Estimation Distribution Algorithm for mixed continuous-discrete optimization problems

Jiří Očenášek, Josef Schwarz

Brno University of Technology, Faculty of Information Technology,
Božetěchova 2, 612 66 Brno, Czech Republic

ocenasek@fit.vutbr.cz, schwarz@fit.vutbr.cz

Summary:
In recent few years expressive progress in the theory and practice of Estimation of Distribution Algorithms (EDA) [1] has appeared, where the classical genetic recombination operators are replaced by probability estimation and stochastic sampling techniques. In this paper we identify some disadvantages of present probabilistic models used in EDAs and propose more general and efficient model for continuous optimization problems based on the decision trees. The new variant of EDA is capable to solve mixed continuous-discrete optimization problems.

Keywords: Estimation Distribution Algorithm, Bayesian Optimization Algorithm, Bayesian network, Gaussian network, decision tree, CART model

1 Introduction

For the discrete optimization problems the Bayesian network [1] and its decision graph refinement [2] are used in Estimation Distribution Algorithms as the general models to encode the parameter dependencies.

For continuous domains there are several approaches used to discover and encode the parameter dependencies – binary encoding [3,4], Gaussian networks [5], Gaussian kernels and mixtures of Gaussians [6].

In the following chapter we show that all present models used for optimization of continuous functions are either inaccurate or do not allow proper mixing of building blocks.

In chapter 3 we proposed the usage of decision trees to remove the disadvantages of present models. In our Mixed Bayesian Optimization Algorithm (MBOA) we implemented the trees with mixed decision nodes, so MBOA is suitable for both continuous and/or discrete optimization problems.

2 Present methods and approaches

In EDA individuals in the population are treated as vectors of instantiations of n random variables X_i, each random variable represents one parameter of a solution

$$\boldsymbol{X} = (X_1, X_2, \ldots, X_n) \tag{1}$$

The goal of optimization is to find an optimal soluton $\boldsymbol{X^*}$ from domain D such that

$$\boldsymbol{X^*} = arg\ max\ fitness(\boldsymbol{X}), \quad \forall \boldsymbol{X} \in D \tag{2}$$

In case of discrete combinatorial problem all variables X_i are discrete. Each generation the space of possible solutions is effectively sampled using the Bayesian network (BN) [1], which encodes the conditional probabilities

$$p(X) = \prod_{i=1}^{n} p(X_i \mid Pa(X_i)) \tag{3}$$

The Bayesian network is constructed in the incremental way - for each variable X_i a large number of statistical tests is performed to form the set of "parent" variables $Pa(X_i)$ which significantly affect the value of variable X_i.

In the case of a continuous optimization problem it is possible to transform each continuous parameter into binary parameters and use the Bayesian network again. However, with the increasing precision of encoding the complexity of BN grows exponentially. The other way of solving continuous optimization problem by EDA is to let the parameters X_i be continuous (like in evolutionary strategies) and find a proper model for this continuous domain.

The basic model is the Gaussian network [5], where the mean value m_i of each random variable X_i is affected by linear combination of "parent" values X_j. This simple regression model is able to describe the fitness function with only one local extreme (the shape of reliable sampling region is elliptical).

Instead of using one probability density function (PDF) it is possible to use each solution as a source of elementary normal PDF - the Gaussian kernel [6]. It seems this may improve the sampling accuracy, but in fact no linkage information necessary for proper mixing of building blocks is provided. The Iterated Density Estimation Evolutionary Algorithm (IDEA) [6] uses clustering strategy to divide the samples into linear clusters and for each cluster one Gaussian network is used. This is more general than usage of the single linear regression for the entire space of parameters, but no mixing of building blocks between clusters is possible.

3 MBOA algorithm

In our approach we propose and implement the EDA algorithm based on the decision trees. Binary decision trees have been successfully used in EDA [2] for discrete optimization problems - with the Bayesian network. In our work we have extra focused on non-binary problems.

3.1 CART model

From the area of data mining we adopted the idea of CART model [7] (Classification and Regression Tree), which is usable also for continuous and categorical splits. For each “target” random variable X_i we build one decision tree. The split nodes of *i*-th decision tree are used to cut the domain of parent variables $Pa(X_i)$ into parts, where the variable X_i is more linear or predictable. We start building the decision trees from empty trees and we recursively add the splitting nodes until no splitting is favourable.

```
Function RecursiveSplitting(Population Pop,
           TargetVariable $X_i$,
           ListOfCandidateSplitVariables Pa)
           : DecisionTreeNode
Begin
  $f_{Temp}$ := EstimateElementaryPDF(Pop,$X_i$,Pa);
  If "model is too detailed" then
     return new LeafNode($f_{Temp}$);
  For Each Variable $X_j$ in Pa do
     $E_j$ := Find_optimal_split_on_$X_j$_with_respect_to_$X_i$;
     ModelGain := Evaluate_the_ split_gain("$X_j \le E_j$", $X_i$);
     Save the $X_j$ and $E_j$ with the highest ModelGain;
  End for
  Pop1 := SelectIndividuals (Pop,"$X_j \le E_j$");
  Pop2 := SelectIndividuals (Pop,"$X_j > E_j$")
  return new SplitNode(new SplitCondition("$X_j \le E_j$"),
              RecursiveSplitting(Pop1,$X_i$, Pa\{$X_j$}),
              RecursiveSplitting(Pop2,$X_i$, Pa\{$X_j$}));
End;
```

The leaf nodes define the elementary models for obtaining the “target” variable X_i. For continuous variables one-dimensional normal PDF is estimated and used as the leaf. We are also able to use Gaussian kernels or the linear regression model known from Gaussian network.

3.2 Metrics for model construction

In the RecursiveSplitting procedure the list of candidate split variables is given in advance, which allows the future parallelization, see [8]. The step of split condition finding and evaluation is essential. In the present version we use more sophisticated metrics to find the optimal split boundary E_j*:

$$E_j^* = \underset{\forall E_j \in X_j}{\arg\max}\ Gain(E_j) \tag{4}$$

$$Gain(E_j) = \max_{\forall E_i \in X_i} \frac{\Gamma\Big(\sum_{r\in\{left,right\}} \sum_{s\in\{up,down\}} (n_{r,s}+1)\Big) \cdot \prod_{r\in\{left,right\}} \prod_{s\in\{up,down\}} \Gamma(n_{r,s}+1)}{\left(\prod_{s\in\{up,down\}} \Gamma\Big(\sum_{r\in\{left,right\}} (n_{r,s}+1)\Big)\right) \cdot \left(\prod_{r\in\{left,right\}} \Gamma\Big(\sum_{s\in\{up,down\}} (n_{r,s}+1)\Big)\right)} \qquad (5)$$

We derived the equation (5) from the Bayes-Dirichlet metrics (BDe, see [9]), meaning of symbols $n_{left,down}$, $n_{left,up}$, $n_{right,down}$, $n_{right,up}$ is shown in Fig. 1.

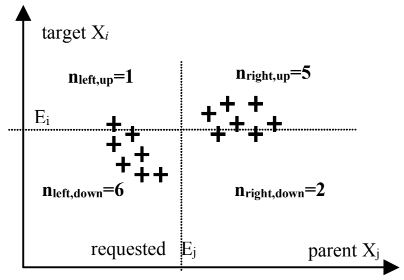
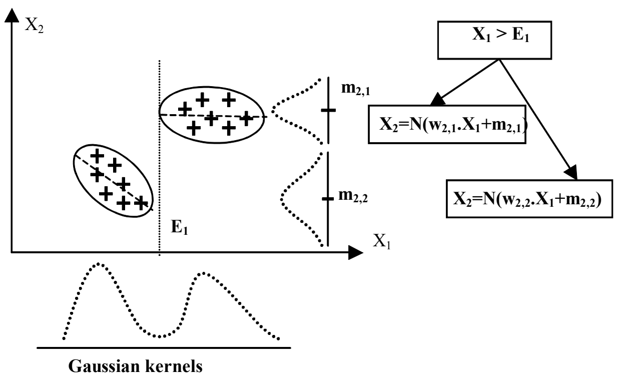


Fig. 1. All individuals in the population are classified into four groups according to conditions *"$X_j \le E_j$"* and *"$X_i \le E_i$"*. The symbols $n_{left,down}$, $n_{left,up}$, $n_{right,down}$, $n_{right,up}$ represent number of individuals in each quadrant.

The next advantage of decision trees is the backward-compatibility with discrete domains. For binary split nodes the split condition is straightforward *"$X_j \le 0$"*, so we can apply the equation (5) for $E_j=0$. Each leaf for binary target determines the frequency of occurrence of value 1 in the part of population, which fulfills the split-conditions corresponding to the path from the root.



X$_2$
m$_{2,1}$
m$_{2,2}$
E$_1$
X$_1$
X$_1$ > E$_1$
X$_2$=N(w$_{2,1}$.X$_1$+m$_{2,1}$)
X$_2$=N(w$_{2,2}$.X$_1$+m$_{2,2}$)
Gaussian kernels

Fig. 2. The population precisely approximated by decision trees.

We are also able to optimize non-binary discrete problems in their natural alphabetical encoding. In this case we use hillclimbing algorithm to split the set of possible X_j values into left and right subset. The variable s in (5) goes through all possible values of X_i instead of only two categories *{up,down}*.

The next advantage of MBOA is the utilization of RTR (Restricted Tournament Replacement) proposed by Martin Pelikan in [10]. In the replacement stage each new individuals competes with the Euclidean-nearest individual among 20 randomly selected individuals. This ensures better niching. As the next improvement we utilized the fitness value in the same way as the continuous parent variable - the population can be split according to certain fitness value into more-homogenous regions to achieve better divergence and model accuracy.

4 Experimental results

In the experimental part we compared the efficiency of our approach with the other mentioned EDA algorithms. First of all we proved the backward-compatibility with the binary domain. We compared the performance of MBOA with the Pelikan's original simple BOA algorithm on several deceptive functions.

Table 1. Results for deceptive$_3$ and trap$_5$ function (see [2])- the average number of fitness evaluations required to reach the global optima in 10 runs, with min. population size N.

Algorithm\Benchmark	deceptive$_3$ function, n=90	trap$_5$ function, n=90
MBOA	43530 evaluations, N=3800	**53280** evaluations, N=5500
Simple BOA	**43510** evaluations, N=3800	54310 evaluations, N=5500

Secondly, we have used some continuous benchmarks to compare MBOA with the best IDEA algorithms in [6]. IDEA1 uses non-clustered normal mixture of 10 PDFs, IDEA2 and IDEA5 use Euclidean 2-means clustering with normal mixtures of 10 PDFs, IDEA3 uses Mahalanobis leader 3½ clustering with normal mixtures of 10 PDFs, IDEA4 uses non-clustered normal mixture of 5 PDFs and IDEA6 uses Mahalanobis leader 5 clustering with normal mixtures of 5 PDFs.

Table 2. Griewank function, problem size n=5, domain= $<-5,5>^5$.

Algorithm	Fitness evaluations to get the global optima (avg. for 10 runs, 7-digit precision)
MBOA with univariate Gaussian leafs	**19790**, N=120
IDEA1	51832, N=1100
IDEA2	72552, N=1750
IDEA3	89718, N=2250

Table 3. Michalewicz function, n=5, domain= $<0,\pi>^5$

Algorithm	Fitness evaluations to get the global optima (avg. for 10 runs, 7-digit precision)
MBOA with univariate Gaussian leafs	**7690**, N=120
IDEA4	28903, N=1300
IDEA5	16596, N=750
IDEA6	22118, N=1000

Thirdly, we tested the scalability of MBOA on continuous two-deceptive function, see [3],[4] and we examined how the number of fitness evaluations grows with the problem size.

Table 4. The scalability for continuous two-deceptive benchmark

Algorithm	Two-deceptive function [3], 5-digit precision
MBOA with Gaussian-kernels leafs	$\mathbf{O(n^{1.79})}$ (n=10-50, N=300-3500)
Simple BOA with fixed-width histogram discretization, see [3]	$O(n^{2.1})$ (4 bins, n=10-40, N=1000-7400)
Simple BOA with fixed-height histogram discretization, see [3]	$O(n^{2.1})$ (16 bins, n=10-30, N=3500-17000)

Fourthly, we prepared our own mixed continuous-discrete benchmark, where the chromozome of length *n=2l* is composed of binary genes b_i and continuous genes c_i. This benchmark is very sensitive to correctness of dependency metric – mixed dependencies have to be discovered to find the global optimum.

$$f_{mixed}(b_0,c_0,b_1,c_1,...,b_{k-1},c_{k-1}) = \sum_{i=0}^{l-1} f_m(b_i,c_i) \tag{6}$$

$$f_m(b_i,c_i) = \begin{cases} 0{,}8 - 0.5 * c_i & b_i = 0 \\ 0.8 - c_i & b_i = 1 \wedge c_i \le 0.8 \\ 5*(c_i - 0.8) & b_i = 1 \wedge c_i > 0.8 \end{cases} \tag{7}$$

Table 5. The average number of evaluation for mixed benchmark

Problem size	n=10	n=20	n=30	n=40	n=50
MBOA with Gaussian-kernels leafs	23100 N=600	67800 N=1200	126000 N=1800	214800 N=2400	291000 N=3000

5 Conclusion and future work

In this paper we have identified the limitations of present probability models [5],[6] and proposed more general and efficient model for continuous optimization problems based on the decision trees. Decision trees are more general model than Gaussian networks and more useful for building blocks evolution than mixtures of Gaussians. Moreover, our approach is compatible with discrete domains, so we introduce the new kind of EDA uniquely capable to solve mixed continuous-discrete optimization problems.

The empirical results show that in the field of discrete optimization problems our MBOA is backward-compatible with the original BOA algorithm and provides similar results. The results for continuous Griewank and Michalewicz function confirm our assumption that MBOA overcomes the IDEA approach [6].

On the two-deceptive continuous benchmark we showed that the scalability of MBOA *$O(n^{1.79})$* is better than the scalability *$O(n^{2.1})$* of discretization approach [3] based on histogram models. The MBOA was also able to solve our mixed continuous-discrete hard benchmark. The results indicate that the main potential of our algorithm lies in solving high-dimensional problems with stronger parameter nonlinearities.

This research has been carried out under the financial support of the Research intention no. CEZ: J22/98: 262200012 - "Research in information and control systems" (Ministry of Education, CZ) and the research grant GA 102/02/0503 "Parallel system performance prediction and tuning" (Grant Agency of Czech Republic).

References

[1] Pelikan, M., Goldberg, D. E., Lobo, F.: A Survey of Optimization by Building and Using Probabilistic Models. IlliGAL Report No.99018, Illinois Genetic Algorithms Laboratory, University of Illinois at Urbana-Champaign, Urbana, IL, September 1999.

[2] Pelikan, M., Goldberg, D. E., Sastry, K.: Bayesian Optimization Algorithm, Decision Graphs, and Bayesian Networks. IlliGAL Report No. 2000020, Illinois Genetic Algorithms Laboratory, University of Illinois at Urbana-Champaign, Urbana, IL, May 2000.

[3] Pelikan, M., Goldberg, D. E., Tsutsui, S.: Combining the Strengths of the Bayesian Optimization Algorithm and Adaptive Evolution Strategies. IlliGAL Report No. 2001023, Illinois Genetic Algorithms Laboratory, University of Illinois at Urbana-Champaign, Urbana, IL, June 2001.

[4] Tsutsui, S., Pelikan, M., Goldberg, D. E.: Evolutionary Algorithm Using Marginal Histogram Models in Continuous Domain. IlliGAL Report No. 2001019, Illinois Genetic Algorithms Laboratory, University of Illinois at Urbana-Champaign, Urbana, IL, March 2001.

[5] Larrañaga, P., Etxeberria, R., Lozano, J. A., Peña, J. M.: Optimization by learning and simulation of Bayesian and Gaussian networks. Technical Report EHU-KZAA-IK-4/99, University of the Basque Country, December 1999.

[6] Bosman, P. A. N., Thierens, D.: Mixed IDEAs, Utrecht University technical report UU-CS-2000-45, December 2000.

[7] Chipman, H., George, E., McCulloch, R.: Bayesian CART model search, Journal Am. Statist. Assoc., 93, pp. 937-960.

[8] Očenášek, J., Schwarz, J.: The Parallel Bayesian Optimization Algorithm, Proceedings of the European Symposium on Computational Inteligence, Physica-Verlag, Košice, Slovak Republic, September 2000, pp. 61-67.

[9] Heckerman, D., Geiger, D., Chickering, M.: Learning Bayesian networks: The combination of knowledge and statistical data. Technical Report MSR-TR-94-09, Microsoft Research, Redmond, WA, 1994.

[10] Pelikan, M., Goldberg, D. E.: Escaping Hierarchical Traps with Competent Genetic Algorithms. IlliGAL Report No. 2001003, Illinois Genetic Algorithms Laboratory, University of Illinois at Urbana-Champaign, Urbana, IL, January 200.