Workgroups Diversity Maximization: A Metaheuristic Approach

Marco Caserta[1] and Stefan Voß[2]

[1] IE Business School, Maria de Molina 12, 28006, Madrid, Spain
mcaserta@faculty.ie.edu
[2] University of Hamburg, Von-Melle-Park 5, 20146, Hamburg, Germany
stefan.voss@uni-hamburg.de

Abstract. Workgroup assignment problems commonly appear in various settings including international business schools. Especially if diverse people, like students, need to be divided into workgroups one may seek environments where diversity is fostered by generating heterogeneous workgroups. We study a problem of workgroups diversity maximization, i.e., the problem of building workgroups with the goal of maximizing intra-group diversity, while minimizing inter-group heterogeneity. For solving this problem with different objectives we propose a hybrid metaheuristic approach which combines local search techniques with a population based metaheuristic, including the cross entropy method as well as path relinking as ingredients. Numerical results are presented on some real-world instances.

1 Introduction

In this paper, we study the problem of creating workgroups with the aim of maximizing intra-group diversity, while minimizing inter-group heterogeneity. Let us consider an organization whose workforce is composed of a large body of workers with a diverse set of skills. Each worker is classified along a set of dimensions, *e.g.*, age, gender, native tongue, educational background, past experience, etc. The goal of the problem is to create teams maximizing diversity within each group, while making the set of teams as homogeneous as possible. In this context, the word "diversity" refers to differences in a range of qualities and characteristics among individuals. On the other hand, a set of groups is homogeneous if the overall set of characteristics of the members of each workgroup is similar, thus distributing people with similar characteristics as much as possible over the workgroups.

This study is motivated by the workgroup assignment problem commonly addressed in international business schools. Typically, business schools are attended by an extremely diverse body of students. Such students are divided in class sections and, within each section, in workgroups. To expose students to a richer experience, business schools attempt to create an environment in which diversity is fostered by generating sections and workgroups with heterogeneity in mind.

M.J. Blesa et al. (Eds.): HM 2013, LNCS 7919, pp. 118–129, 2013.

Although the main application of the problem presented in this paper is the workgroup assignment problem faced by business schools, a number of authors have pointed out that similar problems are encountered in other realms of application, *e.g.*, when assigning employees to project teams, work packets (tours, routes, etc.) to workers, when scheduling final exams at universities, or in the VLSI design.

The workgroup assignment problem has been object of study for over two decades. Previous works on this problem in more or less different settings can be found, e.g., in [2,11,7,8,12,1,5,6]. To clarify those different settings, let us consider two among those papers. First, the seminal paper [11] which presents a decision support system designed to address the students assignment problem. The authors thoroughly examined the problem in the format arising at one of the major European business schools and they proposed a constructive heuristic approach for the assignment of students to class sections and, within each section, to workgroups. The guiding criterion was a measure of similarity, *i.e.*, they iteratively assigned students in such a way that similar students were placed in different sections and workgroups. Secondly, [12] provided a comparison of five different heuristic rules for the creation of maximally diverse groups. The authors compared and contrasted one constructive method arising from the students workgroup assignment problem and four switching methods drawn from the final exam scheduling problem. All methods were driven by the same objective function, *i.e.*, a measure of overall diversity of the resulting partitioning. In order to have a measure of the goodness of the final results, they also proposed an integer bound.

In this paper, we present a hybrid metaheuristic approach for the workgroup diversity problem that combines local search techniques with a population based metaheuristic. The major contributions of this paper are:

- We introduce a set of "hard" constraints that limit the way in which teams are created by preventing individuals with certain attributes to belong to the same team. In the context of the student assignment problem, it could be the case that students that were together in the same teams in previous semesters are not allowed to be assigned to the same team in subsequent semesters. Such limitations actually change the problem itself, since it is no longer possible to ensure that instances of such problem have at least a feasible solution.
- We consider the "general" assignment problem, in which class sections do not need to be of the same size due to, *e.g.*, limitations in rooms availability. In line with that, we develop a set of fitness functions that take into account the relative size of each group.

2 A Mathematical Model for the Workgroup Diversity Problem

The problem studied in this paper can be seen as belonging to the class of assignment problems, in which a (larger) set of entities, *e.g.*, students, is assigned

to a (smaller) set of tasks, *e.g.*, class sections. The objective function of the assignment problem is defined in such a way that maximal diversity is achieved. Such objective function can be expressed in a number of ways, measuring, *e.g.*, the distance among students, the variance of the assignment, etc.

Due to the original motivation behind this work, in the following, we will use the terms students to refer to the entities to be assigned, and, interchangeably, teams or sections to identify the groups students are assigned to. Let us assume we are given a pool of n students, each described by m attributes. Such students must be assigned to a set of K teams. The basic information about each student $i = 1, \ldots, n$ is collected via a matrix $A = \{a_{ij}\}$, where $a_{ij} = 1$ indicates that student i has characteristic, or attribute, j, while $a_{ij} = 0$ indicates that student i does not have attribute j. Without loss of generality, we assume that each attribute is of binary nature, given that, whenever an ordinal attribute is given, this can always be transformed in a set of mutually exclusive binary attributes. For example, let us assume that we use $a'_{iw} \in \{1, \ldots, 10\}$ to indicate the value of a given nominal attribute w, where the attribute can only take values in the set $\{1, \ldots, 10\}$. It is always possible to express such nominal attribute as a collection of ten binary attributes $a_{ij} \in \{0, 1\}$, with $j = 1, \ldots, 10$, with the additional constraints that $\sum_{j=1}^{10} a_{ij} = 1$ for every student. Thus, in the following, we assume that all nominal attributes have been transformed into a set of corresponding binary attributes.

Given the set of attributes $\mathcal{A} = \{1, \ldots, m\}$, we partition such set into $\mathcal{A} = \mathcal{A}_d \cup \mathcal{A}_f$, where A_d is the set of desirable attributes, *i.e.*, students with these attributes should be dispersed over different teams or sections as much as possible, while A_f is the set of forbidden attributes, *i.e.*, students with these attributes cannot be assigned to the same team or section. In a similar fashion, the attributes' matrix A can also be partitioned into two submatrices A_d and A_f, in such a way that A_d is a matrix of size $n \times |\mathcal{A}_d|$ that contains all the values of the desirable attributes, while A_f is a matrix of size $n \times |\mathcal{A}_f|$ containing all the values of the forbidden attributes.

The following set of constraints can be used to model the workgroup assignment problem:

$$\sum_{k=1}^{K} x_{ik} = 1, \quad i = 1, \ldots, n \tag{1}$$

$$\sum_{i=1}^{n} x_{ik} \geq L_k, \quad k = 1, \ldots, K \tag{2}$$

$$\sum_{i=1}^{n} x_{ik} \leq U_k, \quad k = 1, \ldots, K \tag{3}$$

$$\sum_{i=1}^{n} a_{ij} x_{ik} \leq 1, \quad k = 1, \ldots, K, \quad j \in \mathcal{A}_f \tag{4}$$

$$x_{ik} \in \{0, 1\}, \quad i = 1, \ldots, n, \quad k = 1, \ldots, K \tag{5}$$

where x_{ik} is a binary decision variables, whose value equal to 1 indicates that student i is assigned to team k. Constraints (1) account for the fact that each student must be assigned to exactly one team; constraints (2)–(3) define the minimum (L_k) and maximum (U_k) number of students assigned to each team, while constraints (4) ensure that no two students with the same "forbidden" attribute can be assigned to the same team. It is worth noting that, due to constraints (4), we cannot even ensure that the problem has a feasible solution.

With respect to the objective function, we follow an approach that resembles that of [1], in which a number of alternative objective functions were proposed and tested. The peculiarity of our study is that, contrary to what is assumed by previous works, we do not assume that the team sizes must be the same. Therefore, we adjust the metric used to define the different objective functions accordingly.

Z_1: Proportional Entropy. This objective function is derived from information theory concepts and resembles the one introduced in [7]. Let us define

$$p_{kj} = \frac{\sum_i a_{ij} x_{ik}}{\sum_i x_{ik}}$$

the proportion, *i.e.*, percentage, of students assigned to team k that enjoy attribute j, for each team and attribute. The objective function is, thus:

$$\max Z_1 = \sum_{k=1}^{K} \sum_{j=1}^{m} -p_{kj} \ln p_{kj} \tag{6}$$

In Equation (6), we assume that the product $p \ln p$ is set to zero whenever the corresponding p is equal to zero. The difference between the proposed Z_1 measure and the one presented, *e.g.*, in [7], is that the proposed measure takes into account the relative size of the team, with respect to the overall population. In other words, to maximize function Z_1, students will be distributed over teams taking into account the size of each team.

Z_2: Total Proportional Deviation. This measure is a variation of the Total Absolute Deviation measure presented in [7]. Let us first define

$$\overline{p}_j = \frac{1}{K} \sum_{k=1}^{K} \frac{\sum_i a_{ij} x_{ik}}{\sum_i x_{ik}}$$

the average percentage of students with attribute j in each group. We thus define the following objective function:

$$\min Z_2 = \sum_{k=1}^{K} \sum_{j=1}^{m} \left| \frac{\sum_i a_{ij} x_{ik}}{\sum_i x_{ik}} - \overline{p}_j \right| \tag{7}$$

Equation (7) computes the total deviation in percentages with respect to each team and attribute. Again, the main advantage of this measure is that it takes

into account the size of each team and, therefore, provides a measure of the percentage of students within each group enjoying a certain attribute. The goal here is to minimize the deviation of each group percentage from the average percentage, in such a way that each group has approximately the same percentage of students with a given attribute.

Z_3: Total Absolute Deviation. This measure is the Total Absolute Deviation measure presented in [7], except for the fact that we take into account the relative size of each workgroup within the total population. We thus define the following objective function:

$$\min Z_3 = \sum_{k=1}^{K} \sum_{j=1}^{m} \left| \sum_i a_{ij} x_{ik} - \sum_i a_{ij} \frac{\sum_l x_{lk}}{n} \right| \tag{8}$$

Equation (8) provides a measure of deviation between the real (first term in Equation (8)) and theoretically optimal (second term in Equation (8)) number of students in a group with a given attribute. The theoretically optimal number of students with a given attribute in a team is computed relative to the size of the group itself. Thus, the larger the group, the larger the number of students with an attribute in that team.

The major difference between Equation (7) and Equation (8) lies in the unit measure: While Equation (8) is expressed in number of people, Equation (7) is a relative measure and is expressed as percentage. The inherent advantage of Equation (7) is that, if we divided Z_2 by $m \times K$, we would get a standardized value, *i.e.*, Z_2 would take values between 0 and 1.

Z_4: Pairwise Distance. This measure is a modified version of the Z_5 presented in [7]. Here again we take into account the number of students in each team. The following objective function is thus defined:

$$\min Z_4 = \sum_{j=1}^{m} \sum_{k=1}^{K-1} \sum_{l=k+1}^{K} \left| \frac{\sum_i a_{ij} x_{ik}}{\sum_i x_{ik}} - \frac{\sum_i a_{ij} x_{il}}{\sum_i x_{il}} \right| \tag{9}$$

Equation (9) accounts for the pairwise difference in the number of students with a given attribute between any two groups. In the equation, the first term accounts for the number of students with attribute j in team k (relative to the size of that group), while the second term computes the number of students with that same attribute j within group l (again weighted with respect to the size of that group). By minimizing the total sum of pairwise differences, we are aiming at minimizing the inter-groups difference.

3 A Pool-Based Metaheuristic Algorithm

In this section, we present the relevant features of the proposed algorithm. The algorithm is composed of four different steps, presented in Figure 1.

In the initialization phase, we define the pool size $|\Omega|$ and the insertion criterion `ic`. With respect to the proposed approach, we define $|\Omega| = 10$ while the insertion criterion `ic` is related to the fitness value, *i.e.*, a solution is inserted into the pool if its fitness value is better than the fitness value of the worst solution currently in Ω.

The second step of the algorithm defines a population-based metaheuristic with the aim of generating a variety of solutions. The best solutions found during this phase are inserted into Ω. We use a cross entropy scheme to pupulate Ω; details are provided below.

Once the pool Ω has been populated, we make an attempt to improve the quality of the solutions in the pool by means of a simple nested neighborhood search. We iteratively select a solution from the pool and we perform all the 2-opt exchanges, using the steepest ascent method. In other words, given the current solution, we try out all the feasible 2-opt exchanges and we select the one that generates the maximum improvement in the fitness value. The 2-opt scheme stops when no further improvement can be obtained with an exchange.

When the 2-opt scheme reaches a local optimum, we perform a 3-opt exchange using the steepest ascent method. Once again, the 3-opt scheme stops when a local optimum is reached.

Finally, the last step of the algorithm implements a path relinking approach, in which a trajectory leading from an incumbent solution $\mathbf{x}^l$ to a target solution $\mathbf{x}^t$ is defined. At each step, the *distance* between incumbent and target solutions is reduced by executing a 2-opt swap over $\mathbf{x}^l$. Given the two solutions $\mathbf{x}^l = \left\{x^l_{ik}\right\}$ and $\mathbf{x}^t = \{x^t_{ik}\}$, we define the distance between the two as a hamming distance:

$$H\left(\mathbf{x}^l, \mathbf{x}^t\right) = \sum_{i=1}^{n} \sum_{k=1}^{K} |x^l_{ik} - x^t_{ik}| \tag{10}$$

At each iteration of the path relinking, we select the 2-opt swap that, while reducing the hamming distance (10) by at least one, maximizes the improvement in the fitness value of the newly obtained solution.

Let us now present the details of the Cross Entropy (CE) scheme used in Step 2. (See [9] and [4] for a tutorial and a comprehensive overview of the CE. See also [3] for an application and fine-tuning technique for CE.) The assignment of a student to a team can be seen as a stochastic process governed by a set of probabilities. Let us suppose we are given an $n \times K$ probability matrix $P = \{p_{ik}\}$, where each term $p_{ik} \in [0, 1]$ represents the probability of assigning student i to team k. Matrix P is a stochastic matrix, since the sum of the probabilities per row is equal to one, *i.e.*, $\sum_k p_{ik} = 1$. Given such probability matrix P, one could generate an assignment of students to teams.

In order to ensure the feasibility of the generated solution, the following rules should be taken into account:

(R_1) Each student i should be assigned to a single team, as imposed by Constraints (1). Therefore, whenever a student i is assigned to a team k, we set $p_{ik} = 1$ and $p_{iw} = 0$, for all $w \neq k$.

S1 : Initialize Pool.
- Setup data structure to collect solutions into solution pool Ω.
- Define pool size $|\Omega|$ and insertion criterion `ic`.

S2 : Populate Pool.
- Apply the Cross Entropy scheme to populate the pool Ω.
- Define CE population size N, quantile ratio ρ, and smoothing factor α using Response Surface Methodology.
- Run the CE algorithm while stopping criteria `sc` are not verified and add solution to Ω if `ic` is satisfied.

S3 : Local Search.
- Apply a 2-opt and 3-opt schemes to improve the quality of the solutions in Ω.
- For each solution in Ω, apply a 2-opt mechanism using steepest-ascent, *i.e.*, as long as the current solution can be improved with a 2-opt exchange.
- Once the current solution can no longer be improved with a 2-opt exchange, apply a 3-opt exchange using steepest-ascent.

S4 : Path Relinking.
- Apply a Path Relinking scheme using all the solutions from the pool Ω.
- Set as *target solution* the current best solution found so far, *i.e.*, $\mathbf{x}^t$.
- For each solution $\mathbf{x}^l \in \Omega$, transform x^i into $\mathbf{x}^t$ via 2-opt swaps.
- If a new best solution is visited, save the new best solution.

Fig. 1. A Pool-based Metaheuristic Algorithm

(R_2) Each team k should have a minimum of L_k and a maximum of U_k students, as indicated by Constraints (2) and (3). Therefore, as long as the minimum number of students per team is not reached, we need to ensure that there still exists a positive probability of assigning students to that team, *i.e.*, $\sum_i p_{ik} > 0$. Conversely, once the maximum number of students per team has been reached, we need to set the probability of assigning further students to that team to zero, *i.e.*, $\sum_i p_{ik} = 0$.

(R_3) As imposed by Constraints (4), students with the same attribute value for attributes in the set of forbidden attributes $\mathcal{A}_f$ should not be assigned to

the same team. We decided to treat this set of hard constraints as "soft" constraints, by penalizing, in the fitness functions, assignments for which Constraints (4) were not satisfied. In other words, Constraints (4) were relaxed in a Lagrangean fashion and appropriate values for the multipliers were determined, to penalize violations of any of the Constraints (4).

To exploit the stochastic nature of the proposed approach, we generate a population of N assignments, *e.g.*, $\mathbf{x}^1, \ldots, \mathbf{x}^w, \ldots, \mathbf{x}^N$, drawn under the probability matrix P. Next, using the basic idea of the CE, we use the "Maximum Likelihood Estimator" method to revise the probability matrix and to generate a new matrix P^1 that better reflects the best individuals within the current population. Thus, we adjust the current probability values p_{ik} to reflect how likely it is that student i is assigned to team k in a high-quality solution. Once we obtain the new probability matrix P^1, we draw a new population of size N. Hopefully, such matrix better describes high quality solutions obtained in the previous generation and, therefore, the chance of obtaining high quality permutations based upon the new matrix is higher.

This process of "probability matrix update" and "population generation" can be iterated until a stopping criterion `sc` is reached, *i.e.*, either the P matrix converges to a binary matrix (therefore, the process converged to a unique solution in the solution space) or a pre-specified maximum number of iterations, say 30 (see Section 4), has been reached.

The "Maximum Likelihood Estimator" method is used to modify the probabilities p_{ik} in such a way that the new stochastic matrix better reflects the chance of obtaining high quality solutions. Let us assume that, based upon the current stochastic matrix P^t, we have generated a population of size N, *i.e.*, assignments $\mathbf{x}^1, \ldots, \mathbf{x}^N$. Let us now find, within the current population, the objective function value of the $(1-\rho)\%$ quantile, *i.e.*, the value γ for which $\rho\%$ of the population has a better objective function value and $(1-\rho)\%$ has a worse objective function value.

We modify the transition probability matrix using the following updating rule:

$$\hat{p}_{ik} = \frac{\sum_{w=1}^{N} I_{\{\mathbf{x}^w : x_{ik}^w = 1\}} \times I_{\{f(\mathbf{x}^w) \geq \gamma\}}}{\rho N} \tag{11}$$

where $f(\mathbf{x}^w)$ is the fitness value of assignment $\mathbf{x}^w$, and $I_{\{\bullet\}}$ is the indicator function, whose value is 1 if condition $\bullet$ is true, and 0 otherwise. Therefore, we use the following two indicator functions:

$$I_{\{\mathbf{x}^w : x_{ik}^w = 1\}} = \begin{cases} 1 & \text{if student } i \text{ is assigned to team } k \text{ in assignment } \mathbf{x}^w, \\ 0 & \text{otherwise;} \end{cases}$$

$$I_{\{f(\mathbf{x}^w) \geq \gamma\}} = \begin{cases} 1 & \text{if } f(\mathbf{x}^w) \geq \gamma, \\ 0 & \text{otherwise.} \end{cases}$$

Remark. As pointed out by [4], in order to prevent the CE from converging too fast to a suboptimal solution, a *smoothing factor* α (typically $0.7 \leq \alpha \leq 0.9$) could be used in the updating rule. Therefore, to foster a more thorough exploration of the solution space, at each iteration t we use the following updating rule:

$$p_{ik}^{t+1} = \alpha \hat{p}_{ik} + (1 - \alpha) p_{ik}^{t}. \tag{12}$$

4 Computational Results and Statistical Analysis

In this section, we summarize the results obtained by the proposed algorithm on real-world instances obtained by IE Business School. We will present how the algorithm performs when used on six large instances derived from the MBA program at IE Business School, Madrid, Spain. Those instances are taken from different semesters, and belong to the international MBA program as well as the Spanish MBA program. The fact that instances belong to different programs allow to test how the algorithm performs when dealing with students with different profiles. The size of each instance is characterized by two values: The number of students n, and the number of teams to be formed m. The testbed is composed of instances whose size spans from $n = 316$ and $m = 10$ (the largest instance) to $n = 57$ and $m = 9$ (the smallest one).

The algorithm proposed in this paper was coded in C++ and compiled using the GNU g++ 4.5.2 compiler on a dual core Pentium 1.8GHz Linux workstation with 4Gb of RAM. Throughout the computational experiment we kept the pool size $|\Omega|$ constant to ten, while the values of the CE parameters, *i.e.*, N, ρ, and α were determined using the Response Surface Methodology, as illustrated in [3]. The maximum number of iterations of the CE was kept constant to 30 throughout the computational experiment phase.

Since each metric is expressed using a different unit measure, a comparison of fitness values among the functions is not really meaningful. Therefore, we decided to test the behaviour of each function with respect to the others. In other words, we wanted to know to which extent the solution found using a given fitness function was able to produce good values for the other fitness functions. If, for example, fitness function Z_1 produces solutions that are of good quality not only when evaluated with respect to Z_1 but also when evaluated using $Z_2, \ldots, Z_4$, we can claim that the fitness function Z_1 is *robust*.

Therefore, to estimate the robustness of a fitness function, we solved each instance of the problem with each function. Next, for each obtained solution, we computed the fitness value using all the functions and we determined the ranking of these solutions with respect to the same function. One might argue about the use of the term robustness in our context as there are various other options to define robustness (see, *e.g.*, [10]). In different words we could also investigate possible correlations between different functions.

As an example, let us consider the case of instance `A1.3-2012`. The table below summarizes the results. In Table 1, each row corresponds to an execution of the

algorithm using the corresponding Z_i function. For example, when Z_1 was used as objective function of problem (1)–(5), the algorithm found a solution, *i.e.*, $\mathbf{x}^1$, whose objective function value, computed using Z_1, was $Z_1(\mathbf{x}^1) = 181.321$. However, when the same solution was evaluated using fitness function Z_2, we obtained $Z_2(\mathbf{x}^1) = 89.5733$. Similarly, we got $Z_3(\mathbf{x}^1) = 537.44$, and $Z_4(\mathbf{x}^1) = 452.867$.

We next ran the algorithm using fitness function Z_2. Let us indicate the best solution obtained by the algorithm with $\mathbf{x}^2$. The second row of the table provides the values of $Z_i(\mathbf{x}^2)$. The same approach was repeated using Z_3 and Z_4 as objective functions, as presented in Table 1.

Table 1. Objective function values of the different fitness functions on instance `A1.3-2012`. Arrows indicate whether a function is to be maximized (Z_1) or minimized (Z_2, Z_3, Z_4).

Fitness Function Used	Functions Evaluation			
	$\uparrow Z_1$	$\downarrow Z_2$	$\downarrow Z_3$	$\downarrow Z_4$
Z_1	181.321	89.5733	537.44	452.867
Z_2	180.137	89.6233	537.74	453.867
Z_3	179.794	89.5467	537.28	451.833
Z_4	180.289	89.4300	636.58	452.233

Using Table 1, we can provide some information about the robustness (as well as possible correlations) of a given fitness function. We now want to rank, columnwise, solutions $\mathbf{x}^1, \ldots, \mathbf{x}^4$, assigning a score of 1 to the best solution and 4 to the worst one. For example, in column Z_1, we observe that the best solution is $\mathbf{x}^1$, followed by $\mathbf{x}^4$, $\mathbf{x}^2$, and $\mathbf{x}^3$. A similar process for each column of Table 1 leads to the creating of the ranking, as presented in Table 2.

Table 2. Ranking of the different fitness functions on instance `A1.3-2012`. Arrows indicate whether a function is to be maximized (Z_1) or minimized (Z_2, Z_3, Z_4).

Fitness Function Used	Functions Ranking				Avg
	$\uparrow Z_1$	$\downarrow Z_2$	$\downarrow Z_3$	$\downarrow Z_4$	
Z_1	1	3	2	3	2.5
Z_2	3	4	3	4	3.75
Z_3	4	2	1	1	2.25
Z_4	2	1	4	2	1.5

Let us now present the computational results obtained over six real-world instances. Table 3 presents the results, in terms of ranking, over these instances. In the table, column one provides the name of the instance, while columns two and three specify the number of students and the number of teams. Columns

four to seven provide the average rank value of each fitness function. The rank value is computed as presented in Tables 1 and 2. Values presented in the table are averaged over 10 runs per instance and fitness function. Therefore, a total of $6 \times 10 \times 4 = 240$ runs and $240 \times 4 = 960$ function evaluations have been carried out to fill out Table 3.

Table 3. Computational results on real-world instances. Ranking is computed cross-evaluating each solution using all the fitness functions. Each instance is solved ten times using the same fitness function. The table contains a total of 240 runs and 960 evaluations.

Name	n	m	Z_1	Z_2	Z_3	Z_4
A1.1-2012	60	9	1.97	2.92	2.31	2.79
F2012a	140	3	2.67	2.77	3.24	2.89
F2012b	80	2	2.75	2.77	3.24	2.95
sA4-2012	57	9	2.23	2.16	2.35	3.25
A1.3-2012	60	9	2.14	2.76	2.37	2.71
F2011	316	10	2.92	3.1	3.15	3.12

From Table 3 we evince that fitness function Z_1 is the most robust, since it produces higher ranking, *i.e.*, higher quality solutions. Interestingly, the user of the algorithm, when called to select different solutions obtained using different fitness functions, expressed a clear preference for the solutions generated by fitness function Z_1. Thus, the empirical evidence obtained using the ranking table and the preference of the user seem to be aligned.

As a final remark, it is worth noting that the best solution found by the algorithm was found either in step 3 (73% of the time) or in step 4 (21% of the time). The remaining 7% of the time, the cross entropy scheme, *i.e.*, step 2 of the algorithm, produced a solution that was not improved by steps 3 and 4.

5 Conclusions and Future Work

In this paper, we presented a model and an hybrid algorithm for the workgroup diversity maximization problem. This problem aims at finding an assignment of workers to groups that minimizes the inter-group heterogeneity while maximizing the intra-group diversity. The problem finds application in different realms, spanning from the business schools assignment problem to the VLSI design.

The solution approach proposed in the paper is hybrid in nature, where local search techniques are intertwined with a population-based method. The algorithm has been tested on six real-world instances provided by a business school. The size of the instances varies, along with the number of teams to be created. The testbed has been used to determine the robustness of different fitness functions in terms of solution quality (which also allows to investigate possible correlations between the functions). The computational results collected from

the testbed are in line with the preferences expressed by the user. Both are in accordance in identifying one fitness function as superior compared with the others.

Future studies should be focused on a few interesting extensions: (i) generating a valid bound to objectively determine the quality of a solution; and (ii) using statistical analysis to rigorously assert whether the proposed fitness functions present statistically significant differences (and, therefore, to create a robust ranking of such criteria).

As our problem definition incorporates different ideas compared to how this type of problem is modeled or operationalized regarding possible objectives and constraints, it is difficult to directly compare the various problems and related solution methods. In future research we would also be interested to compare the proposed concepts once transfered between problem settings. Moreover, a valid extension would be to combine the first fitness function with one of the other functions using a bi-level programming approach.

References

1. Baker, K.R., Powell, S.G.: Methods for Assigning Students to Groups: A Study of Alternative Objective Functions. Journal of the Operational Research Society 53(4), 397–404 (2002)
2. Beheshtian-Ardekani, M., Mahmood, M.A.: Development and Validation of a Tool for Assigning Students to Groups for Class Projects. Decision Sciences 17(1), 92–113 (1986)
3. Caserta, M., Quiñonez, E.: A Cross Entropy-Lagrangean Hybrid Algorithm for the Multi-item Capacitated Lot-sizing Problem with Setup Times. Computers & Operations Research 36(2), 530–548 (2009)
4. De Boer, P., Kroese, D.P., Mannor, S., Rubinstein, R.Y.: A Tutorial on the Cross-Entropy Method. Annals of Operations Research 134, 19–67 (2005)
5. Desrosiers, J., Mladenovic, N., Villeneuve, D.: Design of Balanced MBA Student Teams. Journal of the Operational Research Society 56(1), 60–66 (2005)
6. Fan, Z.P., Chen, Y., Zeng, S.: A Hybrid Genetic Algorthmic Approach to the Maximally Diverse Grouping Problem. Journal of the Operational Research Society 62(7), 1423–1430 (2011)
7. Mingers, J., O'Brien, F.A.: Creating Students Groups with Similar Characteristics: A Heuristic Approach. Omega 23(3), 313–321 (1995)
8. O'Brien, F.A., Mingers, J.: A Heuristic Algorithm for the Equitable Partitioning Problem. Omega 25(2), 215–223 (1997)
9. Rubinstein, R.Y., Kroese, D.P.: The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization, Monte Carlo Simulation, and Machine Learning. Springer, Berlin (2004)
10. Scholl, A.: Robuste Planung und Optimierung: Grundlagen - Konzepte und Methoden - Experimentelle Untersuchungen. Physica, Heidelberg (2004)
11. Weitz, R.R., Jelassi, M.T.: Assigning Students to Groups: A Multi-Criteria Decision Support System Approach. Decision Sciences 23(3), 746–757 (1992)
12. Weitz, R.R., Lakshminarayanan, S.: An Empirical Comparison of Heuristic Methods for Creating Maximally Diverse Groups. Journal of the Operational Research Society 49(6), 635–646 (1998)