Diversity Considerations in Team Formation Design, Algorithm, and Measurement

Bowen Hui
The University of British Columbia
Kelowna, Canada
bowen.hui@ubc.ca

Opey Adeyemi
The University of British Columbia
Kelowna, Canada
opeyadeyemi@gmail.com

Kiet Phan
The University of British Columbia
Kelowna, Canada
phankiet.at.ubc@gmail.com

Justin Schoenit
The University of British Columbia
Kelowna, Canada
justsch@student.ubc.ca

Seth Akins
The University of British Columbia
Kelowna, Canada
sethlinc@student.ubc.ca

Keyvan Khademi
The University of British Columbia
Kelowna, Canada
keyvankhademi@gmail.com

Abstract

Building teams that foster equitable interaction provides the foundation for a positive collaborative learning experience. Existing literature shows that many context-specific algorithms exist to help instructors form teams automatically in large classes, but the field lacks general guidelines for selecting a suitable algorithm in a given pedagogical context and lacks a general evaluation approach that allows for the methodological comparison of these algorithms. This paper presents a general-purpose team formation algorithm that considers diversity and inclusion in its design. We also describe an evaluation framework with diversity metrics to assess team compositions using synthetically generated student data and real class data. Our simulation and classroom experiments show that our algorithm performs competitively against three state-of-the-art algorithms. We hope this work contributes to building a more equitable and collaborative learning environment for students.

CCS Concepts

• **Human-centered computing** → **Collaborative and social computing design and evaluation methods**; • **General and reference** → **Metrics**; • **Social and professional topics** → *User characteristics.*

Keywords

Team formation, evaluation, diversity metrics, hill climbing

ACM Reference Format:
Bowen Hui, Opey Adeyemi, Kiet Phan, Justin Schoenit, Seth Akins, and Keyvan Khademi. 2025. Diversity Considerations in Team Formation Design, Algorithm, and Measurement. In *LAK25: The 15th International Learning Analytics and Knowledge Conference (LAK 2025), March 03–07, 2025, Dublin, Ireland*. ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3706468.3706473

LAK 2025, Dublin, Ireland

ACM ISBN 979-8-4007-0701-8/25/03
https://doi.org/10.1145/3706468.3706473

1 Introduction

Collaborative learning is central to many educational settings that foster critical thinking, problem-solving, and teamwork. The compositional structure of a team, or group, plays a fundamental role in determining the quality of the work output and team experience. This work focuses on design considerations in team formation to facilitate positive collaborative outcomes. We define the task of *team formation* in the classroom as the act of allocating all the students into non-overlapping groups. In the computer-supported collaborative learning (CSCL) literature, this task is also called *team assembly* [18] and *group formation* [7]. This task is NP-hard because finding an optimal solution requires computing all the possible combinations which cannot be done in polynomial time [17, 29]. Forming effective teams in large classes is challenging due to the complexity of project needs, the available learner characteristics, and the varying criteria suitable in different pedagogical settings. Studies show forming groups randomly can generate unbalanced teams that result in disproportional individual participation and self-assembled teams can cause discrimination among students with poor social relationships [16, 22]. Educators must be mindful of creating teams that foster balanced interactions so that students can maximize their learning gains. Building teams manually is not viable with large class sizes (e.g., a researcher spent 60 hours to form teams for 240 students [44]). Thus, we turn to automated solutions.

A long-standing question in CSCL research is: How should teams be formed? Choosing appropriate criteria impacts the learning opportunities in a given pedagogical context [25]. Studies found that heterogeneous and homogeneous teams afford different learning benefits in varying contexts [31]. Research also reports using hybrid criteria – a mix of heterogeneous and homogeneous criteria – to form teams simultaneously [7, 34, 36]. Moreover, choosing a suitable team formation technique remains at the discretion of the researcher. The literature reveals numerous algorithms proposed for team formation, including genetic algorithms, clustering, hill climbing, multiagent systems, and heuristic techniques [13, 14, 34, 36]. A recent review identifies a knowledge gap in the field where there are no clear guidelines to help educators select an appropriate team formation technique for their pedagogical context [34]. Conversely, general-purpose algorithms that handle multiple types of criteria and learning contexts are rare (though we report on an exception in Section 2).

Moreover, while educators generally agree team diversity is important, studies reveal conflicting results that diversity has on team outcomes and how team diversity is defined (e.g., skills, demographics, goal-orientation) [22]. In particular, gender and racially diverse teams often result in more conflict, and minoritized members are confronted with microaggressions [37], perceived as less skillful than their peers in homogeneous teams [6, 38], and treated with bias by other group members as they are not heard, not given leadership roles, and pressured to change behaviors stereotypical of their identity [20, 33, 44]. These problems are exacerbated when minoritized individuals are disproportionately represented – a notion called *tokenism* that embodies the interaction effects between dominant and minority group members [26, 41, 44, 46]. We refer to these minority group members as *tokens*. Tokenism is sometimes called *solo status* [41, 46], though its effects may surface even when multiple tokens exist in a group [26, 44]. From this view, we join the call to action in enabling equitable collaborative experiences as a necessary educational change [27, 48].

In developing team formation tools, we must consider factors that impact the team formation process and the students' collaborative learning experience. Our main objective is to explore the design of flexible, general-purpose algorithms that are sensitive to and allow the customization of diversity parameter settings. The specific research questions are:

RQ1: What are the diversity considerations in designing a general-purpose team formation algorithm that accounts for multiple types of criteria simultaneously, such as project requirements, learner characteristics, and social preferences?

RQ2: How should diversity metrics be defined to facilitate the verification of team compositions and performance comparisons across algorithms?

Section 2 reviews team formation literature with an eye toward diversity and evaluation methodology. Section 3 presents a novel general-purpose algorithm called the *priority* algorithm that forms diverse teams, avoids tokenizing minority students, considers social preferences, and matches student skills to project requirements. We developed an open evaluation framework to compare the performance of our priority algorithm in Section 4 using both simulated and real class data. Lastly, Section 5 reports on the competitive performance of the priority algorithm based on three formal diversity metrics that measure team compositions. Our work contributes to the learning analytics community by (i) introducing a metric-driven team diagnosis phase that enables users to evaluate and modify team compositions; (ii) supporting educators in building equitable teams that avoid tokenizing minorities, and (iii) enhancing the research community with an open evaluation framework that enables a methodological comparison of team formation algorithms.

Ultimately, we wish to embed the learning analytics loop in team formation tools. To our knowledge, existing tools do not yet close the loop by gathering team outcomes, summarizing them with meaningful metrics, and tweaking future team formation strategies in the same pedagogical context as experimental interventions. Doing so offers future students an improved learning experience and equips instructors with knowledge on how best to form teams in their classes. Evaluating algorithmic performance systematically is our first step toward this goal.

2 Related Work

2.1 Learner Characteristics, Team Formation Criteria, and Algorithmic Approaches

The CSCL literature reports a large variety of learner characteristics that include interest, knowledge, personality traits, demographics, learning styles, and social preferences to form teams via a mix of heterogeneous, homogeneous, and hybrid criteria combinations with those characteristics [7, 13, 14, 34, 36]. Additionally, software engineering techniques commonly use team roles, historical data, project requirements, and student skills to form programming teams [11].

Separately, extensive research in AI has tackled instances of the group formation problem with an emphasis on hardness results. The *Group Activity Selection Problem* (GASP) [15] presents a setting where individuals are assigned to groups based on their preferences over those groups and their potential teammates [24]. Solutions to GASP do not consider task requirements or student skills and, therefore, cannot handle scenarios that match student skills to project requirements.

The optimization community defines the *Team Formation Problem* (TFP) as the creation of a single team to complete one task by matching individual skills to task requirements while minimizing their communication costs [29]. Subsequent attempts generalized to solving multiple TFP instances using constraint programming and discrete optimization [21]. However, these approaches are limited to modeling each person with one skill so that a talented individual may be assigned multiple tasks sequentially. In a classroom where we wish to assign all the students to non-overlapping teams, a more general approach is warranted. A similar method is used in the multiagent community called *Fair Division*, where the objective is to assign resources (i.e., projects) to agents (i.e., students) so that the resulting allocation is fair and agents do not envy each other's assignment [5].

Across these areas, some of the team formation algorithms can flexibly handle multiple attributes and attributes of varying data types [13, 14, 34, 36]. These algorithms mostly belong to the search and optimization category [11], including genetic algorithms, ant colony optimization, particular swarming optimization, and hill climbing.

The choice of learner characteristics and how individuals are represented in an algorithm can have negative consequences. One study models students using a vector representation where each characteristic is a multivariate attribute (e.g., gender with values 1 for woman, 2 for man, 3 for non-binary, etc.). [19]. The authors use Euclidean distance to measure the similarity between two vectors and define the *goodness of heterogeneity* based on a "middle" student in the team. While this approach benefits from a clear mathematical formulation, comparing nominal variables that do not have a natural ordering can result in a biased interpretation. Using the same example, the distance calculation suggests students with gender value 2 (man) are equally similar to students of gender values 1 (woman) and 3 (non-binary), while students of gender value 1 (woman) are twice as different as those of gender value 3 (non-binary). Other approaches that use a similar representation and distance measure suffer from the same issue (e.g., [10, 28, 39]).

2.2 Evaluation Mechanisms

The literature reveals group formation algorithms suffer from poor evaluation methodology, where some papers propose an idea without implementing it [14] and some do not provide any evaluation at all [34]. In many cases, the technique is applied to a toy example (e.g., [3, 42, 45, 47]) or larger classes without comparing it against other algorithms (e.g., [27, 39]). Recent studies began comparing algorithmically generated teams to manually built teams or to random and self-assembled teams by measuring learning gains and student perceptions [2, 12, 27, 30, 31, 43]. Comparisons with non-trivial algorithms are limited to simulations only [25, 49, 50]. A notable exception exists where the authors evaluated their novel algorithm in simulation against other state-of-the-art algorithms and applied their algorithm to large classes to assess project coverage, team diversity, and student perceptions [8]. In the same spirit, we advocate for a comparative evaluation in simulation before applying a new algorithm to classrooms. The lack of comparative evaluations points to the need for a general framework for evaluating group formation algorithms [14]. Our work offers an open simulation framework that serves as a starting point to address this issue.

2.3 Comparison Metrics

Before running classroom studies, we should ideally use analytics to inspect the well-formedness of the generated teams and change the team compositions as needed before testing with human participants. Some papers are limited in presenting a listing or a visual of the generated teams and student characteristics, leaving the reader to decipher the goodness of the team composition (e.g., [2, 3, 39]). We also found research that reported technique-specific metrics, such as fitness values for genetic algorithms [35, 43] and objective values for hill climbing techniques [25]. However, these values do not carry meaning beyond the specific algorithm or experimental setting. On the contrary, most AI papers use objective metrics that are relevant to the application domain and generalizable across algorithms. Example metrics include runtime, the communication cost of a team, project requirements coverage called *activity cover*, and *envy-freeness* to assess the fairness and stability of teams [9, 15, 21, 24, 29]. Our work adopts this approach by proposing generalizable metrics to diagnose team compositions and measure algorithm performance.

A major discovery from our literature review is that very few algorithms verify the diversity of learner characteristics when creating heterogeneous teams. The notion of forming *intra-heterogenous* groups has been explored by many researchers (e.g., [3, 19, 32, 35]) but no metric was used to validate the heterogeneity of the generated teams. In a class where multiple teams are formed, researchers also suggest that groups should be similarly diverse by describing the teams as *inter-homogenous* [35, 42], but we did not find a formal definition for it.

Finally, we are only aware of four algorithms that specifically address team diversity to foster equitable collaboration — three of these do not provide a generalizable measurement of team diversity in their results. An early integer programming approach considered students' gender and language preference in a class of 22 students and showed that the women were split across multiple teams and the two non-native English speakers were grouped together [3]. However, only a listing of the teams was given and no metrics were used to quantify the generated output.

In another study, Bulmer et al. adapted a fair division algorithm to implement their greedy solution called greedy round robin (GRR) that gives all the agents a chance to select their preferred resources (more details in Section 4) [8, 9]. The authors used GRR to create teams that diversify women in a male-dominant class, in addition to matching student skills to project requirements and considering social preferences. To measure gender diversity in teams, the authors used the Gini-Simpson index to represent the proportional frequencies of the different genders [40].

Kalantzi et al. aimed to form teams where members have equal opportunity to benefit from collaborative work [25]. The authors developed an interesting concept that quantifies a member's *individual benefit* from others in the team based on their skill levels. Given specific *protected* attributes (e.g., gender, race), they defined the *group benefit* of a protected attribute to be the average individual benefit of the members in that group. These concepts were embedded in the objective function of their hill climbing algorithm which aimed to find a fair allocation. The results were presented in terms of objective values, which makes it difficult to interpret and generalize.

The only work we are aware of that addresses tokenism in teams algorithmically is group matcher [27], which creates an initial partition of peer mentoring groups based on time availability and iteratively removes tokens by merging partitions to produce the final allocation. However, the authors did not provide any analysis of the solo status or common times on the generated teams.

3 The Priority Algorithm

We propose the priority algorithm which was inspired by a human-centered process of finding the best student teams while considering the highest-priority criterion, then repeating the process with the second-highest priority, and so forth, in a greedy fashion. The implementation adopts a hill-climbing approach that, given an arbitrary solution (Line 1), iteratively improves upon it (Lines 7 to 11). We add to this idea the ability to simultaneously climb multiple paths chosen stochastically (called SPREAD) as a way to escape local minima. Team sets are pruned to produce the K highest scoring candidates (Line 13) and only the best team set is returned in the end (Line 17). Algorithm 1 presents the pseudocode and Figure 1 illustrates the hill climbing process.

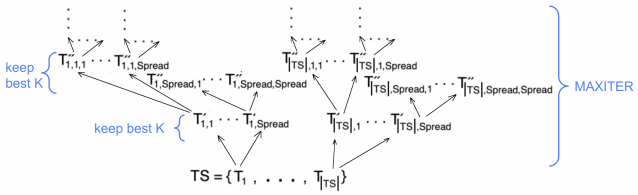

Figure 1: A schematic of the hill climbing process showing the team set expansion at each iteration.

Algorithm 1 Generate an initial team set and incrementally improve it based on an ordered list of criteria.

Input: A set of students with characteristics S, an ordered list of criteria, C
Output: A set of teams, $\hat{T}$

```
T ← random(S)                          ▷ An initial arbitrary solution
TS ← {T}                               ▷ A set of team sets
Let num = 1
while num < MAXITER do
    Candidates ← TS
    for each T ∈ TS do                 ▷ Improve each TS element
        for i from 1 to SPREAD do
            T'_i ← improve(T, C)
            s_i ← score(T'_i, C)
            Candidates ← Candidates ∪ T'_i
        end for
    end for
    Keep K best Candidates w.r.t. C
    TS ← Candidates                    ▷ Rename variables
    num++
end while
Return T̂ ← arg max TS                  ▷ highest scoring team set
```

3.1 Objective Function: Priority Scoring

The flexibility of this algorithm comes from the objective function captured in score(), which evaluates a team set T on how well it satisfies each criterion in an ordered list, C_1, ..., C_J, with J being the index to the most important criterion, $J-1$ for the second most important, and so on. The score() function is defined as the sum of the satisfaction scores weighted against the importance of each criterion, i.e., $\sum_{j=1}^{J}(B^{j-1} \times normalize(satisfy(T, C_j)))$, where B^{j-1} expresses the importance of the j^{th} criterion, satisfy() computes a score in the range $[0, 1]$ to express how well T satisfies C_j, and normalize() maps that score to an integer in $[0, B-1]$. The purpose of normalize() is to categorize the satisfaction scores into B bins so that we can compare their values with less granularity. For example, two team sets will be treated the same even if one team set has a slightly higher satisfaction score than another (e.g., 0.451 versus 0.4512). Also, the bin levels express the degree of importance that a higher-priority criterion has over a lower-priority criterion.

The specific implementation of satisfy() depends on the type of criterion, such as scoring the team based on the presence of tokenized minorities or checking the percentage of skillful members who meet the necessary project requirements. Currently, our algorithm handles criteria for matching skills to project requirements, satisfying social preferences, clustering learner characteristics, and diversifying learner characteristics with the option to avoid creating tokens (e.g., diversifying women while placing more than some number of them together).

3.1.1 Representing Student Attributes. Rather than defining a learner variable as a multiattribute vector that suffers from the bias issue raised in Section 2, we take the relevant values of the variable and model them as bit vectors. For example, for each value of gender, we create separate bits to represent is-woman, is-man, is-nonbinary, etc. These values combine into a bit vector to represent a student's gender. Thus, comparing the distance between two bit vectors will give equitable treatment in assessing gender differences. Intersectionality can be handled analogously (e.g., is-racialized-woman).

3.1.2 Calculating Diversity. To measure how well a set of teams T satisfies a diversity criterion involving a learner characteristic such as gender, score() uses the Gini-Simpson index to determine the proportion of individuals with different values on the same team and averages this result across all the teams. Given N students in a class and A values in a learner characteristic, the time complexity for this calculation is $O(NA)$.

3.1.3 Calculating Token Minorities. We extend the diversity criterion by specifying that the user can diversify a characteristic value while having either 0 or at least m minorities in a team. This formulation gives the user the ability to customize the minimum number of tokens to have on a team. We include the option of 0 because the number of minorities and teams needed may not divide evenly. In such cases, we prefer to form teams without any minorities over creating a team with a token minority. The detailed function is defined as a piecewise function that rewards solutions for placing m minorities in a team relative to the size of the team, n, and a minimum threshold score, θ, as $f(x) = (\frac{2\theta-1}{n-m-1})(x-n)+\theta$ illustrated in Figure 2. In essence, when no minorities are present on the team, the score is $f(0) = \theta$. When the number of minorities on a team is fewer than m, we assign a low score of $f(0 < x < m) = 0$ because we anticipate the team to experience tokenism effects. The ideal case is when we have m minorities in it, so we set $f(m) = 1$. Adding more minorities to the same team decreases the score from $f(m+1) = 1-\theta$ to $f(n) = \theta$ to show the preference for more optimized teams. Setting $f(n) = \theta$ avoids a solution that simply puts all the minorities on the same team. Note that θ must be less than 0.5 otherwise the negative slope in the function becomes positive. Currently, our algorithm sets $\theta = 0.2$. Computing the value for $f(x)$ takes $O(1)$ and determining if each student belongs to a tokenized group takes $O(N)$.

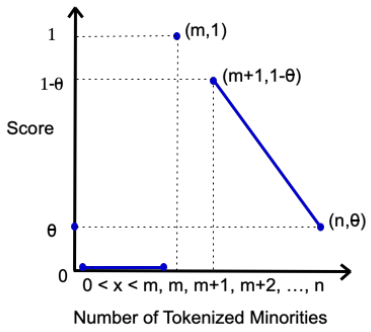

Figure 2: The piecewise function to measure how good it is to have m minorities on a team of size n.

3.2 The Improvement Function

Given an allocation of students to teams, T, improve() augments the teams to obtain T'. The current implementation naively swaps two students from different teams in T randomly. While we do not guarantee the swap results in a better solution, we note that since *Candidates* includes the original team set and the pruning step only considers the highest-scoring teams, the final score of $\hat{T}$ will be no less than the initial score of T. We leave alternative implementations as future work.

3.3 Understanding Algorithmic Behaviors

Recall Algorithm 1 has three parameters: MAXITER, SPREAD, and K. The purpose of this section is to assess the impact of these parameters. First, we generated 120 synthetic students with simple characteristic distributions to ensure it was possible to form reasonably balanced teams. Specifically, each student has 5 binary attributes with 50% of them randomly assigned to each attribute value. The scenario specified that all the characteristics should be diversified with a minimum of 2 individuals of each attribute in each team. We created additional scenarios with 30 projects each with 1 requirement, and only 80% of the projects can be satisfied by the available students.

Due to the presence of multiple priorities, we created an overall performance metric called **priority satisfaction**. This metric combines the satisfaction scores of each criterion weighted by the importance of each priority, i.e., $\sum_{j=1}^{J}(B^{j-1} \times satisfy(T, C_j))$, where $B = 2$. Since this metric closely resembles our algorithm's scoring function, we limit its use to internal validation only.

Figure 3 shows the priority satisfaction results with two different starting algorithms – random (red dashed line) and GRR [9] (blue solid line). We chose to use GRR due to its generality. Each subfigure shows a fixed value of MAXITER while varying K and SPREAD, evaluated on priority satisfaction. Figure 3(a-b) shows the results with the diversity criteria described earlier and Figure 3(c-d) shows the results in a project scenario. Due to space limitations, we only include the snapshots with MAXITER=1 in Figure 3(a) and (c), and with MAXITER=30 in Figure 3(b) and (d).

Generally, Figure 3(a-b) shows increasing the parameter values improves the algorithm performance. When MAXITER=1, varying K and SPREAD offers little improvement. When MAXITER=30, increasing SPREAD slightly improves the priority algorithm with a random initial solution, but offers little benefit when GRR is used initially. This pattern suggests MAXITER, which controls the depth of the search, has more influence over the algorithmic performance than SPREAD and K, which control the number of candidates to evaluate simultaneously and keep for future iterations.

The project scenario results in Figure 3(c-d) show that a random initial solution achieves very low performance but can improve as K, SPREAD, and MAXITER increase. On the other hand, a GRR initial solution gives us an immediate advantage of a near-optimal solution such that increasing K and SPREAD results in little benefits. Overall, the priority algorithm achieves similar levels of performance regardless of the type of team formation criteria involved.

We increased the search complexity to investigate the impact on performance by introducing a new scenario with social constraints that are difficult to satisfy: each student in a class of 120 has only 1 mutual friend and we form teams by satisfying their friendships. Figure 4(a) shows reasonable priority satisfaction results after increasing the parameters to MAXITER=250, SPREAD=100, and K=30. The rest of Figure 4 shows other combinations of team formation criteria added to this social criterion. Generally, increasing the parameter values improves the overall performance.

4 Evaluation Setup

We create an evaluation framework for comparing algorithmic performance across real and synthetic scenarios based on the metrics provided. There are three inputs. First, the user specifies the team formation criteria by identifying the relevant learner characteristics, potential project requirements, and social preferences. Second, the Class Composition Module provides a class of (synthetic or real) students to the Simulation Controller. The user can specify a class distribution for every learner characteristic so that a class of students can be randomly generated according to the distribution parameters. Real-class data can be preprocessed to match the expected syntax and used as input. The third input is the set of team formation algorithms (see Section 4.3) made available through an API endpoint to our repository[1]. Given these inputs, the Simulation Controller runs all the algorithms over a specified number of trials. The output is a resulting allocation from each trial for each configuration of team formation criteria, distribution of students, and algorithms.

4.1 Scenarios for Synthetic Data

Overall, we create scenarios involving projects and diversity criteria to demonstrate the flexibility of the algorithms and their performance in building diverse teams. Specifically, Scenario 1 matches student skills to project requirements, diversifies women without tokenizing them ($m = 2$), and diversifies students with an African background without tokenizing them ($m = 2$). We created 5 unique projects, each with 3 to 5 requirements, and made an equal number of duplicate projects needed to form teams of 4 for class sizes of 20, 100, 240, 500, and 1000. The student population had the following characteristic distributions: 20% women, 80% men; 15% African background, 85% European background; and each project requirement had 10-80% students with the skill to satisfy it.

Scenario 2 clusters common times, diversifies women without tokenizing them ($m = 2$), and diversifies those with an African cultural background without tokenizing them ($m = 2$). We used the same class sizes and population distributions from Scenario 1 and aimed to form teams with 4 students. There were a total of 10 time slots and each student had 3 to 5 available times. For simplicity, gender and ethnicity are defined as binary attributes, but the representation and calculations generalize beyond binary values.

Additionally, we designed Scenario 3 to investigate different approaches to handling tokenism constraints. Although we formulated the tokenism criterion as a diversification constraint, we could alternatively frame this objective by clustering minorities together. Thus, Scenario 3 clusters common times, women, and African students. The same population distributions and class sizes

[1]The Algorithms repository is available at https://github.com/Teamable-Analytics/algorithms.

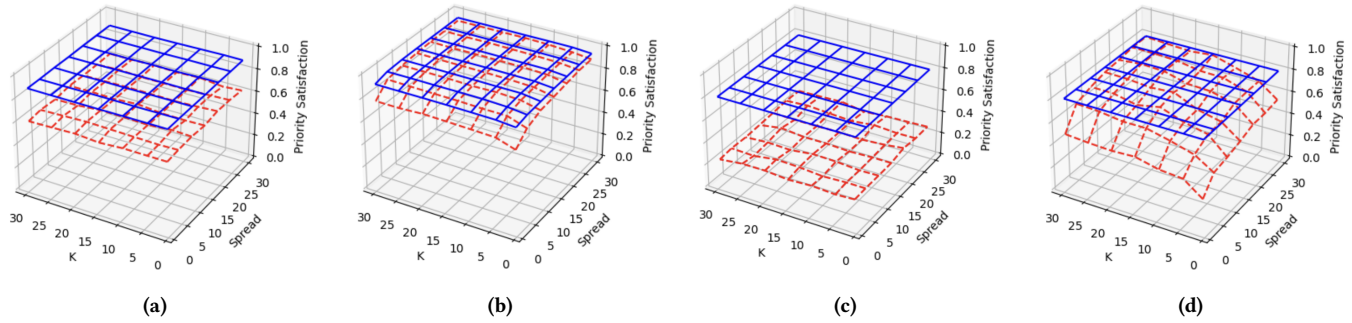

Figure 3: Scenario with 5 diversity criteria while minimizing tokenism (a) using MAXITER=1 and (b) MAXITER=30. Scenario with projects where at best 80% of the students can satisfy the requirements (c) using MAXITER=1 and (d) MAXITER=30. Results using the weight algorithm initially are shown as blue solid lines and those using random initially are shown as red dashed lines. Generally, performance improves as MAXITER increases.

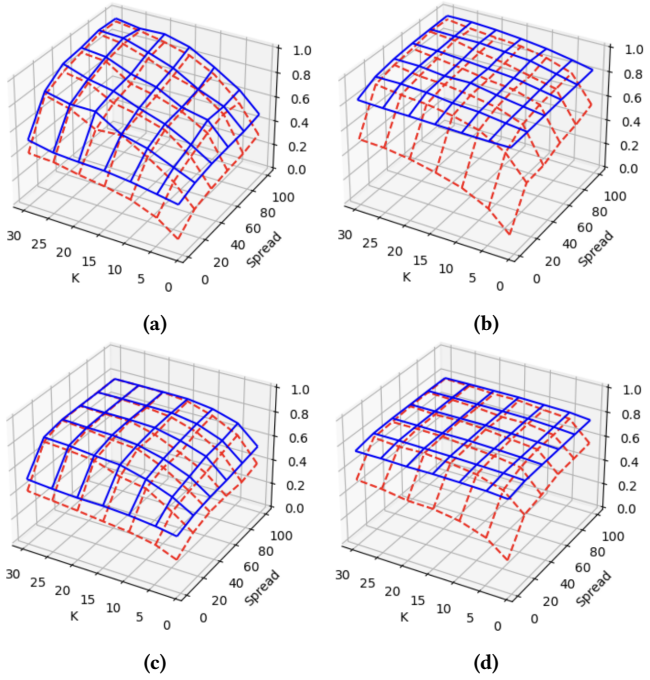

Figure 4: Scenario with sparse social relationships using MAXITER=250 and measuring priority satisfaction along the z-axis: (a) social criterion only, (b) project requirements and social criteria, (c) diversify females while minimizing token individuals and social criteria, and (d) projects, diversify females while minimizing token individuals, and social criteria. Priority that uses random initially is shown as red dashed lines and priority that uses weight initially is shown in blue solid lines.

from Scenarios 1 and 2 are used, with the goal of forming teams of 4 students.

4.2 Real Class Data

We use real student data to evaluate the algorithms in a realistic setting. We collected data from two university classes in 2022 and 2023. The first had 215 students, although only 175 students responded to a survey that elicited background information. We chose to omit students without a survey response from this dataset because different algorithms handle unspecified data differently. The remaining 175 students had the following characteristics distributions: 168 undergraduate students and 7 graduate students; 37 women, 135 men, 2 non-binary, and 1 preferred not to answer; and six timeslot availabilities (t_1, ..., t_6) with 31 students available in t_1, 82 in t_2, 88 in t_3, 103 in t_4, 138 in t_5, and 96 in t_6. In this class, the instructor wished to formed teams of 4 by grouping students with common times, spreading the graduate students across the teams, and diversifying women without tokenizing them ($m = 2$).

The second class had 9 client-sponsored projects, each with 2 to 5 requirements. Seven of these projects were unique and two were duplicates. (Thus, two teams worked on identical projects.) Fifteen or more students can meet all but one requirement, and nobody has the skill to meet the last requirement. The class had 41 students with the following characteristic distributions: for grades we have 4 A+ average, 16 A's, 18 B's, 3 D's, 0 F's (C's were mistakenly omitted); six time availabilities (t_1, ..., t_6) with 13 students available in t_1, 28 in t_2, 28 in t_3, 33 in t_4, 29 in t_5, and 22 in t_6; and every student had the opportunity to name up to 3 friends to be on the same team and up to 3 enemies to avoid. We formed teams of 4 or 5 students by matching project requirements, diversifying grades, clustering timeslots, and satisfying social preferences.

4.3 Comparison Algorithms

We chose **random** to mark the baseline performance. We also selected a fair division algorithm called **double round robin** (DRR) [4] to compare against an AI algorithm designed to handle project scenarios. The formulation involves a utility function between each resource and agent that expresses how desirable it is for an agent to have that resource. DRR assigns resources to agents by having teams take turns selecting the most favorable chore (with negative utility) and then the most favorable good (with positive utility)

in reverse order. This algorithm matches skillful individuals to projects, but cannot address diversity and preferences over projects or teammates.

We selected **greedy round robin** (GRR), introduced in Section 2, as a comparison algorithm due to its generality [9]. In essence, GRR extended the utility model of DRR and allows for multiple round-robin iterations implemented in a greedy fashion. GRR runs in quadratic time as a function of the number of projects and students.

Additionally, we selected a computer science education algorithm called **group matcher** [27] that directly addresses tokenism in study groups, as mentioned in Section 2. Group matcher was designed to create peer mentoring groups so that not all students need to be allocated to teams and it cannot be set to form a fixed number of teams or teams with a fixed size.

4.4 Evaluation Metrics

We define three diversity metrics to describe team compositions:

- **Intra-Heterogeneity**: Within a team, how diverse are the students for a given learner characteristic x? Let x be represented as a bit vector. We define intra-heterogeneity as the cosine difference between two vectors, $1 - cos(\theta)$ which ranges in $[0, 1]$, with 1 being the highest heterogeneity and 0 representing two identical vectors. The class-level intra-heterogeneity is the average of the intra-heterogeneity values of each team.
- **Inter-Homogeneity**: How similarly diverse are the teams in the allocation? This metric is defined as the standard deviation of the intra-heterogeneity values across all the teams to reflect the distribution of the diversity of teams in the class.
- **Solo Status**: Are there minority tokens in the teams? Solo status is the number of token minorities in teams regarding characteristic x divided by the total number of students in the class. This definition allows a team to have more than one token.

We chose to use cosine difference to measure intra-heterogeneity because its values range in [0,1] which offers a consistent and intuitive interpretation. In contrast, the range of the Gini-Simpson index is sensitive to the number of classification types (i.e., the number of values of a learner characteristic), thus, affording less intuitive interpretations.

Although our definition of intra-heterogeneity above is defined in terms of a single learner characteristic x, our student vector representation allows for a straightforward extension to describe the diversity of a team with respect to multiple learner characteristics simply by combining the individual vectors into one. Likewise, solo status can also be extended by summing the number of token minorities based on the characteristics of interest in the numerator.

Beyond measuring diversity, we define domain metrics relevant to other team formation criteria:

- **Project Coverage**: The proportion of requirements met by at least one team member, averaged across all projects.
- **Common Times**: The number of time slots all members can attend divided by the total possible slots, averaged over all the teams.
- **Social Satisfaction**: The percentage of members with satisfied social preferences, averaged over all the teams. A member's social preferences are satisfied if they have at least one friend and no enemies on their team.
- **Run Time**: How long does it take to generate a team allocation for a given class?

5 Results

5.1 Synthetic Data Comparison with Other Algorithms

We report on a runtime comparison in Figure 5 using an Intel i9-13900K processor. Generally, all the algorithms are fast. When the class size reaches 1,000 students, the priority algorithm at the lower parameter setting of K=15, Spread=15, and MAXITER=30, takes 2.67 minutes. We did not plot the priority algorithm with higher parameter settings because they took significantly longer and would skew the graph. In particular, when we increased the parameters to MAXITER=250, Spread=100, K=30, the speed on average increased to 11.6 minutes for a class of 100 students and 134.7 minutes for a class of 1,000 students. Although this runtime seems slow, we note that some algorithms reported on average 2 hours to 167 hours [32, 50]. Furthermore, it is possible to optimize the speed of the priority algorithm by implementing a more intelligent search and using pruning techniques, which we leave for future investigation.

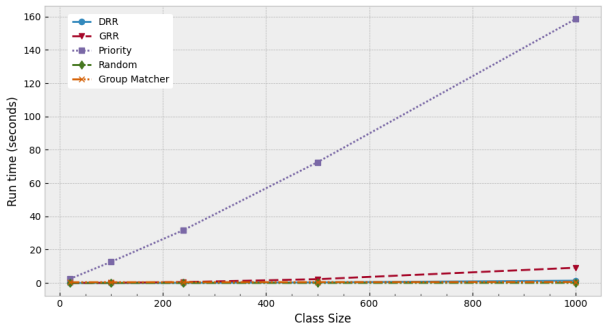

Figure 5: Runtimes with priority parameters set to MAXITER=30, Spread=15, K=15. Priority with the higher parameters is excluded for clarity.

Before presenting the simulation results, we note that DRR and group matcher created a different number of teams and sometimes had different team sizes. For example, in a class of 200 students where we wish to form teams of 4, we expect the algorithms to generate 50 teams. In our scenarios, random, GRR, and priority all formed teams with 4 students. For Scenario 1, DRR generated 50 teams but their sizes varied between 1 and 10 students. In Scenarios 2 and 3, DRR put everyone into one team because it does not handle scenarios without projects. Across all three scenarios, group matcher generated fewer than 50 teams of varying sizes between 1 and 6 students. Thus, DRR and group matcher are not suitable for group formation in a class where teams have the same number of students for equitable grading reasons. Moreover, since DRR was designed to match students to projects, group work without projects should not use DRR to form teams. With this in mind, we

present the simulation results averaged over 100 trials, where the priority algorithm parameters are MAXITER=250, SPREAD=100, K=30, and the initial algorithm is GRR.

Figure 6 shows that priority and GRR perform best on project coverage. Random also does fairly well because the project requirements have a good chance of being matched. Relatively, group matcher comes close to random. Contrary to expectations, DRR performs the worst because it is unable to handle duplicate projects that are equally good for people to choose from.

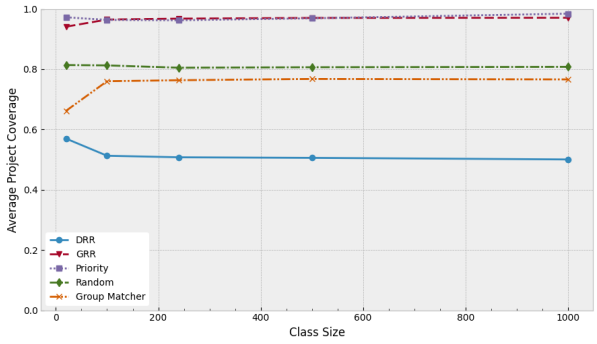

Figure 6: Scenario 1: Project coverage performance averaged over 100 trials.

Figure 7(a-c) shows the results for Scenario 2. GRR and priority do well on common times, likely because they are better at handling competing constraints. They also do better than random on intra-heterogeneity but worse on solo status. We suspect this performance is due to group matcher's flexibility in team size and the lack of competing constraints.

When we explore the effect of clustering minorities as a way to avoid tokens in teams in Scenario 3, Figure 8 shows that GRR and priority gain their competitive advantage on solo status and perform just as well as group matcher.

5.2 Real Class Data Comparisons

We ran each algorithm once using real data from two classes as described in Section 4.2. For the first class, we form teams with 175 students and report the performance in Table 1. We did not apply DRR because there are no projects in this scenario. As a result, group matcher formed 38 teams with 2 to 6 students, while the others formed 43 teams with 4 students and 1 team with 3 students. Due to the varying team sizes, we ran four versions of the priority algorithm that all concentrated on timetables and diversified year levels, plus one of the following gender constraints:

- **P4Div:** Priority, teams of 4, diversify women without tokenizing them ($m = 2$)
- **P6Div:** Priority, teams of 6, diversify women without tokenizing them ($m = 2$)
- **P4Conc:** Priority, teams of 4, concentrate women together
- **P6Conc:** Priority, teams of 6, concentrate women together

Generally, we expect good performance with intra-heterogeneity, but doing well in that criterion will compete with solo status. We expect an improvement in solo status by increasing the team size

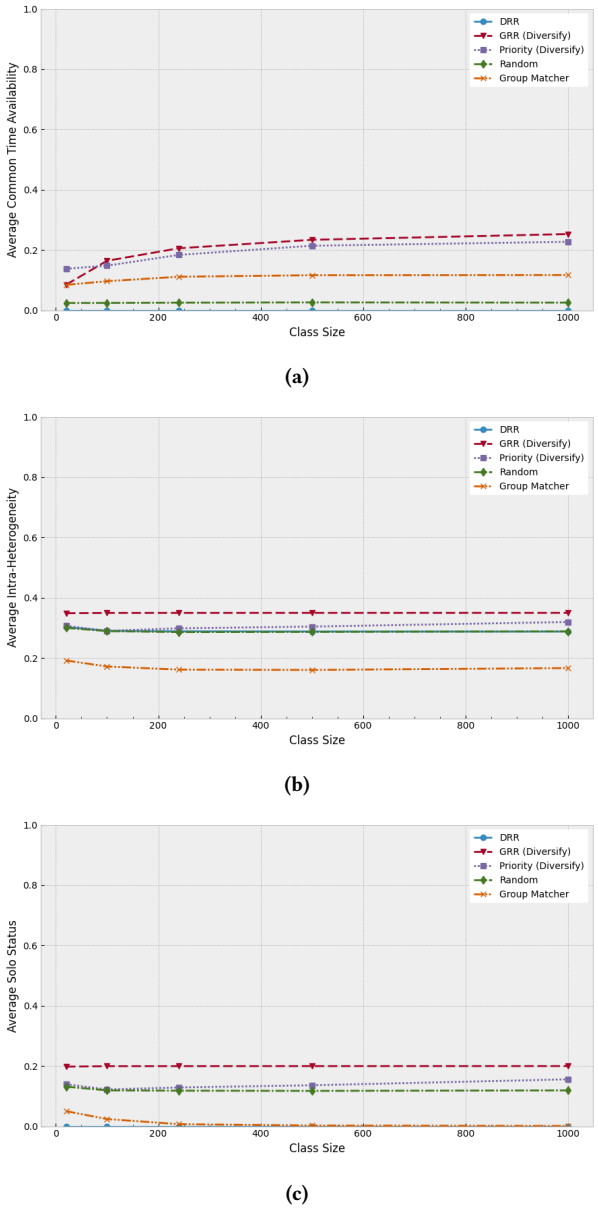

Figure 7: Scenario 2 performance based on: (a) social criterion only, (b) intra-heterogeneity, and (c) solo status.

and further by concentrating on women. Also, we expect poor performance by algorithms that are not designed to meet certain constraints (e.g., DRR does not handle diversity and group matcher does not consider projects). These expectations are reflected in the results in Table 1.

For the second class, we formed teams by matching student skills to project requirements, diversifying grades (GPA), clustering common times, and satisfying social preferences. Group matcher built 9 teams with 3 to 6 students while DRR made 7 teams with 8 to 4 students, and the others formed 9 teams with 4 or 5 students.

Table 1: Results for a scenario that considers common times, diversifies women but avoids tokenizing them, and diversifies graduate students. The best percent is bolded for each metric.

Class 1 Results	Random	Group Matcher	GRR	P4div	P6div	P4conc	P6conc
Common Times	11.4	**25.9**	23.9	23.1	17.8	23.5	18.9
Intra-Heterog. (Gender)	33.3	27.4	**45.8**	34.9	35.1	11.7	11.3
Inter-Homog. (Gender)	29.2	33.6	**14.9**	29.2	22.1	24.0	22.7
Solo Status (Women)	12.57	1.7	22.9	12.57	5.7	1.1	**0.6**
Intra-Heterog. (Year)	**7.9**	0.0	**7.9**	**7.9**	7.8	6.1	7.3
Inter-Homog. (Year)	18.5	**0.0**	18.5	18.5	14.3	17.3	15.3

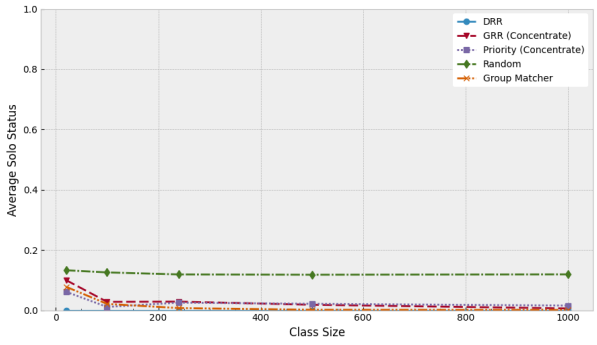

Figure 8: Scenario 3: Performance on intra-heterogeneity and solo status, averaged over 100 trials by concentrating on females and African students.

The results in Table 2 show that all the algorithms perform well on project coverage, although DRR performed the worst because it could not handle the two duplicate projects. Due to the competing constraints, GRR did not do as well as priority on most of the metrics. We also see priority performing well in all respects.

Table 2: Results for a scenario that considers project requirements, diversifies GPA, concentrates time availabilities, and satisfies social preferences. The best percent score for each metric is bolded.

Class 2 Results	Random	DRR	Group Matcher	GRR	Priority (P4Div)
Project Coverage	**97.8**	87.6	**97.8**	**97.8**	**97.8**
Common Times	13.0	4.76	**33.3**	20.4	**33.3**
Intra-Heterog. (GPA)	70.0	61.2	66.3	71.9	**77.4**
Inter-Homog. (GPA)	15.4	14.2	12.3	12.6	**11.7**
Social Satisfaction	0.0	0.0	0.0	11.1	**44.4**

While both GRR and priority can handle multiple types of constraints, priority has the added ability for users to specify the relative importance of each attribute and avoid tokenized minorities.

6 Discussion and Conclusions, and Future Work

Our work is situated within an open-source context that enables a methodological comparison of team formation algorithms using novel diversity metrics. We hope this work contributes to building a more equitable and collaborative learning environment for students. Our next step focuses on optimizing the naive swamping technique in the priority algorithm to generate higher-quality solutions faster. A near-term goal is to evaluate these algorithms in classroom studies to explore their impact on student learning.

6.1 RQ1: Diversity Considerations

We identified several aspects relevant to the design of team formation algorithms: (i) Student representation and distance measures used to compare individuals, (ii) Incorporation of diversity criteria in the algorithm's objective function, (iii) Customizability of diversity parameters to minimize tokenism effects in different pedagogical settings, and (iv) Diversity metrics to evaluate team heterogeneity. We proposed one solution to these diversity considerations in implementing and evaluating the priority algorithm. Further experimentation is needed to investigate how different formulations of these aspects impact team formation.

When forming teams, our priority algorithm can be used to diversify specific learner characteristics while minimizing tokenized minorities. We also saw from Scenario 3 that concentrating on the same learner characteristics serves as an alternative approach to minimizing tokenism in teams. One must bear in mind that the more criteria used to form teams, the more constraints are added to the search problem, making it more difficult to find a good solution. Thus, educators must choose carefully which criteria they want to use and how to prioritize them.

6.2 RQ2: Diversity Metrics

We provided a formal definition for three diversity metrics that assess the heterogeneity of a team, how similarly diverse the teams are in a class, and the presence of single tokens in a team. These metrics facilitate the comparative performance of algorithmic fairness, irrespective of the type of algorithms used. The availability of such metrics has strong implications for how learning technologies could be designed. Educators who use AI-assisted software to generate teams may not fully trust the results and want a way to easily verify AI-generated output. Thus, learning technologies can incorporate a diagnosis phase, as suggested by others as part of a general team formation process [23], where these metrics guide educators on the well-formedness of the teams and identify which teams need to be tweaked. However, the design must avoid expecting users to have high data literacy so the interface is simple and usable by multidisciplinary educators.

6.3 Threats to Validity

The main limitations of our results include using more comparison algorithms and testing on more student distributions. Although science and engineering teams typically focus on women minorities, more diverse pedagogical contexts are needed to continue evaluating the utility of our proposed metrics. A recently published dataset with realistic distributions of skills and communication costs could be used as a benchmark [1]. Our work also did not involve classroom studies that measured the effectiveness of AI-generated teams against other ways of forming teams.

References

[1] B.R. Addanki and B.S. Durga. 2023. Realistic Benchmark Datasets for Team Formation Problem in Social Networks. In *International Conference on Recent Advances in Information Technology (RAIT)*. 1–6.

[2] J. Agarwal, E. Piatt, and P. K. Imbrie. 2022. Team formation in engineering classrooms using multi-criteria optimization with genetic algorithms. In *IEEE Frontiers in Education Conference (FIE)*. 1–6.

[3] I. Amarasinghe, D. Hernández-Leo, and A. Jonsson. 2017. Intelligent group formation in computer supported collaborative learning scripts. In *Proceedings of the IEEE International Conference on Advanced Learning Technologies (ICALT)*. 201–203.

[4] H. Aziz, I. Caragiannis, A. Igarashi, and T. Walsh. 2022. Fair allocation of indivisible goods and chores. *Autonomous Agents and Multi-Agent Systems* 36, 3 (2022), 1–21.

[5] H. Aziz, G. Rauchecker, G. Schryen, and T. Walsh. 2017. Algorithms for Max-Min Share Fair Allocation of Indivisible Chores. In *Proceedings of the AAAI Conference on Artificial Intelligence*. 335–341.

[6] S.G. Baugh and G.B. Graen. 1997. Effects of Team Gender and Racial Composition on Perceptions of Team Performance in Cross-Functional Teams. *Group & Organization Management* 22, 3 (1997), 366–383.

[7] S. Borges, R. Mizoguchi, I.I. Bittencourt, and S. Isotani. 2018. Group Formation in CSCL: A Review of the State of the Art. In *Proceedings of Higher Education for All (HEFA)*. 71–88.

[8] J. Bulmer. 2018. *Team Formation Using Fair Allocation*. Master's thesis. Computer Science, University of British Columbia, Canada.

[9] J. Bulmer, M. Fritter, Y. Gao, and B. Hui. 2020. FASTT: Team Formation Using Fair Division. *Lecture Notes in Computer Science: Advances in Artificial Intelligence* 12109 (2020), 92–104.

[10] C.E. Christodoulopoulos and K.A. Papanikolaou. 2007. A group formation tool in an e-learning context. In *IEEE international conference on tools with artificial intelligence (ICTAI)*. 117–123.

[11] A. Costa, F. Ramos, M. Perkusich, E. Dantas, E. Dilorenzo, F. Chagas, A. Meireles, D. Albuquerque, L. Silva, H. Almeida, and A. Perkusich. 2020. Team Formation in Software Engineering: A Systematic Mapping Study. *IEEE Access* (2020), 145687–145712.

[12] A. Costa, F. Ramos, M. Perkusich, A. de Sousa Neto, L. Silva, F. Cunha, T. Rique, H. Almeida, and A. Perkusich. 2022. A Genetic Algorithm-Based Approach to Support Forming Multiple Scrum Project Teams. *IEEE Access* 10 (2022), 68981–68994.

[13] R. Costaguta. 2015. Algorithms and Machine Learning Techniques in Collaborative Group Formation. In *Proceedings of the Mexican International Conference on Artificial Intelligence (MICAI), LNAI 9414*. 249–258.

[14] W.M. Cruz and S. Isotani. 2014. Group Formation Algorithms in Collaborative Learning Contexts: A Systematic Mapping of the Literature. In *Proceedings of Collaboration and Technology, CRIWG 2014, Lecture Notes in Computer Science*, Vol. 8658, Springer, Cham. 199–214.

[15] A. Darmann, E. Elkind, S. Kurz, J. Lang, J. Schauer, and G. Woeginger. 2012. Group Activity Selection Problem. In *Proceedings of the International Workshop on Internet and Network Economics (WINE)*. 156–169.

[16] P. Dillenbourg. 2002. Over-scripting CSCL: The risks of blending collaborative learning with instructional design. In *Three Worlds of CSCL. Can we support CSCL?* 61–91.

[17] M. Eftekhar, F. Ronaghi, and A. Saberi. 2015. Team Formation Dynamics: A Study Using Online Learning Data. In *Proceedings of the ACM Conference on Online Social Networks (COSN)*. 257–267.

[18] D. Gómez-Zará, L.A. Dechurch, and N.S. Contractor. 2020. A Taxonomy of Team-Assembly Systems: Understanding How People Use Technologies to Form Teams. In *Proceedings of the ACM on Human-Computer Interaction*, Vol. 4. 1–36. Issue 181.

[19] S. Graf and R. Bekele. 2006. Forming Heterogeneous Groups for Intelligent Collaborative Learning Systems with Ant Colony Optimization. In *Proceedings of the International Conference on Intelligent Tutoring Systems (ITS)*. 217–226.

[20] K. Grindstaff and M. Mascarenhas. 2019. 'No One Wants to Believe It': Manifestations of White Privilege in a STEM-Focused College. *Multicultural Perspectives* 21, 2 (2019), 102–111.

[21] J.H. Gutiérrez, C.A. Astudillo, P. Ballesteros-Pérez, D. Mora-Melià, and A. Candia-Véjar. 2016. The multiple team formation problem using sociometry. *Computers & Operations Research* 75 (2016), 150–162.

[22] S.K. Horwitz and I.B. Horwitz. 2007. The Effects of Team Diversity onTeam Outcomes: A Meta-Analytic Review of Team Demography. *Journal of Management* 33, 6 (2007), 987–1015.

[23] B. Hui. 2022. Design Guidelines and Research Directions for Team Analytics. *The International Journal of Information and Learning Technology (IJILT)* 39, 5 (2022), 466–479.

[24] A. Igarashi, D. Peters, and E. Elkind. 2017. Group Activity Selection on Social Networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*. 565–571.

[25] M. Kalantzi, A. Polyzou, and G. Karypis. 2022. FERN: Fair Team Formation for Mutually Beneficial Collaborative Learning. *IEEE Transactions on Learning Technologies* 15, 6 (2022), 757–770.

[26] R.M. Kanter. 1977. Some Effects of Proportions on Group Life: Skewed Sex Ratios and Responses to Token Women. *Amer. J. Sociology* 82, 5 (1977), 965–990.

[27] S. Kohli, N. Ramachandran, and A. Tudor. 2023. Inclusive Study Group Formation At Scale. In *Proceedings of the ACM Special Interest Group on Computer Science Education (SIGCSE)*. 11–17.

[28] J. Konert, D. Burlak, and R. Steinmetz. 2014. The Group Formation Problem: An Algorithmic Approach to Learning Group Formation. In *Proceedings of the European Conference on Technology Enhanced Learning (EC-TEL)*. 221–234. Issue LNCS 8719.

[29] T. Lappas, K. Liu, and E. Terzi. 2009. Finding a team of experts in social networks. In *Proceedings of the ACM SIGKDD international conference on Knowledge discovery and data mining*. 467–476.

[30] X. Li, F. Ouyang, and W. Chen. 2022. Examining the effect of a genetic algorithm-enabled grouping method on collaborative performances, processes, and perceptions. *Journal of Computing in Higher Education* 34 (2022), 790–819.

[31] C. Liang, R. Majumdar, and H. Ogata. 2021. Learning log-based automatic group formation: system design and classroom implementation study. *Research and Practice in Technology Enhanced Learning (RPTEL)* 16, 14 (2021), 1–22.

[32] S.L. Lim and P.J. Bentley. 2019. Diversity Improves Teamwork: Optimising Teams using a Genetic Algorithm. In *Proceedings of the IEEE Congress on Evolutionary Computation (CEC)*. 2848–2855.

[33] C. Lott, A. McAuliffe, and S. Kuttal. 2021. Remote Pair Collaborations of CS Students: Leaving Women Behind?. In *IEEE Symposium on Visual Languages and Human-Centric Computing (VLHCC)*. 1–11.

[34] N. Maqtary, A. Mohsen, and K. Bechkoum. 2019. Group formation techniques in computer-supported collaborative learning: A systematic literature review. *Technology, Knowledge and Learning* 24, 2 (2019), 169–190.

[35] J. Moreno, D.A. Ovalle, and R.M. Vicari. 2012. A genetic algorithm approach for group formation in collaborative learning considering multiple student characteristics. *Computers & Education* 58 (2012), 560–569.

[36] C. Odo, J. Masthoff, and N. Beacham. 2019. Group Formation for Collaborative Learning: A Systematic Literature Review. In *Proceedings of the International Conference on Artificial Intelligence in Education (AIED), LNAI 11626*. 206–212.

[37] M. Ong, C. Wright, L. Espinosa, and G. Orfield. 2011. Inside the Double Bind: A Synthesis of Empirical Research on Undergraduate and Graduate Women of Color in Science, Technology, Engineering, and Mathematics. *Harvard Educational Review* 81, 2 (2011), 172–209.

[38] L.H. Pelled. 1996. Demographic diversity, conflict, and work group outcomes: An intervening process theory. *Organizational Science* 7 (1996), 615–631.

[39] I.M.M. Ramos, D.B. Ramos, B.F. Gadelha, and E.H.T. de Oliveira. 2021. An Approach to Group Formation in Collaborative Learning Using Learning Paths in Learning Management Systems. *IEEE Transactions on Learning Technologies* 14, 5 (2021), 555–567.

[40] E.H. Simpson. 1949. Measurement of Diversity. *Nature* 163 (1949), 688.

[41] E. Spangler, M.A. Gordon, and R.M. Pipkin. 1978. Token Women: An Empirical Test of Kanter's Hypothesis. *Amer. J. Sociology* 84, 1 (1978), 160–170.

[42] Z. Sun and M. Chiarandini. 2021. An Exact Algorithm for Group Formation to Promote Collaborative Learning. In *International Learning Analytics and Knowledge Conference (LAK)*. 546–552.

[43] O.R. Sánchez, C.A.C. Ordóñez, M.Á.R. Duque, and I.I.B.S. Pinto. 2021. Homogeneous Group Formation in Collaborative Learning Scenarios: An Approach Based on Personality Traits and Genetic Algorithms. *IEEE Transactions on Learning Technologies* 14, 4 (2021), 486–499.

[44] G.E. Sæter, V. Stray, S. Almås, and Y. Lindsjørn. 2024. The Role of Team Composition in Agile Software Development Education: A Gendered Perspective. In *Agile Processes in Software Engineering and Extreme Programming (XP)*. 179–194.

[45] R. Terazawa and K. Fujita. 2022. Allocation Considering Agent Importance in Constrained Robust Multi-Team Formation. In *International Conference on Agents and Artificial Intelligence (ICAART)*. 131–138.

[46] M. Thompson and D. Sekaquaptewa. 2002. When Being Different Is Detrimental: Solo Status and the Performance of Women and Racial Minorities. *Analyses of Social Issues and Public Policy* 2, 1 (2002), 183–203.

[47] N. Ugarte, A. Aranzabal, A. Arruarte, and M. Larrañaga. 2022. Using the Behavioural Tendency of Students in a Team Environment for Team Formation. In *IEEE Frontiers in Education Conference (FIE)*. 1–9.

[48] S. Uttamchandani, A. Bhimdiwala, and C.E. Hmelo-Silver. 2020. Finding a place for equity in CSCL: ambitious learning practices as a lever for sustained educational change. *International Journal of Computer-Supported Collaborative Learning* 15 (2020), 373–382.

[49] H. Wang, J. Li, Y. Song, J. Huang, J. Li, and Y. Chen. 2022. An Improved Genetic Algorithm for Team Formation Problem. In *IEEE Symposium Series on Computational Intelligence (SSCI)*. 774–781.

[50] P.Y. Yeh, S.C. Li, H.S. Ma, and J.W. Huang. 2022. Greedy-Based Precise Expansion Algorithm for Customized Group Team Formation Problem. In *2022 International Conference on Technologies and Applications of Artificial Intelligence (TAAI)*. 119–124.