilde{f}_t^{(i)}$.
   - New population $P(t) \leftarrow \tilde{P}(t)$.
   - Set $t \leftarrow t+1$ and go to step 2.

Selection Strategies

- Proportionate Selection
  - Reproduce offspring in proportion to fitness f_i.

$$p_s\left(x_t^{(i)}\right) = \frac{f\left(x_t^{(i)}\right)}{\sum_{j=1}^{\lambda} f\left(x_t^{(j)}\right)}$$

- Ranking Selection
  - Select individuals according to $rank(f_i)$.

$$p_s\left(x_t^{(i)}\right) = \begin{cases} \dfrac{1}{\mu}, & 1 \le i \le \mu \\ 0, & \mu \le i \le \lambda \end{cases}$$

- Tournament Selection
  - Choose q individuals at random, the best of which survives.

$$p_s\left(x_t^{(i)}\right) = \frac{1}{\lambda^q}\left(\left(\lambda - i + 1\right)^q - \left(\lambda - i\right)^q\right)$$

Genetic Operators for Bitstrings

- Reproduction: make copies of chromosome (the fitter the chromosome, the more copies)

1 0 0 0 0 1 0 0 $\longrightarrow$ 1 0 0 0 0 1 0 0

1 0 0 0 0 1 0 0

- Crossover: exchange subparts of two chromosomes

1 0 0 | 0 0 1 0 0 $\longrightarrow$ 1 0 0 1 1 1 1 1

1 1 1 | 1 1 1 1 1 $\quad$ 1 1 1 0 0 1 0 0

- Mutation: randomly flip some bits

0 0 0 0 0 1 0 0 $\longrightarrow$ 0 0 0 0 0 0 0 0

Mutation

- For a binary string, just randomly “flip” a bit.
- For a more complex structure, randomly select a site, delete the structure associated with this site, and randomly create a new sub-structure.
- Some EAs just use mutation (no crossover).
- Normally, however, mutation is used to search in the “local search space”, by allowing small changes in the genotype (and therefore hopefully in the phenotype).

Recombination (Crossover)

- Crossover is used to swap (fit) parts of individuals, in a similar way to sexual reproduction.
- Parents are selected based on fitness.
- Crossover sites selected (randomly, although other mechanisms exist), with some probability.
- Parts of the parents are exchanged to produce children.

Variants of Evolutionary Algorithms

- Evolutionary Programming (Fogel et al., 1960’s)
  - FSMs, mutation only, tournament selection
- Evolution Strategy (Rechenberg et al., 1960’s)
  - Real values, mainly mutation, truncation selection
- Genetic Algorithm (Holland et al., 1970’s)
  - Bitstrings, mainly crossover, proportionate selection
- Genetic Programming (Koza, 1990)
  - Trees, mainly crossover, proportionate selection
- Hybrids: BGA (Muehlenbein et al., 1993), BGP (Zhang et al., 1995)
  - Tree + real values, crossover + mutation + local search, truncation selection

Some More Variants … and Evolutionary Monte Carlo

- Differential Evolution (DE)
  - Storn & Price, 1995
- Particle Swarm Optimization (PSO)
  - Kennedy & Eberhart, 1995
- Ant Colony Optimization (ACO)
  - Dorigo et al., 1996
- Estimation of Distribution Algorithms (EDA)
  - Muehlenbein & Paass, 1996
- Bayesian evolutionary algorithms (BEA)
  - Zhang, 1999
- Evolutionary MCMC (eMCMC)
  - Zhang & Cho, 2001

Probabilistic Evolutionary Algorithms

Evolutionary Monte Carlo

EDA

- EDAs build a probabilistic model of population of samples (parameter vectors).
- Instead of using genetic operators, EDAs generate samples from the probability model.
- Many variants in probability estimation models
  - UMDA, BMDA, FDA, BOA, …
- Use machine learning techniques to perform evolutionary optimization

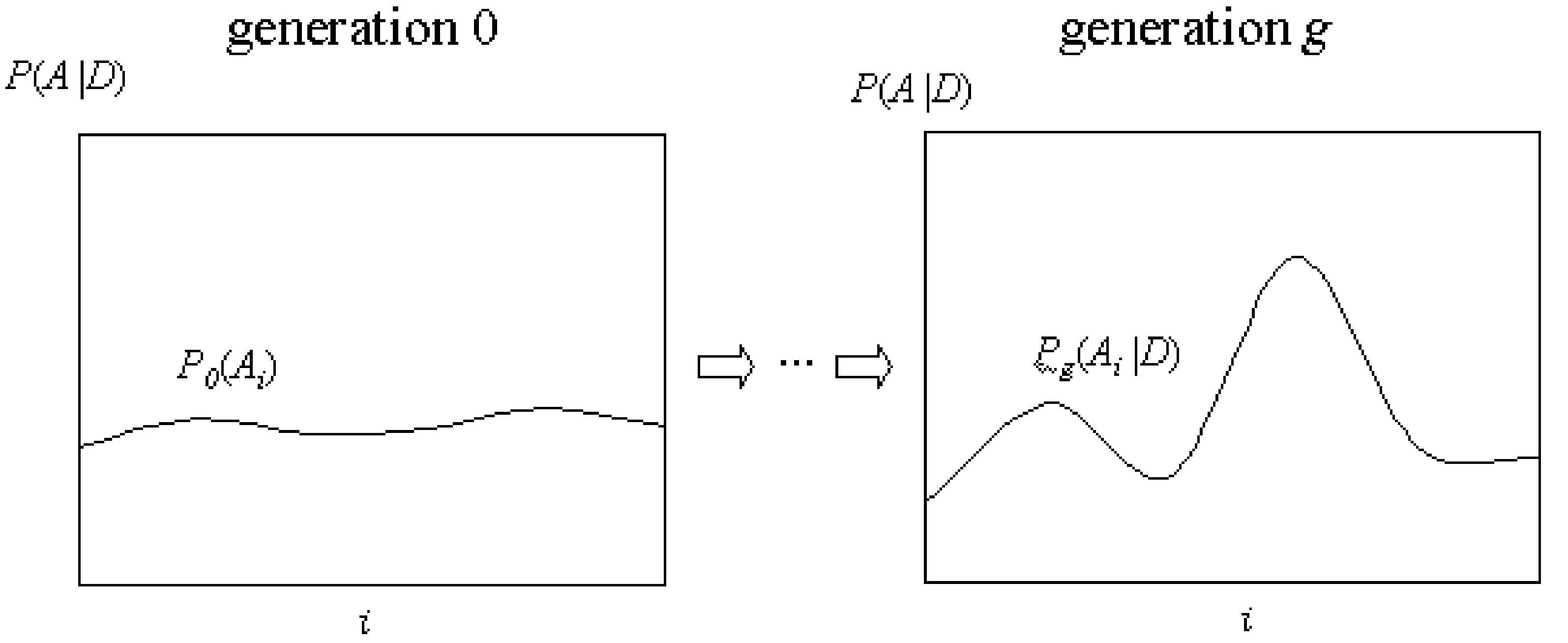
PSO

- Originally motivated to simulate social behavior of insect swarms.
- Starting with a random population, the particles are flown through the problem space.
- Each point is assigned a randomized velocity.
- Maintains the global optima, pbest and gbest.
- Not a probabilistic evolutionary model

BEA: Basic Idea

- "Evolutionary computation (EC) is viewed as an iterative process of *generating the individuals of ever higher posterior probabilities* from the priors and the observed data" (Zhang, 1999)

$$P(A|D)=\frac{P(D|A)P(A)}{P(D)}=\frac{P(D|A)P(A)}{\int P(D|A)P(A)dA}\approx\frac{P(D|A)P(A)}{\sum_{A'\in\mathbf{A}(g)}P(D|A')P(A')}$$



BEA: Algorithm (Zhang, 1999)

1. Sample M individuals A_i ($i=1,\ldots,M$) from $P_0(A)$. Set $g=1$.
2. Compute the posterior fitness $P_i(g) = P_g(A_i|D)$ for $i=1,\ldots,M$:

$$P_g(A_i \mid D) = \frac{P(D \mid A_i) P_{g-1}(A_i)}{\sum_{A_j \in A(g)} P(D \mid A_j) P_{g-1}(A_j)}$$

3. Generate offspring A_i' by sampling from the posterior distribution using variation operators, such as mutation and recombination:

$$P'_{g+1}(A_i' \mid D) = \sum_{A_i \in A(g)} P_g(A_i \mid D) P(A_i' \mid A_i)$$

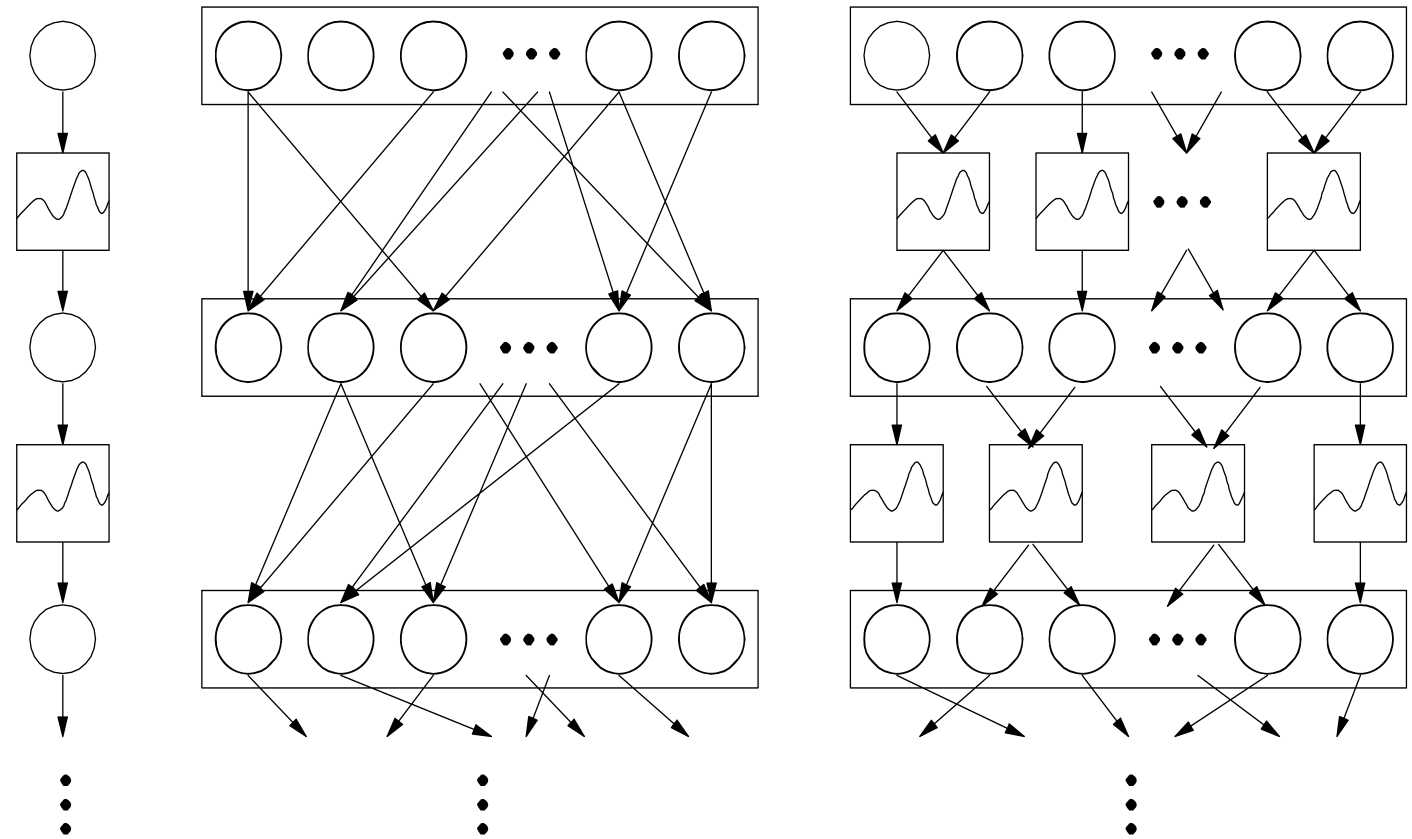
4. Select the individuals into the next generation with acceptance probability

$$a_g(A_i' \mid A_i) = \min\left\{1, \frac{P_g(A_i' \mid D)}{P_g(A_i \mid D)}\right\}$$

5. Revise the priors $P_g(A) = h(P_{g-1}(A), P_g(A \mid D))$.
   Set $g=g+1$ and go to step 2.

eMCMC: An Implementation of BEA

(Zhang & Cho, 2001)



(a) MCMC (b) EA (c) eMCMC

EC and PF: Similarities

- Many similarities between probabilistic evolutionary algorithms, especially EDA and BEA, on one hand and particle filters (PFs) on the other hand.
- Both EC and PF use population-based estimation for searching the parameter space.
- BEA and PF are both based on the Bayesian inference framework.
- As in PF, BEA generates samples using a proposal distribution explicitly.
- Like PF, BEA uses importance weights measured by an acceptance function.

EMC and PF: Differences

- The goal of EC is optimization via population-based estimation while the goal of PF is estimation (filtering) itself.
- BEA dynamically re-estimates the proposal distribution while PF uses a fixed proposal distribution given at the outset.
- BEA recombines multiple samples (crossover) as well as locally adapts a single sample (mutation) while PF performs local adaptation only (mutation).

Cf. Sampling Importance Resampling (SIR) = Sequential Monte Carlo = Particle Filter

1. Initialize $t \leftarrow 0$

- For $i = 1,\dots,N$: sample $x_t^{(i)} \sim p(x_0)$, $\quad t \leftarrow 1$.

2. Importance sampling

- For $i = 1,\dots,N$: sample $x_t^{(i)} \sim q(x_t \mid x_{t-1}^{(i)}, y_t) = p(x_t \mid x_{t-1}^{(i)})$

  Let $x_{0:t}^{(i)} \triangleq (x_{0:t-1}^{(i)}, x_t^{(i)})$

- For $i = 1,\dots,N$: compute weights $\quad w_t^{(i)} = p(y_t \mid x_t^{(i)})$

- Normalize the weights: $\tilde{w}_t^{(i)} = w_t^{(i)} \Big/ \sum_{j=1}^{N} w_t^{(j)}$

3. Resampling

- Resample with replacement N particles $x_{0:t}^{(i)}$ according to the importance weights $w_t^{(i)}$, resulting in $\{\tilde{x}_{0:t}^{(i)}, N^{-1}\}_{i=1}^{N}$.

- New particle population $\{x_{0:t}^{(i)}\}_{i=1}^{N} \leftarrow \{\tilde{x}_{0:t}^{(i)}\}_{i=1}^{N}$.

- Set $t \leftarrow t+1$ and go to step 2.

References

- PSO:
  - Kennedy, J. and Eberhart, R. C., Particle swarm optimization, Proc. of ICNN, 1995.
  - Eberhart, R. C. and Shi, Y., Particle swarm optimization: developments, applications and resources, Proc. of CEC-2001, 2001.
- EDA:
  - Mühlenbein, H. and Paass, G., From recombination of genes to the estimation of distributions I. Binary parameters, Proc. of PPSN-1996, 1996.
  - Larranaga, P. and Lozano, J. A., Estimation of distribution algorithms: A new tool for evolutionary computation, 2002.
  - Cho, D. Y., and Zhang, B..-T., Continuous estimation of distribution algorithms with probabilistic principal component analysis, Proc. of CEC-2001, 2001.
- Evolutionary Monte Carlo (BEA & eMCMC)
  - Zhang, B.-T.,, 2001. A Bayesian framework for evolutionary computation, Proc. of CEC-99, 1999.
  - Zhang, B.-T., Paass, G., Mühlenbein, H., Convergence properties of incremental Bayesian evolutionary algorithms with single Markov chains, Proc. of CEC-2000, 2000.
  - Zhang, B.-T., and Shin, S.-Y., Bayesian evolutionary optimization using Helmholtz machines, Proc. of PPSN VI, 2000
  - Zhang, B.-T. and Cho, D. Y., System identification using evolutionary Markov chain Monte Carlo, Journal of Systems Architecture, 2001.