Proceeding Series of the Brazilian Society of Applied and Computational Mathematics, v. 6, n. 1, 2018.

Trabalho apresentado no XXXVII CNMAC, S.J. dos Campos - SP, 2017.

Proceeding Series of the Brazilian Society of Computational and Applied Mathematics

Clustering Search with Estimation of Distribution

Marcelo Branco do Nascimento[1]
Instituto de Ciência e Tecnologia, UNIFESP, São José dos Campos, SP
Alexandre César Muniz de Oliveira[2]
Departamento de Informática, UFMA, São Luis, MA

Abstract. Clustering Search (CS) is a hybrid optimization method which assists the discovery of promising search areas by dividing the search space. The search process is guided by a grouping mechanism that indicates where to perform local search. This paper proposes the use of estimation distribution to assist both the clustering and local search processes in order to reduce the computational effort to optimize combinatorial problems. Computational experiments and theoretical analyzes are used to validate the proposal.

Keywords. Clustering algorithms, Probability distribution, Optimization methods

1 Introduction

Clustering Search (CS) has been proposed as a generic way of combining metaheuristics and local search in which the search is intensified only in areas of the search space that deserve special attention [4,11]. The main idea is to identify promising areas of the search space by generating solutions through a metaheuristic and clustering them into groups that are further explored with local search heuristics. Center of clusters can be as a reference point or simple model to obtain the best solution on the search space framed by it.

Despite the greater number of CS applications, employing different metaheuristics to generate candidate solutions [3, 5, 7, 12], the local search algorithms are still computationally expensive problem-specific procedures. By the other hand, approaches based on Estimation Distribution Algorithms (EDA) [2, 8, 13, 16] claim to be able to find good solutions with less objective function calls [13]. In EDA, the evolutionary operators are replaced by probabilistic models that represent the set of promising solutions.

This work proposes the use of estimation distribution to assist the local search process, providing a general purpose, adaptive way of exploiting promising areas, also reducing the computational time for it. Clusters of candidate solution are represented now by probabilistic models, capable to generate other solutions inside the search subspace, meaning as a kind of non-problem-specific local search procedure. One can see the whole metaheuristic as multiples local estimation of distribution algorithms, exploring distinct promising search areas.

[1]marcelo.branco01@unifesp.br
[2]acmo@deinf.ufma.br

DOI: 10.5540/03.2018.006.01.0335

2 Previous approaches

Evolutionary Clustering Search (ECS) is a hybrid evolutionary algorithm that employs clustering to better explore search space. The ECS attempts to locate promising search areas by framing them by clusters that are represented by a *center*, c. The number of clusters $\mathcal{NC}$ can be fixed a priori [4] or dynamically determined according to the width of the search areas (size of the problem at hand) [12]. The cluster coverage is determined by a *distance metric* that computes the similarity between a given solution and the cluster center. Hamming and Euclidean distance metrics are popular ones [12].

ECS can be better understood, observing the four conceptually independent parts. *Evolutionary Algorithm component* (EA) works as a continuous generator of candidate solutions, evolving a population independently of the remaining components. Individuals (candidate solutions) are selected, recombined, and updated for future generations. *Iterative Clustering component* (IC) uses a distance metric to create and maintain clusters, based on the similarity of individuals generated by EA component. Each activated cluster receives proportionally votes. Non-activated clusters can be removed and the respective search area is forgotten. *Clustering Analyzer component* (CA) checks each the votes received by each cluster at regular generation intervals, indicating which are promising based on their *density* (number of votes received by the cluster). CA is also responsible for the removal of low density clusters. Finally, the *Local Search component* (LS) provides an exploitation mechanism in alleged promising areas, framed by the more voted clusters.

Estimation Distribution Algorithms replace evolutionary operators by building and sampling probabilistic models. The types of EDA can also be defined based on the need for explicit conservation of the population in each iteration, making the population preserved or eliminated completely during iterations of the algorithm [14]. A major advantage of using incremental EDA is the complexity of memory is significantly reduced, since the population is not completely stored, i.e, only representative individuals are maintained [1].

3 Estimation Distribution Clustering Search - EDCS

Consider an individual X consisting of $x_1, x_2, ..., x_i$ discrete values. EDCS maintains a population $P(t) = \{X_1(t), ..., X_N(t)\}$ and a set of clusters $C = \{c_1, c_2, ..., c_K\}$, where N is the number of individuals in Population at time t, and K is the number of clusters. Each cluster maintains a probability matrix given by:

$$\Omega_k(t) = \begin{pmatrix} p_{11k}(t) & p_{12k}(t) & ... & p_{1jk}(t) \\ p_{21k}(t) & p_{22k}(t) & ... & p_{2jk}(t) \\ ... & ... & & ... \\ p_{i1k}(t) & p_{i2k}(t) & ... & p_{ijk}(t) \end{pmatrix}, \quad (1)$$

where $\Omega_k(t)$ models the distribution of probability in search space of cluster k and $p_{ijk}(t)$ denotes the probability that x_i is assigned to position j in cluster k; the sum of the probabilities, shown in Equation 2, between discrete values for a position j must be equal to 1.

DOI: 10.5540/03.2018.006.01.0335

$$\sum_{i=1}^{n} p_{ijk} = 1 \tag{2}$$

The cluster status can be *PROMISING* or *EXHAUSTED*. The *PROMISING* state says the cluster is able to performing up to θ times the problem-specific local search. *EXHAUSTED* is set when the algorithm decides it is no longer possible to find a better solution. The value of θ indicates the amount of times that is held local search in each cluster configuration.

3.1 Initialization

Initialization is divided into two parts: population and clusters. To initialize the population, the algorithm generates randomly N solutions (individuals). In the second part, each cluster is initialized with a uniform probability distribution whose value is $1/n$. Furthermore, each cluster should store the best individual, represented by $B_k(t) = \{b_1(t), b_2(t), b_3(t), ..., b_K(t)\}$, found in that search area based on the local search engines applied. The status of the cluster to perform the local search is set, always beginning with status *PROMISING*.

3.2 Update of Probability Matrix

Each individual presented to the Iterative Clustering is assimilated by the more similar cluster and the probability matrix is updated.

Each individual of the population is grouped in a cluster based on a rank of similarity. This rank is calculated using a distribution probability with independent variables shown in Equation 3. This probability specifies the chance of a solution belong to a cluster k.

$$p(X|c_k) = p(x_1, ..., x_n|c_k) = \prod_{j=1}^{n} p_{ijk} \tag{3}$$

The individual is assimilated to the cluster which has the highest rank (more similar). The probability of X belonging to a cluster c_k is represented by $p(X|c_k)$.

When the individual is assimilated to a cluster, the probability matrix is updated based on the following rule:

$$\begin{cases} p_{ijk}(t+1) = p_{ijk}(t) \times \frac{D_k(t)}{D_k(t)+1} + \frac{1}{D_k(t)+1}, & \forall\ x_i = j \\ p_{ijk}(t+1) = p_{ijk}(t) \times \frac{D_k(t)}{D_k(t)+1}, & \forall\ x_i \neq j \end{cases} \tag{4}$$

The value $D_k(t)$ is the density of cluster k in time t, in other words, the amount of assimilated individuals.

This rule of Equation 4 is based on Population-Based Incremental Learning (PBIL) [1]:

$$p(t+1) = p(t) \times (1-\gamma) + \gamma \times \sigma. \tag{5}$$

DOI: 10.5540/03.2018.006.01.0335

for $\gamma \in (0, 1]$. The value γ is the learning rate.The bigger γ is,the more contribution of strings parents to the updated probability vector. The value σ is related to the sum of occurrences of the value guided by $p(t)$.

Unlike the PBIL, Equation 4 also updates the probabilities where $x_i \neq j$, for the purpose of maintaining the condition in Equation 2. This condition is necessary to calculate rank in Equation 3.

3.3 Guided Mutation to generate of new individuals

Guided Mutation [16] is used to mutate an existing individual to provide a new individual on the basis of a set of probabilities p_k. The factor $0 < \alpha < 1$ controls the amount of change that the individual receives.

Randomly two groups are formed . The first group has αN values. The second has $(1 - \alpha)N$ values. The values of the first group are assigned directly to the new individual, while the second group elements are rearranged based on probability matrix of each cluster.

The guided mutation is performed in two different situations. The first happens when a cluster is to status EXHAUSTED, and the other is on the remaining individuals of the population. This strategy allows to search other solutions in different regions not yet explored in the search space. If the cluster is *EXHAUSTED* Guided Mutation can bring it from the local minimum and allow it to turn on again.

4 Computational Experiments

To validate the proposal, a recently explored NP-Hard permutation problem was chosen: Minimization of Open Stacks Problem (MOSP). MOSP is strongly linked with other industrial problems [9]. Evolutionary Clustering Search is applied to Gate Matrix Layout Problem (GMLP), besides MOSP and the best results found in literature were reached. An application of EDCS to MOSP/GMLP allows a straight comparison with the best approach found in literature [11].

A Minimization Open Stack Problem (MOSP) occurs in certain situations in industrial production, where items with different specifications (shapes, sizes, etc.) need to be produced and temporarily kept in yards during production. The items are cut and placed in stacks according to their size (a stack for each type of item). Each stack is open as long as there are elements to be cut of the same item. Its goal is to arrange a set of items such that the number of open stack at same time is minimized [9].

GMLP is quite similar to MOSP, but it is related to other conceptual elements, semantically different. Its goal is to arrange a set of gates such that the number of tracks necessary to cover the gates interconnection (nets) is minimized [9].

One must determine the sequence of cutting patterns that minimizes the maximum of stacks open during the cutting process. Typically, this is a problem that occurs due the limitations of physical space, since the accumulation of stacks may cause the need temporarily removing one or the other stack, slowing the production process.

DOI: 10.5540/03.2018.006.01.0335

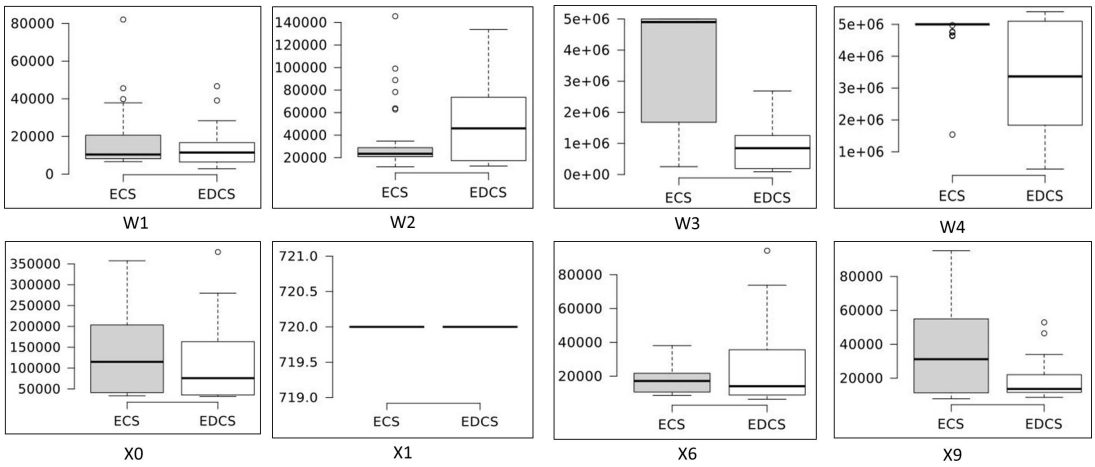

Figure 1: Boxplot Results for objective function calls

4.1 Description of Experiments

In this section, early Evolutionary Clustering Search (ECS) and newly proposed Estimation Distribution Clustering Search (EDCS) are both applied to MOSP/GMLP instances found in the literature and their performance are compared. ECS represent clusters by a candidate solution coded by a permutation of integer, and performing local search by a 2-Opt hill-climbing strategy. The ECS results are up to now the best found in literature [11] .

The hardest MOSP/GMLP problem instances reported in literature [10] are: w1(21), w2(33), w3(70) and w4(140), x0(48), x1(10), x6(32), x9(32). The number of gate/pattern, i.e., instance size appears in round brackets. The ECS parameters were set similarly to [11], while EDCS had the following set: $N = 720$, $K = 30$, $\alpha = 0.8$, $\theta = 10$.

4.2 Results

The results were obtained allowing all approaches to perform up to 5×10^6 objective function calls (FC) in each one of 30 trials. The success rate (SR) was calculated, as well as the objective function calls average $\overline{FC}$.

By ANOVA [15] analysis in Table 1, one can verify that there is significant statistic difference on instances with large numbers of variables (w3 and w4) [6]. In Figure 1, the statistic difference is highlighted.

In Table 1, SR and $\overline{FC}$ are presented and the best ones are highlighted for each problem instance. One can observe that EDCS's SR is better for the two largest instances: w3 and w4; for which $\overline{FC}$, evidently, is the best too. For the other instances, when both has the same SR, the ECS's computational effort to reach the best solution ($\overline{FC}$) is lesser for w2 and x6 instances.

DOI: 10.5540/03.2018.006.01.0335

Table 1: One-way ANOVA with SR and $\overline{FC}$ by ECS and EDCS

Instance	Lenght	F statistic	p-value	$SR(\%)$ ECS	$\overline{FC}$ ECS	$SR(\%)$ EDCS	$\overline{FC}$ EDCS
W1	21	1.2868	0.2613	100	17872.433	100	**13804.533**
W2	33	3.9513	0.0516	100	**35996.4**	100	53200
W3	70	50.9758	1.7072e-09	40	3651220.3	**86.667**	**960320.77**
W4	141	18.9383	5.5474e-05	3.333	4831960.1	**60**	**3310243.9**
X0	48	0.9609	0.3310	100	135056,267	100	**110957.2**
X1	10	0	1	100	720	100	720
X6	32	3.1211	0.0825	100	**17639.267**	100	25598.3
X9	32	13.1676	0.0006	100	38625.833	100	**18172.033**

5 Conclusion

Clustering Search (CS) is a framework for combining exploration and exploitation optimization methods and incorporates explicitly the concept of promising regions of search space. Despite the meaningful number of existing CS applications, the CS main features as cluster representation, assimilation and local search procedures are all problem-specific proposals. This work goes in the direction of replacing some of these problem-specific elements for general general purpose and adaptive way of exploiting promising areas. For this reason, a Clustering-Search based on Estimation Distribution Algorithm is proposed and applied to a permutation problem found in literature. Computational experiments validate the initial assumption that new approach is competitive for dealing with permutation problems.

In future works, new refinements are needed to consolidate the proposal. For example, new applications for unconstrained continuous optimization as well as other combinatorial optimization problem. Local search procedure must be reformulated to avoid definitely problem-specific heuristics.

Acknowledgment

The authors thank CNPq, Proc.481845/2013-5 for the partial funding of this research.

References

[1] S. Baluja. Population-based incremental learning: A method for integrating genetic search based function optimization and competitive learning. Technical report, Pittsburgh, PA, USA, 1994.

[2] E. Bengoetxea. Inexact graph matching by means of estimation of distribution algorithms. *Pattern Recognition*, 35(12):2867-2880, 2002.

[3] A. A. Chaves and L. A. N. Lorena. Hybrid evolutionary algorithm for the capacitated centered clustering problem, *Expert Systems with Applications*, 38(5):5013-5018, 2011.

[4] A. A. Chaves and L. A. N. Lorena, Hybrid metaheuristic for the prize collecting travelling salesman problem, *Lecture Notes in Computer Science*, 4972:123-134, 2008. DOI: 10.1007/978-3-540-78604-7_11.

[5] T. S. Costa and A. C. M. de Oliveira, Artificial bee and differential evolution improved by clustering search on continuous domain optimization, *Soft Computing*, 19(9):2457-2468, 2015. DOI: 10.1007/s00500-014-1500-9.

[6] J. Demšar. Statistical comparisons of classifiers over multiple data sets, *The Journal of Machine Learning Research*, 7:1-30, 2006.

[7] G. R. Filho, M. S. Nagano, and L. A. N. Lorena. Evolutionary clustering search for flowtime minimization in permutation flow shop. In *Proceedings of the 4th international conference on Hybrid metaheuristics*, 4771:69-81. Springer, 2007.

[8] P. Larraanaga and J. A. Lozano. Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation. In *Genetic Algorithms and Evolutionary Computation*. Springer, 2001. ISSN:1568-2587.

[9] A. Linhares and H. H. Yanasse. Connections between cutting-pattern sequencing, VLSI design, and flexible machines. *Computers & Operations Research*, 29(12):1759-1772, 2002.

[10] A. Linhares, H. H. Yanasse, and J. R. A. Torreao Linear gate assignment: a fast statistical mechanics approach, *IEEE Trans. on CAD of Integrated Circuits and Systems*, 18(12):1750-1758, 1999. DOI: 10.1109/43.811324.

[11] A. C. M. de Oliveira, A. and L. A. N. Lorena, Hybrid evolutionary algorithms and clustering search, *Hybrid Evolutionary Algorithms of the Studies in Computational Intelligence*, 75:77-99, 2007. DOI: 10.1007/978-3-540-73297-6_4

[12] A. C. M. de Oliveira and L. A. N. Lorena, Detecting Promising Areas by Evolutionary Clustering Search, *Lecture Notes in Computer Science*, 3171:385-394, 2004. DOI: 10.1007/978-3-540-28645-5_39.

[13] M. Pelikan, Analysis of estimation of distribution algorithms and genetic algorithms on NK landscapes, *Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation*, 10:1033-1040, 2008. DOI: 10.1145/1389095.1389287

[14] M. Pelikan, K. Sastry, and D. E. Goldberg, iBOA: The Incremental Bayesian Optimization Algorithm, *Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation*, 10:455-462, 2008. DOI: 10.1145/1389095.1389177.

[15] W. Penny and R. Henson, Analysis of variance, *Statistical Parametric Mapping: The analysis of functional brain images*, volume 1, chapter 13, pages 166-177, 2006.

[16] Q. Zhang, J. Sun, and E. Tsang, An evolutionary algorithm with guided mutation for the maximum clique problem, *IEEE Transactions on Evolutionary Computation*, 9(2):192-200, 2005. DOI: 10.1109/TEVC.2004.840835.

