Model-Based Evolutionary Algorithms
Part 1: Estimation of Distribution Algorithms

Dirk Thierens

Universiteit Utrecht
The Netherlands

What ?

Evolutionary Algorithms

- Population-based, stochastic search algorithms
- Exploitation: selection
- Exploration: mutation & crossover

Model-Based Evolutionary Algorithms

- Population-based, stochastic search algorithms
- Exploitation: selection
- Exploration:
  1. Learn a model from selected solutions
  2. Generate new solutions from the model (& population)

What ?

Probabilistic Model-Based Evolutionary Algorithms (MBEA)

- a.k.a. Estimation of Distribution Algorithms (EDAs)
- a.k.a. Probabilistic Model-Building Genetic Algorithms
- a.k.a. Iterated Density Estimation Evolutionary Algorithms

MBEA = Evolutionary Computing + Machine Learning

Note: model not necessarily probabilistic

Why ?

Goal: Black Box Optimization

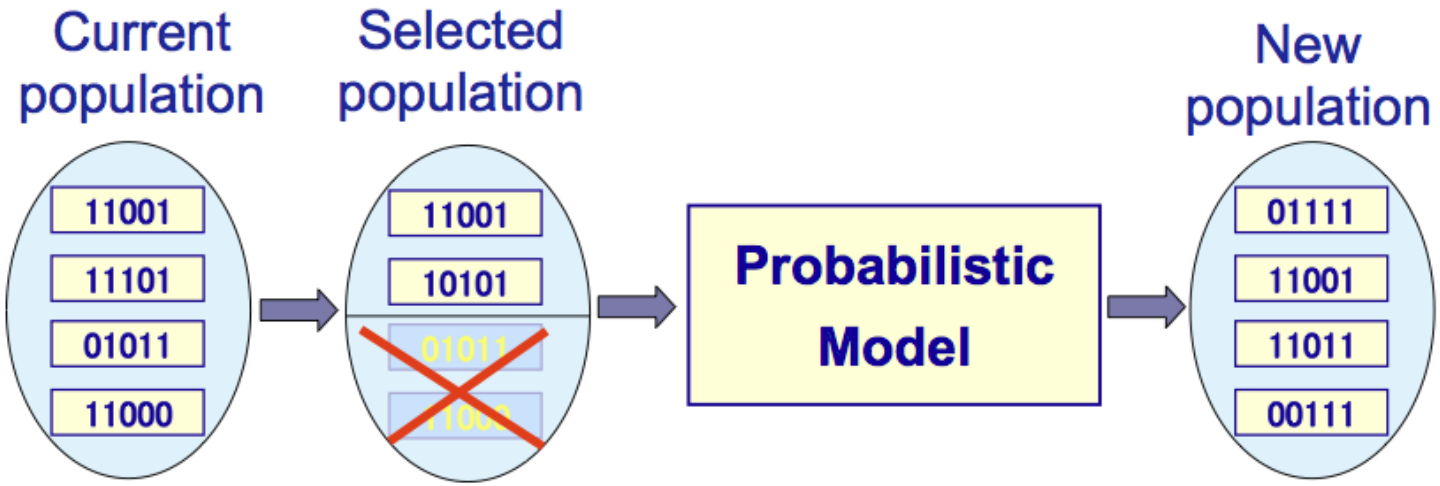
- Little known about the structure of the problem
- Clean separation optimizer from problem definition
- Easy and generally applicable

Approach

* Classical EAs: need suitable representation & variation operators
* Model-Based EAs: learn structure from good solutions

Discrete Representation

- Typically binary representation
- Higher order cardinality: similar approach



Probabilistic Model-Building Genetic Algorithm

Type of Models

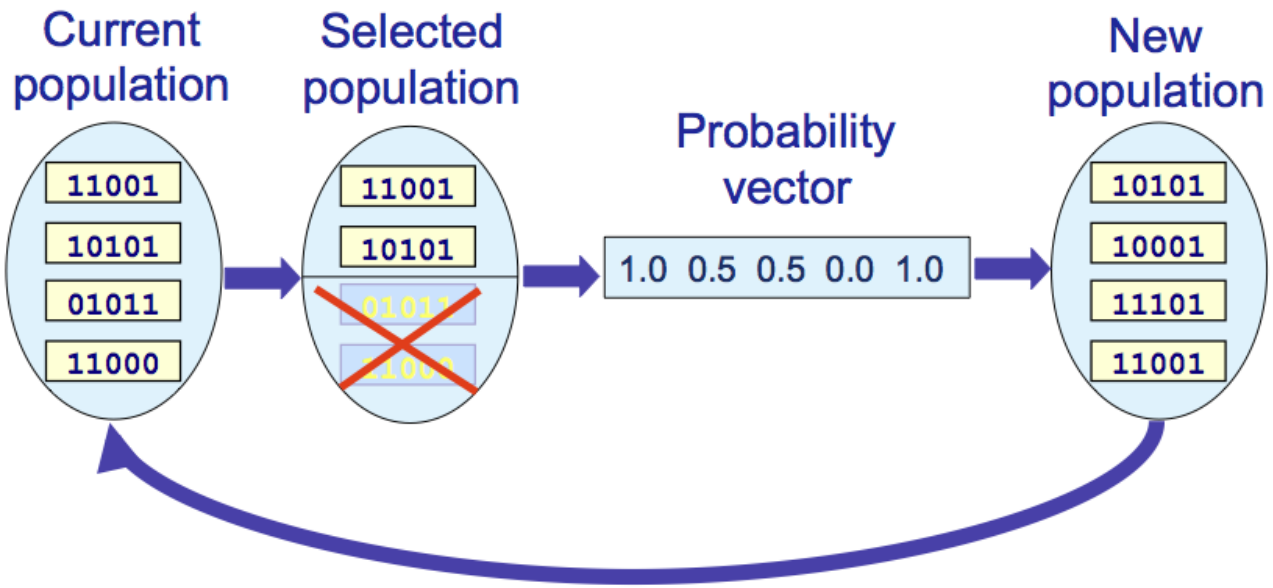
- Univariate: no statistical interaction between variables considered.
- Bivariate: pairwise dependencies learned.
- Multivariate: higher-order interactions modeled.

Univariate PMBGA

Model

* Model: probability vector $[p_1, \ldots, p_\ell]$ (ℓ: string length)
* p_i: probability of value 1 at string position i
* $p(X) = \prod_{i=1}^{\ell} p(x_i)$ ($p(x_i)$: univariate marginal distribution)

- Learn model: count proportions of 1 in selected population
- Sample model: generate new solutions with specified probabilities



Univariate PMBGA

Different Variants

- PBIL (Baluja; 1995)
  - Prob. vector incrementally updated over successive generations
- UMDA (Mühlenbein, Paass; 1996)
  - No incremental updates: example above
- Compact GA (Harik, Lobo, Goldberg; 1998)
  - Models steady-state GA with tournament selection
- DEUM (Shakya, McCall, Brown; 2004)
  - Uses Markov Random Field modeling

A hard problem for the univariate model

Data
000000
111111
010101
101010
000010
111000
010111
111000
000111
111111

Marginal Product model		
	$\hat{P}(X_0X_1X_2)$	$\hat{P}(X_3X_4X_5)$
000	0.3	0.3
001	0.0	0.0
010	0.2	0.2
011	0.0	0.0
100	0.0	0.0
101	0.1	0.1
110	0.0	0.0
111	0.4	0.4

Univariate model						
	$\hat{P}(X_0)$	$\hat{P}(X_1)$	$\hat{P}(X_2)$	$\hat{P}(X_3)$	$\hat{P}(X_4)$	$\hat{P}(X_5)$
0	0.5	0.4	0.5	0.5	0.4	0.5
1	0.5	0.6	0.5	0.5	0.6	0.5

- What is the probability of generating 111111?
- Univariate model: $0.5 \cdot 0.6 \cdot 0.5 \cdot 0.5 \cdot 0.6 \cdot 0.5 = 0.0225$
- MP model: $0.4 \cdot 0.4 = 0.16$ (7 times larger!)

Learning problem structure on the fly

- Without a "good" decomposition of the problem, important partial solutions (building blocks) are likely to get disrupted in variation.
- Disruption leads to inefficiency.
- Can we automatically configure the model structure favorably?
- Selection increases proportion of good building blocks and thus "correlations" between variables of these building blocks.
- So, learn which variables are "correlated".
- See the population (or selection) as a data set.
- Apply statistics / probability theory / probabilistic modeling.

Bivariate PMBGA

Model

- Need more than just probabilities of bit values
- Model pairwise interactions: conditional probabilities

- MIMIC (de Bonet, Isbell, Viola; 1996)
  - Dependency Chain
- COMIT (Baluja, Davies; 1997)
  - Dependency Tree
- BMDA (Pelikan , Mühlenbein; 1998)
  - Independent trees (forest)

Entropy

- Random variable X with probability distribution function $p(X)$
- Entropy $H(X)$ is a measure of uncertainty about a random variable X:

$$H(X) = \sum_{x \in X} -p(x) \log p(x)$$

- Conditional entropy $H(Y|X)$ is a measure of uncertainty remaining about Y after X is known (what X does not say about Y):

$$H(Y|X) = \sum_{x \in X} \sum_{y \in Y} p(x,y) \log \frac{p(x)}{p(x,y)}$$

Mutual information

- The mutual information $I(X, Y)$ of two random variables is a measure of the variables' mutual dependence.
- Mutual information is more general than the correlation coefficient (= linear relation between real-valued variables)
- Mutual information determines how similar the joint distribution $p(X, Y)$ is to the products of factored marginal distribution $p(X)p(Y)$:

$$I(X, Y) = \sum_{x \in X} \sum_{y \in Y} p(x, y) \log \frac{p(x, y)}{p(x)p(y)}$$

Mutual information and entropy

- Mutual information in relation to entropy:

$$\begin{aligned} I(X,Y) &= H(Y) - H(Y|X) \\ &= H(X) - H(X|Y) \\ &= H(X) + H(Y) - H(X,Y) \end{aligned}$$

- Mutual information can thus be seen as the amount of uncertainty in Y, minus the amount of uncertainty in Y which remains after X is known, which is equivalent to the amount of uncertainty in Y which is removed by knowing X
- Mutual information is the amount of information (that is, reduction in uncertainty) that knowing either variable provides about the other

Bivariate PMBGA

MIMIC

- Model: chain of pairwise dependencies.
- $p(X) = \prod_{i=1}^{\ell-1} p(x_{i+1}|x_i)p(x_1)$.
- MIMIC greedily searches for the optimal permutation of variables that minimizes Kullack-Leibler divergence.

Bivariate PMBGA

MIMIC

- Joint probability distribution over a set of random variables, $X = X_i$ is:
  $p(X) = p(X_1|X_2...X_n)p(X_2|X_3...X_n)...p(X_{n-1}|X_n)p(X_n)$
- Given only pairwise conditional probabilities, $p(X_i|X_j)$ and unconditional probabilities, $p(X_i)$, we want to approximate the true joint distribution as close as possible
- Given a permutation of numbers between 1 and n: $\pi = i_1 i_2 ... i_n$ define a class of probability distributions $p_\pi(X)$:

$$p_\pi(X) = p(X_{i_1}|X_{i_2})p(X_{i_2}|X_{i_3})...p(X_{i_{n-1}}|X_{i_n})p(X_{i_n})$$

Bivariate PMBGA

MIMIC

- Goal is to find a permutation π that maximizes the agreement between $p_\pi(X)$ and the true joint distribution $p(X)$
- Agreement between distributions can be measured by the Kullback-Leibler divergence:

$$\begin{aligned} D(p(X)||p_\pi(X)) &= \sum_{x \in X} p(x) \log \frac{p(x)}{p_\pi(x)} \\ &= -H(p) + H(X_{i_1}|X_{i_2}) + \ldots + H(X_{i_{n-1}}|X_{i_n}) + H(X_{i_n}) \end{aligned}$$

- The optimal permutation π minimizes the sum of the conditional entropies:

$$H(X_{i_1}|X_{i_2}) + \ldots + H(X_{i_{n-1}}|X_{i_n}) + H(X_{i_n})$$

Bivariate PMBGA

MIMIC: algorithm

1. $i_n = argmin_j \ H(X_j)$
2. $i_k = argmin_t \ H(X_t|X_{i_{k+1}})$, where $t \neq i_{k+1}...i_n$ and $k = n-1, n-2, ..., 2, 1$

Generating samples from the distribution:

1. Choose a value for X_{i_n} based on the probability $p(X_{i_n})$
2. for $k = n-1, n-2, ..., 2, 1$, choose an element X_{i_k} based on the conditional probability $p(X_{i_k}|X_{i_{k+1}})$

Both algorithms run in $O(n^2)$

Bivariate PMBGA

COMIT

- Optimal dependency tree instead of linear chain.
- Compute fully connected weighted graph between problem variables.
- Weights are the mutual information $I(X, Y)$ between the variables.
- $I(X, Y) = \sum_{y \in Y} \sum_{x \in X} p(x, y) \log \frac{p(x,y)}{p(x)p(y)}$.
- COMIT computes the maximum spanning tree of the weighted graph.

Bivariate PMBGA

COMIT

- The approximating probability model is restricted to factorizations in which the conditional probability distribution for any random variable depends on the value of at most one other random variable:

$$p(X) = \prod_{i=1}^{n} p(X_i | X_{parent(i)})$$

- $p(X)$ is the class of distributions with a tree as graphical model

Bivariate PMBGA

COMIT

- The optimal tree model (Chow and Liu, 1968):
  1. Create fully connected, weighted graph G
  2. Each vertex V_i corresponds to random variable X_i
  3. Each weight W_{ij} for the edge between V_i and V_j is equal to the mutual information $I(X_i, X_j)$ between X_i and X_j
  4. The edges in the maximum spanning tree of G determine an optimal set of $n - 1$ conditional probabilities with which to approximate the joint probability distribution.
  5. Note that all ordered trees conforming the unordered spanning tree model identical distributions.

Bivariate PMBGA

COMIT: algorithm

1. Calculate unconditional and conditional probabilities $p(X_i)$ and $p(X_i, X_j)$, and the mutual information $I(X_i, X_j)$.
2. Select arbitrary random variable X_r as root of the tree
3. Find the pair of variables X_{in} and X_{out}, where X_{in} is already in the tree and X_{out} is not, with the maximum mutual information $I(X_{in}, X_{out})$
4. Add X_{out} to the tree with X_{in} as parent, repeat until all variables are in the tree

Bivariate PMBGA

COMIT

- Algorithm runs in $O(n^2)$ (same as MIMIC)
- Because it is a variant of Prim's algorithm for finding maximum spanning trees the resulting tree maximizes the sum:

$$\sum_{i=1}^{n} I(X_i|X_{parent(i)})$$

- Therefore it minimizes the Kullback-Leibler divergence between the joint probability distribution and the second order approximation probability model (proof: Chow and Liu, 1968)

Bivariate PMBGA

BMDA

- BMDA also builds tree model.
- Model not necessarily fully connected: set of trees or forrest.
- Pairwise interactions measured by Pearson's chi-square statistics.

Multivariate PMBGA

Marginal Product Model

- Extended Compact GA (ECGA) (Harik; 1999) was first EDA going beyond pairwise dependencies.
- Greedily searches for the Marginal Product Model that minimizes the minimum description length (MDL).
- $p(X) = \prod_{g=1}^{G} p(X_g)$
- Choose the probability distribution with the lowest MDL score.
- Start from simplest model: the univariate factorization.
- Join two groups that result in the largest improvement in the used scoring measure.
- Stop when no joining of two groups improves the score further.

Multivariate PMBGA

Minimum Description Length (MDL)

- MDL is a measure of complexity (Information Theory).
- $MDL(M, D) = D_{Model} + D_{Data}$
  1. Model complexity D_{Model}: complexity of describing the model.
  2. Compressed population complexity D_{Data}: complexity of describing the data within the model (= measure of goodness of the probability distribution estimation).
- Best model = the one with the lowest MDL score.

Minimum Description Length score

MDL

- Model Complexity $D_{Model} = log_2(N+1)\sum_i(2^{S_i}-1)$
- Compressed Population Complexity $D_{Data} = N\sum_i H(M_i)$
- Combined Complexity = Model Complexity + Compressed Population Complexity

N : Population size
S_i : size of partition i
M_i : marginal distribution of the partition i
$H(M_i)$: entropy of the marginal distribution of the partition i

Multivariate PMBGA

Learning MP model

1. Start from univariate FOS:
$$\{\{0\},\{1\},\{2\},\ldots,\{l-2\},\{l-1\}\}$$
2. All possible pairs of partitions are temporarily merged:
$$\{\{0,1\},\{2\},\ldots,\{l-2\},\{l-1\}\}$$
$$\{\{0,2\},\{1\},\ldots,\{l-2\},\{l-1\}\}$$
$$\vdots$$
$$\{\{0\},\{1,2\},\ldots,\{l-2\},\{l-1\}\}$$
$$\vdots$$
$$\{\{0\},\{1\},\{2\},\ldots,\{l-2,l-1\}\}$$
3. Compute MDL score of each factorization.
4. Choose the best scoring factorization if better than current.
5. Repeat until no better scoring factorization is found.

Small example

population size $N = 8$, string length $l = 4$

1	0	0	0
1	1	0	1
0	1	1	1
1	1	0	0
0	0	1	0
0	1	1	1
1	0	0	0
1	0	0	1

- Marginal Product Model: $[I_1], [I_2], [I_3], [I_4]$

$[I_1]$		$[I_2]$		$[I_3]$		$[I_4]$	
1	5/8	1	4/8	1	3/8	1	4/8
0	3/8	0	4/8	0	5/8	0	4/8

- Marginal Product Model: $[I_1, I_3], [I_2], [I_4]$

$[I_1, I_3]$		$[I_2]$		$[I_4]$	
11	0/8	1	4/8	1	4/8
10	5/8	0	4/8	0	4/8
01	3/8				
00	0/8				

Entropy calculations:

1. Marginal Product Model: $[I_1], [I_2], [I_3], [I_4]$

$$Entropy([I_1]) = -(5/8)log_2(5/8) - (3/8)log_2(3/8) = 0.954$$

$$Entropy([I_2]) = -(4/8)log_2(4/8) - (4/8)log_2(4/8) = 1$$

$$Entropy([I_3]) = -(3/8)log_2(3/8) - (5/8)log_2(5/8) = 0.954$$

$$Entropy([I_4]) = -(4/8)log_2(4/8) - (4/8)log_2(4/8) = 1$$

2. Marginal Product Model: $[I_1, I_3], [I_2], [I_4]$

$$Entropy([I_1, I_3]) = -(5/8)log_2(5/8) - (3/8)log_2(3/8) = 0.954$$

$$Entropy([I_2]) = -(4/8)log_2(4/8) - (4/8)log_2(4/8) = 1$$

$$Entropy([I_4]) = -(4/8)log_2(4/8) - (4/8)log_2(4/8) = 1$$

- Marginal Product Model: $[I_1], [I_2], [I_3], [I_4]$
  Model Complexity = $log_2(9)(1 + 1 + 1 + 1) = 12.7$
  Compressed Population Complexity = 8 (0.945 + 1 + 0.954 + 1) = 31.3
  Combined Complexity = 12.7 + 31.3 = 44
- Marginal Product Model: $[I_1, I_3], [I_2], [I_4]$
  Model Complexity = $log_2(9)(3 + 1 + 1) = 15.8$
  Compressed Population Complexity = 8 (0.945 + 1 + 1) = 23.6
  Combined Complexity = 15.8 + 23.6 = 39.4

MPM	Combined Complexity
$[I_1, I_2][I_3][I_4]$	46.7
$[I_1, I_3][I_2][I_4]$	39.4
$[I_1, I_4][I_2][I_3]$	46.7
$[I_1][I_2, I_3][I_4]$	46.7
$[I_1][I_2, I_4][I_3]$	45.6
$[I_1][I_2][I_3, I_4]$	46.7

MPM	Combined Complexity
$[I_1, I_3, I_2][I_4]$	48.6
$[I_1, I_3, I_4][I_2]$	48.6
$[I_1, I_3][I_2, I_4]$	41.4

The Marginal Product Model: $[I_1, I_3], [I_2], [I_4]$ has the lowest combined complexity so it is the best model to compress the population and therefore captures the most dependencies in the set of solutions.

Example: Deceptive Trap Function

Building block length $k = 4$; Number of building blocks $m = 10$.

GA: uniform crossover, tournament selection ($s = 16$):

Population size	subfunctions solved	function evals.
100	3.9	740
500	5.2	5100
1000	6.1	15600
5000	6.8	100000
10000	7.3	248000
20000	8.0	614000
50000	7.9	1560000
100000	8.8	3790000

Example: Deceptive Trap Function

Extended Compact GA:

Population size	subfunctions solved	function evals.
100	4.0	750
200	5.2	1460
300	7.1	2610
500	9.3	4000
600	9.9	5040
1000	10.0	7300

Conclusion

Simple Genetic Algorithms are limited in their capability to mix or recombine non-linked building blocks

1. Design linkage into problem representation and recombination operator

or

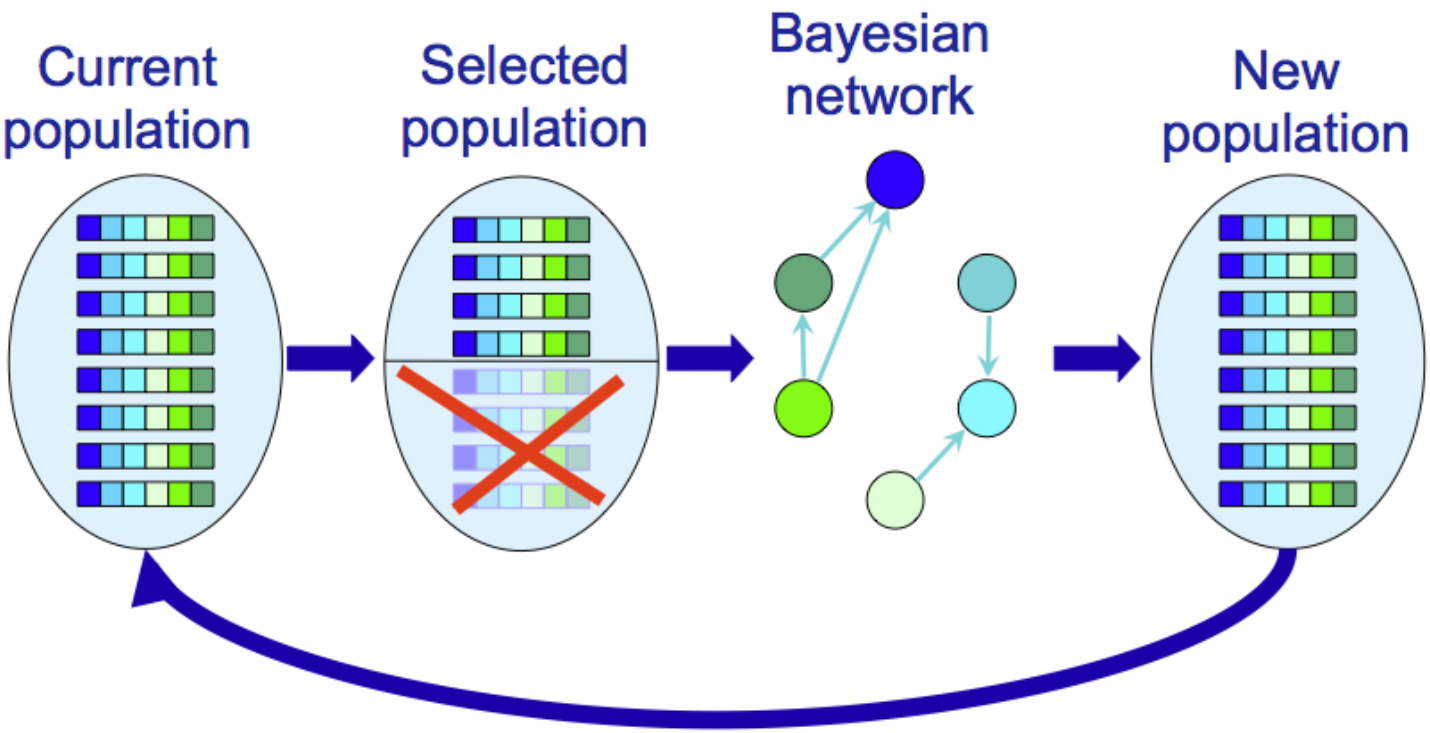
2. Learn linkage by using probabilistic model building genetic algorithm

Multivariate PMBGA

Bayesian Network

- Probability vector, dependency tree, and marginal product model are limited probability models.
- Bayesian network much more powerful model.
  - Acyclic directed graph.
  - Nodes are problem variables.
  - Edges represent conditional dependencies.

Multivariate PMBGA



Multivariate PMBGA

Bayesian network learning

- Similar to ECGA: scoring metric + greedy search
- Scoring metric: MDL or Bayesian measure
- Greedy search:
  - Initially, no variables are connected.
  - Greedily either add, remove, or reverse an edge between two variables.
  - Until local optimum is reached.

Multivariate PMBGA

Bayesian Network PMBGAs variants

- Bayesian Optimization Algorithm (BOA)
  (Pelikan, Goldberg, Cantú-Paz; 1998)
- Estimation of Distribution Networks Algorithm (EBNA)
  (Etxeberria, Larrañaga; 1999)
- Learning Factorized Distribution Algorithm (LFDA)
  (Mühlenbein, Mahnig, Rodriguez; 1999)

- Similarities: All use Bayesian Network as probability model.
- Dissimilarities: All use different method to learn BN.

Hierarchical BOA

- hBOA (Pelikan, Goldberg; 2001)
- Decomposition on multiple levels.
  - Bayesian network learning by BOA
- Compact representation.
  - Local Structures to represent conditional probabilities.
- Preservation of alternative solutions.
  - Niching with Restricted Tournament Replacement

Multivariate PMBGA

Markov Network

- Markov Netwok EDA
  (MN-EDA: Santana, 2005) (DEUM: Shakya & McCall, 2007).
- Probability model is undirected graph.
- Factorise the joint probability distribution in cliques of the undirected graph and sample it.
- Most recent version: Markovian Optimisation Algorithm (MOA) (Shakya & Santana, 2008).
- MOA does not explicitly factorise the distribution but uses the local Markov property and Gibbs sampling to generate new solutions.