DMC-GRASP: A Continuous GRASP hybridized with Data Mining

Raphael Gomes Santos, Alexandre Plastino
Instituto de Computação
Universidade Federal Fluminense (UFF)
Niterói, Brazil
raphael.gomes@id.uff.br, plastino@ic.uff.br

Alexandre C. M. de Oliveira
Departamento de Informática
Universidade Federal do Maranhão (UFMA)
São Luís, Brazil
acmo@deinf.ufma.br

Confirm Centre for Smart Manufacturing
School of Computer Science
University College Cork
Cork, Ireland
amunizdeoliveira@ucc.ie

2022 IEEE Congress on Evolutionary Computation (CEC) | 978-1-6654-6708-7/22/$31.00 ©2022 IEEE | DOI: 10.1109/CEC55065.2022.9870264

***Abstract*—The hybridization of metaheuristics with data mining techniques has been successfully applied to combinatorial optimization problems. Examples of this type of strategy are DM-GRASP and MDM-GRASP, hybrid versions of the Greedy Randomized Adaptive Search Procedure (GRASP) metaheuristic, which incorporate data mining techniques. This type of hybrid method is called Data-Driven Metaheuristics and aims at extracting useful knowledge from the data generated by metaheuristics in their search process. Despite success in combinatorial problems like the set packing problem and maximum diversity problem, proposals of this type to solve continuous optimization problems are still scarce in the literature. This work presents a data mining hybrid version of C-GRASP, an adaptation of GRASP for problems with continuous variables. We call this new version DMC-GRASP, which identifies patterns in high-quality solutions and generates new solutions guided by these patterns. We performed computational experiments with DMC-GRASP on a set of well-known mathematical benchmark functions, and the results showed that metaheuristics for continuous optimization could also benefit from using patterns to guide the search for better solutions.**

***Index Terms*—Metaheuristics, Data Mining, Continuous Optimization.**

I. Introduction

Many real-world problems can be formulated as optimization problems with variables in continuous domains, i.e., as continuous optimization problems. These problems, in general, are challenging to solve. This difficulty has given rise to several proposals for optimization techniques that seek sub-optimal solutions. In this context, metaheuristics arise, which are general-purpose heuristics to solve complex optimization problems that can not be solved using approximate or gradient-based optimization techniques [2].

In this work, we focus on Global Optimization. Formally, in minimization form, the continuous global optimization seeks for a solution $\mathbf{x}^* \in S \subseteq \mathbb{R}^n$ such that $f(\mathbf{x}^*) \leq f(\mathbf{x}), \forall \mathbf{x} \in S$, where S is a subset of $\mathbb{R}^n$ and the objective function is $f : S \rightarrow \mathbb{R}$. The solution $\mathbf{x}^*$ is called the global minimum. A solution $\mathbf{x}$' is called a local minimum in a neighborhood $S_0 \subseteq S$ if $f(\mathbf{x}') \leq f(\mathbf{x})$, $\forall \mathbf{x} \in S_0$. We consider there are no requirements on the properties of f. In particular, it may have a nonlinear or non-differentiable form, for which there are no classical gradient-based optimization techniques for obtaining optimal solutions. Global optimization problems appear in many areas such as materials science, biology, chemistry and genetics, electrical engineering, robotics, and transportation science [1]. One example of metaheuristic proposed to solve continuous global optimization is the Continuous GRASP (C-GRASP), which is an adaptation of GRASP metaheuristic to problems with continuous variables.

In this work, to deal with continuous optimization problems, we explore the hybridization of metaheuristics methods with data mining techniques. In this context, data mining aims at finding patterns in good solutions generated by heuristics. This idea has been successfully applied to different combinatorial optimization problems [12]–[16]. In [10], Plastino et al. proposed the DM-GRASP, the hybridization of GRASP [8], a multi-start metaheuristic, with data mining techniques, achieving good and promising results. Inspired by this work, we present the DMC-GRASP, a hybridization of C-GRASP [5] with Data Mining, to be applied to optimization problems with continuous variables.

This paper is organized as follows. In section II, we present the related work. In section III, we describe the C-GRASP metaheuristic. In section IV, we describe our proposal – the hybridization of C-GRASP with data mining, called DMC-GRASP. In section V, the different versions of DMC-GRASP are compared with each other and against C-GRASP. Section VI presents our conclusions and points out directions for future work.

II. Related Work

Recently, hybrid metaheuristics have been proposed to use explicit knowledge discovered during the search using advanced machine learning (ML) models or data mining (DM) techniques. Metaheuristics generate many data in the search

process. The data can be static – when it concerns the target problem and instance features. Moreover, several dynamic data are generated during the iterative search process, such as solutions in the decision and the objective spaces, sequence of solutions or trajectories, successive populations of solutions, moves, recombinations, local optima, elite solutions, and bad solutions [22]. Thus ML and DM can help analyze these data to extract useful knowledge. In [22] metaheuristics that use built-in ML or DM techniques in their search are called data-driven metaheuristics.

Following these ideas, some works have proposed the hybridization of metaheuristics with DM to solve complex optimization problems. The hybridization of GRASP metaheuristic with a data mining process was first introduced and successfully applied to the set packing problem [10], and to the maximum diversity problem [11]. The data mining technique was applied in the construction phase in both works. The hybrid proposal was divided into two parts. In the first one, a number of GRASP iterations are executed, and the best solutions are stored in an elite set. Then, a data mining algorithm is used to extract patterns from this set of sub-optimal solutions. In the second part, the GRASP iterations use the mined patterns to construct new solutions. In this strategy, the data mining process is performed only once, after exactly half of the GRASP iterations. This strategy was called DM-GRASP. A similar hybridization strategy was applied to other metaheuristics, as Iterated Local Search (ILS), to solve the set covering with pairs problem and a state-of-art hybrid heuristic for solving the classical p-median problem [13].

Plastino et al [12] proposed the Multi Data Mining GRASP (MDM-GRASP), an extension of DM-GRASP which executes the data mining procedure not just once but many times in an adaptive way during the heuristic execution. The main idea of MDM-GRASP is to execute the mining process: (a) as soon as the elite set becomes stable – which means that no change in the elite set occurred throughout a given number of iterations – and (b) whenever the elite set has been changed and again has become stable. Their hypothesis is that mining more than once will explore the gradual evolution of the elite set and allow the extraction of refined patterns.

In the context of data-driven metaheuristics to continuous optimization problems, the Estimation of Distribution Algorithms (EDAs) stand out. EDAs are stochastic optimization techniques that explore the solution space by building and sampling explicit probabilistic models of promising candidate solutions. EDAs use the models to guide the search for better solutions. The works [25]–[27] generate new solutions based on the Gaussian distribution model. Others [28]–[30] are based on histogram models. Although these works are based on the approach of following the previously generated data to sample new solutions, they differ from this present work because they do not present the idea of patterns. Our proposal explicitly uses pattern, which will compose the start point of the search in iterations after the data mining. The underlying idea is that we can discover features of good solutions in the data and build new good solutions directly upon these features. This kind of proposal to solve continuous optimization problems is still scarce.

III. Continuous GRASP

C-GRASP [5]–[7] is a version of GRASP adapted to continuous optimization. As GRASP [3], [4], [8], C-GRASP is a multi-start metaheuristic that uses a greedy randomized construction procedure to generate starting solutions and a local search procedure to improve them. C-GRASP works by discretizing the solution domain into a uniform grid. The construction and local search procedures move along points on the grid. As the algorithm iterates, the grid adaptively becomes denser. Algorithm 1 shows the pseudocode of C-GRASP.

C-GRASP procedure takes as input the problem dimension n, lower and upper bound vectors $\mathbf{l}$ and $\mathbf{u}$, the objective function $f(.)$, as well as the parameters h_s, h_e and ρ_{lo}. Parameters h_s and h_e define the starting and ending grid discretization densities, while ρ_{lo} defines the portion of the neighborhood of the current solution that is explored during the local improvement procedure.

The algorithm processes the iterations while the stopping criteria are not satisfied. At each iteration, an initial solution is set to a random point distributed uniformly over the box defined by $\mathbf{l}$ and $\mathbf{u}$. The h parameter, which represents the current density of the grid, is reset to h_s. The construction and local search procedures are called sequentially, in a loop, generating *internal iteration*.

The solution returned from the local improvement procedure is compared against the current best solution. If the returned solution is better than the current best solution, then the current best solution is updated with the returned solution. If the variables `ImprC` and `ImprL` are still set to false, the grid density is decreased by halving h.

The variable `ImprC` and `ImprL` are false when no improvement is made in the construction and local search procedures, respectively. An *external iteration* ends when $h < h_e$. At the time the stopping criteria are satisfied, the best solution found is returned.

IV. Continuous Grasp hybridized with Data Mining

In the previously developed hybrid data mining metaheuristics [13], the patterns are composed of a set of solution components that frequently appear together in the elite solutions. The pattern search characterizes, therefore, a frequent itemset mining application. A frequent itemset mined with support $s\%$ represents a set of elements that occur in $s\%$ of the elite solutions [24]. In our hybridization proposal, differently, patterns are clusters of points highly concentrated, that is, with low dispersion. We search for these clusters in the set of the best solutions generated by the heuristic. The low dispersion indicates that the good solutions tend to be close to a determined point in space. Thus, these good solutions are similar, representing a pattern. The similarity between solutions can be computed using the euclidian distance.

Algorithm 1 Continuous GRASP

```
1: procedure C-GRASP(n, l, u, f(.), h_s, h_e, ρ_lo)
2:     f* ← ∞
3:     while Stopping criteria are not met do                    ▷ external iteration
4:         x ← UNIFRAND(l, u)
5:         h ← h_s
6:         while h ≥ h_e do                                      ▷ internal iteration
7:             ImprC ← false
8:             ImprL ← false
9:             [x, ImprC] ← CONSTRUCTGREEDYRANDOMIZED(x, f(.), n, h, l, u, ImprC)
10:            [x, ImprC] ← LOCALIMPROVEMENT(x, f(.), n, h, l, u, ρ_lo, ImprL)
11:            if f(x) < f_* then
12:                x* ← x
13:                f* ← f(x)
14:            end if
15:            if ImprC = false and ImprL = false then
16:                h ← h/2                                       ▷ making grid denser
17:            end if
18:        end while
19:    end while
20:    return x*
21: end procedure
```

Algorithm 2 Construct new solution from pattern p

```
1: procedure CONSTRUCTFROMPATTERN(p, l, u)
2:     for i ∈ D do
3:         if i ∈ p then
4:             x_i = p_i
5:         else
6:             x_i = UNIFRAND(l_i, u_i)
7:         end if
8:     end for
9:     return x
10: end procedure
```

As in the proposed hybridization of GRASP with data mining techniques [10] to solve combinatorial problems, the new hybridized C-GRASP will have two phases: The first one is called the *elite set generation phase*, which consists of executing pure C-GRASP iterations to obtain a set of different solutions. The d best solutions from this set compose the elite set. Next, the second phase, called *hybrid phase*, is performed. In this part, different C-GRASP iterations are executed. In these iterations, the construction of a new solution is guided by a mined pattern. Initially, all elements of the selected pattern are inserted into the partial solution. Then, a complete solution will be built, choosing the uniform random values for variables not present in the pattern. This procedure called CONSTRUCTFROMPATTERN is described in Algorithm 2. This way, all initial solutions will contain the elements of the selected pattern. Then the iteration proceeds as a standard C-GRASP iteration.

We called the C-GRASP hybridized with Data Mining by DMC-GRASP. The Algorithm 3 shows the entire DMC-GRASP procedure. Note that for DMC-GRASP, we will need a stopping criterion for the *elite set generation phase* and another to *hybrid phase*. The stop criterion for the *elite set generation phase* will be to reach a percentage (ρ_{fe}) of the maximum resource budget of the algorithm. The *hybrid phase* will consume the leftover budget. This resource budget can be a maximum number of iterations or a maximum of objective function evaluations. In the next subsections, we will specify the proposed ideas to define the procedure EXTRACTPATTERN in line 23 of DMC-GRASP, which is supposed to extract patterns from the elite set.

A. Average and Standard Deviation

The average value of a set of numbers indicates the central tendency of the data. It is the most likely value to be observed. The dispersion of data around the mean is characterized by the standard deviation measure. A low standard deviation indicates the values tend to be close to the mean of the set, while a high standard deviation indicates that the values are spread out over a broader range [17]. Thus, we can use the mean and standard deviation as concentration measures and then capture the central tendency of data.

Therefore, we can characterize the solution pattern as the mean of the solutions in the elite set. We call pattern here the vector $\mathbf{p}$ of size $|D|$, in which, for each $i \in D$, $\mathbf{p}_i = \mu(E_i)$, that is, the coordinate i of pattern is the mean of values in coordinate i of solutions in the elite set E. Where D is the set of dimensions of the problem and $\mu(E_i)$ is the average of values in dimension i of E. Considering this pattern, the hybrid C-GRASP, in his second phase, builds the new solution

Algorithm 3 Continuous GRASP + Data Mining

procedure DMC-GRASP($n, \mathbf{l}, \mathbf{u}, f(.), h_s, h_e, \rho_{lo}, \rho_{fe}$)
  $f^* \leftarrow \infty$
  $E \leftarrow \emptyset$
  while Stopping criteria 1 are not met **do** ▷ elite set generation phase
    $\mathbf{x} \leftarrow$ UNIFRAND($\mathbf{l}, \mathbf{u}$)
    $h \leftarrow h_s$
    while $h \geq h_e$ **do**
      ImprC $\leftarrow$ **false**
      ImprL $\leftarrow$ **false**
      [$\mathbf{x}$, ImprC] $\leftarrow$ CONSTRUCTGREEDYRANDOMIZED($\mathbf{x}, f(.), n, h, \mathbf{l}, \mathbf{u}$, ImprC)
      [$\mathbf{x}$, ImprC] $\leftarrow$ LOCALIMPROVEMENT($\mathbf{x}, f(.), n, h, \mathbf{l}, \mathbf{u}, \rho_{lo}$, ImprL)
      if $f(x) < f_*$ **then**
        $\mathbf{x}^* \leftarrow \mathbf{x}$
        $f^* \leftarrow f(\mathbf{x})$
      end if
      $E \leftarrow$ UPDATEELITESET($E, \mathbf{x}$)
      if ImprC = false **and** ImprL = false **then**
        $h \leftarrow h/2$
      end if
    end while
  end while
  while Stopping criteria 2 are not met **do** ▷ hybrid phase
    $\mathbf{p} \leftarrow$ EXTRACTPATTERN(E)
    $\mathbf{x} \leftarrow$ CONSTRUCTFROMPATTERN($\mathbf{p}, \mathbf{l}, \mathbf{u}$)
    $h \leftarrow h_s$
    while $h \geq h_e$ **do**
      ImprC $\leftarrow$ **false**
      ImprL $\leftarrow$ **false**
      [$\mathbf{x}$, ImprC] $\leftarrow$ CONSTRUCTGREEDYRANDOMIZED($\mathbf{x}, f(.), n, h, \mathbf{l}, \mathbf{u}$, ImprC)
      [$\mathbf{x}$, ImprC] $\leftarrow$ LOCALIMPROVEMENT($\mathbf{x}, f(.), n, h, \mathbf{l}, \mathbf{u}, \rho_{lo}$, ImprL)
      if $f(\mathbf{x}) < f_*$ **then**
        $\mathbf{x}^* \leftarrow \mathbf{x}$
        $f^* \leftarrow f(\mathbf{x})$
      end if
      $E \leftarrow$ UPDATEELITESET($E, \mathbf{x}$)
      if ImprC = false **and** ImprL = false **then**
        $h \leftarrow h/2$
      end if
    end while
  end while
  return $\mathbf{x}^*$
end procedure

starting from the average of the best solutions found in his first phase.

In some cases, we could have some dimensions that are not sufficiently concentrated. We do not have enough evidence to define a pattern in this case. We can use the standard deviation to select only dimensions where the solution set is sufficiently concentrated around the average value to avoid this problem. Now, the pattern will be the sub-vector of $\mathbf{p}$ composed of coordinates i where $\sigma(E_i) \leq \tau$. Here, $\sigma(E_i)$ is the standard deviation in values of dimension i of the elite set E, and the parameter τ is the threshold that indicates low dispersion. This mechanism aims at selecting only the variables sufficiently concentrated for composing the pattern. We call the version of DMC-GRASP with this pattern extraction strategy by DMC-GRASP.

B. Clustering by dimension

In some cases, the mean and standard deviation may not be adequate to identify patterns in the data. The data could be distributed in clusters spread out in the search space. In the problems we are focusing on, the search space, in

general, has many minimums, and we expect that the set of good solutions will be positioned near these minimums. We propose then using clustering algorithms to identify groups of points concentrated in different regions of the search space. Clustering is an unsupervised machine learning task that aims at partitioning the input data set into subsets (clusters), so these data in each subset share common aspects [18].

Therefore, following this idea to extract patterns, for each dimension i, we first cluster the data in dimension i of solutions in the elite set. From the greatest cluster, we get the centroid κ_i, which will be part of the pattern. So, $\mathbf{p}_i = \kappa_i$. This process aims at getting the mean of a more dense set of points, eliminating the influence of outliers to the mean. To cluster the data we use the X-means algorithm [21], which is an extension of the K-means algorithm [19] that estimates the best number of clusters k to fit the data. This estimation is based on Bayesian Information Criterion (BIC) or the Akaike Information Criterion (AIC) measure. They are well-known model selection approaches that evaluate how well a model fits the data. They penalize models for having more parameters, so these criteria will select models with fewer parameters. The DMC-GRASP with this pattern extraction strategy will be called DMC-GRASP-X.

C. Multidimensional Clustering

The previous methods focused on the view and analysis of each dimension, separately, to extract patterns. Now, we propose clustering in the multidimensional space of the solutions. This strategy tends to better reflect the distribution of elite solutions. So, now, before extracting the pattern, we cluster the elite set in the n-dimensional space, getting the centroid κ^n from a selected cluster to be the pattern. To select the cluster, we have three proposals. The first one is to select the greatest cluster, and we call this approach by DMC-GRASP-MX. The second selection strategy is randomly choosing a cluster, and we call this by DMC-GRASP-RX. The third one is to put all clusters in a queue to be used in a round-robin way. In each iteration, in the second phase of DMC-GRASP, we get the first element in the cluster queue, and after this, that cluster goes to the end of the queue, This third approach is named DMC-GRASP-QX. From the selected cluster, we get the centroid as the pattern. We use the standard deviation threshold to filter only the dimensions with low dispersion.

V. Computational Experiments

In this section, we present the computational experiments performed to evaluate the DMC-GRASP algorithm. We compare all versions of DMC-GRASP with C-GRASP. To tune the parameters of DMC-GRASP, we used irace [23]. Table I shows the parameters used in the following experiments. In these experiments, we introduced an extra parameter, denoted here by $|\mathbf{p}|$ to determine the portion of the pattern which can be used. The idea is to verify if using parts of patterns is better than using the whole pattern.

Initially, we compare C-GRASP with the five versions of DMC-GRASP described in Section IV. The comparison was made in a set of 16 well-known benchmark functions. Table II shows the comparison among the algorithms, considering the average of the best solution found. Each algorithm was run 30 times, with a different seed to each execution. To this experiment, the parameters h_s, h_e and ρ_{lo} were set to 0.5, 0.01, 0.7 respectively. C-GRASP was set to stop when it reaches 20 external iterations or finds a solution $\mathbf{x}$ such that $f(\mathbf{x})$ is significantly close to the global minima $f(\mathbf{x}^*)$. As in [5], [6], [9] be significantly close means:

$$|f(\mathbf{x}^*) - f(\mathbf{x})| \leq \epsilon_1 |f(\mathbf{x}^*)| + \epsilon_2 \quad (1)$$

where $\epsilon_1 = 10^{-4}$ and $\epsilon_2 = 10^{-6}$. After running C-GRASP, we catch his average number of objective function evaluations and set it as stop criterion for all DMC-GRASP versions as well as the best solution found to be significantly close to global minima.

TABLE I
DMC-GRASP PARAMETERS

algorithm	parameters			
	$\|E\|$	ρ_{fe}	$\|\mathbf{p}\|$	τ
DMC-GRASP	40	0.4	1	0.685
DMC-GRASP-X	40	0.1	0.8	0.0814
DMC-GRASP-MX	70	0.1	0.9	0.0712
DMC-GRASP-RX	70	0.4	0.9	0.3723
DMC-GRASP-QX	60	0.6	1	1

The numbers in bold indicate the best result for a function among the six algorithms in Table II. The underlined numbers indicate for which function the corresponding DMC-GRASP version performed better than C-GRASP. In turn, the results marked with '*' are statistically significant. We have used the *Wilcoxon signed-rank Test* [20]. The results have shown that, except for the BOHACHESKY function, for all functions, at least one version of DMC-GRASP has performed better than C-GRASP in the direct comparison of means.

We highlight the DMC-GRASP-QX results, which are better than C-GRASP ones in 12 test functions, including 10 with statistical significance. We also highlight that DMC-GRASP-QX version wins C-GRASP with statistical significance to all functions with $|D| \geq 5$.

Table III shows the comparison of the best solutions found by the algorithms. In this case, only in the POWERSUM function the C-GRASP finds the better result, however tied with three versions of the DMC-GRASP. For all other functions, at least one DMC-GRASP version has found a better solution than C-GRASP. In this evaluation, DMC-GRASP-X has found a better solution than C-GRASP in 14 test functions.

To analyze the effect of data mining on the algorithm behaviour, we plot the mean cost values of solutions generated by the construction procedure and local search with respect to the 30 seeds used. Figures 1 and 2 shows the behaviours of C-GRASP and DMC-GRASP-QX concerning the test functions $\text{HARTMANN}_{6,4}$ and RASTRINGIN, respectively. The vertical blue line in the graphs of DMC-GRASP-QX marks the average iteration where the mining starts. The red regions show

TABLE II
COMPARISON OF MEANS

f	$\|D\|$	C-GRASP	DMC-GRASP	DMC-GRASP-X	DMC-GRASP-MX	DMC-GRASP-RX	DMC-GRASP-QX
BOHACHEVSKY	2	**5.9394e-05**	8.8393e-05	9.0882e-05	1.2660e-04	9.1861e-05	7.5870e-05
GOLDSTEINPRICE	2	3.0017e+00	3.0004e+00*	3.0016e+00	3.0014e+00	3.0015e+00	**3.0003e+00***
COLVILLE	4	4.5945e-04	**2.0047e-04***	4.8941e-04	4.4075e-04	4.9144e-04	2.5488e-04*
POWERSUM	4	3.8587e-04	7.0311e-04	4.7032e-04	**3.2270e-04**	5.3069e-04	6.2174e-04
SHEKEL$_{4,5}$	4	-9.9809e+00	-9.9013e+00	-9.0680e+00	-8.5699e+00	-9.9022e+00	**-9.9822e+00***
SHEKEL$_{4,7}$	4	-1.0399e+01	-1.0223e+01	-9.7004e+00	-9.5249e+00	**-1.0401e+01***	-1.0400e+01*
SHEKEL$_{4,10}$	4	-1.0532e+01	-9.9938e+00	-8.7981e+00	-9.0625e+00	-1.0131e+01	**-1.0533e+01***
PERM	4	8.8594e-03	**8.4339e-03**	6.9338e-02	1.3755e-01	8.2989e-02	9.8245e-03
PERM$_0$	4	8.8594e-03	**8.4339e-03**	6.9338e-02	1.3755e-01	8.2989e-02	9.8245e-03
ROSENBROCK	5	2.2598e-02	2.3066e-02	**6.3716e-03***	8.7730e-03*	1.1202e-02*	1.4140e-02*
HARTMANN$_{6,4}$	6	-3.3219e+00	-3.3221e+00*	-3.3220e+00*	-3.3220e+00*	**-3.3221e+00***	-3.3221e+00*
ROSENBROCK	10	6.0933e-02	6.1261e-02	6.4494e-02	1.3559e-01	**3.3287e-02***	3.9672e-02
GRIEWANK	10	5.6184e-03	**3.1773e-05***	5.2173e-05*	5.5775e-05*	4.8557e-05*	1.1137e-03*
ROSENBROCK	20	9.6348e-02	1.0589e-01	4.1610e-01	3.8107e-01	**7.0546e-02***	7.8893e-02
GRIEWANK	20	9.0563e-03	**3.6800e-05***	1.4525e-03*	6.5679e-05*	8.8092e-04*	3.0862e-03*
ACKLEY	30	1.1034e-02	5.8176e-03*	7.1033e-03*	6.8517e-03*	6.0661e-03*	**5.5458e-03***
Wins	-	-	8	6	8	9	**12**

TABLE III
COMPARISON OF BEST

f	$\|D\|$	C-GRASP	DMC-GRASP	DMC-GRASP-X	DMC-GRASP-MX	DMC-GRASP-RX	DMC-GRASP-QX
BOHACHEVSKY	2	1.4933e-06	2.2356e-06	2.7544e-07	4.1318e-07	1.1469e-06	**1.6384e-07**
GOLDSTEINPRICE	2	3.0002e+00	**3.0000e+00**	3.0001e+00	3.0002e+00	3.0001e+00	3.0000e+00
COLVILLE	4	8.5447e-05	**2.1366e-05**	7.5327e-05	6.0435e-05	9.3411e-05	2.4957e-05
POWERSUM	4	**7.9648e-06**	**7.9648e-06**	3.0893e-05	2.8253e-05	**7.9648e-06**	**7.9648e-06**
SHEKEL$_{4,5}$	4	-1.0152e+01	-1.0153e+01	-1.0153e+01	-1.0153e+01	**-1.0153e+01**	-1.0153e+01
SHEKEL$_{4,7}$	4	-1.0401e+01	-1.0403e+01	-1.0403e+01	-1.0403e+01	-1.0403e+01	**-1.0403e+01**
SHEKEL$_{4,10}$	4	-1.0536e+01	**-1.0536e+01**	**-1.0536e+01**	-1.0536e+01	-1.0536e+01	-1.0536e+01
PERM	4	1.9945e-03	4.8515e-03	**1.4043e-03**	3.6150e-03	1.4778e-03	2.9955e-03
PERM$_0$	4	1.9945e-03	4.8515e-03	**1.4043e-03**	3.6150e-03	1.4778e-03	2.9955e-03
ROSENBROCK	5	2.9203e-03	1.6199e-03	9.8446e-04	1.3650e-03	2.5237e-03	**9.5073e-04**
HARTMANN$_{6,4}$	6	-3.3222e+00	**-3.3223e+00**	-3.3223e+00	-3.3223e+00	-3.3223e+00	-3.3223e+00
ROSENBROCK	10	1.3456e-02	4.0260e-03	2.4733e-03	5.6096e-03	**1.9466e-03**	5.3127e-03
GRIEWANK	10	2.4863e-05	**0.0000e+00**	1.9827e-05	2.9522e-05	2.4863e-05	2.0045e-05
ROSENBROCK	20	3.8979e-02	2.4384e-02	1.6369e-02	1.9877e-02	1.5783e-02	**1.5226e-02**
GRIEWANK	20	3.3826e-05	**0.0000e+00**	3.8813e-05	4.6140e-05	8.8439e-06	4.1625e-05
ACKLEY	30	9.6810e-03	4.6564e-03	5.6677e-03	5.7473e-03	5.0666e-03	**4.4856e-03**
Wins	-	-	12	**14**	10	13	12

where the algorithms tend to find their best solutions. The black star marks the point with the best mean cost value. We can observe that there is a quick convergence in average to the lower values in the hybrid phase of DMC-GRASP. The region in which it tends to find the best solutions is shorter than C-GRASP and placed after starting the data mining. We can also note that the DMC-GRASP performed more iterations than C-GRASP with the same number of function evaluations, which means it executes fewer function evaluations by iteration.

We also have undertaken an analysis using the time-to-target plot [31] for some functions to compare the convergence speed of C-GRASP and DMC-GRASP-QX, which has been the best version of DMC-GRASP in the previous experiments. Figure 3 shows the graph for the HARTMANN$_{6,4}$ function. The time-to-target plot expresses that DMC-GRASP has a faster convergence than C-GRASP. The DMC-GRASP algorithm presented about 90% of probability to find a solution at least as good as the target within about $7,500$ function evaluations. C-GRASP reached the same probability after about $17,500$ evaluations. Figure 4 shows the graph for the RASTRINGIN function, for which DMC-GRASP also has presented a faster convergence, reaching about 100% of probability to find the target solution before C-GRASP in one order of magnitude of function evaluations.

VI. CONCLUSION

In this paper, we proposed the hybridization of the C-GRASP metaheuristic with Data Mining, called DMC-GRASP. The DMC-GRASP uses the patterns extracted from good solutions by the Data Mining techniques to construct new solutions, possibly better ones.

We compared the performance of five versions of DMC-GRASP with C-GRASP in a set of well-known benchmark functions. The results of the experiments showed that DMC-GRASP can converge faster to better solutions than C-GRASP.

We can conclude from this work that pattern mining has great potential to guide C-GRASP – or other metaheuristics proposed for continuous optimization – to high-quality solu-

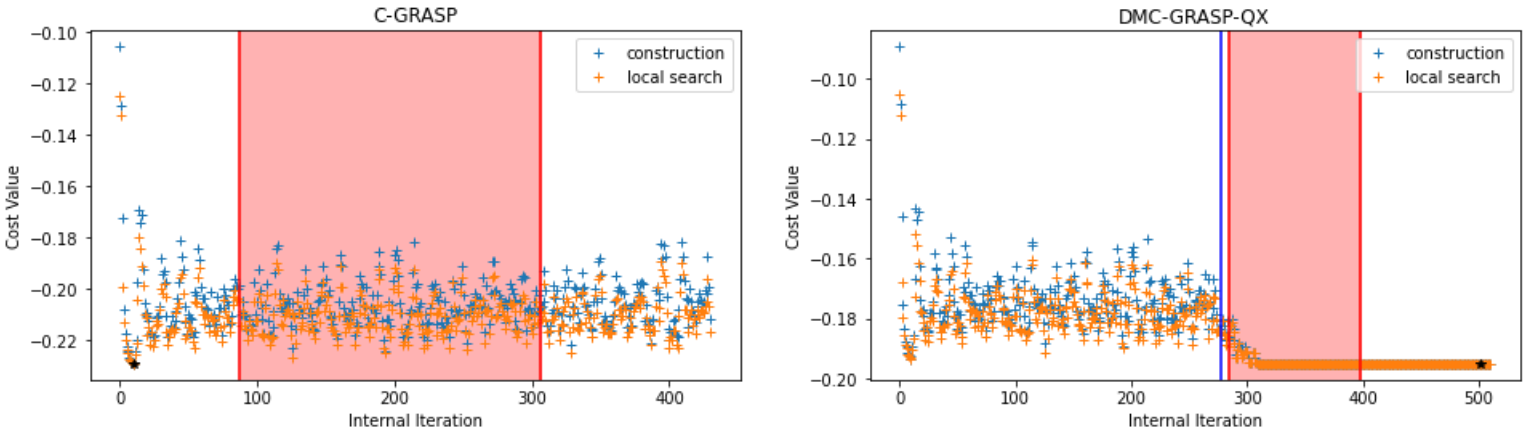

Fig. 1. Analysis of average behaviour of C-GRASP and DMC-GRASP-QX, over the internal iterations for the $\text{HARTMANN}_{6,4}$ function ($|D| = 6$).

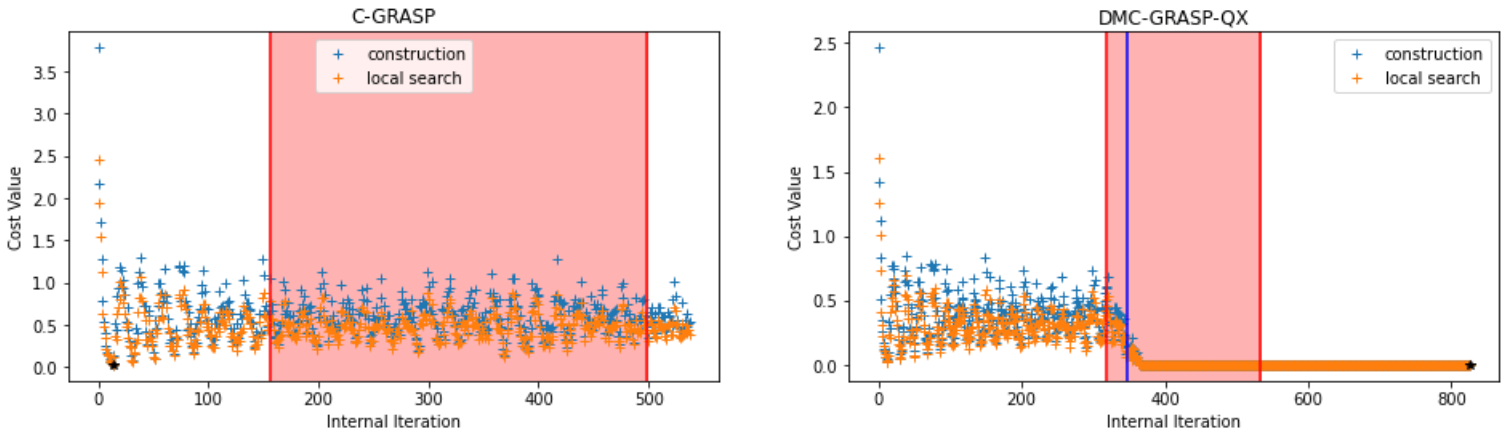

Fig. 2. Analysis of average behaviour of C-GRASP and DMC-GRASP-QX, over the internal iterations for the RASTRIGIN function ($|D| = 20$).

tions. In future work, we intend to use pattern extracted by data mining techniques to improve other state-of-art heuristics in the context of continuous optimization.

VII. Acknowledgements

This work was supported by a research grant from Science Foundation Ireland (SFI) under grant no. SFI/16/RC/3918 (CONFIRM) and Marie Skłodowska-Curie grant agreement no. 847.577 cofunded by the European Regional Development Fund. This work was also supported by CNPq – research grant 315750/2021-9.

References

[1] P.M. Pardalos and M.G.C. Resende, editors. Handbook of Applied Optimization. Oxford University Press, New York, 2002.

[2] Absalom E. Ezugwu, Olawale J. Adeleke, Andronicus A. Akinyelu, and Serestina Viriri . “A conceptual comparison of several metaheuristic algorithms on continuous optimization problems”. Neural Computing and Applications. 32, 2020. 10.1007/s00521-019-04132-w.

[3] P. Festa and M.G.C. Resende. “An annotated bibliography of GRASP—part I: algorithms”. Int. Trans. Oper. Res. 16(1), 2009a, pp. 1–24.

[4] P. Festa and M.G.C. Resende. “An annotated bibliography of GRASP—part I: applications”. Int. Trans. Oper. Res. 16(2), 2009b, pp. 131–172.

[5] M.J. Hirsch, C.N. Meneses, P.M. Pardalos, and M.G.C. Resende. “Global optimization by continuous GRASP”. Optim. Lett. 1, 2007, pp. 201–212.

[6] M.J. Hirsch, C.N. Meneses, P.M. Pardalos, M.A. Ragle, and M.G.C. Resende. “A continuous GRASP to determine the relationship between drugs and adverse reactions”. In: Data Mining, Systems Analysis e Optimization in Medicine. AIP Conference Proceedings, vol. 952, 2008, pp. 106–121.

[7] M.J. Hirsch, P.M. Pardalos, and M.G.C. Resende. “Speeding up continuous GRASP”. Eur. J. Oper. Res. 205, 2010, pp. 507–521.

[8] T.A. Feo and M.G.C. Resende. “Greedy randomized adaptive search procedures”. J. Glob. Optim. 6, 1995, pp. 109–133.

[9] T.M.U. Aráujo, L.M.M.S. Andrade, C. Magno, L.A.F. Cabral, R.Q. Nascimento, and C.N. Meneses. “DC-GRASP: Directing the search on continuous-GRASP”. Journal of Heuristics, 2015. doi: 10.1007/s10732-014-9278-6

[10] M.H.F. Ribeiro, A. Plastino, and S.L. Martins. “Hybridization of grasp metaheuristic with data mining techniques”. . Journal of Mathematical Modelling and Algorithms, 5, 2006, pp. 23–41.

[11] L.F. Santos, S.L. Martins, and A. Plastino. “Applications of the DM-GRASP heuristic: a survey”. Int Trans Oper Res 15, 2008, pp. 387–416.

[12] A. Plastino, H. Barbalho, L.F.M. Santos, R. Fuchshuber, and S.L. Martins. “Adaptive and multi-mining versions of the DM-GRASP hybrid metaheuristic”. Journal of Heuristics, 20, 2014, pp. 39-74.

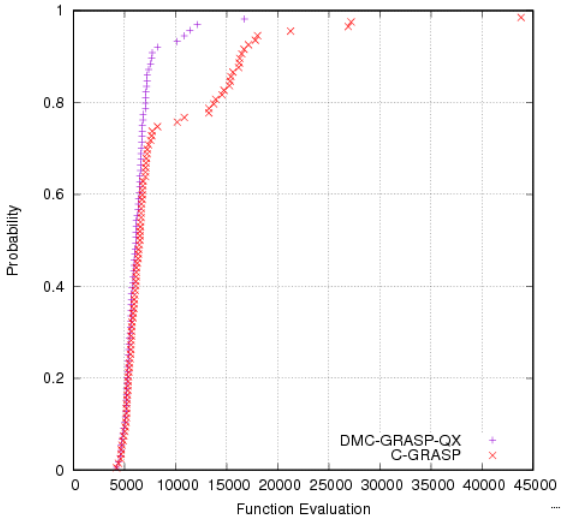

Fig. 3. Analysis of convergence of DMC-GRASP-QX for the RASTRIGIN function ($|D| = 20$), using a time-to-target plot.

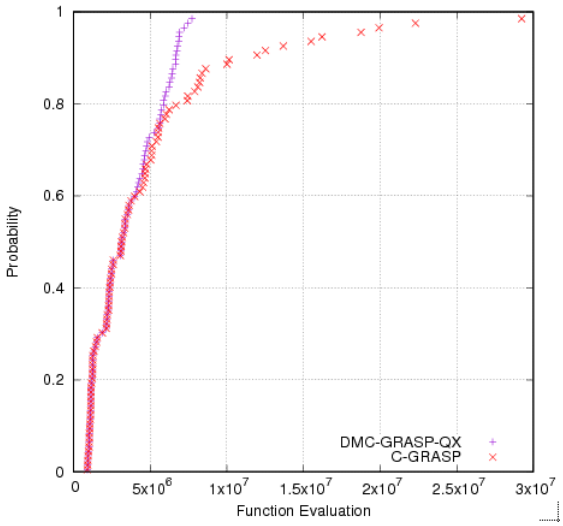

Fig. 4. Analysis of convergence of DMC-GRASP-QX for the RASTRIGIN function ($|D| = 20$), using a time-to-target plot.

[13] S.L. Martins, I. Rosseti, and A. Plastino. "Data Mining in Stochastic Local Search". In: Martí R., Pardalos P., Resende M. (eds) Handbook of Heuristics. Springer, 2018.

[14] D. Martins, G.M. Vianna, I. Rosseti, S. L. Martins, and A. Plastino. "Making a state-of-the-art heuristic faster with data mining". Ann Oper Res 263, 2018, pp. 141–162. https://doi.org/10.1007/s10479-014-1693-4.

[15] M. Guerine, I. Rosseti, and A. Plastino. "A hybrid data mining heuristic to solve the point-feature cartographic label placement problem". Intl. Trans. in Op. Res., 27, 2020, pp. 1189-1209. https://doi.org/10.1111/itor.12666.

[16] M.R.H. Maia, A. Plastino and P.H.V. Penna. "MineReduce: An approach based on data mining for problem size reduction". Comput. Oper. Res. 122: 104995, 2020.

[17] D.P. Bertsekas and J.N. Tsitsikis. Introduction to Probability. 2nd ed. Athena Scientific, 2008. ISBN: 9781886529236.

[18] Adil Fahad, Najlaa Alshatri, Zahir Tari, Abdullah Alamri, Ibrahim Khalil, Albert Y. Zomaya, Sebti Foufou, and Abdelaziz Bouras. "A survey of clustering algorithms for big data: Taxonomy and empirical analysis". IEEE Transactions on Emerging Topics in Computing 2, 3, pp. 267–279, 2014.

[19] R. Duda and P. Hart. Pattern Classification and Scene Analysis. John Wiley & Sons, 1973.

[20] Wilcoxon, Frank.. "Individual Comparisons by Ranking Methods." Biometrics 1, 1945, pp. 196-202.

[21] D. Pelleg and A. W. Moore. "X-means: Extending K-means with Efficient Estimation of the Number of Clusters". In Proceedings of the Seventeenth International Conference on Machine Learning (ICML '00). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2000, pp. 727–734.

[22] El-Ghazali Talbi. "Machine learning into metaheuristics: A survey and taxonomy of data-driven metaheuristics". 2020. ffhal-02745295.

[23] M. López-Ibáñez, J. Dubois-Lacoste, L. P. Cáceres, T. Stützle, and M. Birattari. "The irace package: Iterated Racing for Automatic Algorithm Configuration". Operations Research Perspectives, 3, 2016, pp. 43–58.

[24] C.C. Aggarwal . "An introduction to Frequent Pattern Mining". In: Aggarwal CC, Han J (eds) Frequent Pattern Mining. Springer, Basel, 2014, pp. 1–14.

[25] M. Sebag and A. Ducoulombier, "Extending Population-based Incremental Learning to Continuous Search Spaces," Parallel Problem Solving from Nature - PPSN V, 1998, pp. 418–427.

[26] P. Bosman and D. Thierens, "Expanding from Discrete to Continuous Estimation of Distribution Algorithms: The IDEA," Parallel Problem Solving from Nature PPSN VI, 2000, pp. 767–776.

[27] P. Larrañaga, R. Etxeberria, J. Lozano, and J. Peña, "Optimization in Continuous Domains by Learning and Simulation of Gaussian Networks," in Proceedings of GECCO-2000, 2000, pp. 201–204.

[28] S. Tsutsui, M. Pelikan, and D. Goldberg, "Evolutionary Algorithm Using Marginal Histogram Models in Continuous Domain", in Proceedings of the Optimization by Building and Using Probabilistic Models OBUPM Workshop at GECCO-2001, 2001, pp. 230–233.

[29] N. Ding, S. Zhou, and Z. Sun, "Optimizing Continuous Problems Using Estimation of Distribution Algorithm Based on Histogram Model," in Proceedings of the 6th Conference of Simulated Evolution and Learning, 2006, pp. 545–552.

[30] N. Ding, S. Zhou, and Z. Sun, "Histogram-Based Estimation Of Distribution Algorithm: A Competent Method for Continuous Optimization", Journal of Computer Science and Technology, vol. 23, no. 1, 2008, pp. 35–43.

[31] R. Aiex, M. Resende, and C. Ribeiro, "TTT plots: A perl program to create time-to-target plots", Optimization Letters, 1, 2007, pp. 355-366. 10.1007/s11590-006-0031-4.