EDA-PSO: A Hybrid Paradigm Combining Estimation of Distribution Algorithms and Particle Swarm Optimization

Endika Bengoetxea[1] and Pedro Larrañaga[2]

[1] Intelligent Systems Group, University of the Basque Country, San Sebastian, Spain
endika@ehu.es
[2] Departamento de Inteligencia Artificial, Universidad Politécnica de Madrid, Madrid, Spain
pedro.larranaga@fi.upm.es

Abstract. Estimation of Distribution Algorithms (EDAs) is an evolutionary computation optimization paradigm that relies the evolution of each generation on calculating a probabilistic graphical model able to reflect dependencies among variables out of the selected individuals of the population. This showed to be able to improve results with GAs for complex problems.

This paper presents a new hybrid approach combining EDAs and particle swarm optimization, with the aim to take advantage of EDAs capability to learn from the dependencies between variables while profiting particle swarm's optimization ability to keep a sense of "direction" towards the most promising areas of the search space. Experimental results show the validity of this approach with widely known combinatorial optimization problems.

Keywords: Swarm intelligence, Estimation Distribution Algorithms, Particle Swarm Optimization, Bayesian Networks.

1 Introduction and Motivation

Despite the popularity of Genetic Algorithms (GAs) [9] when applied to combinatorial optimization problems their behavior depends largely on the adequate setting of parameters –crossing and mutation operators, probabilities of crossing and mutation, size of the population, rate of generational reproduction...– and GAs show a poor performance in some problems where the designed operators of crossing and mutation do not guarantee that the building block hypothesis is preserved.

Estimation of Distribution Algorithms (EDA) [16,13] were proposed in the aim of making easier to predict the movements of the populations in the search space as well as to avoid the need for so many parameters as in GAs. EDA are population-based search algorithms based on probabilistic modeling of promising solutions. It has been underlined that their higher execution time pays off when optimization problems contain dependencies between variables which EDAs can learn on their learning step and propose the new generation of individuals accordingly [14,15]. Many papers in the literature improve performance focusing on several aspects of EDAs such as adding local optimization to improve each individual, the capability to learn more complex probabilistic graphical models and hybridation of EDAs with other known paradigms.

On the other hand, other evolutionary computation techniques such as Ant Colony Optimization (ACO) and Particle Swarm Optimization (PSO) focus the search strategy on maintaining a set of entities –ants or particles– which are able to keep a sense of *historical memory.* In the particular case of PSO [10,8], a population of particles is kept moving around the search space at given velocities that are adjusted stochastically according to the historical best position –*gbest*– and the neighborhood best position –*lbest.* If we compare PSO and GAs, PSO have fewer parameters to set and keeps memory capabilities since each particle remembers its best value through *gbest.*

In EDAs this notion of *memory* is not explicitly applied since the newly generated population can be very different from the previous and the probabilistic graphical model learned is based on the characteristics of the selected individuals of the current generation only. This lack of sense of memory in EDAs could be a drawback for concrete optimization problems with local optima having a similar fitness value. Here we propose to combine EDA's capability of learning dependencies between variables together with the memory or directional sense of ACO and PSO, through a new hybrid approach between EDA-PSO.

Hybrid PSO systems are nowadays an active research trend [22]. Examples include hybrids with GAs [7], cooperative approaches [4], self-organizing hierarchical techniques [19], and deflection, stretching and repulsion techniques [18]. Other approaches introduce into PSO techniques inspired in biology to prevent the swarm from crowding too closely and to locate as many optimal solutions as possible [5,17]. In the literature we can also find examples of combining PSO and EDA algorithms, although hybridizing occurs either partially or it is only applied as an improvement of PSO's steps. In [23,21] discrete EDAs are successfully applied to learn a probability model out of the particle current position each generation, in such a way that EDAs are applied to improve PSO in a similar way as GA hybrid variants in [2,7]. Finally, [11] presents a paradigm closer to a full hybrid: For every particle in the swarm two candidate particles are generated, one applying PSO and another with EDA. Every iteration the location to move a particle is determined using the PSO position update equation, and EDA-version particle is calculated by sampling a Gaussian distribution from the kernel in all n dimensions of the problem. The fitness of both particles is calculated, and the the fittest is selected for the next iteration.

Our proposal intends hybridizing beyond the [11] approach by building the new population instantiating the probabilistic graphical model of EDA as well as by maintaining a number P or particles that will create P new individuals each generation. Our approach is presented to continuous domains, which generalises discrete domain approaches. To our knowledge there is no such an approach in the literature combining both paradigms. The outline of the article is as follows:

Section 2 presents EDAs' theoretical background, and Section 3 proposes our hybrid approach EDA-PSO. Section 4 describes the experiments performed and the results obtained, and Section 5 provides conclusions.

2 Estimation of Distribution Algorithms

EDAs [13] are non-deterministic, stochastic heuristic search strategies that form part of the evolutionary computation approaches. The characteristic that most differentiates EDAs from other evolutionary search strategies is that they evolve by estimating the probability distribution of the fittest individuals and then sampling the induced model. EDAs are capable to underlying interdependencies among the encoded variables and to express them explicitly through the joint probability distribution associated with the individuals selected every generation.

More formally, let $\boldsymbol{X} = (X_1, \ldots, X_n)$ be a set of random variables, and let x_i be a value of X_i. Then, a probabilistic graphical model for $\boldsymbol{X}$ is a graphical factorization of the joint generalized probability distribution, $\rho(\boldsymbol{X} = \boldsymbol{x})$ (or simply $\rho(\boldsymbol{x})$). The model induced every generation in EDAs is expressed in the form of a directed acyclic graph (DAG) describing conditional interdependencies between the variables on $\boldsymbol{X}$. If $\boldsymbol{Pa}_i$ represents the set of parent variables of variable X_i in the probabilistic graphical model, the factorization of the joint distribution could be written as $\rho(\boldsymbol{x}) = \rho(x_1, \ldots, x_n) = \prod_{i=1}^{n} \rho(x_i \mid \boldsymbol{pa}_i)$.

EDAs are classified depending on the maximum number of dependencies between variables that they consider, typically divided into three categories [13]. The Univariate Marginal Distribution Algorithm for continuous domains (UMDA$_c$) is a representative example of **univariate** EDAs, where all variables are considered to be independent. In the second category, we have EDAs that can take into account **bivariate** conditional dependencies, where each variable can have a maximum of one parent variable. An example for continuous domains is the greedy algorithm called MIMIC (Mutual Information Maximization for Input Clustering), MIMIC$_c$. The third category is **multivariate** EDAs, where variables can have multiple parent variables. A representative of this category is EGNA (Estimation of Gaussian Network Algorithm) [12].

3 Hybridation of Estimation of Distribution Algorithms and Particle Swarm Optimization

The novelty of our approach consists on ensuring a full hybridation of both techniques by merging the sub-populations generated by EDA and PSO to take advantage of their respective advantages in optimization. Let consider that P is the number of particles in the search process, and that $R > P$ is the number of individuals in the population, out of which each generation a total of $N \leq R$ individuals will be selected to evolve to the next generation in EDAs. At the same time, the population will be divided in P chunks of same size R/P, and each chunk will be assigned to a different PSO particle for all the search process.

Since the population in EDAs is ordered according to the individuals' fitness values, each particle will represent a different swarm from the fittest to the least fit chunk of individuals. This configuration attempts to be a means to exit from local optima in case of too quick convergence, particularly for problems in which the fitness function is ambiguous (i.e there are different individuals with same fitness) or the fitness difference between not equally promising individuals does not sufficiently reflect it to guide the search process adequately.

The EDA-PSO approach will have the next steps:

1.-Initialization: Generate the first population D_0 of R individuals.

2.-Selection & partition: Select a number N $(N \leq R)$ of individuals for EDAs, and partition the population in R/P P chunks. Let's call $chunk_i$ to the i^th chunk, for $i = 1, 2, \cdots P$.

3.-EDA-Learning: Induce the n–dimensional probabilistic model of the N individuals.

4.-EDA-Simulation: Generate D_{l+1}^{EDA} of $R - P$ new individuals by simulating the probability distribution learned in the previous step.

5.-PSO-Position update: For each particle P_i $i_1 \ldots P$, update the $gbest_i$ and $lbest_i$ values using $chunk_i$.

6.-PSO-Create population: Generate the portion D_l^{PSO} of the new population of P new individuals from the new position of each particle.

7.-Fusion of populations: Build the new population $D_{l+1} = D_{l+1}^{EDA} \cup D_{l+1}^{PSO}$.

Steps 2 to 7 are repeated until a stopping condition is verified. The steps to generate the two parts of the new full population D_l are different for D_l^{EDA} and D_l^{PSO}. In the former the classical EDA approach is used, , while the P particles are kept separated ensuring that each is assigned to each chunk, so that all particles represent a different fitness category of individuals' swarm. Under this approach both PSO particles and EDA will take into account the individuals generated by both paradigms for the next generation.

As regards the PSO side, our approach is defined in such a way that when the search process is closer to the optimum the population should be more uniform in each chunk. This approach complements the ongoing research on the provision on PSO for the most adequate inertia weight in [6] and acceleration coefficients to particles during the search [19,22], since in our EDA-PSO algorithm these are regulated by means of the uniformity of the different population chunks.

We formulate this approach using the PSO's canonical version and the classical EDA approach for continuous domains, although the random sampling of uniform distributions is removed since the random component will be provided by the simulation of EDA's probabilistic graphical model every generation. Thus, given the velocity vector $\boldsymbol{V_i} = \left[v_i^1, v_i^2, \ldots v_i^n\right]$ and the position vector $\boldsymbol{POS_i} = \left[POS_i^1, POS_i^2, \ldots POS_i^n\right]$, then each generation the velocity and position of each particle will be updated according to the PSO standard formula.

Also, since EDAs usually include the best individual from D_l also in D_{l+1}, we propose a different *lbest* and *gbest* notion regarding the canonical PSO, such that $\boldsymbol{gbest_i}$ will be the global best of $chunk_i$, and to consider $\boldsymbol{lbest_i}$ as a combination of the current individuals of $chunk_i$.

4 Experiments

Experiments were carried out in order to measure both the behavior and performance of EDA-PSO when applied to classical optimization problems. The selected optimization problems are the ones proposed in [3] to compare evolutionary computation algorithms in continuous domains: Ackley [1], Griewangk [20], and Sphere model. These functions have been chosen due to their very different nature: the Sphere model does not contain any local optimum, Ackley presents several local optima, and Griewangk has still many more local optima although with smaller *valleys*:

1. **Ackley:** The fitness function of this minimization problem is defined as $F(\boldsymbol{x}) = -20 \cdot \exp\left(-0.2\sqrt{\frac{1}{n} \cdot \sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n} \cdot \sum_{i=1}^{n} \cos(2\pi x_i)\right) + 20 + \exp(1)$. The problem is defined with $-20 \leq x_i \leq 30$, $i = 1, \ldots, n$.
2. **Griewangk:** The fitness function of this minimization problem is defined as $F(\boldsymbol{x}) = 1 + \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right)$, where the range of all the components of the individual is $-600 \leq x_i \leq 600$, $i = 1, \ldots, n$.
3. **Sphere model:** This is a simple minimization problem defined for $-600 \leq x_i \leq 600$, $i = 1, \ldots, n$, with the fitness function $F(\boldsymbol{x}) = \sum_{i=1}^{n} x_i^2$.

For all problems the optimum value is 0, which is obtained when the individual has all its variables at 0.

For EDA-PSO the initial population was generated using random generation based on a uniform distribution, and the following combination of settings where tested for the different parameters: EDA type (UMDA$_c$, MIMIC$_c$, EGNA$_{ee}$); Problem size (n= 10, 30, 50); Number of particles (P=5, 10, 30, 50, 100); Population size (R=250); Selection size (N=100, 50). As regards the PSO part, the values set were the standard $\omega = 0.7$, $c1 = 0.1$ and $c2 = 0.2$. In EDA-PSO we choose that the the best individual of D_l is also included in D_{l+1}, so that 249 new individuals are generated per generation. The stopping criterion for all problems was satisfied when the optimum solution was found (assuming this case to be the case when the fittest individual has a fitness smaller than 10^{-6}), or when a maximum of 150 generations were reached.

Each algorithm and combination of parameters was run 20 times for each of the optimization problems, and the main results per algorithm and optimization problem are shown in Table 1. Due to lack of space, we show here the most representative results, which resulted in N=50 for UMDA$_c$ and MIMIC$_c$, and $N = 100$ for EGNA$_{ee}$.

The number of particles shows an important effect in the performance. In the case of Ackley, EDA-PSO results do not improve the standard EDA, but in the case of Griewangk and Sphere results are improved when $P = 10$, although they are worse for $P = 100$. It must be noted that for EGNA$_{ee}$ the values of R and N are too low to allow the EDA to learn adequately the dependencies between variables required to guide search adequately, and the table shows that in PSO contributes to help the algorithm to improve; however, in Ackley its performance is also very dependent on the number of particles P.

Table 1. Mean results of each of the problems and algorithms. Here are cases of $N = 100$ for EGNA$_{ee}$, and $N = 50$ for UMDA$_c$ and MIMIC$_c$. The *Gen.* column is the mean generations and *Eval.* corresponds to the mean number of evaluations. In cases in which not all executions converged, *Gen.* shows the convergence percentage and *Eval.* is the mean fitness obtained out of all executions.

	Ackley				Griewangk				Sphere			
	$n = 30$		$n = 50$		$n = 30$		$n = 50$		$n = 30$		$n = 50$	
EDA	Gen.	Eval.	Gen.	Eval	Gen.	Eval	Gen.	Eval	Gen.	Eval	Gen.	Eval
UMDA$_c$	42.9	86007	57.3	114193	26.1	52424	34.2	68616	24.8	49825	34	68216
MIMIC$_c$	42.9	86007	57.2	114592	25.9	52024	34.2	68616	24.4	49026	33.6	67416
EGNA$_{ee}$	70%*	0.72	0%*	2.9	28.8	57821	37.2	74613	27.2	54623	36.2	72614
EDA-PSO	Gen.	Eval.	Gen.	Eval	Gen.	Eval	Gen.	Eval	Gen.	Eval	Gen.	Eval
UMDA$_c$ & $P = 10$	42.8	86683	56.9	115151	25.3	51351	33.2	67301	24	48726	32.8	66493
MIMIC$_c$ & $P = 10$	42.7	86481	56.8	114949	25.4	51553	33	66897	24	48726	32.6	66089
EGNA$_{ee}$ & $P = 10$	90%*	0.6	0%*	1.8	28.2	57205	35.8	72550	26.9	54581	35.8	72550
UMDA$_c$ & $P = 50$	43	90607	57.2	120413	25	52825	32.6	68777	24	49466	32	67518
MIMIC$_c$ & $P = 50$	42.8	90187	61	128389	24.7	52195	32.8	67938	23	48627	32	67518
EGNA$_{ee}$ & $P = 50$	48.7	102571	0%*	1.8	27.7	58492	34.8	73395	26	54924	80%*	-5E-5
UMDA$_c$ & $P = 100$	42.8	94567	57.1	126013	25	55425	32.6	72137	23.6	52346	32	71258
MIMIC$_c$ & $P = 100$	42.6	94127	57.1	126013	25.3	56085	32.8	72577	23.7	52566	32.2	70818
EGNA$_{ee}$ & $P = 100$	90%*	1.4	0%*	0.6	27.7	61362	34.8	76975	26	57624	80%*	-1.2E05

This table also illustrates the complementarity of this hybrid EDA and PSO approach for optimization problems of different complexity. When there are no no local optima like in Sphere, the contribution of our hybrid approach does not show all its potential. For the case of Ackley where local optima have bigger *valleys* than in Griewangk, in all cases an increase of performance is shown with EDA-PSO over EDA, although the choice $P = 50$ appears the best for EGNA while UMDA performs better with $P = 10$. Finally, when having small local optima such as in Griewangk EDA-PSO allows convergence in less generations as well as a small reduction on the number of individuals evaluated to reach converge. In order to check statistical significance between algorithms' behavior the non-parametric tests of Kruskal-Wallis and Mann-Whitney were applied. The null hypothesis of the same distribution densities was tested between EDA and all EDA-PSO executions obtaining $p = 0.706$ for the number of generations and $p < 0.001$ for such of evaluations respectively. On the other hand, considering the cases of EDA and EDA-PSO with $P = 10$ $p < 0.001$ was obtained both for the number of generations and evaluations.

5 Conclusions and Future Work

This work demonstrates the validity of a new full hybrid EDA-PSO approach, analysing its performance depending on the nature and complexity of the optimization problem. Preliminary experimental results to compare the performance of this new approach on typical optimization problems in continuous domains have been shown, and they have been compared with such of continuous EDAs. At the light of the results we can conclude that EDA-PSO has proved to be a new paradigm capable of improving the results of continuous EDAs, where the number of particles have an important influence on the efficiency of the algorithm and can contribute to better efficiency when set adequately.

There is a lot of space for improvement and future research by applying diverse variants on how to hybridate EDA and PSO when updating each generation $\boldsymbol{lbest_i}$ which we are currently studying. Future work also is to be done with other well known optimisation problems such as Schwefel or Griewangk, and also other problems in which both EDA and PSO are known not to perform efficiently (such as satisfactibility problems).

Acknowledgments. This work has been partially supported by the Saiotek and Research Groups 2007-2012 (IT-242-07) programs (Basque Government), TIN2008- 06815-C02-01 and Consolider Ingenio 2010 - CSD2007- 00018 projects (Spanish Ministry of Science and Innovation) and COMBIOMED network in computational biomedicine (Carlos III Health Institute).

References

1. Ackley, D.H.: A Connectionist Machine for Genetic Hillclimbing. Kluwer, Dordrecht (1987)
2. Angeline, P.: Using selection to improve particle swarm optimization. In: Proceedings of the IEEE Congress on Evolutionary Computation, Anchorage AK, pp. 84–89 (1998)
3. Bengoetxea, E., Miquélez, T., Larrañaga, P., Lozano, J.A.: Experimental results in function optimization with EDAs in continuous domain. In: Larrañaga, P., Lozano, J.A. (eds.) Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation, pp. 181–194. Kluwer Academic Publishers, Dordrecht (2001)
4. van den Bergh, F., Engelbrecht, A.: A cooperative approach to particle swarm optimization. IEEE Transactions on Evolutionary Computation 8(3), 225–239 (2004)
5. Brits, R., Engelbrecht, A., van den Bergh, F.: Locating multiple optima using particle swarm optimization. Applied Mathematics and Computation 189(2), 1859–1883 (2007)
6. Chatterjee, A., Siarry, P.: Non linear inertia weight variation for dynamic adaptation in particle swarm optimization. Computers and Operations Research 33(3), 859–871 (2004)
7. Chen, Y., Peng, W., Jian, M.: Particle swarm optimization with recombination and dynamic linkage discovery. IEEE Transactions on Systems, Man, and Cybernetics 37(6), 1460–1470 (2007)
8. Eberhart, R., Kennedy, J.: A new optimizer using particle swarm theory. In: Proceedings of the 6th IEEE Symposium MIcromachines and Human Science (MHS), pp. 39–43 (1995)
9. Goldberg, D.E.: Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley, Reading (1989)
10. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings of the IEEE Conference on Neural Networks (ICNN), vol. 4, pp. 1942–1948 (1995)
11. Kulkani, R., Venayagamoorthy, G.: An estimation of distribution improved particle swarm optimization algorithm. In: Proceedings of the Third International Conference on Intelligent Sensors, Sensor Networks and Information (ISSNIP), pp. 539–544 (2007)

12. Larrañaga, P., Etxeberria, R., Lozano, J.A., Peña, J.M.: Optimization in continuous domains by learning and simulation of Gaussian networks. In: Proceedings of the Workshop in Optimization by Building and Using Probabilistic Models. A Workshop within the 2000 Genetic and Evolutionary Computation Conference, GECCO 2000, Las Vegas, Nevada, USA, pp. 201–204 (2000)
13. Larrañaga, P., Lozano, J.A.: Estimation of Distribution Algorithms. In: A New Tool for Evolutionary Computation. Kluwer Academic Publishers, Dordrecht (2001)
14. Lozano, J., Larrañaga, P., Inza, I., Bengoetxea, E.: Towards a New Evolutionary Computation. In: Advances in Estimation of Distribution Algorithms. Springer, Heidelberg (2006)
15. Miquélez, T., Bengoetxea, E., Mendiburu, A., Larrañaga, P.: Combining Bayesian classifiers and estimation of distribution algorithms for optimization in continuous domains. Connection Science 19(4), 297–319 (2007)
16. Mühlenbein, H., Paaß, G.: From recombination of genes to the estimation of distributions: I. Binary parameters. In: Ebeling, W., Rechenberg, I., Voigt, H.-M., Schwefel, H.-P. (eds.) PPSN 1996. LNCS, vol. 1141, pp. 178–187. Springer, Heidelberg (1996)
17. Parrott, D., Li, X.: Locating and tracking multiple dynamic optima by a particle swarm model using speciation. IEEE Transactions on Evolutionary Computation 10(4), 440–458 (2006)
18. Parsopoulos, K., Vrahatis, M.: On the computation of all global minimizers through particle swarm optimization. IEEE Transactions on Evolutionary Computation 8(3), 211–224 (2004)
19. Ratnaweera, A., Halgamuge, S., Watson, H.: Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients. IEEE Transactions on Evolutionary Computation 8(3), 240–255 (2004)
20. Törn, A., Žilinskas, A. (eds.): Global Optimization. LNCS, vol. 350. Springer, Heidelberg (1989)
21. Wang, J., Kuang, Z., Xu, X., Zhou, Y.: Discrete particle swarm optimization based on estimation of distribution for polygonal approximation problems. Expert Systems with Applications 36(5), 9398–9408 (2009)
22. Zhan, Z.H., Zhang, J., Li, Y., Chung, H.H.: Adaptive particle swarm optimization. IEEE Transactions on Systems, Man, and Cybernetics 39(6), 1362–1381 (2009)
23. Zhou, Y., Wang, J., Yin, J.: A discrete estimation of distribution particle swarm optimization for combinatorial optimization problems. In: Proceedings of the Third International Conference on Natural Computation (ICNC), vol. 4, pp. 80–84 (2007)