gruepr: An Open Source Program for Creating Student Project Teams

Dr. Joshua L. Hertz, Northeastern University

Dr. Hertz earned a B.S. in Ceramic Engineering from Alfred University in 1999 and then a Ph.D. in Materials Science and Engineering from the Massachusetts Institute of Technology in 2006. Following this, he worked at the National Institute of Standards and Technology as a National Research Council postdoctoral fellow. He joined the Department of Mechanical Engineering at the University of Delaware as an Assistant Professor in September 2008, leading a lab that researched the effects of composition and nanostructure on ionic conduction and surface exchange in ceramic materials. In 2014, he moved to Northeastern University to focus on teaching and developing curriculum in the First Year Engineering program.

Prof. Duncan Davis, Northeastern University

Duncan Davis is an Assistant Teaching Professor in First Year Engineering. His research focuses on using gamification to convey course content in first year classes. Mostly recently, he has implemented a series of escape room projects to teach engineering to first year students through the process of designing, prototyping, and building these play experiences.

Brian Patrick O'Connell, Northeastern University

Dr. O'Connell is an assistant teaching professor in the First-Year Engineering group at Northeastern University. His undergraduate degree in Mechanical Engineering came from the University of Massachusetts at Amherst in 2006. He then worked for Kollmorgen Electro/Optical as a mechanical engineer developing periscopes and optronic masts. In 2011, he returned to academia at Tufts University, earning his MS and Ph.D. in Mechanical Engineering for his work with low-cost educational technologies and his development and use of technologies to aid usage tracking in makerspaces to examine them as interactive learning environments.

Dr. Constantine Mukasa, Northeastern University

Constantine Mukasa received a B.S. degree in Computer Engineering from Bethune-Cookman University, Daytona Beach, Florida, USA in 2007, and his M.Sc. and Ph.D. degrees in Electrical Engineering from Florida Atlantic University, Boca Raton, Florida, USA, in 2013 and 2017, respectively. He is currently an Assistant Professor at Northeastern University in Boston, MA. His research interests include Engineering Education, Wireless Communications, satellite and mobile communication Systems, vehicular networks, wireless network connectivity, and interference modeling.

gruepr: An Open Source Program for Creating Student Project Teams

Abstract

This complete, evidence-based practice paper presents a study on a new open-source software tool that was written by the authors for creating optimal student teams. Creating effective teams can be a difficult task for an instructor, especially when creating them from first year students who have yet to establish preferences for work habits or teammates. In many cases, instructors of first year courses will form student teams by randomly assigning students to teams, by allowing students to self-select teammates, or, at best, by trying to optimize teams using whatever limited knowledge they have about these new students. Here, we present and analyze *gruepr*, a new software tool for partitioning up to 200 students into optimal project teams. The tool has been released under an open source license so that the C++ source code is freely available. The software tool runs on the instructor's computer using survey data entered by the students into, and then downloaded from, a Google Form. The instructor has considerable flexibility in choosing the content of the survey questions as well as their definition of a quantitatively optimal team.

The optimization problem is computationally difficult, and the search space is immense. For example, there are over 10^{19} ways to partition 32 students into 8 teams of 4. Thus, manually assembling a set of teams is time consuming and highly unlikely to provide an optimal result. It is for this reason that gruepr was created. Gruepr uses a genetic algorithm to perform the optimization. In this paper, we first discuss the need for such a program, then describe how an instructor uses gruepr, then present the design of the program, and finally provide initial analysis of gruepr and results from its recent use by 6 faculty members in the formation of teams within 18 sections of a project-based, first year engineering course.

Introduction

It is no coincidence that engineering colleges are striving to provide the appropriate environment to nurture and support team-based pedagogies to meet learning outcomes and prepare students to function as effective team members before entering the workforce. Research shows that team-oriented projects are becoming a customary pedagogy in both first-year and capstone undergraduate engineering courses [1]. Additionally, team-oriented coursework is one of the required criteria for accreditation of engineering colleges [2]. Micheaelsen et al. [3] argued that team-based learning transforms the classroom experience into one that is enjoyable for both instructors and students.

At the core of this pedagogy is the creation of effective teams to exploit the benefits of peer-to-peer interaction and instruction. Team formation is a complex task that has been extensively

studied in psychology [4], management [5], and related fields [6]. In these studies, several characteristics including prior knowledge, student's skills, motivation, competence, homophily [4], diversity, familiarity with other students, personality, and scheduling, have been suggested to significantly influence the effectiveness of the team and its members. However, the findings of these studies are not necessarily transferable to engineering education.

In the past decade, there has been a substantial body of research on team formation and the characteristics that make teams successful in engineering education, especially for first year engineers (FYE). For instance, Borrego et al. [7] performed a survey on the accumulated knowledge and theories on team formation and effectiveness from all fields and critically applied the findings to an engineering education setting. Morgeson et al. [8] highlighted that students placed on heterogeneous teams, where resources are distributed evenly among the teams, were more successful than those placed on homogenous teams. The impact of teaming female FYE students was studied by Dasgupta and colleagues [9]. They showed that female FYE students who were placed on female-majority teams felt less threatened and more positively challenged when working in groups than ones placed on female-minority or sex-parity teams. Additionally, it was observed that the female students assigned to female-majority teams expressed higher confidence and enthusiasm, and they verbally participated more during the group work. Such findings presage increased retention numbers and career aspiration in engineering for female students.

In general, there are four approaches that are commonly used by instructors to formulate groups: self-enroll, random assignment, instructor-selected, and computer-aided formation. Each approach has advantages and disadvantages that we discuss below.

Self-enrollment, where students choose their own teammates, offers more flexibility and control to the students about their learning, and it has been used successfully in upper-level course to take advantage of students' familiarity with each other [10]. However, this approach may not be suitable for FYE students who are not familiar with each other. Using self-enroll in FYE course leads students to rely on their past experiences and opt for what is familiar to them. This leads to excessive homogeneity and a lack of balance on academic ability and diversity within teams during the first years of college [11]. Additionally, this approach increases groupthink, less accountability, off-task chatting, and suboptimal team member evaluations, while alienating introverted students or those who have yet to establish relationships with their fellow students.

In random assignment, students are selected and placed on teams randomly. This option can be done easily using a free, web-based system; a downloadable application specifically designed for this; or a similar component commonly built into a learning management system such as Moodle, Blackboard, or Canvas. Random team formation is time-efficient and advantageous when working with large classes with broad and diverse backgrounds [12] and in cases where the

instructor is not previewed with prior information about the students. Beyond that, teams formed using random assignment do not necessarily offer any benefits compared to self-enrolled teams when it comes to diversity, balanced resources, and personalities [10]. Michaelson and colleagues [13] posited that random formation leaves too much to chance. Due to these reasons, random assignment is mainly used for expediency or makeshift short projects with fewer learning outcomes such that complex team forming approaches are not justified.

Instructor-selected team formation is cumbersome but McKeachie and Svinicki [14] suggest that this approach provides better odds in balancing member resources across the groups. This approach requires that the instructor has some prior knowledge about each student. This is often impractical for FYE instructors who mainly instruct students who are new to them and to the college. To obtain useful information, FYE instructors can use a pre-class questionnaire to collect information about the various characteristics that can be used in team formation. With the questionnaire information, the instructor must then seek to create the right balance within each team, a challenging task in a typical FYE class with many students. In other words, FYE instructors find it challenging to account for most of the characteristics important to team formation as the class size and number of characteristics to be considered grow [10]. Consequently, just a few characteristics are typically considered in the interest of time and complexity of the process. Additionally, in an effort to balance resources within the team, the process may yield homogenous teams compared to random formation [12]. These conditions minimize the clear advantages of using this approach.

Maximizing the advantages of the instructor-based team formation approaches, computer-aided team formation approaches facilitate the use of a range of student characteristics while simultaneously reducing the complexity and time required of the instructor. In the authors' search for computer-aided team formation programs, only the well-known Comprehensive Assessment of Team Member Effectiveness (CATME) system was found by the authors. CATME is an online tool used by instructors to create and manage groups or teams. CATME was developed in 2003 [10], released to the academic community in 2005, and initially provided at no cost to faculty. Currently, CATME comprises four tool sets namely: CATME Team-Maker, CATME Peer Evaluation, CATME Rater Calibration, and CATME Meeting support. Since its development, there has been a significant growth in its use in education. Several studies describe CATME as a tool for both team formation and team evaluation in engineering education [10, 15-18]. Most relevant to this paper is Team-Maker the CATME tool that facilities formation of effective teams using several characteristics.

Within Team-Maker, faculty upload classroom rosters and create questionnaires that can include characteristics such as demographic data, interests, grades (GPA), and weekly schedules. Students answer the questionnaire, and the data collected is then used to find an optimal set of teams based on given criteria set by the instructor. CATME Team-Maker offers the instructor

many options during the team forming process that make it an effective tool. For instance, the instructor may distribute resources homogeneously or heterogeneously across all teams. In 2017, a per-student fee was introduced by CATME in order to recoup the costs of human and computer support.

No competitors to CATME Team-Maker were found by the authors, who decided to write their own software to assist in team formation. The resulting program, gruepr, is written in C++ and compiled as a Windows program. Gruepr facilitates the logistics of optimal team formation in a similar manner to CATME Team-Maker. The source code has been released under an open source license and can be freely downloaded from the project's homepage [19]. We have to state that during our initial search and development of gruepr, we had no knowledge of the Individual and Team Performance Metrics Lab (ITP Metrics), a free team-forming web-based tool. It is until recently, when reading a paper by Jamieson and Shaw [15] that the authors came to learn about ITP Metrics. For that reason, our work does not account for this tool.

Software Usage

The use of gruepr by an instructor proceeds as follows. First, the instructor creates a Google Form to collect relevant data from the students. There is considerable flexibility in the questions within the form. A few questions are required and must be placed within a specific order in the survey, such as asking for the student's name and email address. Other questions are optional, such as asking for the student's gender. The instructor can write and include up to 9 "attribute" questions with multiple choice answers. The "attribute" questions are highly flexible and could be skills assessments, work preferences, attitudes, or any other question the instructor wishes to ask the students, provided that the answers are (or can be made to be) categorical or Likert-scale. A model Google Form has been made openly accessible [20] and has been transcribed in Appendix A. Multiple sections of students can use the same Google Form, even if they are to be teamed separately. After all students have submitted their survey response, the results are downloaded onto the instructor's computer directly from Google Forms as a single comma-separated-value (.csv) text file.

The instructor then runs gruepr on their computer and opens within it the text file of survey results. Gruepr automatically determines from the file what questions were asked in the survey. If there are multiple sections of students in the data, the instructor can choose which section is to be formed into teams during this run of the program. Next, the instructor chooses a series of teaming preferences and weighting options. The instructor's choice in determining what constitutes a quantitatively optimal team can be based on: 1) achieving a high degree of overlap in the free time in students' weekly schedules, 2) achieving within each team either homogeneity or heterogeneity of the responses to each of "attribute" questions, 3) preventing an isolated woman on a team, 4) preventing any chosen pairs or sets of students from being on the same team, and/or 5) requiring any chosen pairs or sets of students to be on the same team. The

relative importance within the optimization scheme of the schedule overlap and of the homogeneity or heterogeneity of each attribute question is individually selectable using a numerical weighting. The number of students on each team can be set using a single goal size (e.g., 4 students on each team), or the size of each team can be individually set. Whenever a perfectly even distribution of students is not possible, a choice is given to make the off teams one student larger or one student smaller.

Next, gruepr runs its genetic optimization algorithm, displaying its progress to the instructor. All of the instructor's chosen teaming options are used in the algorithm's fitness function. After the optimization algorithm operates for some time, the set of teams with the quantitatively highest score is shown to the instructor. The instructor can choose to keep these teams, make minor tweaks by swapping one or more pairs of students between teams, or get rid of the teams and restart the optimization scheme from the beginning. If the instructor chooses to restart the optimization, they may also choose to adjust the teaming options and/or team size(s) at that time. Since genetic algorithms are, in general, not guaranteed to find the global maximum within a search space [21], restarting the optimization scheme typically results in a very similar but non-identical set of teams even if the teaming options are left unchanged.

Whenever the instructor is satisfied with the set of teams, they may save the team data as text files. One of the text files is intended for the instructor, as it includes all of the students' data from their surveys. An example of the data included in this text file is shown in Appendix B. Another text file is intended to be split and distributed to the teams, as it shows for each team the students' names and email addresses, and a summary table of the team's schedule availability throughout the week. A third text file provides the names and email addresses of the students on each team in a format that is easy to open in Excel or import into other locations.

Optimization Algorithm

Gruepr uses a genetic algorithm to search for an optimal partitioning of the students into teams. Genetic algorithms begin by defining a genome, a way to represent a particular solution to the problem as a finite set of parameters. In gruepr, each student is represented internally using a sequential ID number. Each genome—i.e., one particular partitioning of the students into teams—is an array containing a permutation of the ID numbers. The order of the ID numbers within the array determines, in conjunction with the chosen team sizes, which students are on which team. For example, if gruepr is partitioning ten students into three teams with team sizes of three, three, and four students, then the array `[3, 9, 0, 4, 5, 1, 8, 7, 6, 2]` would represent that the first team consists of the students with ID numbers 3, 9, and 0; the second team consists of the students with ID numbers 4, 5, and 1; and the third team consists of the students with ID numbers 8, 7, 6, and 2.

There is significant redundancy in this permutation-of-teammate-ID method of encoding the genome. For example, the array `[1, 5, 4, 0, 3, 9, 2, 7, 8, 6]` is a different genome from the previous example, but it represents an identical partitioning of students into teams, since the order of the teams and the order of the students within a team does not matter. This redundancy suggests a future means to improve the genetic algorithm's efficiency.

Gruepr's algorithm uses a population of 30,000 genomes. In the initial population, Generation 1, every genome is simply a random permutation of the ID numbers. From each generation of 30,000 genomes, a succeeding generation of 30,000 is created. Each genome in the population is quantitatively scored according to the fitness function described in more detail below. Genomes with a higher score are more likely to pass part of their genome onto genomes in the next generation. The mechanism used to accomplish this is for two tournament-selected [22] parent genomes to create a child genome using an order crossover operation. The "OX1" order crossover operation [23] is used, but modified such that the two crossover points are always at locations in the genome that represent where a team boundary exists. For example, if the first team consists of four students, then a crossover point would never be after the first, second, or third array element, though one could be before the first element or after the fourth array element.

During the evolution process, genomes are allowed to undergo random mutations, manifested as a swapping of two randomly chosen positions on the genome. Evolution proceeds for at least 40 generations, stopping when either a population stability metric is reached or 300 generations have passed. This process typically takes the computer about a minute or two. The best set of teams found—i.e., the genome with the highest score—is then displayed on the screen.

Genetic algorithms require a fitness function to quantify how "good" a genome is and thereby determine the probability that it will propagate its genes to the next generation. In gruepr, a "compatibility score" is calculated for each team within a genome, and then a net "team set score" is calculated for this genome. The team set score is used as the fitness function. Below, we first describe the calculation of one team's compatibility score, then we describe how the compatibility scores for all of the teams in one team set are aggregated into a single team set score. A number of concepts used in the calculation of a team's compatibility score were first described by Layton, et al. in [10].

A team's compatibility score is based on several sub-scores. The first set of sub-scores is based on the "attribute" questions that the instructor has included in the survey. Specifically, each attribute score is based on the range of attribute values that the teammates have relative to the total range of possible values found among all students. If heterogeneity is desired on a team, such as when a range of skill levels on a particular topic is desired within a team, this ratio is directly used. The attribute score is thus 1 if the entire spectrum of values is found on this team,

and is 0 if all students on this team have the same value. If homogeneity is desired on a team, such as when grouping students based on level of preference to work on weekends vs. weekdays, then this range is flipped by subtracting the ratio from 1. Calculation of the attribute score is summarized in equation 1, with option *a* used when heterogeneity within a team is preferred for this attribute and option *b* used when homogeneity is preferred instead.

$$attribute\ score = \begin{cases} \dfrac{range\ of\ values\ within\ the\ team}{range\ of\ values\ among\ all\ students}, & (a) \\ 1 - \dfrac{range\ of\ values\ within\ the\ team}{range\ of\ values\ among\ all\ students}, & (b) \end{cases} \quad (1)$$

After calculating the attribute score(s), a schedule score is calculated for each team. The schedule score is based on how many times throughout the week all teammates have an overlapping free block in their schedule. The instructor chooses values for the: (a) minimum number of meeting times and (b) desired number of meeting times. The instructor also chooses whether a meeting time must be at least two hours long or, alternatively, that any single hour throughout the week can be used as a meeting time. The schedule score is calculated in such a way to prioritize first eliminating any team whose teammates can meet for fewer than the required minimum number of meeting times. After that, the next priority is to get each team to have as many available meeting times as possible up to the desired number of meeting times. After that, a third priority is to further increase the number of available meeting times.

The weekly availability matrix from each teammate is compared to count the number of times during the week where every teammate has listed themselves as free. Each of these is considered an available team meeting time. If a team has fewer than the required minimum number of meeting times, then the schedule score is set to a negative penalty value. The size of this penalty is chosen to be just large enough to ensure that the team's final compatibility score will be less than or equal to 0. If a team can meet for at least the minimum number of meeting times but fewer than the desired number of meeting times, then the schedule score is calculated as the ratio of the number of available meeting times relative to the desired number of meeting times. Finally, if a team has more than the desired number of meeting times, the schedule score instead is calculated as before except that the additional meeting times beyond the desired number are only counted at a 25% rate. Thus, the schedule score ranges between roughly 0 and 1, but can fall outside those bounds for a very low or very high number of available meeting times. Calculation of the schedule score is summarized in equation 2, with option *a* used when the team has fewer than the minimum required number of times where all teammates are free to meet, option *b* is used when the team has more than the minimum number but fewer than the desired number of meeting times, and option *c* is used when the team has more than the desired number of meeting times.

$$schedule\ score = \begin{cases} -\frac{1+number\ of\ attributes}{schedule\ score\ weight}, & (a) \\ \frac{\#\ of\ available\ meeting\ times}{desired\ \#\ of\ meeting\ times}, & (b) \\ 1 + \frac{(\#\ of\ available\ meeting\ times) - (desired\ \#\ of\ meeting\ times)}{4 \cdot desired\ \#\ of\ meeting\ times}, & (c) \end{cases} \quad (2)$$

If the instructor wishes, pairs or sets of students can be either prevented from being placed or required to be placed on the same team. A compatibility score penalty is applied for any team that has two or more students that are prevented from being teammates or that has one student on it without one or more of their required teammates. The size of one of these penalties is, like the previously described schedule score penalty, chosen to be just large enough to ensure that the team's final compatibility score will be less than or equal to 0. Calculation of the prevented teammate penalty is summarized in equation 3, with option *a* used when one or more pairs of prevented teammates are present on the team and option *b* used when there are no prevented pairs. Calculation of the required teammate penalty is summarized in equation 4, with option *a* used when one or more teammates is present without their required teammate(s) and option *b* is used when no student on the team is present without their required teammate(s).

$$prevented\ teammates\ penalty = \begin{cases} -(1 + \#\ of\ attributes), & (a) \\ 0, & (b) \end{cases} \quad (3)$$

$$required\ teammates\ penalty = \begin{cases} -(1 + \#\ of\ attributes), & (a) \\ 0, & (b) \end{cases} \quad (4)$$

If the instructor wishes, a gender isolation penalty can be calculated to account for teams that have one (and only one) woman student. Future versions of gruepr may expand on the idea behind this penalty to allow consideration of gender non-binary students and/or teams with only one man. It also may allow the calculation of analogous penalties to prevent isolation of students from underrepresented minority groups, non-English speakers, first generation college students, or any other student qualities that an instructor wants to consider. The size of the gender isolation penalty is the same as the penalty related to prevented teammates or required teammates. Calculation of the isolated gender penalty is summarized in equation 5, with option *a* used when the number of women on the team is exactly 1 and option *b* is used when the number of women on the team is either 0 or greater than 1.

$$gender\ isolation\ penalty = \begin{cases} -(1 + \#\ of\ attributes), & (a) \\ 0, & (b) \end{cases} \quad (5)$$

The compatibility score for one team is calculated as a weighted sum of each of the attribute scores and the schedule score. As previously described, each attribute score ranges between 0 and 1, and the schedule score ranges roughly between 0 and 1. These values are multiplied by the instructor's chosen weighting factors in order to reflect their relative importance. Onto this compatibility score is added any prevented teammates penalty, required teammates penalty, and/or gender isolation penalty. This final sum is then normalized in order to give a score that lies generally within the range 0 to 100. A team score can go outside this range only by becoming negative because one or more of the penalties applies, or by going over 100 because the schedule score was greater than 1. Calculation of the compatibility score is summarized in equation 6.

$$\begin{aligned} compatibility\ score = \left(\frac{100}{schedule\ weight + \sum attribute\ weights}\right) \cdot \\ \sum attribute\ score \cdot attribute\ weight + \\ schedule\ score \cdot schedule\ weight + \\ prevented\ teammates\ penalty + \\ required\ teammates\ penalty + \\ gender\ isolation\ penalty \end{aligned} \tag{6}$$

The net team set score is not a simple arithmetic mean of the individual team compatibility scores. Using the arithmetic mean would not account for a highly uneven distribution of compatibility scores, and might lead to "runaway" problems, where the average improves by giving one team an extremely high score at the expense of fairly poor scores for the other teams in the set. Therefore, in gruepr the team set score is calculated as the harmonic mean of the individual team compatibility scores. The harmonic mean, which is the reciprocal of the arithmetic mean of the reciprocals, is skewed towards the lowest values in a sample. Thus, while the algorithm in gruepr always moves in the direction of increasing any team's compatibility score, the algorithm moves *fastest* towards increasing the scores of the *lowest* scoring teams. If one or more teams in a team set have a compatibility score that is less than or equal to zero, then the harmonic mean is mathematically problematic. In these cases, the team set score used is the arithmetic mean "punished" by subtracting half the absolute value of the arithmetic mean. Calculation of the team set score is summarized in equation 7, where option *a* is used whenever one or more teams have a compatibility score ≤ 0 and option *b* is used when every team has a compatibility score > 0.

$$team\ set\ score = \begin{cases} \frac{\sum compatibility\ score}{\#\ of\ teams} - \left| \frac{\frac{\sum compatibility\ score}{\#\ of\ teams}}{2} \right|, & (a) \\ \left(\frac{\sum \frac{1}{compatibility\ score}}{\#\ of\ teams} \right)^{-1}, & (b) \end{cases} \quad (7)$$

Methods and Results

To create a controlled measure of the effectiveness of gruepr in forming teams using a typical population of students, data was collected from 32 first year engineering students. These students were then formed into teams using three different methods: random partitioning, instructor formation, and gruepr. For the latter two methods, complementary team schedules—the highest overlap of free time to ensure the highest number of opportunities for the teams to meet—was used as the most important factor. Specifically, when using gruepr, a weight of 70 was used for schedule and 20 for the other attributes. Isolated woman students were prevented as an additional constraint in all three cases, but no additional constraints, such as required or prevented teammates, were set. Statistical analysis comparing the teams generated using each of the three methods were compiled in **Table 1**. In order to have an equivalent metric to compare the three methods, we used gruepr's team set score. Although harmonic mean is more accurate for gruepr's analysis, some of the randomized team sets contained negative scores, so we were forced to use the modified arithmetic means in the team set score for this analysis. For more discussion about when gruepr uses the harmonic mean versus the arithmetic mean, please see the previous section.

Randomized

We randomly partitioned the students onto teams using a C++ algorithm. In order to ease the creation of teams that were random yet maintained the constraint of not allowing any isolated women, these teams were gender segregated. As one might expect, random generation created the team set with the lowest team set score. These teams would have significantly fewer available times throughout the week to meet as well as less desired skill compatibility compared to the teams produced using the other two methods.

Instructor-selected

Three of the authors made teams manually. On average, it took the authors about 30 minutes to make their team set. The manually created team sets scored better than randomized teams but worse than those produced by gruepr. These teams have moderate schedule and skill overlap.

Gruepr-generated

When gruepr created the teams, the resulting teams had the most schedule and skill compatibility. Inputting the data, setting the parameters, and running the algorithm in gruepr took about 10 minutes total. Therefore, using gruepr both saved time and made quantitatively more desirable team sets.

Table 1: Using actual student survey data, three team forming methods are compared: randomized, instructor-selected, and gruepr.

	Average Team Compatibility Score	Median Team Compatibility Score	Standard Deviation of Team Compatibility Scores	Best Team Set Score
Randomized	88.43	111.08	62.98	39.54
Instructor-Selected	99.66	109.50	66.50	50.59
gruepr	143.34	148.68	39.58	127.16

Next, we give a demonstration of how, over the course of tens or hundreds of generations, the 30,000 team sets in gruepr's population have an improving team set score. **Figure 1** shows the generational progression of the team score by plotting the population's median and mean team set score over about 60 generations. Also plotted is the maximum team set score within the population.

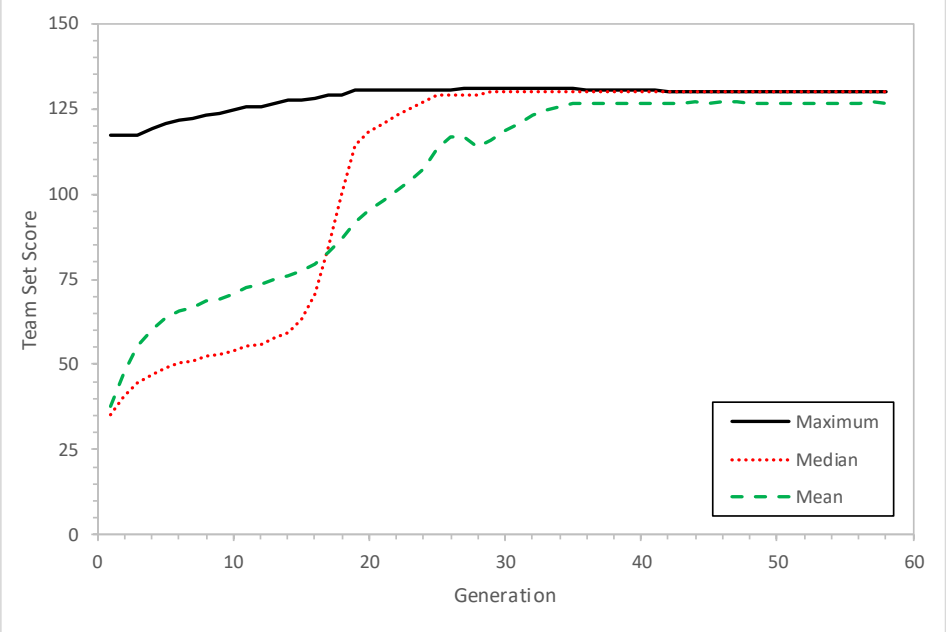

Figure 1: As the genetic algorithm proceeds, the population evolves towards a maximization of the team set scores. Here, the maximum, mean, and median team set scores within the 30,000 genomes in each generation are displayed. Since genomes with higher team set scores are generally used to seed future generations, eventually the population collapses to a stable value for the data set.

To further examine the long term effects of this technology on student's FYE experience, experiential analysis is being pursued, investigating the effectiveness of gruepr as an educational tool and its influence on the student experience. Currently, the interview and survey protocols focus on perceptions of the student design team dynamics and effectiveness based on teams formed with high weighting of complementary schedules. Collection has already begun and the final results will be reported at a later time. Some preliminary data is already available. FYE faculty have participated in some early focus group questioning about their initial perceptions of this tool. 6 faculty members have responded to the initial requests for feedback and have provided some insight into gruepr's usefulness.

In these preliminary interviews, the 6 faculty members who participated were asked individually to comment generally on the following topics:

- What are your common issues with forming student design teams in your courses?
- Do your students have opinions on forming teams and what are they?
- Was gruepr a useful tool for you in this task?
- Do you have any feedback and thoughts on your implementation of this tool and what are they?

The data collected is limited to short responses from each of the participating faculty. The data is intended to further develop and update gruepr and should not be considered complete evidence of its success or failure. Their opinions were found to be consistent enough by the author collecting the feedback to be noteworthy, sufficient to be worthwhile in providing with the rest of this paper's description and discussion of gruepr. The following is a summary of that feedback.

Faculty reported that students generally want the pressure of term length group selection to be placed on the instructor and that this is a non-trivial task. The complexity of this task is understood to be great but not a problem that is insurmountable, and experience helps. Having a freely available open-source option to automate this task is highly preferred, particularly one that focuses on the agreed upon main concern, schedule matching. There are inherent limitations with the tool though, not in its capability of taking on the task but due to the human effect of the faculty using it. Gruepr allows for a wide range of selection criteria and has no limitations on the distribution of importance. This can lead to a crippling effect where everything is important so nothing is important. Untenable chains of student's self-selecting required or prevented teammates can also negatively affect the outcome and lead to diminishing returns from the algorithm. These problems are not the fault of the application per se, but at the same time the application does not provide context or insight into how these faulty decisions will affect the outcomes. The need for guidance or limitations and what exactly they should entail is still an open question.

Faculty are split on their opinions towards gender factoring into group formations. Some espouse the argument that female students should be provided this safer environment of being at least paired on teams while others argue that they may not be so protected in their future work environments. This conversation exposed desires for less binary options for this complex criteria and the ability to be able to provide faculty-determined individual criteria. Currently, faculty cannot easily include their own individual student criteria beyond specifying required/prevented pairings of students and choosing whether to accept gruepr's final configuration of teams. The source data set is based on students self-reporting their preferences and demographics, and faculty cannot easily add to that data set any additional indicators or other demographics that can't be collected via student self-reporting. For example, an instructor may wish to add data such as "likelihood to not pull their weight" or "capability to succeed as an isolated woman in a group." When it comes to subsequent rounds of group forming, once the instructor has gotten a better sense of their students, these factors begin to take root but are not readily usable in gruepr.

There was a fair amount of general shared praise as well. The tool uses a familiar environment in that a Google Form survey is easily customizable and readily accessible by the students, and the final data is simple to retrieve for the faculty. These factors reduce the mental load and time commitment required for this task. Faculty have responded that there is a reduction in the formerly common student complaint that an inability to meet has been detrimental to their project success. Overall, there is agreement that gruepr has a shallow learning curve and builds great teams.

Although the reception to gruepr was very positive, it has some limitations that should be fixed in future versions of the software. When using the pairing mechanism to prevent or ensure particular teams, there is no ability to fix a mistake. For example, if you pair the wrong students, you have to run the program from the beginning and input the data a second time. This can cost instructors time when allowing students to pick other students that they do or do not want to work with. Once the teams are created, there is not an easy way to switch students between the teams and get a new team score and schedule overlap table. This makes last minute tweaks to the teams difficult to make. Both problems could be solved by updating the user interface. Work currently underway involves switching gruepr to a graphical user interface. An initial screenshot is shown in Appendix C. With this interface, the teamings are more fluid and instructors can more easily tweak them once created.

Conclusions

Our work presented here announces and demonstrates the utility of gruepr, a new software program to form students into teams. Under the assumption that one can quantify the arrangement of student qualities that are more likely to result in a successful team, gruepr facilitates the finding of an optimized partitioning of students. Using gruepr, an instructor has significant flexibility to base the optimization upon individual schedules, students skills, and

other attributes. It is shown here to save time compared to making teams manually and to save money compared to competing software. The authors continue to work on the program to make it easier to use. Anyone is welcome to contribute to this project or adapt the program for their own use, since the project source code is freely available under an open source license.

References

[1] J. E. Froyd, *The engineering education coalitions program*, In National Academy of Engineering (Ed.), Educating the engineer of 2020: Adapting engineering education to the New Century, National Academies Press, Washington, DC, 2005.

[2] ABET, *Accreditation policy and procedure manual (2019–2020)*. Baltimore, MD, 2019.

[3] L.K. Michaelsen, N. Davidson, and C.H. Major, "Team-based learning practices and principles in comparison with cooperative learning and problem-based learning,". *Journal on Excellence in College Teaching*, vol. 25(3&4), 2014.

[4] J.L. Bailey and J.Skvoretz, "The Social-psychological Aspects of Team Formation: New Avenues for Research," *Sociology Compass*, vol. 11, no. 6, 2017.

[5] S. El Baroudi, S. N. Khapova, P. G.W. Jansen, and J. Richardson, "Individual and contextual predictors of team member proactivity: what do we know and where do we go from here?" *Human Resource Management Review*, early access, 2018.

[6] M.-I. Sanchez-Segura, M. Hadzikadic, G.-L. Dugarte-Peña, and F. Medina-Dominguez, "Team Formation Using a Systems Thinking Approach," *Systems Research and Behavioral Science*, vol. 35, no. 4, 2018.

[7] M. Borrego, J. Karlin, L. D. McNair and K. Beddoes, "Team Effectiveness Theory from Industrial and Organizational Psychology Applied to Engineering Student Project Teams: A Research Review," Journal of Engineering Education, vol. 102, no. 4, Nov., 2013.

[8] F. Morgeson, M. Reider, & M. Campion, "Selecting individuals in team settings: The importance of social skills, personality characteristics, and teamwork knowledge," *Personnel Psychology*, vol. 58, no.3, 2005.

[9] N. Dasgupta, M. M. Scircle, and M. Hunsinger, "Female peers in small work groups enhance women's motivation, verbal participation, and career aspirations in engineering*," in Proceedings of the National Academy of Sciences*, Apr. 2015, 112 (16).

[10] R. A. Layton, M. L. Loughry, M. W. Ohland, and G. D. Ricco, "Design and Validation of a Web-Based System for Assigning Members to Teams Using Instructor-Specified Criteria," *Advances in Engineering Education*, Spring 2010.

[11] B. Oakley, R.M. Felder, R. Brent, and I. Elhajj, "Turning student groups into effective teams," *Journal of Student Centered Learning*, vol. 2, no. 1, 2004.

[12] E.F. Barkley, C.H. Major, and K.P. Cross, *Collaborative learning techniques: A handbook for college faculty*, 2nd ed., Jossey-Bass, San Francisco, CA, 2014.

[13] L.K. Michaelson, A.B. Knight, and L.D. Fink, (Eds.), *Team-based learning: A transformative use of small groups in college teaching*. 1st ed., Stylus Publishing, Sterling, VA, 2004.

[14] W. McKeachie, and M. Svinicki, *McKeachie's teaching tips,* 14th ed., Wadsworth, Cengage Learning, Belmont, CA, 2014.

[15] M. V. Jamieson and J. M. Shaw, "CATME or ITP Metrics? Which One Should I Use for Design Team Development and Assessment?" *2018 American Society for Engineering Education (ASEE) Annual Conference & Exposition,* Salt Lake City, UT, June 24-27 2018.

[16] C. Kiassat, and X. Jiang, "Systematic Team Formation Leading to Peer Support and Leadership Skills Development," *123rd American Society for Engineering Education, conference & Exposition*, New Orleans, LA, June 26-29, 2016.

[17] A. M. Lucietto, A. S. Scott, K. A. Connor, and F. C. Berry, "Initial Survey of Engineering Technology Capstone Courses and Teamwork Building Using CATME," *2017 American Society for Engineering Education Annual Conference & Exposition, Columbus, OH,* June 25-28, 2017.

[18] M. W. Ohland, R. A. Layton, D. M. Ferguson, M. L. Loughry, and H. R. Pomeranz, "SMARTER Teamwork: System for Management, Assessment, Research, Training, Education, and Remediation for Teamwork," *Proceedings of the 118th ASEE Annual Conference,* 2014.

[19] http://bit.ly/Gruepr

[20] http://bit.ly/TeamFormSurv

[21] D. Whitley, "A genetic algorithm tutorial," *Statistics and Computing*, vol. 4, pp. 65-85, 1994.

[22] D.E. Goldberg and K. Deb, "A Comparative Analysis of Selection Schemes Used in Genetic Algorithms," *Foundations of Genetic Algorithms,* vol. 1, pp. 69-93, 1991.

[23] T. Starkweather, S. McDaniel, K. Mathias, D. Whitley, and C. Whitley, “A Comparison of Genetic Sequencing Operators,” *Proceedings on the Fourth International Conference on Genetic Algorithms*, pp. 69-76, 1991.

Appendix A - Prototypical set of questions used to survey students using a Google Form

There is considerable flexibility in the set of questions asked of the students. This example survey includes three attribute questions, listed as questions 5-7.

1. What is your first name (or the name you prefer to be called)?

_____

2. What is your last name?

_____

3. What is your school email address?

_____

4. With which gender do you identify?

[Woman / Man / Non-binary / Prefer not to answer]

5. What is your personal experience with computer programming?

[1. I have NEVER written a computer program. / 2. I have VERY LIMITED prior experience writing computer programs. / 3. I have written computer programs A FEW TIMES but would need some time to relearn before doing it again. / 4. I have written computer programs A NUMBER OF TIMES and could do so now if asked.]

6. What is your personal experience with hands-on build or repair tasks?

[1. I have NEVER built or repaired anything by hand. / 2. I have VERY LIMITED prior experience building or repairing things. / 3. I have built or repaired things by hand A FEW TIMES but would feel somewhat unsure to do so now. / 4. I have built or repaired things by hand A NUMBER OF TIMES and would feel comfortable doing so now.]

7. What is your preferred role in a group setting?

[1. I prefer to LEAD my team. / 2. I like a BALANCE between leading and following. / 3. I like to FOLLOW the lead of someone else.]

8. Check the times that you are BUSY and will be UNAVAILABLE for group work.

[A clickable matrix of 7 days by 14 working hours is offered]

9. In which section are you enrolled?

[Section A / Section B / Section C / Section D]

10. Any additional things we should know about you before we form the teams?

_____

Appendix B - Example results output from gruepr

Only one team's data is shown; actual output would also show the results for the other teams in the team set. Below the team number is the team's compatibility score, and below that are four rows of data summarizing the four students on the team. Specifically, a "W" indicates a student who is a woman, and the three numerical values represent the student's responses to the three attribute questions (see the attribute questions in Appendix A). Finally, a student's name and email address is shown. In this demonstration, women were prevented from being isolated, homogeneity was desired in the first two attributes and heterogeneity was desired in the third attribute.

The output shown below is what is intended to be used by the instructor. Gruepr also produces output that is intended to be given to the students. That output is identical except it omits the team's compatibility score and the students' gender and attribute values.

```
Team 1

Score = 118.82

    W  3  3  2  Abby Doe         doe.a@college.edu
       3  3  2  Brian Smith      smith.b@college.edu
       3  3  1  Chris Williams   williams.c@college.edu
    W  3  4  2  Delilah Jones    jones.d@college.edu

Availability:
                 Sun    Mon    Tue    Wed    Thu    Fri    Sat
8am              25%    25%    25%    25%    25%    25%    25%
9am              50%    25%    25%    25%    25%    50%    50%
10am             75%     0%    25%     0%     0%    50%    75%
11am            100%     0%    50%    25%     0%    75%   100%
noon            100%     0%   100%    25%     0%   100%   100%
1pm             100%     0%   100%     0%     0%   100%   100%
2pm             100%     0%    75%     0%     0%   100%   100%
3pm             100%    50%    75%    50%    75%   100%   100%
4pm             100%    50%    75%    25%    75%   100%   100%
5pm              75%   100%    75%   100%   100%   100%   100%
6pm              75%   100%    75%   100%   100%   100%    75%
7pm              50%    75%    75%    75%    75%    75%    50%
8pm              50%    50%    75%    75%    75%    50%    25%
9pm              50%    50%    75%    75%    75%    50%    25%
```

Appendix C - Screenshot of the next version of gruepr

A new version of gruepr is currently being written by the authors. It has a graphical user interface, shown below.