Distance-Based Exponential Probability Models for Constrained Combinatorial Problems

Josu Ceberio[1(✉)], Alexander Mendiburu[1], and Jose A. Lozano[1,2]

[1] University of the Basque Country UPV/EHU,
Paseo Manuel Lardizabal 1, 20018 Donostia, Spain
josu.ceberio@ehu.eus
[2] Basque Center for Applied Mathematics (BCAM),
Alameda Mazarredo 14, 48009 Bilbao, Spain

Abstract. Estimation of Distribution Algorithms (EDAs) have already demonstrated their utility when solving a broad range of combinatorial problems. However, there is still room for methodological improvement when approaching problems with constraints. The great majority of works in the literature implement repairing or penalty schemes, or use ad-hoc sampling methods in order to guarantee the feasibility of solutions. In any of the previous cases, the behavior of the EDA is somehow denaturalized, since the sampled set does not follow the probability distribution estimated at that step. In this work, we present a general method to approach constrained combinatorial optimization problems by means of EDAs. This consists of developing distance-based exponential probability models defined exclusively on the set of feasible solutions. In order to illustrate this procedure, we take the 2-partition balanced Graph Partitioning Problem as a case of study, and design efficient learning and sampling methods to use distance-based exponential probability models in EDAs.

Keywords: Constraint · Estimation of Distribution Algorithm
Distance-based exponential model · Graph Partitioning Problem

1 Introduction

A Combinatorial Optimization Problem (COP), $\mathbf{P} = (\Omega, f)$, consists of a domain of solutions Ω also known as *search space* (finite or infinite countable), and an *objective function* f that is defined as

This work has been partially supported by the Research Groups 2013–2018 (IT-609-13) programs (Basque Government), and TIN2016-78365R and TIN2017-82626R (Spanish Ministry of Economy, Industry and Competitiveness). Jose A. Lozano is also supported by BERC 2014–2017 and Elkartek programs (Basque government) and Severo Ochoa Program SEV-2013-0323 (Spanish Ministry of Economy and Competitiveness).

F. Herrera et al. (Eds.): CAEPIA 2018, LNAI 11160, pp. 187–197, 2018.
https://doi.org/10.1007/978-3-030-00374-6_18

$$f : \Omega \rightarrow \mathbf{R}$$
$$x \mapsto f(x)$$

Given an instance of the problem, the aim is to find a solution $x \in \Omega$ such that f is maximized (or minimized).

The work on computers and intractability by Garey and Johnson [6] showed that many COPs are NP-*hard*, denoting the difficulty of finding the optimum solution of a problem of this type (motivated principally by the size of Ω). In this sense, literature on evolutionary computation has proposed a broad range of heuristic and metaheuristic algorithms that are able to provide acceptable solutions in reasonable computation times. Among those algorithms, Estimation of Distribution Algorithms (EDAs) [11,16] have proved to be a powerful evolutionary algorithm for solving either artificial or real-world COPs [3,13].

EDAs are a type of population-based evolutionary algorithm designed for solving combinatorial and numerical optimization problems. Based on machine learning techniques, at each iteration, EDAs learn a probabilistic model from a subset of the most promising solutions, trying to explicitly express the dependencies between the different variables of the problem. Then, by sampling the probabilistic model learnt in the previous step, a new population of solutions is created. The algorithm iterates until a certain stopping criterion is met, such as a maximum number of iterations, homogeneity of the population, or a lack of improvement in the solutions.

Among different types of COPs, there is one where there is room for methodological improvement in the design of EDAs. We refer to problems with constraints (also known as restrictions). In general, a n dimensional constrained COP, or from now on, *constrained problem* [10] consists of (following the same notation as for COPs)

$$\begin{array}{lll} \text{minimizing} & f(\mathbf{x}), & \mathbf{x} = (x_1, \ldots, x_n) \in \Omega \\ \text{subject to,} & g_i(\mathbf{x}) \leq 0, & i = 1, \ldots, r \\ & h_j(\mathbf{x}) = 0, & j = r+1, \ldots, m \end{array}$$

where g and h are linear functions that are used to describe, respectively, inequality and equality constraints. Some relevant examples of constrained problems are the knapsack problem [12], capacitated arc routing problems [5] or graph partitioning problem [15].

Constrained problems introduce a challenging characteristic in the definition of the solutions in the search space: feasibility. A solution $\mathbf{x}$ is *feasible* if it holds all the constraints (functions g and h), otherwise it is *unfeasible*. It must be noted that the codification used to describe the solutions may induce solutions that are unfeasible. For instance, if the codification used to formalize solutions of size n is the binary coding, then the set of all solutions that fit in the codification are all the binary vectors of size n (2^n in total). Nonetheless, many of those solutions might be unfeasible, and thus, procedures that discriminate between feasible and unfeasible solutions are needed within the algorithms.

This point represents a serious drawback for EDAs, since these algorithms learn a probability distribution defined on the whole set of solutions induced

by the codification (either feasible or unfeasible solutions), and then the new solutions are sampled from that distribution. As a consequence, both type of solutions are usually generated. In this sense, in the case of EDAs, in order to hold the feasibility of the solutions, three trends have been followed: (1) repair unfeasible solutions, (2) use penalty functions to punish unfeasible solutions[1], or (3) implement sampling procedures that update the probability distribution while constructing the solutions [18]. When following any of these trends, the behavior of EDAs is somehow denaturalized as the obtained sample of solutions does not follow the probability distribution defined at that iteration.

Works in the literature with regard to the design of EDAs for solving permutation problems have shown that introducing probability models defined exclusively on the space of feasible solutions (the set of all permutations of a given size) can be a step forward in terms of performance [2]. Inspired by that work, recently, Ceberio et al. [4] proposed a lattice-based probability model defined exclusively on the set of feasible solutions of the 2-partition balanced Graph Partitioning Problem (GPP). Despite its novelty, the proposed approach is ad-hoc for that problem, and can hardly be extended to other problems.

Based on the work of Irurozki [8] on probability models for permutation spaces, in this manuscript we address constrained problems in the framework of EDAs following a more general research line: design and implement new distance-based exponential probability models that define a probability distribution on the set of feasible solutions. In order to design such types of models, there are three key aspects related to the model to consider, and it is desirable to give computationally accurate, and efficient, implementations: (1) calculate the probability of any x in Ω (feasible set), (2) given a set $\{\mathbf{x}^1, \ldots, \mathbf{x}^n\}$ of solutions, estimate the parameters of the probability model, and (3) sample solutions given the parameters of the probability model.

With illustrative purposes, we consider the GPP as a case of study (in Sect. 2). Then, in Sect. 3, one by one we address the three key-aspects enumerated above. In Sect. 4, we carried out a set of experiments to evaluate the performance of the proposed EDA. Lastly, conclusions and future works are listed in Sect. 5.

2 Graph Partitioning Problem

The Graph Partitioning Problem (GPP) is the problem of finding a k-partition of the vertices of a graph while minimizing the number (or sum of weights) of the edges between sets [15]. Among its many variants, in this paper we considered the balanced 2-partition case, i.e., a solution for the problem groups the vertices into two sets of equal size. So, any solution is encoded as a binary vector $\mathbf{x} \in \{0,1\}^n$, where x_i indicates the set to which vertex i is assigned, n stands for the size of the problem, and is subject to the restriction of having the same number of zeros

[1] Trends (1) and (2) have been proposed for a broad range of algorithms that include EDAs.

and ones, i.e., $\sum_{i=1}^{n} x_i = n/2$. Taking this restriction into account, the search space of solutions Ω is composed of $\binom{n}{n/2}$ binary vectors.

Given a weighted undirected graph $G = (V, E)$ with $n = |V|$ vertices, and the weights w_{ij} between each pair of vertices in G, the objective function is calculated as the total weight of the edges between the sets (sum of the weights of the edges in the cut), and it is denoted as

$$f(\mathbf{x}) = \sum_{i=1}^{n} \sum_{j=1}^{n} x_i(1 - x_j)w_{ij}$$

Figure 1 introduces a GPP instance of $n = 8$, where two solutions are illustrated. As we can see, $\mathbf{x}^1$ is better than $\mathbf{x}^2$ since the sum of the weights of the edges of the cut (in bold) is lower: $f(\mathbf{x}^1) = 1 + 3$ and $f(\mathbf{x}^2) = 2 + 2 + 1$.

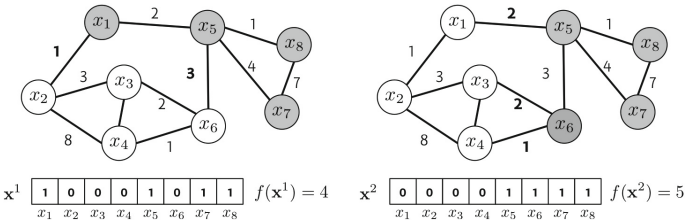

Fig. 1. Example of $n = 8$ instance of GPP with two different solutions $\mathbf{x}^1$ and $\mathbf{x}^2$.

The GPP is a very well-known constrained problem and has been studied extensively. When the problem is balanced (as in our case), the problem is NP-hard [1]. This problem is present in many real world applications [14], in fields such as tracking areas in cellular networks [19] or image segmentation [7].

3 Distance-Based Exponential Model

Let us consider a distance-based exponential probability model P defined on a finite (or infinite countable) domain of solutions Ω. Under this model, the probability value of every solution in the domain, $x \in \Omega$, is calculated as

$$P(x) = \frac{e^{-\theta d(x,\bar{x})}}{\psi(\theta)} \tag{1}$$

where $\psi(\theta)$ is the normalization constant, θ denotes the spread of the distribution, and $d(x, \bar{x})$ is the distance of x to a reference solution $\bar{x}$.

This model assigns every solution x a probability that decays exponentially with respect to its distance to $\bar{x}$. When θ equals 0, the model assigns equal probability to every solution in Ω. The larger the value of θ, the more concentrated the distribution becomes around the reference solution.

As previously mentioned, in order to introduce a model of this type in EDAs to deal with constrained problems, there are three key aspects that are mandatory to be solved efficiently. First, we need to calculate the probability of any given solution in Ω, i.e., compute Eq. 1, in closed form. Next, given a set of solutions, it is necessary to calculate the parameters of the model: θ and $\bar{x}$. In this work, we decided to calculate their maximum likelihood estimators (MLE). Finally, given the parameters of the model, we need to be able to obtain a sample of solutions that follows the inferred probability distribution.

It is worth remarking that, as EDAs are iterative algorithms, and learning/sampling procedures are repeated innumerable times in that framework, developing low complexity methods for each procedure is critical for the general feasibility of the algorithm. For the sake of illustrating the approach presented in this manuscript, we considered the balanced 2-partition GPP as a case of study, and we individually address each of the three key aspects described above.

3.1 A Case of Study: The Graph Partitioning Problem

The first decision to make is the election of the distance-metric d, which directly depends on the codification of solutions. As described in Sec. 2, the solutions of the 2-balanced GPP are codified as binary vectors of size n with an equal number of zeros as ones. Among binary vectors, the most natural distance-metric is the Hamming distance, i.e., given two solutions x and y, the Hamming distance is calculated as $d_H(x, y) = \sum_{i=1}^{n} \mathbf{1}_{[x(i) \neq y(i)]}$. It is worth noting that the codification employed to describe the solutions is highly redundant. For instance, the solutions $x^1 = (0, 0, 0, 1, 1, 1)$ and $x^2 = (1, 1, 1, 0, 0, 0)$ are equal. One is a relabeling of the other, but in terms of the Hamming distance, they are at maximum distance from each other. Therefore, we define a new distance for this case that is based on the Hamming distance.

Definition 1. *Let x be a solution for a 2-balanced GPP problem, and $\neg x$ is the negated or complementary solution of x, then the distance-metric $d(x, \bar{x})$ is calculated as*

$$d(x, \bar{x}) = \min\{d_H(x, \bar{x}), d_H(\neg x, \bar{x})\}$$

Under this metric, the maximum distance K to which a solution can be defined is $\lfloor n/2 \rfloor$. The minimum distance is 0 (when x or $\neg x$, are equal to $\bar{x}$). Solutions can only be at paired distances in order to hold the constraint of the balanced 2-partition case. The number of solutions at distance k (being k even) is $\binom{n/2}{k/2}^2$. Note that each possible solution can be coded with two different binary vectors (one the negation of the other).

Computing the Probability Value. Given a solution x and the parameters θ and $\bar{x}$, computing $P(x)$ is given by an efficient computation of the normalization constant $\psi(\theta)$. This function, which is roughly computed as

$$\psi(\theta) = \sum_{y \in \Omega} e^{-\theta d(y, \bar{x})},$$

considering the previous observations, can be reformulated as follows:

$$\psi(\theta) = \sum_{l=0}^{K/2} \binom{n/2}{l}^2 e^{-\theta 2l}$$

Learning. Given a set $\mathbf{x} = \{x^1, \ldots, x^N\}$ of solutions, the learning step consists of estimating the parameters of the proposed model: $\bar{x}$ and θ. In this work, we decided to estimate the MLE of the parameters. To that end, we define the likelihood expression as

$$L(\theta, \bar{x}|\mathbf{x}) = \prod_{i=1}^{N} \frac{e^{-\theta d(x_i, \bar{x})}}{\psi(\theta)},$$

The aim is to find the parameters that maximize that function. To that end, we apply the logarithm, as calculus is more simple, but the MLE parameters are not affected. So we have the function

$$\log L(\theta, \bar{x}|\mathbf{x}) = -\theta N \bar{d} - N \log \left[\sum_{l=0}^{K/2} \binom{n/2}{l}^2 e^{-\theta 2l} \right] \tag{2}$$

where $\bar{d} = \frac{1}{N} \sum_{i=1}^{N} d(x_i, \bar{x})$.

The term related to the estimation of $\bar{x}$ is independent to that of θ, so by minimizing $\sum_{i=1}^{N} d(x_i, \bar{x})$, the likelihood is maximized. Thus, first we define the computation of $\bar{x}$ as

$$\bar{x} = \arg\min_{x \in \Omega} \sum_{i=1}^{N} d(x_i, x)$$

Such an estimation is an optimization task itself. Therefore, we propose estimating $\bar{x}$ as in [3]. First, we calculate the average values at each position individually in the N solutions, and next, the n/2 positions with largest values in the resulting vector are assigned 1, and the rest 0.

Once the $\bar{x}$ is estimated, the MLE of the spread parameter θ is calculated. In order to find the maximum value of θ that maximizes Eq. 2, we compute its derivative and equate it to 0:

$$-N\bar{d} + N \frac{\sum_{l=0}^{K/2} \binom{n/2}{l}^2 2l e^{-\theta 2l}}{\sum_{l=0}^{K/2} \binom{n/2}{l}^2 e^{-\theta 2l}} = 0,$$

and finally,

$$\sum_{l=0}^{K/2} \binom{n/2}{l}^2 (2l - \bar{d}) e^{-\theta 2l} = 0. \tag{3}$$

θ cannot be calculated exactly from Eq. 3 and, thus, we propose using numerical methods, such as Newton-Raphson [3] to estimate it.

Sampling. Once the parameters of the model have been estimated, the next step in EDAs is to sample a set of solutions that follow the probability distribution defined by the parameters calculated in the previous step. In this case, we propose using a distance-based sampling algorithm based on the following two statements [8]: (1) every solution has the same number of solutions at each distance, and (2) every solution at same distance from $\bar{x}$ has same probability.

First, a distance $k = 2l$ at which generate a solution is sampled. The probability under the proposed model to generate a solution at distance k is

$$P(k) = \sum_{x|d(x,\bar{x})=k} P(x) = \binom{n/2}{k/2}^2 \frac{e^{-\theta k}}{\psi(\theta)}$$

Once k has been decided, we generate uniformly at random one solution among those at distance k from the reference solution $\bar{x}$. To that end, we choose u.a.r. k/2 zeros and k/2 ones, and the chosen items are bit-flipped.

4 Experiments

The experimental study has been organized in two parts. First, the feasible ranges to estimate the θ parameter are calculated. Then, a performance evaluation of the EDA on a benchmark of instances of GPP is carried out.

4.1 Feasible Ranges for θ

In order to incorporate the model to the framework of EDAs, and avoid scenarios in which the proposed model is not suitable for optimization, the spread parameter θ needs to be controlled in a range of feasible values. To that end, we calculate for different $n = \{124, 250, 500, 1000\}$, in the range [0,10] for θ with step size 0.1, the probability assigned to the reference solution. Results are introduced in Fig. 2.

Without performing previous experiments, for each problem size, we set the lower and upper bounds that assign to $\bar{x}$ the probabilities 10^{-9} and 0.1 respectively. The values are presented in Table 1.

4.2 Performance Analysis

In order to obtain some hints about the performance of the proposed exponential probability model (abbreviated as **Exp**) for solving the GPP, we carried some experiments on the set of 22 GPP instances [9]. In addition, with comparison purposes, we included three EDAs in the analysis: the univariate marginal distribution algorithm (UMDA) [16], the tree EDA (Tree) [17] and the Lattice EDA [4] (Lattice). The UMDA assumes that the n-dimensional joint probability distribution factorizes as a product of n univariate and independent probability distributions[2]. The second EDA, Tree, estimates, at each iteration, a dependency-tree

[2] Despite this strong assumption, due to its good general performance and low time consumption it is appropriate to include UMDA as baseline in the experimentation.

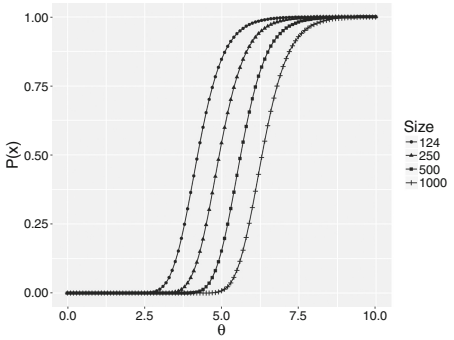

Fig. 2. Probability assigned to $\bar{x}$ for different n and θ.

Table 1. Lower and upper bounds for the estimation of θ parameters.

n	θ_{lower}	θ_{upper}
124	1.6	3.5
250	2.4	4.2
500	3.1	4.9
1000	3.8	5.6

which encodes conditional dependencies between the problem variables. Finally, in [4], the authors of the paper made the first attempt of developing EDAs for constrained COPs, and proposed a square lattice probability model to optimize the GPP.

As this is a preliminary work, we considered using the same parameter settings as in [4]:

- Population size is set to 10 n.
- 5 n solutions are selected to learn the probabilistic model.
- At each iteration, 10 n solutions are sampled.
- A maximum number of 100 n^2 evaluations are considered as the stopping criterion.

Equally, the initial population is generated uniformly at random.

Particular Settings of the Exponential EDA. A preliminary analysis of the quality of the reference solution estimated at the learning step pointed out that the proposed method, although it provides MLE of the reference solution $\bar{x}$, it is not competitive. In this sense, we decided to set the best solution of the population as $\bar{x}$.

In addition, as sampling at distance 0 provides the reference solution, we avoid sampling at this distance. Finally, due to convergence and diversity considerations, the solutions that do not exist in the population are only accepted when sampling.

Results. Results are presented in Table 2 as Average Relative Percentage Deviation (ARPD) and Average Execution Times (AET) of 10 executions of each algorithm-instance pair.

Looking at the results, we can conclude that with regard to the ARPD, Tree is the best performing algorithm followed by Lattice and Exp. Finally, UMDA

Table 2. ARPD and AET results for Johnson's benchmark instances. The results in bold denote the approach with the lowest ARPD and AET. (*) denotes the instances in which 10^6 evaluations were used instead of $100n^2$.

Instance	ARPD					AET			
	Best	Exp	Lattice	UMDA	Tree	Exp	Lattice	UMDA	Tree
$G_{124,0.02}$	*13*	0,32	0,32	0,61	**0,19**	**33**	145	57	464
$G_{124,0.16}$	*449*	0,04	0,02	0,05	**0,01**	**32**	144	60	496
$G_{250,0.01}$	*31*	0,40	0,33	0,49	**0,20**	**511**	2835	879	19543
$G_{250,0.02}$	*118*	0,10	0,07	0,14	**0,06**	**506**	2698	882	22610
$G_{250,0.04}$	*360*	0,04	0,04	0,10	**0,03**	**515**	2271	802	21963
$G_{250,0.08}$	*830*	0,05	**0,01**	0,05	**0,01**	**507**	2187	818	20652
$G_{500,0.005}$	*61*	0,38	0,30	0,40	**0,08**	**7768**	64873	12760	771427
$G_{500,0.01}$	*234*	0,14	0,09	0,21	**0,07**	**7948**	54905	14025	758556
$G_{500,0.02}$	*642*	0,06	**0,03**	0,11	**0,03**	**7918**	58294	13992	833603
$G_{500,0.04}$	*1754*	0,05	**0,02**	0,06	**0,02**	**7612**	44498	14366	768596
$G_{1000,0.0025}$*	*131*	1,16	2,96	3,20	**0,74**	**1420**	2832	234724	250136
$G_{1000,0.005}$*	*496*	**0,52**	1,22	1,28	0,88	**1395**	2775	229806	276535
$G_{1000,0.01}$*	*142*	**0,28**	0,56	0,66	0,62	**1358**	2859	239480	256820
$G_{1000,0.02}$*	*3450*	**0,18**	0,35	0,40	0,39	**1349**	2775	237858	252978
$U_{500,0.05}$	*23*	1,34	1,17	1,89	**0,57**	**7754**	58430	14541	721097
$U_{500,0.1}$	*61*	1,50	1,05	1,12	**0,57**	**7698**	45947	13065	745215
$U_{500,0.2}$	*185*	0,89	0,56	0,87	**0,44**	**7767**	43988	12518	745311
$U_{500,0.4}$	*412*	0,81	0,41	0,38	**0,28**	**7728**	48326	12867	764009
$U_{1000,0.05}$*	*77*	4,82	**1,62**	12,83	2,39	**1364**	2938	2757	268978
$U_{1000,0.1}$*	*170*	5,09	**1,67**	11,67	3,73	**1349**	3451	2734	271070
$U_{1000,0.2}$*	*352*	5,23	**1,67**	10,58	4,94	**1322**	3379	2648	263100
$U_{1000,0.4}$*	*862*	4,09	**1,53**	3,24	2,29	**1322**	3348	2494	264821

is the worst performing algorithm. Note that Exp obtained the best result in three of the largest instances while it is by far the fastest algorithm for all sizes considered.

5 Conclusions and Future Work

In this work we proposed using distance-based exponential probability models defined exclusively on the set of feasible solutions. In this sense, we took as a case of study the 2-balanced Graph Partitioning Problem, and designed step by step a probability model to be used in the framework of EDAs. In addition, some experiments to fine-tune the probability model have also been carried out.

Finally, in order to evaluate the performance of the proposed algorithm, some benchmark experiments were carried out. Results revealed that the proposed methodology is a promising research line for future work.

In this work, we used an heuristic method to calculate the reference solution $\bar{x}$ as it is a cheap algorithm to approach the estimation problem. However, we have no intuition about its precision, in this sense, for the future, we should (1) perform experiments to estimate its accuracy, and (2) investigate other estimation methods for MLE of $\bar{x}$.

Regarding the experimental study, in future research, we aim to provide a more general experimental study in which the parameters of the compared algorithms are fined-tuned, and are evaluated on a larger benchmark of instances. We also consider to hybridize Exp in order to perform a fair comparison of the algorithms.

References

1. Bui, T.N., Jones, C.: Finding good approximate vertex and edge partitions is NP-hard. Inf. Process. Lett. **42**(3), 153–159 (1992)
2. Ceberio, J.: Solving permutation problems with estimation of distribution algorithms and extensions thereof. Ph.D. thesis, University of the Basque Country, December 2014
3. Ceberio, J., Irurozki, E., Mendiburu, A., Lozano, J.A.: A distance-based ranking model estimation of distribution algorithm for the flowshop scheduling problem. IEEE Trans. Evol. Comput. **18**(2), 286–300 (2014)
4. Ceberio, J., Mendiburu, A., Lozano, J.A.: A square lattice probability model for optimising the graph partitioning problem. In: 2017 IEEE Congress on Evolutionary Computation (CEC), pp. 1629–1636. IEEE (2017)
5. Dror, M.: Arc Routing: Theory Solutions and Applications. Springer, Heidelberg (2012). https://doi.org/10.1007/978-1-4615-4495-1
6. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman & Co., New York (1979)
7. Grady, L., Schwartz, E.L.: Isoperimetric graph partitioning for image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **28**(3), 469–475 (2006)
8. Irurozki, E.: Sampling and learning distance-based probability models for permutation spaces. Ph.D. thesis, University of the Basque Country, November 2014
9. Johnson, D.S., Aragon, C.R., McGeoch, L.A., Schevon, C.: Optimization by simulated annealing: an experimental evaluation. part i, graph partitioning. Oper. Res. **37**(6), 865–892 (1989)
10. Kusakci, A.O., Can, M.: Constrained optimization with evolutionary algorithms: a comprehensive review. Southeast Eur. J. Soft Comput. **1**(2) (2012)
11. Larrañaga, P., Lozano, J.A.: Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation. Kluwer Academic Publishers (2002)
12. Martello, S., Toth, P.: Knapsack Problems: Algorithms and Computer Implementations. Wiley Inc., Hoboken (1990)
13. Mendiburu, A., Miguel-Alonso, J., Lozano, J.A., Ostra, M., Ubide, C.: Parallel EDAs to create multivariate calibration models for quantitative chemical applications. J. Parallel Distrib. Comput. **66**(8), 1002–1013 (2006)

14. Menegola, B.: A Study of the k-way graph partitioning problem. Ph.D. thesis, Institute of Informatics, Federal University of Rio Grande Do Sul (2012)
15. Mezuman, E., Weiss, Y.: Globally optimizing graph partitioning problems using message passing. In: Proceedings of the 15th (AISTATS), pp. 770–778 (2012)
16. Mühlenbein, H., Paaß, G.: From recombination of genes to the estimation of distributions I. Binary parameters. In: Voigt, H.-M., Ebeling, W., Rechenberg, I., Schwefel, H.-P. (eds.) PPSN 1996. LNCS, vol. 1141, pp. 178–187. Springer, Heidelberg (1996). https://doi.org/10.1007/3-540-61723-X_982
17. Pelikan, M., Tsutsui, S., Kalapala, R.: Dependency trees, permutations, and quadratic assignment problem. Technical report, Medal Report No. 2007003 (2007)
18. Santana, R., Ochoa, A.: Dealing with constraints with estimation of distribution algorithms: the univariate case. In: Second Symposium on Artificial Intelligence. Adaptive Systems. CIMAF 99, La Habana, pp. 378–384 (1999)
19. Toril, M., Luna-Ramírez, S., Wille, V.: Automatic replanning of tracking areas in cellular networks. IEEE Trans. Vehic. Technol. **62**(5), 2005–2013 (2013)