Synergies Between Probabilistic Graphical Models and Evolutionary Computation

Pedro Larrañaga

Department of Computer Science and Artificial Intelligence
University of the Basque Country
http://www.sc.ehu.es/isg/

Mexican International Conference on Artificial Intelligence 2002 · MICAI2002 · Mérida · 25th April 2002
Outline

- Introduction
- Probabilistic Graphical Models
- Evolutionary Computation
- Genetic Algorithms Based Approaches to Combinatorial Optimization in Bayesian Networks
- Evolutionary Computation Based on Learning and Simulation of Bayesian Networks
- Summary
Double direction synergy

<table>
<thead>
<tr>
<th>Evolutionary Computation</th>
<th>Probabilistic Graphical Models</th>
</tr>
</thead>
<tbody>
<tr>
<td>genetic algorithms</td>
<td>Bayesian networks</td>
</tr>
<tr>
<td>estimation of distribution algorithms</td>
<td>Gaussian networks</td>
</tr>
</tbody>
</table>
Probabilistic Graphical Models

- Limitations of the rule based systems
- Human intuition some similarity with probability theory
- At the beginning of the AI very simple probabilistic expert systems
- Probabilistic reasoning in complex systems: Pearl (1988) and Lauritzen and Spiegelhalter (1988) solve the difficulties
- Probabilistic graphical models (Bayesian networks, Gaussian networks, ...) based on the conditional independence concept
• $X$ is conditionally independent of $Y$ given $Z$, $I(X,Y|Z)$, if and only if:
  • $p(x|y,z) = p(x|z)$ $\forall x, y, z$

• Representation used to encode uncertainty in expert systems

• In between these two opposite cases:
  • $p(\mathbf{x}) = p(x_1, x_2, \ldots, x_n)$ needs $2^n - 1$ parameters
  • $p(\mathbf{x}) = \prod_{i=1}^{n} p(x_i)$ needs $n$ parameters
\( X = (X_1, \ldots, X_n) \) a \( n \)-dimensional random variable

- \( x_i \) value of \( X_i \) (the \( i \)-th component of \( X \))
- \( y = (x_i)_{X_i \in Y} \) value of \( Y \subseteq X \)

- A probabilistic graphical model for \( X \) produces a factorization of the joint probability distribution \( \rho(x) \)
  - \( p(x) \) discrete case (Bayesian network)
  - \( f(x) \) continuous case (Gaussian network)

- Two components
  - Structure \( S \): directed acyclic graph (DAG) representing conditional independencies between triplets of variables of \( X \)
  - A set of local densities
Factorization of the generalized density function:

- $\mathcal{Pa}_i$: set of parents of $X_i$ in $S$

$$\rho(\mathbf{x}) = \rho(x_1, x_2, \ldots, x_n) = \prod_{i=1}^{n} \rho(x_i | \mathcal{Pa}_i^S)$$

- If the probability distribution depends on a finite set of parameters, then

$$\boldsymbol{\theta}_S = (\theta_1, \ldots, \theta_n)$$

$$\rho(\mathbf{x} | \boldsymbol{\theta}_S) = \prod_{i=1}^{n} \rho(x_i | \mathcal{Pa}_i^S, \theta_i)$$

- Probabilistic graphical model $M = (S, \boldsymbol{\theta}_S)$
Bayesian networks

- For all $i$, $X_i$ is discrete (multinomial)
- $x_i^1, \ldots, x_i^{r_i}$ denote the $r_i$ different values of $X_i$
- $p(x_i^k|pa_i^j, S_i, \theta_i) = \theta_{ijk}$ is the conditional probability that $X_i$ takes its $k^{th}$ value, given that its parents are in the $j^{th}$ value
- $q_i = \prod_{X_g \in Pa_i} r_g$ denotes the number of different instantiations of $Pa_i$
- Local parameters $\theta_i = (((\theta_{ijk})_{k=1}^{r_i})_{j=1}^{q_i})$

Synergies Between Probabilistic Graphical Models and Evolutionary Computation– p.8/46
Figure 1: Structure, local probabilities and resulting factorization for a Bayesian network with 4 variables ($X_1$, $X_3$ y $X_4$ with two different values, and $X_2$ with three possible values)
Different problems:

- **Structure learning**: how to obtain the Bayesian network structure from a dataset containing cases

- **Probabilistic reasoning**: to propagate some evidence through the network
  
  \[ p(S_i = s_i \mid X_0 = x_0) \]
  with \( X_0 \) observed variables
  
  Example: \( p(S_2 = s_2 \mid X_4 = x_4^1) \)

- **Abductive inference**: determination of the most probable state of the system, given the value of any subset of variables
  
  - Maximum a posteriori explanation (MAPE):
    
    \( X_U = X \setminus X_O \) unobserved variables
    
    \( x_U^* = \arg \max_{x_U} p(X_U = x_U \mid X_O = x_O) \)
    
    Example: \( x_U^* = \arg \max_{x_U} p(X_U = x_U \mid X_4 = x_4^1) \) with
    
    \( X_U = (X_1, X_2, X_3) \)
• Paradigms in Artificial Intelligent that imitate process in nature: neural networks, simulated annealing, tabu search, ant colonies, evolutionary computation, ...

• Evolutionary computation:
  • Adaptive methods used for solving problems (search and optimization)
  • Populations evolve in nature according to the principles of natural selection and survival of the fittest (Darwin 1859)
  • Genetic algorithms (Holland 1975), evolution strategies, evolutionary programming, genetic programming, ...
begin AGA
Generate initial population at random
WHILE NOT stop DO
BEGIN
Select parents from the population
Produce children by the selected parents
Mutate the individuals
Extend the population by adding children to it
Reduce the extended population
END
Return the best individual found
end AGA

Figure 2: The pseudo-code of an abstract genetic algorithm
Some problems with evolutionary computation paradigms:

- Several parameters’ values to be fixed
- Difficult to predict the evolution of the population
- Good blocks can be broken by the effect of crossover operators
- The relation between the variables (linkage learning) remain implicit
- Deceptive problems
Genetic Algorithms Based Approaches to Combinatorial Optimization in Bayesian Networks

- Structure learning from data
  - Space of orderings
  - Space of structures
- Triangulation of the moral graph
- Abductive inference
Structure learning from data: An alternative to model the expert’s knowledge (expensive, unreliable, time consuming ...) is to automatically induce the structure

structure learning $\equiv$ model induction $\equiv$ model search

- Detecting conditional independencies vs. score + search
- Assuming ordering between the variables vs. no ordering
- Trees or polytrees vs. multiply connected structures
- Space of orderings vs. space of structures (DAGs) vs. space of equivalence classes
Structure learning from data

- Scores:
  - Penalized maximum likelihood
  - Bayesian scores
  - Information theory based scores

- Search:
  - Greedy, simulated annealing, tabu search, ant colonies, genetic algorithms, .......
Structure learning from data. Bayesian score

Under some general assumptions Cooper and Herskovits (1992) obtained:

\[
p(S, D) = p(S) \prod_{i=1}^{n} g(X_i, Pa_i) = p(S) \prod_{i=1}^{n} \prod_{j=1}^{q_i} \frac{(r_i - 1)!}{(N_{ij} + r_i - 1)!} \prod_{k=1}^{r_i} N_{ijk}!
\]

- **S**: DAG that is being evaluating
- **D**: data set
- **n**: number of variables
- **r_i**: number of states \(X_i\) can have
- **q_i**: number of possible state combinations of \(Pa_i\)
- **N_{ijk}**: number of cases in \(D\) where \(X_i\) takes its \(k^{th}\) value and the parent set of \(X_i\) are on their \(j^{th}\) combination of values
- **N_{ij}**: \(\sum_{k=1}^{r_i} N_{ijk}\)
Structure learning from data. Searching in the space of orderings

The K2 algorithm (Cooper and Herskovits 1992)

- An ordering between the nodes is assumed
- An upper bound is set on the number of parents for any node
- For every node K2 searches for the set of parent nodes that maximizes:

\[ g(X_i, Pa_i) = \prod_{j=1}^{q_i} \frac{(r_i - 1)!}{(N_{ij} + r_i - 1)!} \prod_{k=1}^{r_i} N_{ijk}! \]

- K2 assumes initially that a node does not have parents
- Every step K2 adds incrementally the parent whose addition more increases \( g(X_i, Pa_i) \)
- K2 stops when adding a single parent to any node cannot increase \( g(X_i, Pa_i) \)
- K2 is a greedy algorithm
Structure learning from data. Searching in the space of orderings

Motivation: K2 algorithm needs an ordering between the variables

- Individuals in the genetic sense are total orderings between the variables
- Cardinality of the search space: $n!$
- The quality of a variable ordering is evaluated with K2
- Genetic operators developed for the TSP with a path representation
**Genetic Algorithms Based Approaches to Combinatorial Optimization in Bayesian Networks**

Structure learning from data. Searching in the space of orderings.

<table>
<thead>
<tr>
<th>Crossover operators</th>
<th>Mutation operators</th>
</tr>
</thead>
<tbody>
<tr>
<td>Partially-Mapped Crossover (Goldberg, Lingle 1985)</td>
<td>Displacement Mutation (Michalewicz 1992)</td>
</tr>
<tr>
<td>Cycle Crossover (Oliver et al. 1987)</td>
<td>Exchange Mutation (Banzhaf 1990)</td>
</tr>
<tr>
<td>Order Crossover (Davis 1985)</td>
<td>Insertion Mutation (Fogel 1990)</td>
</tr>
<tr>
<td>Order-Based Crossover (Syswerda 1991)</td>
<td>Simple-Inversion Mutation (Holland 1975)</td>
</tr>
<tr>
<td>Position-Based Crossover (Syswerda 1991)</td>
<td>Inversion Mutation (Fogel 1990)</td>
</tr>
<tr>
<td>Edge Recombination Crossover (Whitley et al. 1989)</td>
<td>Scramble Mutation (Syswerda 1991)</td>
</tr>
<tr>
<td>Voting Recombination (Mühlenbein 1989)</td>
<td></td>
</tr>
<tr>
<td>Alternating-Position Crossover (Larrañaga et al. 1994)</td>
<td></td>
</tr>
</tbody>
</table>

Structure learning from data. Searching in the space of structures

A Bayesian network structure can be represented by a $n \times n$ connectivity matrix $C$

$$c_{ij} = \begin{cases} 
1 & \text{if } X_j \text{ is a parent of } X_i \\
0 & \text{otherwise}
\end{cases}$$

$c_{11} c_{21} \cdots c_{n1} c_{12} c_{22} \cdots c_{n2} \cdots c_{1n} c_{2n} \cdots c_{nn}$

- **Assuming an ordering between the variables**
  - Connectivity matrices of the network structures are triangulated. The genetic operators are closed operators under DAG conditions

- **Without assuming an ordering between the variables**
  - The genetic operators are not closed operators regarding the DAG conditions. To ensure the closeness a repair operator is introduced
Structure learning from data. Searching in the space of structures. Without assuming an ordering:

- Structures of Figure 3 represented by: 001001000 and 000000110

![Diagram of structures](image)

Figure 3: The parents structures

- The crossover point is chosen in between the sixth and the seventh bit. The offspring strings are: 001001110 and 000001100 in Figure 4

![Diagram of crossover](image)

Figure 4: Crossover does not always result in legal structures
Structure learning from data. Searching in the space of structures. Without assuming an ordering

- The DAG of Figure 5 (a) is represented by the string 010001000
- Suppose that the seventh bit is modified due to mutation
- This gives the string 010001100, which corresponds with the cyclic graph of Figure 5 (b)

Figure 5: Mutation is not a closed operator
Structure learning from data. Searching in the space of structures. Without assuming an ordering

Triangulation of the moral graph

Propagation of the evidence: updating the probability distributions of the unobserved variables according to the newly available evidence

1. Exact propagation algorithms
   - Conditioning method (Pearl 1986)
   - Clustering method (Lauritzen and Spiegelhalter 1988)

2. Approximate propagation algorithms
   - Probabilistic logic sampling (Henrion 1988)
   - Markov sampling method (Pearl 1987)
Triangulation of the moral graph

The algorithm of Lauritzen and Spiegelhalter (1988):

1. Moralization of the DAG
2. Triangulation of the moral graph
3. Creation of the junction graph
4. Creation of the junction tree
5. Modification of the evidence propagation algorithm for trees to the junction tree

The only problematic step in the process from DAG to junction tree is the triangulation
Triangulation of the moral graph

- Moral graph: variables with a common child are linked, after which all directions on the arcs are deleted
- A graph is triangulated if any cycle of length greater than 3 has a chord
- Basic technique to triangulate a moral graph $G$ is through successive elimination of the nodes of $G$
- Eliminating node $v$ consists of adding edges to the graph in such a way that all nodes adjacent to $v$ become pairwise adjacent, and subsequently deleting $v$ and its adjacent edges
- The cliques –maximal complete subgraphs– obtained during the triangulation define a decomposition of the Bayesian network
Triangulation of the moral graph

- The graph triangulation problem is equivalent to the search for an *optimal vertex elimination sequence*

- The optimal vertex elimination sequence is the one that minimizes:

\[
    w(G^t_{\#}) = \log_2 \sum_C 2^{w(C)}
\]

- \(w(G^t_{\#})\) weight of the triangulated graph \(G^t_{\#}\) obtained by eliminating the vertices of \(G\) in the order defined by the ordering \(#\)

- \(w(C) = \sum_{i=1}^{k} w(v_i)\), weight of clique \(C\), and \(w(v_i) = \log_2 n_i\) weight of variable \(v_i\) as a function of the number of its states \(n_i\)
Genetic Algorithms Based Approaches to Combinatorial Optimization Problems in Bayesian Networks

Triangulation of the moral graph

Figure 6: (a) DAG, (b) moral graph. Nodes are eliminated in order: $v_1, v_5, v_3, v_4, v_2, v_6$. Let $n_i = i + 1$ ($i = 1, 2, \ldots, 6$), (c) eliminate $v_1$: $C_1 = \{v_1, v_2, v_3, v_4\}$, added edges: $\{v_2, v_3\}, \{v_3, v_4\}$, (d) eliminate $v_5$: $C_2 = \{v_4, v_5\}$, (e) eliminate $v_3$: $C_3 = \emptyset$, (f) eliminate $v_4$: $C_4 = \{v_2, v_4, v_6\}$, added edge: $\{v_2, v_6\}$, (g) eliminate $v_2$: $C_5 = \emptyset$, (h) eliminate $v_6$: $C_6 = \emptyset$. Total weight of the triangulated graph: $\log_2 (2^3 \cdot 3 \cdot 4 \cdot 5 + 5 \cdot 6 + 3 \cdot 5 \cdot 7) = \log_2 255$. 
Triangulation of the moral graph

- Approach based in genetic algorithms
- Each individual represents one elimination ordering
- Fitness function: $G^t_B$
- Same crossover and mutation operators as in the structure learning in the space of orderings
Abductive inference

- Determination of the most probable state of the system, given the value of any subset of variables
- Maximum a posteriori explanation (MAPE):

$$x^*_U = \arg \max_{x_U} p(x_U | X_O = x_O)$$

$$X_U = X \setminus X_O$$ unobserved variables

- Optimization problem. Cardinality of the search space: $$\prod_{i=1}^{\mid U \mid} r_i$$
  - $$\mid U \mid$$ the number of variables not instantiated
  - $$r_i$$ the number of discrete states for the $$i$$-th node

- First approach with genetic algorithms
New approach to Evolutionary Computation

- Population based
- There are neither crossover nor mutation operators
- Every generation the underlying probability distribution of the selected individuals is estimated
- From this probability distribution a new population of individuals is simulated
- Repeat the two previous steps until a stopping criterion is met

EDAs (Estimation of Distribution Algorithms) (Mühlenbein and Paail, 1996)
Illustrating EDAs with one example

$$\max h(x) = \sum_{i=1}^{6} x_i \text{ with } x_i = 0, 1$$

(a) $D_0$

$p_0(X_i = 1) = 0.5$ for $i = 1, \ldots, 6$
### Evolutionary Computation Based on Learning and Simulation of Bayesian Networks

<table>
<thead>
<tr>
<th></th>
<th>$X_1$</th>
<th>$X_2$</th>
<th>$X_3$</th>
<th>$X_4$</th>
<th>$X_5$</th>
<th>$X_6$</th>
<th>$h(x)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>11</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>12</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>13</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>14</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>15</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>16</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>17</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>18</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>19</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>20</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>3</td>
</tr>
</tbody>
</table>

Synergies Between Probabilistic Graphical Models and Evolutionary Computation– p.34/46
(b) $|D_{0}^{S_{e}}| = 10$ truncation

<table>
<thead>
<tr>
<th></th>
<th>$X_1$</th>
<th>$X_2$</th>
<th>$X_3$</th>
<th>$X_4$</th>
<th>$X_5$</th>
<th>$X_6$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>11</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>12</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>15</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>17</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>18</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>19</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Evolutionary Computation Based on Learning and Simulation of Bayesian Networks iv

(c) $p_1(\mathbf{x}) = p_1(x_1, \ldots, x_6) = \prod_{i=1}^{6} p(x_i | D_0^{Se})$

learned model

$\hat{p}(X_1 = 1 | D_0^{Se}) = 0.7$

$\hat{p}(X_2 = 1 | D_0^{Se}) = 0.7$

$\hat{p}(X_3 = 1 | D_0^{Se}) = 0.6$

$\hat{p}(X_4 = 1 | D_0^{Se}) = 0.6$

$\hat{p}(X_5 = 1 | D_0^{Se}) = 0.8$

$\hat{p}(X_6 = 1 | D_0^{Se}) = 0.7$
### Evolutionary Computation Based on Learning and Simulation of Bayesian Networks

#### (d) Sampling $p_1(\mathbf{x})$

<table>
<thead>
<tr>
<th></th>
<th>$X_1$</th>
<th>$X_2$</th>
<th>$X_3$</th>
<th>$X_4$</th>
<th>$X_5$</th>
<th>$X_6$</th>
<th>$h(\mathbf{x})$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>11</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>12</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>13</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>14</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>15</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>16</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>17</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>18</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>19</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>20</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>5</td>
</tr>
</tbody>
</table>

Synergies Between Probabilistic Graphical Models and Evolutionary Computation– p.37/46
Evolutionary Computation Based on Learning and Simulation of Bayesian Networks

(e) \(|D_1^{Se}| = 10\) truncation

<table>
<thead>
<tr>
<th></th>
<th>X₁</th>
<th>X₂</th>
<th>X₃</th>
<th>X₄</th>
<th>X₅</th>
<th>X₆</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>15</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>17</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>20</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
(f) Repeat

- Select $S_e$ individuals from $D_l$ obtaining $D_l^{S_e}$
- Learn the joint probability distribution of selected individuals

$$p_l(x) = \prod_{i=1}^{6} p(x_i|D_l^{S_e}_{i-1})$$
- Sample $p_l(x)$ obtaining $D_l$
### Evolutionary Computation Based on Learning and Simulation of Bayesian Networks

#### Synergies Between Probabilistic Graphical Models and Evolutionary Computation

#### Table 1: Data Distribution

<table>
<thead>
<tr>
<th></th>
<th>$X_1$</th>
<th>$X_2$</th>
<th>$X_3$</th>
<th>$X_4$</th>
<th>eval</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>13.25</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>32.45</td>
</tr>
<tr>
<td>N</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>34.12</td>
</tr>
</tbody>
</table>

#### Table 2: Selection of $S_e < N$ Individuals

<table>
<thead>
<tr>
<th></th>
<th>$X_1$</th>
<th>$X_2$</th>
<th>$X_3$</th>
<th>$X_4$</th>
<th>eval</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>32.25</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>33.78</td>
</tr>
<tr>
<td>Se</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>34.12</td>
</tr>
</tbody>
</table>

#### Figure 1: Induction of the Probability Model

The figure illustrates the process of selecting $S_e < N$ individuals from the initial data distribution $D_0$, using $D_L$ and $D_L^{S_e}$, and sampling from the probability model $\rho_l(x)$. The equations $\rho_l(x)$ and $\rho_l(x|D_L^{S_e})$ are also shown.
**EDA**

\[ D_0 \leftarrow \text{Generate } N \text{ individuals (the initial population) at random} \]

**Repeat** for \( l = 1, 2, \ldots \) until the stopping criterion is met

\[ D_{l-1}^{Se} \leftarrow \text{Select } Se \leq N \text{ individuals from } D_{l-1} \text{ according to a selection method} \]

\[ p_l(x) = p(x|D_{l-1}^{Se}) \leftarrow \text{Estimate the probability distribution of an individual being among the selected individuals} \]

\[ D_l \leftarrow \text{Sample } N \text{ individuals (the new population) from } p_l(x) \]
Depending on $p_t(x)$, EDAs can be classified as:

- **Without dependencies**
  - UMDA (Univariate Marginal Distribution Algorithm) (Mühlenbein, 1998)
    \[
    p_t(x) = \prod_{i=1}^{n} p_t(x_i)
    \]

- **Bivariate dependencies**
  - COMIT (Combining Optimizers with Mutual Information Trees) (Baluja and Davies, 1997)
    \[
    p_t(x) = \prod_{i=1}^{n} p_t(x_i | x_j(i))
    \]

- **Multiple dependencies**
  - EBNA (Estimation of Bayesian Network Algorithm) (Etxeberria and Larrañaga, 1999). Every generation a Bayesian network is learned
Probabilistic Logic Sampling (PLS) Henrion, 1988

Given an ancestral ordering, $\pi$, of the nodes
For $j = 1, 2, \ldots, N$
For $i = 1, 2, \ldots, n$

$x_{\pi(i)} \leftarrow \text{generate a value from } p(x_{\pi(i)} | pa_{\pi(i)})$

- In ancestral orderings variables are ordered in such a way that the values for $Pa_{\pi(i)}$ must be assigned before $X_{\pi(i)}$ is sampled
- The cases are generated one variable at each time in a forward way
Some applications of EDAs:

- **Optimization**
  - Travelling salesman problem
  - Knapsack problem
  - Job shop scheduling problem
  - Inexact graph matching

- **Machine learning**
  - Feature subset selection
  - Feature weighting for K-NN
  - Rule induction
  - Partial abductive inference in Bayesian networks
  - Partitional clustering
  - Adjusting weights in artificial neural networks
Evolutionary Computation Based on Learning and Simulation of Bayesian Networks xiii

References

Summary

- Cooperation between probabilistic graphical models (Bayesian networks) and evolutionary computation (genetic algorithms)
- Genetic algorithms solving combinatorial optimization problems related with Bayesian networks (structure learning, triangulation of the moral graph, abductive inference)
- Learning and simulation of Bayesian networks as the foundation of a new evolutionary computation paradigm (estimation of distribution algorithms)