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

Josu Ceberio, Alexander Mendiburu, Jose A. Lozano

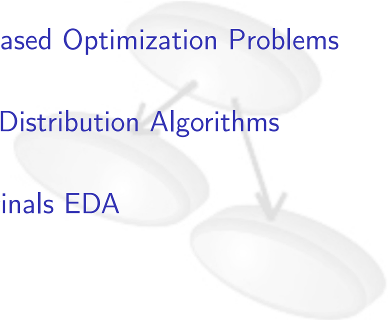
Intelligent Systems Group
Department of Computer Science and Artificial Intelligence
The University of the Basque Country

Intelligent Systems Group

UPV - EHU

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

Outline

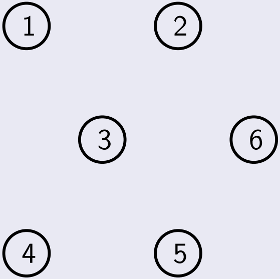


1. Permutation-based Optimization Problems
2. Estimation of Distribution Algorithms
3. *K*-Order Marginals EDA
4. Experiments
5. Conclusions and Future Work

Outline

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

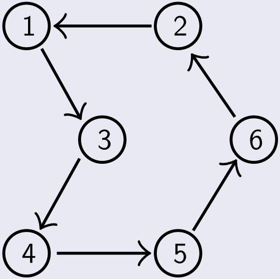
Travelling Salesman Problem



- 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:

$$\sigma = (1\ 3\ 4\ 5\ 6\ 2)$$

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

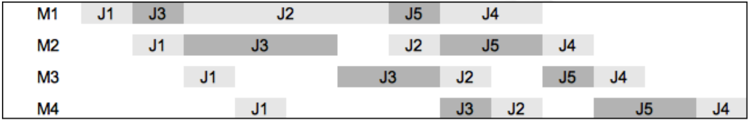


Figure 1: Example of a solution for an instance of 5 jobs on 4 machines

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

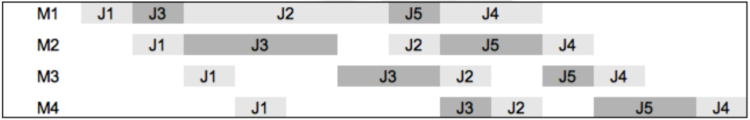
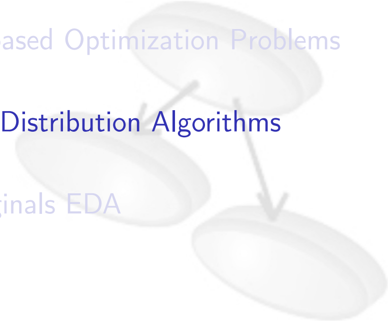


Figure 1: Example of a solution for an instance of 5 jobs on 4 machines

A solution: σ=(1 3 2 5 4)

Outline



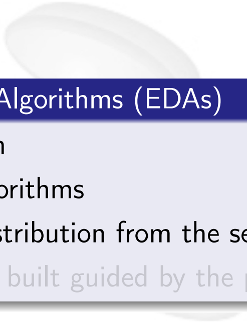
1. Permutation-based Optimization Problems
2. Estimation of Distribution Algorithms
3. *K*-Order Marginals EDA
4. Experiments
5. Conclusions and Future Work

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

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 $\Omega_i = \{1, 2, \ldots, r_i\}, \;\; r_i \in \mathbb{N} \;\; i = 1, \ldots, n$

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 $\Omega_i = \{1, 2, \ldots, r_i\}, \;\; r_i \in \mathbb{N} \;\; i = 1, \ldots, n$

The sampling of these models may not provide permutations

EDAs designed for real-valued domain

Basics

- Algorithms: EGNA, IDEA, UMDA*c*, MIMIC*c*, ...
- 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

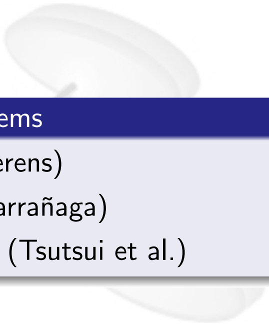
- Algorithms: EGNA, IDEA, UMDA*c*, MIMIC*c*, . . .
- Random Keys (Bean 1994)
- Given a real vector $(x_1, x_2, \ldots, x_n)$ of length n, a permutation can be obtained by ranking the positions using the values x_i, $(i = 1, \ldots, n)$
- Example

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

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

Redundancy in the codification

Specific designs

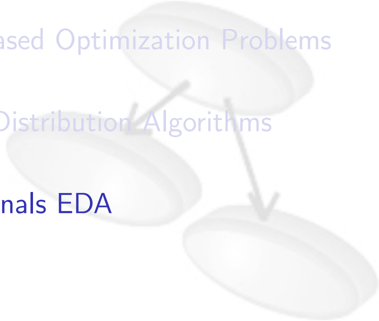


EDAs for pemutation problems

- 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



K-Order Marginals EDA

Learning

- 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

		Index Combinations											
		(1,2)	(1,3)	(1,4)	(2,1)	(2,3)	(2,4)	(3,1)	(3,2)	(3,4)	(4,1)	(4,2)	(4,3)
Positions	(1,2)	0.20	0.05	0.10	0.05	0.10	0.05	0.05	0.10	0.05	0.05	0.10	0.10
	(1,3)	0.05	0.20	0.10	0.10	0.05	0.05	0.05	0.05	0.10	0.10	0.10	0.05
	(1,4)	0.10	0.10	0.15	0.05	0.05	0.10	0.10	0.05	0.05	0.10	0.05	0.10
	(2,3)	0.05	0.05	0.05	0.10	0.15	0.15	0.10	0.10	0.05	0.05	0.05	0.10
	(2,4)	0.05	0.05	0.05	0.10	0.15	0.15	0.10	0.05	0.10	0.05	0.10	0.05
	(3,4)	0.05	0.10	0.10	0.10	0.05	0.05	0.05	0.10	0.15	0.10	0.05	0.10

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 $i = 1, \ldots, k$

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:

		Index Combinations			
		1	*2*	*3*	*4*
Positions	*1*	0.41	0.16	0.16	0.25
	2	0.09	0.50	0.25	0.16
	3	0.25	0.16	0.33	0.25
	4	0.25	0.16	0.25	0.33

- 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:

		Index Combinations											
		(1,2)	(1,3)	(1,4)	(2,1)	(2,3)	(2,4)	***(3,1)***	***(3,2)***	***(3,4)***	(4,1)	(4,2)	(4,3)
Positions	*(1,2)*	0.20	0.05	0.10	0.05	0.10	0.05	0.05	0.10	0.05	0.05	0.10	0.10
	(1,3)	0.05	0.20	0.10	0.10	0.05	0.05	0.05	0.05	0.10	0.10	0.10	0.05
	(1,4)	0.10	0.10	0.15	0.05	0.05	0.10	0.10	0.05	0.05	0.10	0.05	0.10
	(2,3)	0.05	0.05	0.05	0.10	0.15	0.15	0.40	0.40	0.20	0.05	0.05	0.10
	(2,4)	0.05	0.05	0.05	0.10	0.15	0.15	0.10	0.05	0.10	0.05	0.10	0.05
	(3,4)	0.05	0.10	0.10	0.10	0.05	0.05	0.05	0.10	0.15	0.10	0.05	0.10

- Sampled indexes combination is $(3, 2)$

K-Order Marginals Sampling

Example - Step 3

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

		Index Combinations											
		(1,2)	(1,3)	(1,4)	***(2,1)***	*(2,3)*	***(2,4)***	(3,1)	(3,2)	(3,4)	(4,1)	(4,2)	(4,3)
Positions	(1,2)	0.20	0.05	0.10	0.05	0.10	0.05	0.05	0.10	0.05	0.05	0.10	0.10
	(1,3)	0.05	0.20	0.10	0.10	0.05	0.05	0.05	0.05	0.10	0.10	0.10	0.05
	(1,4)	0.10	0.10	0.15	0.05	0.05	0.10	0.10	0.05	0.05	0.10	0.05	0.10
	(2,3)	0.05	0.05	0.05	0.10	0.15	0.15	0.10	0.10	0.05	0.05	0.05	0.10
	(2,4)	0.05	0.05	0.05	0.10	0.15	0.15	0.10	0.05	0.10	0.05	0.10	0.05
	(3,4)	0.05	0.10	0.10	0.66	0.05	0.33	0.05	0.10	0.15	0.10	0.05	0.10

- Sampled indexes combination is $(2, 1)$

K-Order Marginals Sampling

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

Parameter	Value
K-order	$\{1, 2, 3\}$
Population size range	$\{10n, 20n, 40n, 80n, 160n, 320n, 640n\}$
Selection size	Population size / 2
Offspring size	Population size - 1
Selection type	Ranking selection method
Elitism selection method	The best individual of the previous generation is guaranteed to survive
Stopping criterion	A maximum number of generations: $100n$

Experiments

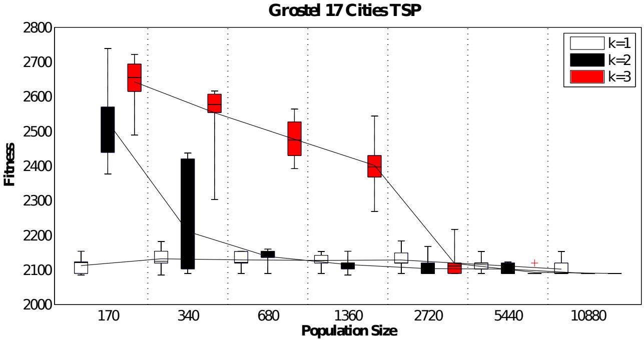


Figure 2: *K*-order marginals EDA solving Grostel 17 TSP instance.

Experiments

Table 2: Error rate of K-order marginals EDA for Grostel 17 TSP instance.

Pop. Size	$k = 1$		$k = 2$		$k = 3$	
	Mean	*Dev*	*Mean*	*Dev*	*Mean*	*Dev*
170	0.0132	0.0096	0.2119	0.0370	0.2675	0.0236
340	0.0226	0.0094	0.0604	0.0634	0.2247	0.0263
680	0.0211	0.0094	0.0259	0.0097	0.1880	0.0219
1360	0.0205	0.0076	0.0146	0.0066	0.1519	0.0231
2720	0.0209	0.0115	0.0090	0.0093	0.0163	0.0114
5440	0.0147	0.0061	0.0083	0.0071	0.0038	0.0026
10880	0.0083	0.0083	**0.0024**	0.0000	**0.0024**	0.0000

Experiments

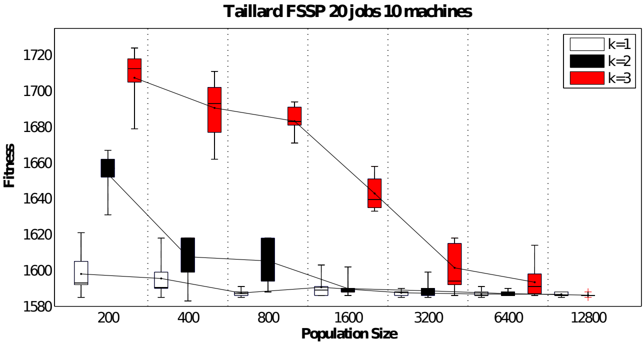


Figure 3: *K*-order marginals EDA solving Taillard 20-10 FSSP instance.

Experiments

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

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

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[a]

[a] J. Ceberio, E. Irurozki, A. Mendiburu, J.A. Lozano. **A review on Estimation of Distribution Algorithms for Permutation-based Combinatorial Optimization Problems**. *Progress in Artificial Intelligence*. 2011.

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]

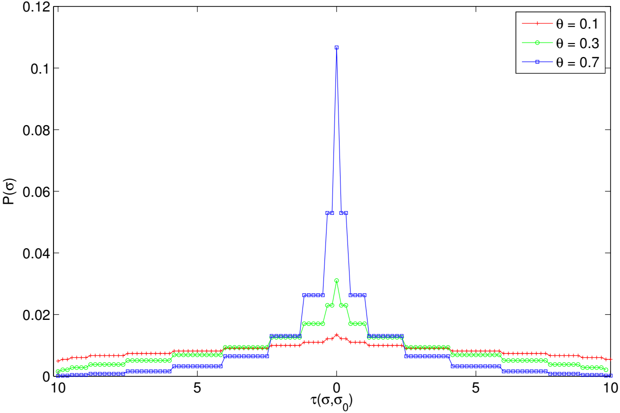
[a] J. Ceberio, E. Irurozki, A. Mendiburu, J.A. Lozano. **A review on Estimation of Distribution Algorithms for Permutation-based Combinatorial Optimization Problems**. *Progress in Artificial Intelligence*. 2011.

We need specific probabilistic models developed for permutation spaces!!

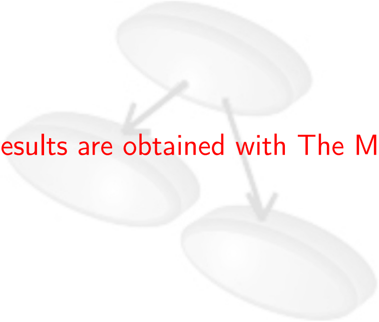
The Mallows Model

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]

[1] J. Ceberio, A. Mendiburu, J.A. Lozano. **Introducing The Mallows Model on Estimation of Distribution Algorithms**. *International Conference on Neural Information Processing*. Submitted. 2011.

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

Intelligent Systems Group

UPV - EHU

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