A Preliminary Study on EDAs for Permutation Problems Based On Marginal Models

Josu Ceberio, Alexander Mendiburu, Jose A. Lozano

Intelligent Systems Group
Department of Computer Science and Artificial Intelligence
The University of the Basque Country

Genetic and Evolutionary Computation Conference (GECCO)
(Dublin, July 12-16, 2011)
Outline

1. Permutation-based Optimization Problems
2. Estimation of Distribution Algorithms
3. $K$-Order Marginals EDA
4. Experiments
5. Conclusions and Future Work
Outline

1. Permutation-based Optimization Problems
2. Estimation of Distribution Algorithms
3. K-Order Marginals EDA
4. Experiments
5. Conclusions and Future Work
A Preliminary Study on EDAs for Permutation Problems Based On Marginal Models
Permutation-based Optimization Problems

Travelling Salesman Problem

- Given a set of $n$ cities and the distances between them
- Find the shortest path that passes for each city once and comes back to the departure city

A solution:

$$\sigma = (1 \ 3 \ 4 \ 5 \ 6 \ 2)$$
A Preliminary Study on EDAs for Permutation Problems Based On Marginal Models
Permutation-based Optimization Problems

**Travelling Salesman Problem**

- Given a set of \( n \) cities and the distances between them.
- Find the shortest path that passes for each city once and comes back to the departure city.
- A solution:

\[
\sigma = (1 \ 3 \ 4 \ 5 \ 6 \ 2)
\]
Flow Shop Scheduling Problem

Definition

- It consists of scheduling \( n \) jobs on \( m \) machines
- A job consists of \( m \) operations and the \( j^{th} \) operation of each job must be processed on machine \( j \) for a specific time
- The goal of the optimization is to minimize the total flow time of processing all jobs
Flow Shop Scheduling Problem

**Definition**

- It consists of scheduling \( n \) jobs on \( m \) machines.
- A job consists of \( m \) operations and the \( j^{th} \) operation of each job must be processed on machine \( j \) for a specific time.
- The goal of the optimization is to minimize the total flow time of processing all jobs.

![Figure 1: Example of a solution for an instance of 5 jobs on 4 machines](image-url)
Flow Shop Scheduling Problem

Definition

- It consists of scheduling \( n \) jobs on \( m \) machines
- A job consists of \( m \) operations and the \( j^{th} \) operation of each job must be processed on machine \( j \) for a specific time
- The goal of the optimization is to minimize the total flow time of processing all jobs

Figure 1: Example of a solution for an instance of 5 jobs on 4 machines

A solution: \( \sigma = (1 \ 3 \ 2 \ 5 \ 4) \)
A Preliminary Study on EDAs for Permutation Problems Based On Marginal Models

Estimation of Distribution Algorithms

Outline

1. Permutation-based Optimization Problems
2. Estimation of Distribution Algorithms
3. $K$-Order Marginals EDA
4. Experiments
5. Conclusions and Future Work
Estimation of Distribution Algorithms (EDAs)

- Evolutionary Algorithm
- Similar to Genetic Algorithms
  - Learn a probability distribution from the selected individuals
  - The new population is built guided by the probabilistic model
Introduction

Estimation of Distribution Algorithms (EDAs)

- Evolutionary Algorithm
- Similar to Genetic Algorithms
- **Learn** a probability distribution from the selected individuals
- The new population is built guided by the probabilistic model
Introduction

Estimation of Distribution Algorithms (EDAs)

- Evolutionary Algorithm
- Similar to Genetic Algorithms
- **Learn** a probability distribution from the selected individuals
- The new population is built guided by the probabilistic model
EDAs designed for discrete domain problems

**Basics**

- **Algorithms:**
  - Univariate: UMDA, PBIL, …
  - Bivariate: MIMIC, COMIT, …
  - Multivariate: EBNA, BOA, …

- **Path representation**

- These algorithms learn a probability distribution over a set (of variables) \( \Omega = \Omega_1 \times \Omega_2 \times \ldots \times \Omega_n \) where \( \Omega_i = \{1, 2, \ldots, r_i\}, \quad r_i \in \mathbb{N} \quad i = 1, \ldots, n \)
EDAs designed for discrete domain problems

Basics

- **Algorithms:**
  - Univariate: UMDA, PBIL, ...
  - Bivariate: MIMIC, COMIT, ...
  - Multivariate: EBNA, BOA, ...

- **Path representation**

These algorithms learn a probability distribution over a set (of variables) \( \Omega = \Omega_1 \times \Omega_2 \times \ldots \times \Omega_n \) where \( \Omega_i = \{1, 2, \ldots, r_i\}, \quad r_i \in \mathbb{N} \quad i = 1, \ldots, n \)

The sampling of these models may not provide permutations
EDAs designed for real-valued domain

Basics

- Algorithms: EGNA, IDEA, UMDAc, MIMICc, ...
- Random Keys (Bean 1994)
- Given a real vector \((x_1, x_2, \ldots, x_n)\) of length \(n\), a permutation can be obtained by ranking the positions using the values \(x_i\), \((i = 1, \ldots, n)\)
- Example

\[(2.35, 3.42, 9.35, 0.32, 11.54, 10.42, 5.23, 4.2, 7.8)\]

the permutation obtained is \(\sigma = (2 \ 3 \ 7 \ 1 \ 9 \ 8 \ 5 \ 4 \ 6)\)
EDAs designed for real-valued domain

Basics

- Algorithms: EGNA, IDEA, UMDAc, MIMICc, …
- Random Keys (Bean 1994)
- Given a real vector \((x_1, x_2, \ldots, x_n)\) of length \(n\), a permutation can be obtained by ranking the positions using the values \(x_i\), \((i = 1, \ldots, n)\)
- Example

\[(2.35, 3.42, 9.35, 0.32, 11.54, 10.42, 5.23, 4.2, 7.8)\]

the permutation obtained is \(\sigma = (2 \ 3 \ 7 \ 1 \ 9 \ 8 \ 5 \ 4 \ 6)\)

Redundancy in the codification
Specific designs

EDAs for permutation problems

- ICE - (Bosman & Thierens)
- REDA - (Romero & Larrañaga)
- NHBSA and EHBSA - (Tsutsui et al.)
Outline

1. Permutation-based Optimization Problems
2. Estimation of Distribution Algorithms
3. K-Order Marginals EDA
4. Experiments
5. Conclusions and Future Work
At each step a matrix of $K$-order marginals is learnt.

Each entry of the probability matrix

$$P(\sigma_{i_1} = j_1, \ldots, \sigma_{i_k} = j_k)$$

is calculated from the number of times that the configuration $(\sigma_{i_1} = j_1, \ldots, \sigma_{i_k} = j_k)$ appears in the selected individuals.
### K-Order Marginals EDA

**Table 1: 2-order marginals matrix**

<table>
<thead>
<tr>
<th>Positions</th>
<th>(1,2)</th>
<th>(1,3)</th>
<th>(1,4)</th>
<th>(2,1)</th>
<th>(2,3)</th>
<th>(2,4)</th>
<th>(3,1)</th>
<th>(3,2)</th>
<th>(3,4)</th>
<th>(4,1)</th>
<th>(4,2)</th>
<th>(4,3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1,2)</td>
<td>0.20</td>
<td>0.05</td>
<td>0.10</td>
<td>0.05</td>
<td>0.10</td>
<td>0.05</td>
<td>0.05</td>
<td>0.10</td>
<td>0.05</td>
<td>0.05</td>
<td>0.10</td>
<td>0.10</td>
</tr>
<tr>
<td>(1,3)</td>
<td>0.05</td>
<td>0.20</td>
<td>0.10</td>
<td>0.10</td>
<td>0.05</td>
<td>0.05</td>
<td>0.05</td>
<td>0.10</td>
<td>0.10</td>
<td>0.10</td>
<td>0.05</td>
<td>0.05</td>
</tr>
<tr>
<td>(1,4)</td>
<td>0.10</td>
<td>0.10</td>
<td>0.15</td>
<td>0.05</td>
<td>0.05</td>
<td>0.10</td>
<td>0.05</td>
<td>0.10</td>
<td>0.05</td>
<td>0.10</td>
<td>0.05</td>
<td>0.10</td>
</tr>
<tr>
<td>(2,3)</td>
<td>0.05</td>
<td>0.05</td>
<td>0.05</td>
<td>0.10</td>
<td>0.15</td>
<td>0.15</td>
<td>0.10</td>
<td>0.10</td>
<td>0.05</td>
<td>0.05</td>
<td>0.05</td>
<td>0.10</td>
</tr>
<tr>
<td>(2,4)</td>
<td>0.05</td>
<td>0.05</td>
<td>0.05</td>
<td>0.10</td>
<td>0.15</td>
<td>0.15</td>
<td>0.10</td>
<td>0.05</td>
<td>0.10</td>
<td>0.05</td>
<td>0.10</td>
<td>0.05</td>
</tr>
<tr>
<td>(3,4)</td>
<td>0.05</td>
<td>0.10</td>
<td>0.10</td>
<td>0.10</td>
<td>0.05</td>
<td>0.05</td>
<td>0.10</td>
<td>0.15</td>
<td>0.10</td>
<td>0.05</td>
<td>0.10</td>
<td>0.10</td>
</tr>
</tbody>
</table>
K-Order Marginals Sampling

Sampling

- The individual is initialized as empty $S = (-, -, -, -)$
- The sampling process is done by sampling a position of the individual at each step in the $M_i$ matrix $i = 1, \ldots, k$
A Preliminary Study on EDAs for Permutation Problems Based On Marginal Models

K-Order Marginals EDA

K-Order Marginals Sampling

Sampling
- The individual is initialized as empty $S = (−, −, −, −)$
- The sampling process is done by sampling a position of the individual at each step in the $M_i$ matrix $i = 1, \ldots, k$

Example - Step 1
- Randomly obtained position is 2
- $M_1$ marginals matrix:

<table>
<thead>
<tr>
<th>Positions</th>
<th>Index Combinations</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>0.41</td>
<td>0.16</td>
<td>0.16</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>0.09</td>
<td>0.50</td>
<td>0.25</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3</td>
<td>0.25</td>
<td>0.16</td>
<td>0.33</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
<td>0.25</td>
<td>0.16</td>
<td>0.25</td>
</tr>
</tbody>
</table>

- Sampled index is 3
K-Order Marginals Sampling

Example - Step 2

- Partially sampled individual is $S = (-, 3, -, -)$
- Randomly obtained combination of positions is $(2, 3)$
- $M_2$ marginals matrix:

<table>
<thead>
<tr>
<th>Positions</th>
<th>(1,2)</th>
<th>(1,3)</th>
<th>(1,4)</th>
<th>(2,1)</th>
<th>(2,3)</th>
<th>(2,4)</th>
<th>(3,1)</th>
<th>(3,2)</th>
<th>(3,4)</th>
<th>(4,1)</th>
<th>(4,2)</th>
<th>(4,3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1,2)</td>
<td>0.20</td>
<td>0.05</td>
<td>0.10</td>
<td>0.05</td>
<td>0.10</td>
<td>0.05</td>
<td>0.10</td>
<td>0.05</td>
<td>0.10</td>
<td>0.05</td>
<td>0.10</td>
<td>0.05</td>
</tr>
<tr>
<td>(1,3)</td>
<td>0.05</td>
<td>0.20</td>
<td>0.10</td>
<td>0.10</td>
<td>0.05</td>
<td>0.05</td>
<td>0.05</td>
<td>0.05</td>
<td>0.10</td>
<td>0.10</td>
<td>0.10</td>
<td>0.05</td>
</tr>
<tr>
<td>(1,4)</td>
<td>0.10</td>
<td>0.10</td>
<td>0.15</td>
<td>0.05</td>
<td>0.05</td>
<td>0.10</td>
<td>0.05</td>
<td>0.05</td>
<td>0.05</td>
<td>0.05</td>
<td>0.05</td>
<td>0.10</td>
</tr>
<tr>
<td>(2,3)</td>
<td>0.05</td>
<td>0.05</td>
<td>0.05</td>
<td>0.10</td>
<td>0.15</td>
<td>0.15</td>
<td>0.40</td>
<td>0.40</td>
<td>0.20</td>
<td>0.05</td>
<td>0.05</td>
<td>0.10</td>
</tr>
<tr>
<td>(2,4)</td>
<td>0.05</td>
<td>0.05</td>
<td>0.05</td>
<td>0.10</td>
<td>0.15</td>
<td>0.15</td>
<td>0.10</td>
<td>0.05</td>
<td>0.10</td>
<td>0.05</td>
<td>0.10</td>
<td>0.05</td>
</tr>
<tr>
<td>(3,4)</td>
<td>0.05</td>
<td>0.10</td>
<td>0.10</td>
<td>0.10</td>
<td>0.05</td>
<td>0.05</td>
<td>0.05</td>
<td>0.10</td>
<td>0.15</td>
<td>0.10</td>
<td>0.05</td>
<td>0.10</td>
</tr>
</tbody>
</table>

- Sampled indexes combination is $(3, 2)$
### Example - Step 3

- Partially sampled individual is $S = (-, 3, 2, -)$
- Randomly obtained combination of positions is $(3, 4)$
- $M_2$ marginals matrix:

<table>
<thead>
<tr>
<th></th>
<th>(1,2)</th>
<th>(1,3)</th>
<th>(1,4)</th>
<th>(2,1)</th>
<th>(2,3)</th>
<th>(2,4)</th>
<th>(3,1)</th>
<th>(3,2)</th>
<th>(3,4)</th>
<th>(4,1)</th>
<th>(4,2)</th>
<th>(4,3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positions</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(1,2)</td>
<td>0.20</td>
<td>0.05</td>
<td>0.10</td>
<td>0.05</td>
<td>0.10</td>
<td>0.05</td>
<td>0.05</td>
<td>0.10</td>
<td>0.05</td>
<td>0.05</td>
<td>0.10</td>
<td>0.10</td>
</tr>
<tr>
<td>(1,3)</td>
<td>0.05</td>
<td>0.20</td>
<td>0.10</td>
<td>0.10</td>
<td>0.05</td>
<td>0.05</td>
<td>0.05</td>
<td>0.05</td>
<td>0.10</td>
<td>0.10</td>
<td>0.10</td>
<td>0.05</td>
</tr>
<tr>
<td>(1,4)</td>
<td>0.10</td>
<td>0.10</td>
<td>0.15</td>
<td>0.05</td>
<td>0.05</td>
<td>0.10</td>
<td>0.10</td>
<td>0.05</td>
<td>0.05</td>
<td>0.05</td>
<td>0.05</td>
<td>0.10</td>
</tr>
<tr>
<td>(2,3)</td>
<td>0.05</td>
<td>0.05</td>
<td>0.05</td>
<td>0.10</td>
<td>0.15</td>
<td>0.15</td>
<td>0.10</td>
<td>0.10</td>
<td>0.05</td>
<td>0.05</td>
<td>0.05</td>
<td>0.10</td>
</tr>
<tr>
<td>(2,4)</td>
<td>0.05</td>
<td>0.05</td>
<td>0.05</td>
<td>0.10</td>
<td>0.15</td>
<td>0.15</td>
<td>0.10</td>
<td>0.05</td>
<td>0.10</td>
<td>0.05</td>
<td>0.10</td>
<td>0.05</td>
</tr>
<tr>
<td>(3,4)</td>
<td>0.05</td>
<td>0.10</td>
<td>0.10</td>
<td>0.66</td>
<td>0.05</td>
<td>0.33</td>
<td>0.05</td>
<td>0.10</td>
<td>0.15</td>
<td>0.10</td>
<td>0.05</td>
<td>0.10</td>
</tr>
</tbody>
</table>

- Sampled indexes combination is $(2, 1)$
Example - Step 4

- Partially sampled individual is $S = (-, 3, 2, 1)$
- Remaining index is placed in position 1
- The new individual is $S = (4, 3, 2, 1)$
Outline

1. Permutation-based Optimization Problems
2. Estimation of Distribution Algorithms
3. K-Order Marginals EDA
4. Experiments
5. Conclusions and Future Work
Experiments

Instances

- TSP (Grostel 17)
- FSSP (Taillard 20 jobs 10 machines)

Execution Parameter Set

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$K$-order</td>
<td>${1, 2, 3}$</td>
</tr>
<tr>
<td>Population size range</td>
<td>${10n, 20n, 40n, 80n, 160n, 320n, 640n}$</td>
</tr>
<tr>
<td>Selection size</td>
<td>Population size / 2</td>
</tr>
<tr>
<td>Offspring size</td>
<td>Population size - 1</td>
</tr>
<tr>
<td>Selection type</td>
<td>Ranking selection method</td>
</tr>
<tr>
<td>Elitism selection method</td>
<td>The best individual of the previous generation is guaranteed to survive</td>
</tr>
<tr>
<td>Stopping criterion</td>
<td>A maximum number of generations: $100n$</td>
</tr>
</tbody>
</table>
Figure 2: $K$-order marginals EDA solving Grostel 17 TSP instance.
Table 2: Error rate of $K$-order marginals EDA for Grostel 17 TSP instance.

<table>
<thead>
<tr>
<th>Pop. Size</th>
<th>$k = 1$</th>
<th></th>
<th>$k = 2$</th>
<th></th>
<th>$k = 3$</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mean</td>
<td>Dev</td>
<td>Mean</td>
<td>Dev</td>
<td>Mean</td>
<td>Dev</td>
</tr>
<tr>
<td>170</td>
<td>0.0132</td>
<td>0.0096</td>
<td>0.2119</td>
<td>0.0370</td>
<td>0.2675</td>
<td>0.0236</td>
</tr>
<tr>
<td>340</td>
<td>0.0226</td>
<td>0.0094</td>
<td>0.0604</td>
<td>0.0634</td>
<td>0.2247</td>
<td>0.0263</td>
</tr>
<tr>
<td>680</td>
<td>0.0211</td>
<td>0.0094</td>
<td>0.0259</td>
<td>0.0097</td>
<td>0.1880</td>
<td>0.0219</td>
</tr>
<tr>
<td>1360</td>
<td>0.0205</td>
<td>0.0076</td>
<td>0.0146</td>
<td>0.0066</td>
<td>0.1519</td>
<td>0.0231</td>
</tr>
<tr>
<td>2720</td>
<td>0.0209</td>
<td>0.0115</td>
<td>0.0090</td>
<td>0.0093</td>
<td>0.0163</td>
<td>0.0114</td>
</tr>
<tr>
<td>5440</td>
<td>0.0147</td>
<td>0.0061</td>
<td>0.0083</td>
<td>0.0071</td>
<td>0.0038</td>
<td>0.0026</td>
</tr>
<tr>
<td>10880</td>
<td>0.0083</td>
<td>0.0083</td>
<td>0.0024</td>
<td>0.0000</td>
<td>0.0024</td>
<td>0.0000</td>
</tr>
</tbody>
</table>
Figure 3: $K$-order marginals EDA solving Taillard 20-10 FSSP instance.
Experiments

Table 3: Error rate of $K$-order marginals EDA for Taillard 20-10 FSSP.

<table>
<thead>
<tr>
<th>Pop. Size</th>
<th>$k = 1$</th>
<th></th>
<th>$k = 2$</th>
<th></th>
<th>$k = 3$</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mean</td>
<td>Dev</td>
<td>Mean</td>
<td>Dev</td>
<td>Mean</td>
<td>Dev</td>
</tr>
<tr>
<td>200</td>
<td>0.0101</td>
<td>0.0063</td>
<td>0.0454</td>
<td>0.0060</td>
<td>0.0793</td>
<td>0.0070</td>
</tr>
<tr>
<td>400</td>
<td>0.0085</td>
<td>0.0056</td>
<td>0.0162</td>
<td>0.0079</td>
<td>0.0686</td>
<td>0.0090</td>
</tr>
<tr>
<td>800</td>
<td>0.0034</td>
<td>0.0010</td>
<td>0.0147</td>
<td>0.0066</td>
<td>0.0640</td>
<td>0.0033</td>
</tr>
<tr>
<td>1600</td>
<td>0.0054</td>
<td>0.0026</td>
<td>0.0050</td>
<td>0.0022</td>
<td>0.0385</td>
<td>0.0052</td>
</tr>
<tr>
<td>3200</td>
<td>0.0035</td>
<td>0.0009</td>
<td>0.0042</td>
<td>0.0018</td>
<td>0.0123</td>
<td>0.0073</td>
</tr>
<tr>
<td>6400</td>
<td>0.0030</td>
<td>0.0008</td>
<td>0.0032</td>
<td>0.0008</td>
<td>0.0077</td>
<td>0.0036</td>
</tr>
<tr>
<td>12800</td>
<td>0.0030</td>
<td>0.0007</td>
<td>0.0027</td>
<td>0.0005</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
Outline

1. Permutation-based Optimization Problems
2. Estimation of Distribution Algorithms
3. K-Order Marginals EDA
4. Experiments
5. Conclusions and Future Work
Conclusions

**K-order Marginals EDA**

- Higher order models are slightly better, however higher populations are needed.
- Computational requirements are huge for high order marginals.
Conclusions

**$K$-order Marginals EDA**
- Higher order models are slightly better, however higher populations are needed
- Computational requirements are huge for high order marginals

**Existing approaches**
- Ad-hoc solutions
- Still far from best known solutions

---

Conclusions

**K-order Marginals EDA**
- Higher order models are slightly better, however higher populations are needed
- Computational requirements are huge for high order marginals

**Existing approaches**
- Ad-hoc solutions
- Still far from best known solutions\(^a\)


We need specific probabilistic models developed for permutation spaces!!
The Mallows Model

Distance-based exponential probability model

\[ P(\sigma) = \frac{1}{Z(\theta)} e^{-\theta d(\sigma, \sigma_0)} \]
The Mallows Model

Much better results are obtained with The Mallows Model\(^1\)

A Preliminary Study on EDAs for Permutation Problems Based On Marginal Models

Josu Ceberio, Alexander Mendiburu, Jose A. Lozano

Intelligent Systems Group
Department of Computer Science and Artificial Intelligence
The University of the Basque Country

Genetic and Evolutionary Computation Conference (GECCO)
(Dublin, July 12-16, 2011)