Optim Lett (2014) 8:705–714
DOI 10.1007/s11590-013-0610-0

ORIGINAL PAPER

Memetic self-adaptive evolution strategies applied to the maximum diversity problem

Alan Robert Resende de Freitas · Frederico Gadelha Guimarães · Rodrigo César Pedrosa Silva · Marcone Jamilson Freitas Souza

Received: 22 February 2012 / Accepted: 21 January 2013 / Published online: 8 February 2013

Abstract The maximum diversity problem consists in finding a subset of elements which have maximum diversity between each other. It is a very important problem due to its general aspect, that implies many practical applications such as facility location, genetics, and product design. We propose a method based on evolution strategies with local search and self-adaptation of the parameters. For all time limits from 1 to 300 s as well as for time to converge to the best solutions known, this method leads to better results when compared to other state-of-the-art algorithms.

Keywords Maximum diversity problem · Metaheuristics · Memetic self-adaptive evolution strategies · Evolutionary algorithms

1 Introduction

The maximum diversity problem (MDP) consists in finding a subset of elements that have maximum diversity according to a function that defines the diversity between any two elements in a set. An example of this problem might be to find a subset of students

A. R. R. de Freitas (✉) · F. G. Guimarães · R. C. Pedrosa Silva
UFMG, Belo Horizonte, Brazil
e-mail: alandefreitas@gmail.com

F. G. Guimarães
e-mail: fredericoguimaraes@ufmg.br

R. C. Pedrosa Silva
e-mail: rcpsilva@gmail.com

M. J. F. Souza
UFOP, Ouro Preto, Brazil
e-mail: marcone.freitas@gmail.com

in a classroom that have maximum diversity among themselves. The diversity between any two students might be stored in a matrix and defined according to parameters such as gender, age, grades and nationality (Sect. 2).

The contributions of this paper are (i) a Memetic self-adaptive evolution strategy (MSES) (Sect. 3), (ii) an analysis of the benchmark instances and the problem difficulty (Sect. 4), (iii) a comparison of results between the MSES and the best methods available in the literature (Sect. 5). In Sect. 6, we draw conclusions from the comparison of the algorithms.

2 Maximum diversity problem

The MDP consists in finding a subset M ($|M| = m$) from a set N ($|N| = n$) in a way that the sum of diversities amongst the m elements is maximized. Many relations of diversity can be used to define diversity values d_{ij} according to the practical application of the MDP. The problem is concisely described [11] by formulation (1), where $x_i = 1$ if element i is in the subset M.

$$\begin{aligned} \text{Maximize} \quad & \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} d_{ij} x_i x_j, \ \text{subject to} \sum_{i=1}^{n} x_i = m \\ \text{where} \quad & x_i \in \{0, 1\} \, \forall i = 1, \ldots, n \end{aligned} \tag{1}$$

Applications of the MDP [14] are location of facilities, environmental systems, medical treatments, genetics and design. The clique problem can be reduced to the MDP [11], justifying the interest on metaheuristics for obtaining solutions in reasonable time. Methods for the problem include Greedy Randomized Adaptive Search Procedures (GRASP) [8,16,18,19], Variable Neighborhood Search (VNS) [3,6,9,18], Simulated Annealing [1], Lotfi-Cerveny-Weitz heuristic [22], Tabu Search [7,15,20], Iterated Greedy Algorithm [12], branch-and-bound [2,13], Hopfield Networks [21], and Scatter Search [17].

In an extensive comparison with methods for the MDP [14], even simple heuristics achieved good results. However, for high level solutions, a VNS [6] and an ITS [15] were the best methods for the benchmark instances. In a more recent paper, a Learnable Tabu Search (LTS) [20] appears as the best approach.

3 Memetic self-adaptive evolution strategies

Evolution strategies (ES) are a class of evolutionary algorithms primarily dependent on mutation [4]. The notation ES(μ, λ) stands for an ES with μ parents and λ children, being each of those individuals a candidate solution to the problem.

The algorithm is based on generations that repeat until a halting criteria is reached. In a generation, μ parents are mutated to produce λ children. Afterwards, the best μ new individuals become the current parents. A common adaptation to ES is to include a step size or mutation strength variable σ_i associated with each individual i [4]. This variable is often adjusted through self-adaptation, that is, by also applying secondary

```
1  Parent $p\_indices \leftarrow [1 \ldots \mu]$; Children $c\_indices \leftarrow [\mu+1 \ldots \mu+\lambda]$;
2  $c_p \leftarrow 0.05$; $\rho \leftarrow 2$; $\alpha_{min} = 1$; $\alpha_{max} \leftarrow min(10, \lfloor 0.1n \rfloor)$;
3  Initial Population $\mathbf{X_i} \leftarrow \mu$ random solutions where $i \in p\_indices$;
4  $\mathbf{X} \leftarrow$ Weak Local Search($\mathbf{X}$) ;
5  $\sigma_{min} \leftarrow 1$; $\sigma_{max} \leftarrow min(m, n-m)$; $\sigma_i$ for $i \in [1 \ldots \mu+\lambda] \leftarrow \sigma_{min}$;
6  while a certain criterion is not met do
7      for $i \leftarrow$ each of the $\lambda$ children indices in c_indices do
8          if Random number in [0,1] $< c_p$ then Crossover $\leftarrow$ true;
9          if Crossover then
10             $\mathbf{z} \leftarrow \rho$ different parents randomly chosen from $p\_indices$;
11             $\mathbf{X}_i \leftarrow$ Crossover($\mathbf{X_z}$);
12             $\sigma_i \leftarrow \lfloor (mean(\sigma_{\mathbf{z}}) + 0.5 \rfloor$ ;
13         else
14             $z \leftarrow 1$ parent randomly chosen from $p\_indices$ ;
15             $\mathbf{X}_i \leftarrow \mathbf{X}_z$;
16             $\sigma_i \leftarrow \sigma_z$;
17         end
18         $\mathbf{X}_i \leftarrow$ Mutation($\mathbf{X}_i$, $\sigma_i$);
19         $\mathbf{X}_i \leftarrow$ Weak Local Search($\mathbf{X}_i$);
20         if Crossover then
21             if $f(\mathbf{X}_i) > min(f(\mathbf{X_z}))$ then
22                 $\sigma_i \leftarrow \sigma_{min}$ ;
23             end
24         else
25             if $f(\mathbf{X}_i) > f(\mathbf{X}_z)$ then
26                 $\sigma_i \leftarrow \sigma_{min}$ ;
27             else
28                 $\sigma_i \leftarrow \sigma_i \% \sigma_{max} + 1$; $\sigma_z \leftarrow \sigma_i$ ;
29             end
30         end
31         if $f(\mathbf{X}_i) > f(best\ solution\ known)$ then $\mathbf{X}_i \leftarrow$ Fast Strong Local
           Search($\mathbf{X}_i$);
32     end
33     $p\_indices \leftarrow \mu$ best unique individuals; $c\_indices \leftarrow$ the $\lambda$ other individuals;
34     $j \leftarrow 1$ parent randomly chosen from $p\_indices$;
35     $\alpha \leftarrow$ random integer in $[\alpha_{min}, \alpha_{max}]$;
36     $\mathbf{X}_i \leftarrow$ Intensive Strong Local Search(Perturbation($\mathbf{X}_i$,$\alpha$));
37 end
38 return The best solution seen by the algorithm;
```

Fig. 1 Memetic self-adaptive evolution strategy- MSES($\mu + \lambda$)

mutation operators on the associated σ_i values and expecting the best configurations to survive over time. This adjustment of the algorithm composes the self-adaptive ES.

We propose a memetic self-adaptive evolution strategy (MSES) presented in Fig. 1 that includes local searches [15,20], self-adaptation of the σ_i mutation parameter [4,6,15] and a low probability crossover operator. The source code of the algorithm is available online from the authors[1] and the next paragraphs explain all details of this pseudocode, adaptable to other problems.

In order to save computational cost in line 1, indices mark which individuals are the parents and which ones are the children. Thus, when new μ individuals are defined as parents, only the indices need to change. In line 2, the crossover probability c_p, the number of parents per crossover ρ, is defined to 2 and the range of perturbation in the strong local search α are initialized.

[1] All additional information mentioned in this paper, such as source codes and results, is available from the authors on www.alandefreitas.com/downloads/problem-instances/maximum-diversity-problem.php.

The initial μ parent individuals are generated in line 3 as uniformly distributed random solutions. Each individual represents a solution to the problem as a vector of n elements x_i where $\sum_{i=1}^{n} x_i = m$, as in Eq. 1. All initial individuals already undergo a weak local search in line 4. Given that we can get to a neighbor of a solution by exchanging one element from M to $N \backslash M$, the weak local search is a first-improvement local search [10] that tests all neighbors in random order and moves to any neighbor that improves the current solution until no neighbor is able to improve the current solution.

For a more efficient local search, only the benefit Δ of each neighbor over the current solution is analyzed [15]. The Δ_{qr} of removing the element q and including the element r is $d_{r.} - d_{q.} - d_{qr}$, where $d_{x.} = \sum_{i=1}^{n} d_{xi}$ for all $i \in M$. The values $d_{i.}$ of all elements can be calculated only once every iteration.

The initial solutions i receive their σ_i in line 5. The range of possible σ values is from $\sigma_{\min} = 1$ to $\sigma_{\max} = \min(m, n - m)$, but all individuals have $\sigma_{\min}$ at this point. Those values are later self-adapted to determine the mutation strength. Iterative evolution begins in line 6 until certain halting criteria are met. In this work, the only criterion considered is elapsed time.

At each generation beginning in line 7, a second loop generates each of the new λ children. In line 8, it defines if the new child is going to be created from only a mutation applied to a parent, with probability $1 - c_p$, or a crossover of ρ parents followed by mutation, with probability c_p. If the new child will be produced through crossover we have a new process from line 9: $\rho = 2$ parents are randomly chosen with uniform distribution and their indices are stored in $\mathbf{z}$. The child i is then generated through crossover on the ρ parents in $\mathbf{z}$.

In the crossover operator, given that each parent j represents a subset M_j, the child's M_i is initially formed by the intersection $\cap M_\rho$ of all M_j for which $j \in \mathbf{z}$. At this point, if $|\cap M_\rho| = |M_i| = m$, the child is returned. Otherwise, if $|\cap M_\rho| = |M_i| < m$, the operator selects $m - |M_i|$ random elements from $\cup M_\rho \backslash \cap M_\rho$ to include in M_i, being $\cup M_\rho$ the union of all M_j for which $j \in \mathbf{z}$.

After the application of crossover to the individuals, the σ_i of the new child i is also inherited from the σ of its parents in line 12. The new σ_i is the mean σ of the parents rounded to the closest integer. On the other hand, if only mutation is used to generate the new individual, a different process begins in line 14: only one individual z is chosen as parent of the child i, which will be a copy of the parent z. The σ_i will also be a simple copy of σ_z.

Independently of the application of crossover, all generated individuals undergo mutation in line 18. The mutation operator consists of randomly exchanging σ_i elements in M_i for elements in $N_i \backslash M_i$. Each generated individual undergoes the weak local search in line 19. The weak local search avoids underestimating the potential of solutions mutated with high σ. From this point, the σ values also have to be adjusted. If crossover was applied and the child is better than any of its parents, its σ_i returns to the minimum value $\sigma_{\min}$ in line 22. If crossover was applied and there was no improvement, nothing happens and σ_i is left as it was with the mean σ of its parents.

On the other hand, if only mutation was applied and the child is better than its parent, its σ_i also returns to $\sigma_{\min}$ in line 26. However, if there is no improvement over the parent, as in line 28, σ_i is adjusted to $\sigma_i + 1$ in the next generation, or returned to $\sigma_{\min}$ if σ_i was already equal to $\sigma_{\max}$.

In any case, a child that improves the best solution known undergoes the fast strong local search [20] in line 31. The strong local search used at this point is a Tabu Search, a local search algorithm that accepts movements to worse solutions as long as they are not in the Tabu list, differently from the search used as weak local search in line 19. This Tabu Search includes or removes only one element in M to find neighbor solutions also based on Δ values. The contribution Δ_j of each neighbor j in this algorithm is $\Delta_j = \sum_{i=1}^{n} d_{ji}$ for all $i \in M$ if $j \notin M$ or $\Delta_j = -\sum_{i=1}^{n} d_{ji}$ for all $i \in M$ if $j \in M$.

This Tabu Search has tenure $t = \sqrt{m}$. This means that once a movement to a neighbor is applied, this movement is marked as Tabu for t iterations and cannot be performed unless it has better objective value than the best feasible solution known so far. The algorithm has the limit of n iterations. At every iteration one movement towards the best neighbor is performed. If the current solution has less than m elements in M, only neighbors with more elements are considered. If the current solution has more than m elements in M, only neighbors with less elements are considered. If the current solution has m elements in M, the solution is feasible and all neighbors are considered.

After generating all children, the indices of the parents become the μ best unique individuals in line 33. In lines 34–36, a randomly chosen parent is perturbed and undergoes an intensive strong local search [15]. The perturbation alters α elements in the solution. At every iteration, each of those α elements is chosen from a list of the best 5 neighbor candidates. The neighbor structure and their measure of Δ are the same as in the weak local search.

Afterwards, this perturbed solution undergoes a second strong local search in line 36. This strong local search is a Tabu Search with the halting criterion of $\max(10{,}000,\ 1{,}000n)$ tested neighbors and tenure $t = n/4$. It uses the same neighbor structure and calculation of Δ as the weak local search. The neighbors are explored in random order and every time a better solution is found, it becomes the current solution and the algorithm goes to the next iteration. Solutions in the Tabu list are only considered if they improve the best solution known. If the best solution known has been improved by the end of the iteration, the new current solution undergoes the weak local search with a best-improvement approach instead of the ordinary first-improvement algorithm.

4 MDP instances

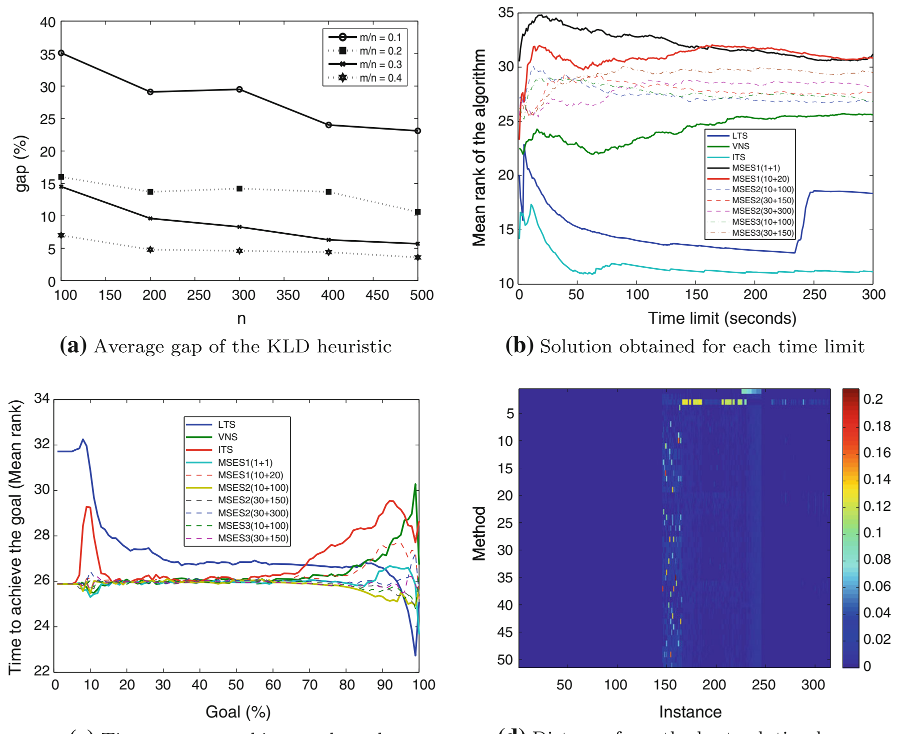
The 315 instances from the MDPLib [14] were used to test the proposed algorithm. The maintenance of the MDPLib is a valuable contribution from OPTSICOM.[2] A contingency table with the configuration of all the instances in the MDPLib is available online from the authors.

[2] http://www.optsicom.es/mdp.

The instances in the MDPLib have $10 < n < 3{,}000$; $0.08 < n/m < 0.8$; and values d_{ij} not defined as a function of n or m, which can make instances with a high value of n become easier as d_{ij} become small in relation to the problem, similarly to what happens to other optimization problems [5]. To demonstrate this, Fig. 2 shows the average gaps between the objective function values found by the constructive heuristic KLD [19] and the best values known for the 20 instances from the library SOM-b [14]. We see that the greater n is and the closer m/n is to $1/2$, the smaller the gap we obtain for those instances.

Increasing the value of n, however, does not necessarily mean that the probability of finding the optimal solution is higher. The number of solutions in the search space is $C_m^n = \frac{n!}{m!(n-m)!}$. Thus, the number of possible solutions increases when n increases or m gets closer to $n/2$ so it does not make sense to define the complexity of the problem only in terms of n.

As for all instances in the MDPLib, the values d_{ij} are not defined as a function of n or m. Given that the search space increases with n and decreases with $|m - n/2|$, the most complex instances in the MDPLib are the ones with $n = 3{,}000$; $m/n = 0.2$, $n = 2{,}000$; $m/n = 0.1$, and $n = 500$; $m/n = 0.4$, which represent 46 (14%) of the 315 instances. All other instances have either smaller n or a m/n (or m) more distant from 0.5 (or $n/2$).



(a) Average gap of the KLD heuristic

(b) Solution obtained for each time limit

(c) Time spent to achieve each goal

(d) Distance from the best solution known

Fig. 2 Performance of the algorithms

Thus, an increase in n may lead to higher probability of getting a solution with a small gap from the optimum. However, in the search space, it is more difficult to achieve the global optimum. Thus, simple algorithms are likely to have good results if the objective gap is used as reference, as shown in Fig. 2. However, even when it is easy to find solutions with quality similar to the optimal, the size of the search space is always the same C_m^n, and the difficulty to find the global optimum does not necessarily change.

5 Experiments

In an extensive comparison among the best methods for the MDP [14], VNS [6] had the best results for the problem except for the largest instances, where the ITS [15] is the method with the best rank. The LTS [20] is a more recent approach that showed better results on 50 instances with size $2{,}000 < n < 5{,}000$.

LTS [20]: An evolutionary strategy involving a k-means clustering algorithm for probability vectors identifies points likely to be good initial solutions for a Tabu Search. At each iteration, the Tabu Search updates an individual in the population and the probability vectors are also updated.

VNS [6]: Perturbations on the best solution generate new solutions. Those new solutions undergo a local search and are compared to the best solution. Perturbation strength increases when better solutions are found.

ITS [15]: A Tabu search is applied to an initial solution until the limit of evaluated neighbors is reached. Then, the current solution undergoes a change operator of random intensity and the Tabu Search restarts.

Three versions of MSES with minor adjustments are considered in the comparison. Having the algorithm in Fig. 1 as reference, MSES1 has the intensive strong local of lines 34–36 removed, MSES2 is exactly the same as described in Fig. 1, and MSES3 has a supplementary fast strong local search, such as the one in line 31, applied to the parent resultant from intensive strong local search in line 36 if it improves the best solution known.

In order to have relevant results, the performance of all the mentioned algorithms is compared for all the 300 time limits possible from 1 to 300 s and all MDPLib instances. Besides, in order to remove any factor that could influence the performance of the algorithms' concepts themselves, (i) the order of all replicates of all experiments were randomized, (ii) the tests were run in computers with the same configurations for the 300 different time limits, and (iii) all the algorithms were implemented in the same language.

The computers used were Intel® Core i5-650 / 4M Cache / 3.20 GHz / 4GB 1333Mhz DDR3 / 500GB (7200 RPM) SATA 3.0Gb/s HD with 16MB DataBurst Cache running Windows® 7 SP1 Professional (32Bit OS). For fair comparison of the algorithms, all of them were implemented in MATLAB and the source codes were made available from our website.

Combinations of MSES($\mu+\lambda$) with $\mu = 1, 10, 30, 50$ and $\lambda/\mu = 1, 2, 5, 10$ were tested for time limits of $1, 2, 3 \ldots 300$ seconds. For each time limit, the efficiency of the algorithms on the MDPLib is compared with a Friedman test, appropriate to compare different treatments (16 combinations for each MSES + 3 algorithms from the literature) on blocks (315 instances) by ranking data within blocks. In this, p-values close to 0 indicate that the treatments do not have the same performance and the ranks indicate the performance of each method. As usual, we declare a result significant if the p-value < 0.05.

For all time limits, the maximum p-value obtained is $7.02 \times 10^{-108} < 0.05$, indicating with confidence that the algorithms do not have the same performance for any time limit. The configurations of MSES with best rank for most time limits are MSES1$(1+1)$, MSES1$(10+20)$, MSES2$(10+100)$, MSES2$(30+150)$, MSES2$(30+300)$, MSES3$(10+100)$, MSES3$(30, 150)$. Figures with the comparison of the MSES were made available online by the authors. Figure 2 shows the performance of each algorithm measured by their mean rank in the Friedman test for all time limits from 1 to 300 s.

The two lines with best rank for all time limits are MSES1$(1+1)$ when the time limit is less than 154 s and MSES1$(10+20)$ when the time limit is greater than 153 s. The other version of MSES are represented in dotted lines. The best ranks among the other algorithms belong to VNS, LTS, and ITS, respectively. The range of possible mean ranks in this graph goes from 10.91 to 34.80 and the confidence intervals, which keep the p-values < 0.05, between the methods are in the range from 12.61 to 14.55.

We also use the data to analyze the methods in relation to goals. That is, for a given goal, which algorithm takes less time to achieve it. In this case, in each Friedman test, the treatments are the algorithms and the blocks are the time spent to achieve a certain goal. Figure 2c shows the mean ranks of the algorithms for each goal. The lower the mean rank, the less time was spent to achieve the goal. Most methods have similar performance for goals lower than 70 %. For goals under 95 %, MSES2$(10+100)$ has the best performance. For goals over 95 %, LTS becomes the best algorithm. However, for goals closest to the best values known, MSES1$(1 + 1)$ is again the best algorithm.

In addition to those comparisons in terms of goal achievements and absolute performance, Fig. 2d shows the gap between the solutions found by the algorithms and the best solutions known. Each line represents a method and each column represents an instance. The figure shows the how most solutions are close to the best solutions and it also demonstrates the importance of using ranks to cognitively perceive the difference between the algorithms. As supplementary material, comparisons of the algorithms for different subsets, the absolute results of all tests, the best solutions known, and the gap between the best solutions and the results on the instances are also available online.

6 Conclusion and future work

All the heuristics considered in previous works [14,20], VNS, LTS, and ITS, are smart approaches that consider specificities of the MDP. Based on their results, we propose here an MSES algorithm for the MDP with three variations in relation to its local search and variation of the parameters μ and λ.

MSES1$(1+1)$ and MSES1$(10+20)$ have the best results for time limits below 154 and above 153 s, respectively. MSES1$(1+1)$ presented the best time to achieve the best results known. For easier goals, LTS and MSES2(10+ 100) have the best results. The fact that the best algorithms for easier goals do not have good results in the first test can be explained by the discussion in Sect. 4, where we show how simple heuristics may be preferred for the MDP when the goal is not to achieve the optimal results.

For the extension of this work, we propose (i) comparing other instances in which d_{ij} is defined in function of n and m is closer to $n/2$, making the problems in fact more difficult; (ii) more tests on the adjustment of λ and μ; (iii) testing MSES(μ, λ), where all parents would be replaced by their children (as opposed to MSES($\mu + \lambda$)); and (iv) taking advantage of the population-based algorithm to evolve the individuals in parallel and accelerate convergence.

Acknowledgments We sincerely thank the reviewers for their valuable contribution to this paper. This work has been supported by the Brazilian agencies CAPES, CNPq, and FAPEMIG; and the Marie Curie International Research Staff Exchange Scheme Fellowship within the 7th European Community Framework Programme.

References

1. Agrafiotis, D.K.: Stochastic algorithms for maximizing molecular diversity. J. Chem. Inf. Comput. Sci. **37**(5), 841–851 (1997). doi:10.1021/ci9700337
2. Aringhieri, R., Bruglieri, M., Cordone, R.: Optimal results and tight bounds for the maximum diversity problem. Found. Comput. Decision Sci. **34**(2), 73–86 (2009)
3. Aringhieri, R., Cordone, R.: Comparing local search metaheuristics for the maximum diversity problem. J. Operational Res. Soc. **62**(2), 266–280 (2011)
4. Back, T.: Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms. Oxford University Press, USA (1996)
5. Berretta, R., Moscato, P.: New ideas in optimization chap. The number partitioning problem: an open challenge for evolutionary computation? pp. 261–278. McGraw-Hill Ltd., UK, Maidenhead, UK, England (1999). http://dl.acm.org/citation.cfm?id=329055.329082
6. Brimberg, J., Mladenovic, N., Urosevic, D., Ngai, E.: Variable neighborhood search for the heaviest-subgraph. Comput. Operat. Res. **36**(11), 2885–2891 (2009). doi:10.1016/j.cor.2008.12.020
7. Duarte, A., Marti, R.: Tabu search and grasp for the maximum diversity problem. Eur. J. Operational Res. **178**(1), 71–84 (2007). doi:10.1016/j.ejor.2006.01.021
8. Ghosh, J.B.: Computational aspects of the maximum diversity problem. Operat. Res. Lett. **19**(4), 175–181 (1996). doi:10.1016/0167-6377(96)00025-9
9. Glover, F., Kuo, C., Dhir, K.: Heuristic algorithms for the maximum diversity problem. J. Inf. Optimization Sci. **19**(1), 109–132 (1998)
10. Hansen, P., Mladenović, N.: First vs. best improvement: an empirical study. Discrete Appl. Math. **154**(5), 802–817 (2006). doi:10.1016/j.dam.2005.05.020
11. Kuo, C.C., Glover, F., Dhir, K.S.: Analyzing and modeling the maximum diversity problem by zero-one programming. Decision Sci. **24**(6), 1171–1185 (1993). doi:10.1111/j.1540-5915.1993.tb00509.x
12. Lozano, M., Molina, D., Garcí-a-Martí-nez, C.: Iterated greedy for the maximum diversity problem. Eur. J. Operational Res. **214**(1), 31–38 (2011). doi:10.1016/j.ejor.2011.04.018
13. Martí, R., Gallego, M., Duarte, A.: A branch and bound algorithm for the maximum diversity problem. Eur. J. Operational Res. **200**(1), 36–44 (2010). doi:10.1016/j.ejor.2008.12.023
14. Martí, R., Gallego, M., Duarte, A., Pardo, E.: Heuristics and metaheuristics for the maximum diversity problem. J. Heuristics 1–25 (2011). doi:10.1007/s10732-011-9172-4
15. Palubeckis, G.: Iterated tabu search for the maximum diversity problem. Appl. Math. Comput. **189**(1), 371–383 (2007). doi:10.1016/j.amc.2006.11.090
16. Resende, M., Martí, R., Gallego, M., Duarte, A.: Grasp and path relinking for the max-min diversity problem. Comput. Operat. Res. **37**(3), 498–508 (2010). doi:10.1016/j.cor.2008.05.011 (Metaheuristics)

17. Resende, M.G.C., Ribeiro, C.C., Glover, F., Martí, R.: Scatter search and path-relinking: Fundamentals, advances, and applications. In: Gendreau, M., Potvin, J.Y. (eds.) Handbook of Metaheuristics, International Series in Operations Research and Management Science, vol. 146, pp. 87–107. Springer US (2010). doi:10.1007/978-1-4419-1665-5_4
18. Silva, G., Ochi, L., Martins, S.: Experimental comparison of greedy randomized adaptive search procedures for the maximum diversity problem. In: Ribeiro, C., Martins, S. (eds.) Experimental and Efficient Algorithms, Lecture Notes in Computer Science, vol. 3059, pp. 498–512. Springer, Heidelberg (2004). doi:10.1007/978-3-540-24838-5_37
19. Silva, G.C., Andrade, M.R., Ochi, L.S., Martins, S.L., Plastino, A.: New heuristics for the maximum diversity problem. J. Heuristics **13**(4), 315–336 (2007). doi:10.1007/s10732-007-9010-x
20. Wang, J., Zhou, Y., Cai, Y., Yin, J.: Learnable tabu search guided by estimation of distribution for maximum diversity problems. Soft Computing **16**(4), 711–728 (2012). doi:10.1007/s00500-011-0780-6
21. Wang, J., Zhou, Y., Yin, J., Zhang, Y.: Competitive hopfield network combined with estimation of distribution for maximum diversity problems. Trans. Sys. Man Cyber. Part B **39**(4), 1048–1066 (2009). doi:10.1109/TSMCB.2008.2010220
22. Weitz, R., Lakshminarayanan, S.: An empirical comparison of heuristic and graph theoretic methods for creating maximally diverse groups, vlsi design, and exam scheduling. Omega **25**(4), 473–482 (1997). doi:10.1016/S0305-0483(97)00007-8