Article

Some Metaheuristics for Multiple Teams Selection Problem

Tung Son Ngo 1,2, \*, Jafreezal Jaafar 1, Izzatdin Abdul Aziz 1, Truong Giang Tong 2, and Ngoc Anh Bui 2

|  |
| --- |
| **Citation:** Lastname, F.; Lastname, F.; Lastname, F. Title. *Informatics* **2021**, *8*, x. https://doi.org/10.3390/xxxxx  Academic Editor: Firstname Lastname  Received: date  Accepted: date  Published: date  **Publisher’s Note:** MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.    **Copyright:** © 2021 by the authors. Submitted for possible open access publication under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/). |

1 Department of Computer and Information Sciences, Universiti Teknologi PETRONAS, 32610 Seri Iskandar, Perak Darul Ridzuan, Malaysia; sonnt69@fe.edu.vn, jafreez@utp.edu.my, izzatdin@utp.edu.my.

2 Information and Communication Department, FPT University,10000 Hanoi, Vietnam; giangtthe153299@fpt.edu.vn, anhbn5@fe.edu.vn.

**\*** Correspondence: sonnt69@fe.edu.vn

**Abstract:** Combinatorial search problems are often found in many different areas. The optimizer needs to scan a large of the available solutions in the search space during the search process. It is usually classified into the class of NP-Hard problems that requires an efficient search algorithm to maintain the quality of the solution with a reasonable computational cost. Team selection is one of the typical issues. This problem becomes more complicated to achieve multiple goals in the decision-making process. This study introduces a multiple cross-functional teams selection model with different skill requirements from candidates who meet the maximum required skills in both deep and wide aspects for the groups. Compromise programming is used to approach the formulated multi-objective optimization problem. We designed metaheuristic algorithms to solve the proposed model, including genetic algorithm (GA) and ant colony optimization (ACO). We also compare the developed algorithms with the MIQP-CPLEX algorithm on datasets of 500 programming contestants on 37 skills to evaluate their effectiveness.

**Keywords:** MOP, Combinatorial Optimization, Compromise Programming, Team Selection, Genetic Algorithm, CPLEX.

1. Introduction

1.1. Background

In management, operational research and other fields, cross-functional team (CFT) selection [1] is an area of ​​interest. Selecting the right teams for the jobs brings success to the organization. A cross-functional team is defined as a group of suitable candidates who have excellent personal skills and can collaborate and support each other in their work. This team's skills are multidisciplinary from many different fields such as biology, math, mechanics, technology, and more. This study is follow-up work from 2020 [2] to develop a methodology for selecting CFTs from available candidates in the organization. The team selection problem is classified into NP-Hard and combinatorial optimization [1, 2,3,4,5]. In a single team selection problem, the solution is represented as where stands for the number of candidates, means that the student is assigned to the team and otherwise. To select a team with members from candidates. The number of available solutions is up to [2].

In practice, the number of groups to be selected is usually more than one, corresponding to different tasks. An increase in the number of candidates or team size or the number of formulated teams significantly increases the search space. In this study, our goal is to choose teams simultaneously to satisfy multiple objectives of business requirements from limited resources. The number of available solutions in the search space is where denotes the team size of the group and . The solution is represented as a graph , where V represents the set of candidates and Groups. Each existing edge in E illustrates the assignment of the corresponding candidate to the group. **Figure 1** shows an example of the CFTs selection problem.



**Figure 1:** Multi-Team Selection Problem with and ;

2.1. Existing Researches

The optimization problems in modern team selection are usually considered to achieve many goals following business requirements. Therefore, it encounters the difficulties of the classical problem and the multi-objective optimization (MOP) problem. The desired goals are often performance, cost, or benefit aspects. The selected members/ teams must cooperate to solve common problems to achieve a specific purpose. Employers need to maximize profits when selecting team members from available candidates [**6**]. Ahmed et al. provide a MOP to choose their cricket team. The model uses objective functions to access three aspects of team performance: batting, bowling, and fielding [**4**]. In [**5**], Chand et al. also use the MOP model to select a cricket team with similar objective functions as [**4**] but with a different goal of minimizing cost. Toledano et al. introduced a bi-objectives optimization to their optimizer, including team valuation and cost [**7**]. Another research also introduced the bi-objectives problem is [**2**]. The model aims to access both aspects of team performance to formulate a new team with many skills and mastering skills. Their approaches are different, while [**7**] uses a dominance-based algorithm to find solutions on praetor-frontier, [**2**] uses Compromise programming to define a compromise solution close to the most referential point. In [**8**], an optimization model is designed to form a team of players for football clubs maximizes the profits from transferring players in degrees. It consists of maximizing the expected net present value of the group, which includes the value of the players owned minus the money spent for buying and borrowing players and paying salaries, plus the income generated by selling and lending players. The problem with these studies is that they were used to select one group. Selecting multiple teams leaves no choice but to resolve the issues repeatedly by eliminating the selected candidates. It leads to the chosen later groups not being treated fairly.

There are many existing approaches to MOP. The most used way for categorizing the methods is by differentiating them into classical forms and evolutionary algorithms. Such classification is mainly based on the working principles for finding Pareto optimal solutions [**9**]. The disadvantage of traditional methods is that it is difficult to find the Pareto frontier. However, when the problem is reduced to a single objective, well-designed resolution techniques can be applied. Evolutionary algorithms (EA) and their version for MOP (MOEA) is often used to solve the MOP and combinatorial optimization, including team selection problem. Ahmed et al. designed an NSGA-II algorithm to search for a Pareto-frontier in 3D space [**4**]. Zhao and Zhang developed some metaheuristic algorithms for team foundations [**10**]. They conclude D-PSO produced better results compare to GA and original PSO in the significant data context. Bello et al. Build an ACO algorithm to select a team based on the preferences of the two decision-makers [**11**]. They perform experiments on small data sets (20 candidates). The MOEA approach yields a set of results that lie on the Pareto frontier. However, it requires higher-level information to trade off the objective functions. In practice, the decision maker's goal is to determine the final solution without this information. This difficulty is a problem in the decision-making process. The optimization with two/three objective functions and the non-dominated solution can be described in a Pareto Front on a two/three-dimensional surface [**12**]. In higher-dimensional space, and the case of decision-maker cannot determine the preferences to the objectives. It is a challenge in the decision-making process.

The resolution techniques for MOP and combinatorial optimization includes mathematical approach and Metaheuristics. In practice, metaheuristics are more suitable for large-scale applications. Chand et al. build an algorithm based on determining the upper and lower bounds of the constraints [**5**]. The solutions are then determined based on the exhaustive method. This prosed algorithm has fewer available solutions to scan than pure exhaust, but it may not be feasible for large-scale problems. Both EA and MOEA have been widely used to solve MOPs because of their advantages by obtaining a set of solutions present in a solution process. They can deal with problems with non-convex Pareto fronts and different types of variables. There are no assumptions of convexity and separability distinction on the objectives and constraints in designing EA algorithms. Besides these advantages, EAs do not ensure the convergence of optimum solutions. They may use a lot of costly function evaluation, which increases computing time. This limitation is particularly crucial when tackling computationally expensive tasks. Therefore, it is vital to design an EA scheme to acquire solutions at accepted computational cost without affecting the quality of the solutions [**13**].

3.1. Contributions

This study presents an optimizer for selecting multiple groups from candidates who match the criteria in skills, which is an improvement from previous research [**2**]. The proposed multi-objective optimization model allows the selection of G groups. Various aspects of team members' skills are set as goals to be achieved by the optimizer. We use the compromise programming (CP) approach for MOP. We propose GA and ACO schemes to solve the proposed model. To evaluate the efficiency of the algorithms, we compare them with CPLEX's MIQP-solver. Our study suggests a new variant of team selection problems. It benefits researchers in the field of management, as well as in empirical research on combinatorial search. This research also contributes to our proposed methodology for Multi objectives scheduling and planning problems [**14**]. The rest of this paper is organized as follows. The proposed model and algorithm are described in Section 2. To evaluate the proposed approach, we display the experiments and discussion in Section 3. Finally, section 4 offers a conclusion.

2. Methodology

2.1. Mathematical Formulation for MOP-TS

There are several variables are used for the model as following:

* is the number of candidates.
* denotes the number of groups.
* represents the number of skills in the skillset.
* stands for the number of members in group .

The decision variables where

* .
* is the rating score for skill of the candidate .
* is the minimum required score for skill for group .

In [**2**], Ngo et al. proposed a two-objective model for selecting their team: "deep" - candidates who are well-versed in the skills they know, and "wide" - the selected candidates to have the number of skills, the more, the better. This is still suitable in the context of choosing a good team. However, we make some adjustments to accommodate the multi-groups selection model.

The objective functions can be defined as follows:

* To select candidates who are fluent in required skills by particular group:
* To select candidates who know many required skills by the group:

Subject to:

* No candidate can join more than one group:

|  |  |
| --- | --- |
|  |  |

* No group is over team size:

|  |  |
| --- | --- |
|  |  |

* Selected groups must respect the minimum requirement on the skills:

|  |  |
| --- | --- |
|  |  |

2.2. Compromise Programming to MOP-TS

The idea of Compromise programming (CP) [**15**] is based on not utilizing any preference information or depending on assumptions about the relevance of objectives. The approach does not strive to discover numerous Pareto optimum solutions. Instead, the distance between a reference point and the feasible objective region is reduced to identify a single optimal solution. There are many researches that have used CP to their MOPs such as university timetabling that including examination timetabling [**16**], teaching task assignment [**17**], student enrollment timetabling [**18**], knowledge based recommender [**19**], project task assignment [**20**]. There are several methods used to select the preferred point or normalize the distance function [**21**]. Ngo et al [**2**]. did not standardize their compromise objective function. This causes deep or wide targets to be biased in distance calculation. The method of selecting a referential point or normalizing the distance function has many variations. The literature suggests that the weighted metrics are used for measuring the distance of any solution from the reference point. The ideal objective vector is often used as the reference point:

Where x is the decision variable and X is feasible set, , can take any value between and (in practice normally ), weight vector , and is the number of objective functions. To normalize the elements of the distance function, we can bring them to range then the objective function in form of norm 2 as: where . It allows to identify the range for weight vector where .

In our case, the and are easy to pre-calculated as:

Where:

and

* is sorted vector of by descending order.
* is sorted vector of by ascending order.

The compromised objective function can be rewritten as:

2.3. Proposed Genetic Algorithm

The Metaheuristic algorithms are often used to solve Combinatorial Optimization and NP-Hard problems. This section describes the design of two evolutionary algorithms [**22**], including the Genetic algorithm (GA) and ant colony optimization (ACO), to solve the proposed model.

2.3.1. Genetic Algorithm

GA is inspired by natural evolution. Its operations including selection, crossover, and mutation. The algorithm begins with a random population, with each individual represents a solution for the problem. The optimal solution is obtained through the adaptation of the new generations. The quality improvement of the solutions is evaluated by their fitness values. The basic flow of GA is shown in **Figure 2**.



**Figure 2:** Basic flow of the GA.

1. Initialize the population: We generate population is the set of indiviuals. The chromosome is represented similar to the decision variables but here we use list as List<List<Integer>> of items each of stores the indexes of selected candidates as **Figure 3**. This mechanism allows to reduce the size of original .



**Figure 3:** Example of the solution in GA with , candidates with id selected for team 1, candidates with id selected for team 2.

1. Fitness function defined for each in as:

Where return 1 if solution violates any constraints in and otherwise.

The violated individuals are punished by removing their effects. In the next genetic operations, these solutions are highly potentially replaced by better quality solutions.

1. Selection: We choose the selection rate of to keep the elite individuals to the next generation.
2. Crossover: denotes the crossover rate where the new individual is constructed as following:

Step 1: Select randomly 2 individuals as parents denoted by

Step 2: Gathers selected members from to list .

Step 3: Randomly select items from to constructs the new solution for the next generation.

**Figure 4** illustrates an example of the three steps of the crossover phase.



**Figure 4:** steps 1 to 3 of the crossoverphase.

1. Mutation: we select the mutation rate to select individuals from the population. These individuals’ genes are modified by randomly selected candidates from the list of candidates.
2. Stop algorithm if after generations, the system cannot find any better solutions. Otherwise, comeback .

2.3.2. Ant Colony Optimization

The ant colony optimization algorithm (ACO) is a technique to solve optimization problems. Using the multi artificial ants can find the right paths on the graphs. The behavior of real ants inspires these ants. They communicate with each other using the pheromone. The basic flow of the ACO is illustrated in **Figure 5**.



**Figure 5:** Basic flow of the ACO.

1. Initialize the cost matrix: we create matrix where represented for the cost if candidate chosen to group as following:

Where: and

1. Initialize the pheromone matrix: we create matrix where
2. Initialize the ant colony: We generate population P is the set of π individuals .
3. Solutions building: in an iteration of ACO each ant choosing points ( Zg point at column g) in Pheromone Matrix based on amount of pheromone at each point and its column as formula below:

Where return 1 if solution violates any constraints in and otherwise.

1. Pheromone actualization: Elite ants: only ants that obtained the best solutions are chosen to update pheromone matrix with rate ϕ with formula:

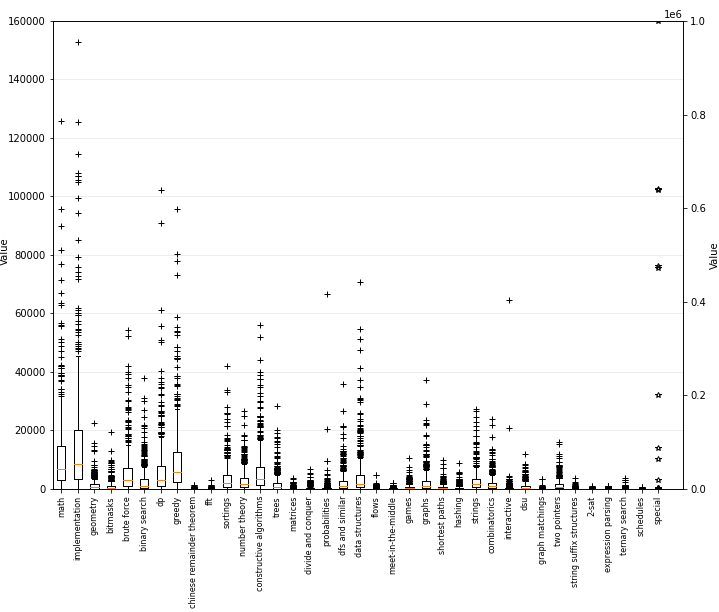
where ant visited point (i,j)

1. Stop algorithm if after generations the best fitness value of the population did not change. Otherwise, comeback .

3. Experiments and Results

3.1. Experimental Design

To evaluate the performance of the proposed algorithms. We use the dataset of 500 programming contestants provided by [2]. Contestants tackle programming exercises, each of which is tagged into different types of exercises in 37 categories. **Figure 6** shows the statistical numbers corresponding to the available skills in the dataset. Each skill has a minimum score of 0, and a max score is not more significant than 160,000 except “special,” which is plotted to the right-vertical axis. It has a maximum score of above 1,000,000, and few candidates got above 160,000 scores in this skill. Some skills have many high-score members like “implementation” with the most extraordinary median is 8354 – it means half of the candidates achieved more than that- and right below is 6834 median score of “math”. In opposition, many skills limit the number of applicants, ex: “2-stat” have not more than 100 candidates scored in this skill before. Moreover, the maximum score of “schedules” is 714, and only 93 of 500 candidates have experience. The ”special” is a particular skill because it has the highest maximum score, and the gap between the best and the worst is huge when not more than 250 candidates have scored in this skill.



**Figure 6:** Statistical numbers on 37 skills in the dataset of 500 programming contestants.

We conduct a quality assessment of the designed algorithm in 2 phases. In the experiment, we set G=1 and took 37 skills as required to compare with the results of the designed GA by Ngo et al. [**2**] (GA-1) and MIQP-CPLEX. In the following experiments, we evaluated these algorithms to select more groups. The assessment involves the quality of the optimal solutions, processing time, and dealing with different decision scenarios. Experiments are performed on computers configured according to **Table 1**.

**Table 1:** System configurations for experiments

|  |  |
| --- | --- |
| **Item** | **Info** |
| CPU | Intel(R) Core (TM) i5-8350U CPU @ 1.70GHz 1.90 GHz |
| RAM | Corsair Vengeance LPX 8GB |
| Programming Platform | Python 3 |
| Operating System | Window 10 |

Metaheuristics algorithms operate according to user customization through parameters. They significantly affect the performance of algorithms. For example, one can bulk order search agents to increase the likelihood of finding a better-quality solution. However, this increases the computational cost. We calibrate the parameter values shown in **Table 2** used in this experiment by re-executing the algorithm several times.

**Table 2:** Parameters to conduct the experiments.

|  |  |  |
| --- | --- | --- |
| **Parameter** | **GA** | **ACO** |
| Population size | 2\* | 0.6\* |
| Crossover rate | 0.9 | None |
| Mutation rate | 0.1 | None |
| Selection rate | 0.1 | 0.1 |
| Stop condition | 40 | 40 |

**Figure 7** shows the query that is used in the experiment. The number of selected groups is 3, along with the required skills required for each group. We use indexes of skills in arrays instead of listing their names. This query is used in sections 3.2.1 and 3.2.2. The heat map illustrates the minimum required scores that need to archive by the particular selected team on the skills. No required skills are displayed in white.

A picture containing timeline

Description automatically generated

**Figure 7:** The query for multiple-teams selection with .

As mentioned in section 2.2, the original objective functions are transformed into a distance function from the actual solution point to the ideal point. It requires the use of  and . The values of their elements are shown in **Table 3**. These values are readily calculated based on the candidate achievement data. We pre-calculated them for different scales of top candidates from the tested dataset. The pair values and of and are respectively represents the best and worst scores of the groups and .

**Table 3:** The ideal points and worst points in different scales of the system.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | | | | | |  | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |
| 500 | 3597600 | 66 | 3454630 | 58 | 1689677 | 25 | 0 | 0 | 0 | 0 | 0 | 0 |
| 300 | 3073288 | 66 | 2930317 | 58 | 1329388 | 25 | 1.83 | 1.83 | 1.83 | 1.83 | 0 | 0 |
| 100 | 2887750 | 66 | 2744861 | 58 | 1165651 | 25 | 11402 | 19 | 9406 | 16 | 1762 | 5.6 |
| 50 | 2617117 | 66 | 2474233 | 58 | 1165651 | 25 | 17511 | 25.45 | 16654 | 22.45 | 3992 | 7.6 |

3.2. Results

3.2.1. Single Team Selection

In [**2**], the author compared GA algorithms (called GA-1), and DCA where their GA-1 showed superior results when selecting three members from the tested dataset. To evaluate the quality of proposed algorithms, we compare our designs to GA-1 in solving the single team selection problem using the same objective function. The results in **Table 4** show that CPLEX could not find a solution from 500 candidates on tested machine. Even when the number of candidates is 300, the objective value obtained by CPLEX was the worst. It is not surprising that GA-1 received a solution quality similar to the proposed algorithms with a better computation time. GA-1 is designed on the principle of excluding retesting of selected members to narrow the search space. It is influential in choosing a single team context. However, this mechanism does not work to select multiple groups at the same time. It trades-off between population diversity and the speed of convergence of the algorithm that is not a solution that satisfies many non-trivial goals and causes the first selected group to take precedence over other groups. Both proposed GA and ACO can find the quality solution as GA-1 for single team selection.

**Table 4:** The results of different algorithms to select a single group that requires 37 skills on the tested dataset.

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** |  | **Objective Value** | **Execution Time (sec)** |
| GA | 500 | 659108 | 10.4 |
| ACO | 500 | 659108 | 50.7 |
| GA-1 | 500 | 659108 | 0.23 |
| MIQP-CPLEX | 500 | - | - |
| GA | 300 | 1005773 | 11.9 |
| ACO | 300 | 1005773 | 20.6 |
| GA-1 | 300 | 1005773 | 0.23 |
| CPLEX | 300 | 1745300 | 3.42 |

3.2.2. Multiple Team Selection

The results of algorithm execution with different scales of the system are shown in **Figure 8**. We can observe that CPLEX produces much worse solutions than ACO and GA algorithms. It cannot handle the size of 300 and 500 candidates due to memory overflow. Note that a fitness value of 0.7 doesn't mean it's only 30% performance. We define ideal values based on the search space for all candidates. If a candidate is excellent, all the groups want this candidate, but the reality is that he is assigned to only one specific group. The z\* score may not be in the search space. The fitness value is only to evaluate which solution is better. ACO and GA are designed to use the same principle for stochastic operations, so they provide similar results even though GAs solutions are slightly better than ACO on the scales of 50 and 300. However, ACO shows a disadvantage when the processing time increases many times with the expansion of the system scale. Even at the small scales of 50 and 100 candidates, its performances are still outstanding. Meanwhile, GA can solve problems with quality solutions without having to compromise much on computational performance. At the 500 candidates scale, the computation time is only slightly more than the 300-candidates scale.

In this experiment, we configured ACO to achieve the same search results as GA. To do this, the processing time of ACO is six times higher compared to GA. With ACO, an ant chooses its path by calculating the probability of each member to be selected for the groups in each iteration. GA does not have to scan the whole candidates for mutation or crossover. This makes ACO takes a longer time to finish an iteration. CPLEX is not capable of handling the size of 300 candidates. Both solution quality and computation time of CPLEX show that it is comprehensively inferior to the proposed algorithms. We only use a single core to executes the algorithms. The executions can be speeded up using parallel mechanism for search agents in both ACO and GA proposed by Ngo et al [**19**].

Chart, line chart

Description automatically generated

**Figure 8:** Obtained results by the algorithms on different system scales.

**Figure 9** shows the value of the fitness functions and the objective function returned through each loop of the search processes. By observing the shape of the two graphs of fitness functions produced by GA and ACO, we find that the convergence of GA is slightly better than that of ACO. Even though ACO's search agents need more time to complete an iteration leads total processing time is longer. GA achieved the optimal solution at the 55th generation. Meanwhile, ACO took 101 iterations to achieve the same result. The objective function values do not always increase even though the fitness function values always decrease through the loops. This phenomenon is due to normalizing the range of the objective functions to [0,1]. The distribution of data affects the reduction of values in the distance-based fitness function. When solutions are projected down to a specific objective function in objective space. If the standard deviation is large, the objective value has a higher impact on the fitness value calculation.

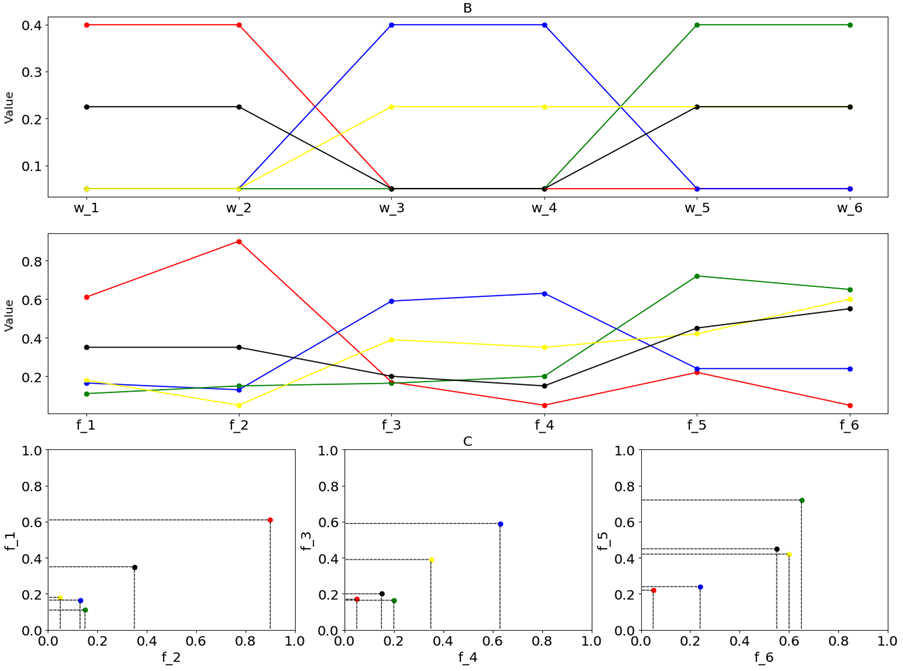
Diagram

Description automatically generated

**Figure 9:** returned values of different functions over iterations/generations on the tested dataset of 500 candidates; A) Fitness values; B) ; C) ; D) ; E) ; F) ; G) ;

3.2.3. Different Decision Scenarios

Approaches to the MOP problem based on the decomposition of multi-objective functions to a ranking function have many advantages. Compromise Programming is more suitable for the decision-maker who cannot indicate the preferences to trade-off the specific goals. In contrast, it is challenging to find optimal solutions on the Pareto frontier as MOEA [**23**]. However, decision-makers use the weight parameters to manipulate the optimizer for different decision scenarios, as shown in **Figure 10**. The combination of compromise programming and evolutionary algorithms allows a seamless transition between model and algorithm design through the use of compromised-objective and fitness functions that are both distance-based functions. We test 5 different sets of weight parameters values, including three priority cases for other teams and the remaining 2 cases with weights not prioritized for a single group, shown in **Figure 10A**. Based on the observed fitness values shown in **Figure 10B** and plotted solutions in the interested objective spaces of the groups in **Figure 10C**, we can see that the user can direct the solver to generate the optimal solutions proportional to the corresponding parameter values. However, it isn't easy to estimate appropriate values for specific scenarios without relying on the experience of decision-makers.

****

**Figure 10:** 5 optimal solutions corresponding to different weight parameters; A) weight parameter values; B) objective values of the obtained solutions; C) the objected solutions projected to interesting objective spaces of the selected groups.

4. Conclusions

In this study, we present an adaptive method to solve the multi-criteria CFTs selection problem. A MOP model was proposed to a new variant of the team selection problem that requires selecting multiple teams among the set candidates. The proposed problem's required search space is much larger than the single team selection introduced by Ngo et al. [**2**]. The non-trivial MOPs always require to trade-off between the objective functions in the decision-making process. In the team selection problems, the decision-makers may not have enough suitable candidates for the teams as expected. It requires involving the higher-level information to assign preferences to each goal. We use the approach of compromise programming to solve this problem. The Solver needs to find the closest solution to the pre-assigned compromise solution instead of solving the original MOP. The designed mathematical optimization models and evolutionary algorithms can be integrated as a system using compromise programming. The compromised-objective function serves as the evaluation function of the search agents. We also designed GA and ACO to solve the proposed model. To evaluate the efficiency of the proposed algorithms, we compare the algorithms with CPLEX-MIQP on the dataset of 500 programming contestants and GA-1 in the context of the single team selection. The results display the ability to find high-quality solutions with a reasonable computational time of the proposed algorithms, especially GA. The optimizers also help the decision-maker in different decision scenarios by driving the search process using weight parameters.

The CP-based approach is beneficial in situations where the decision-maker cannot indicate preferences in the decision-making process. However, it may not be easy to define Pareto Frontier in other cases. In different decision-making strategies, the decision-makers need to use their own experience to choose the values of the parameters. The proposed model does not involve team communications and other soft skills but only technical skills. Therefore, our future research around developing the generic model for multiple CFTs selection and improving the performance of the algorithms.

**Author Contributions:** Conceptualization, N.T.S., T.T.G. and B.N.A.; methodology, N.T.S., J.J. and I.A.A.; software, N.T.S. and T.T.G.; validation, N.T.S. and T.T.G. and B.N.A.; formal analysis, N.T.S. and T.T.G.; investigation, N.T.S. and T.T.G.; resources, N.T.S., T.T.G. and B.N.A.; data curation, N.T.S. and T.T.G. and B.N.A.; writing— N.T.S., X.X.; writing—review and editing, N.T.S., J.J. and I.A.A.; visualization, N.T.S. and T.T.G.; supervision, J.J., I.A.A. and B.N.A.; project administration, N.T.S. and B.N.A.; funding acquisition, N.T.S. and B.N.A..

**Funding:** This research was funded by FPT University, under decision number QĐ1097/QĐ-ĐHFPT and QĐ 1393/QĐ-ĐHFPT for Project HO-CPDT2021.

**Conflicts of Interest:** The authors declare no conflict of interest.

References

1. Bo Feng; Zhong-Zhong Jiang; Zhi-Ping Fan; Na Fu (2010). A method for member selection of cross-functional teams using the individual and collaborative performances. , 203(3), 652–661. doi:10.1016/j.ejor.2009.08.017
2. Ngo, T.S.; Bui, N.A.; Tran, T.T.; Le, P.C.; Bui, D.C.; Nguyen, T.D.; Phan, L.D.; Kieu, Q.T.; Nguyen, B.S.; Tran, S.N. Some Algorithms to Solve a Bi-Objectives Problem for Team Selection. Appl. Sci. 2020, 10, 2700. https://doi.org/10.3390/app10082700.
3. Pantuso G. The Football Team composition problem: a stochastic programming approach. Journal of Quantitative Analysis in Sports. 2017: 13(3):113–129. https://doi.org/10.1515/jqas-2017-0030
4. Ahmed, Faez; Deb, Kalyanmoy; Jindal, Abhilash (2013). Multi-objective optimization and decision making approaches to cricket team selection. Applied Soft Computing, 13(1), 402–414. doi:10.1016/j.asoc.2012.07.031.
5. S. Chand, H. K. Singh and T. Ray, "Team Selection Using Multi-/Many-Objective Optimization with Integer Linear Programming," 2018 IEEE Congress on Evolutionary Computation (CEC), 2018, pp. 1-8, doi: 10.1109/CEC.2018.8477945.
6. Wang, Jian; Zhang, Jia (2015). A win–win team formation problem based on the negotiation. Engineering Applications of Artificial Intelligence, 44(), 137–152. doi:10.1016/j.engappai.2015.06.001.
7. Pérez-Toledano, Miguel Ángel; Rodriguez, Francisco J.; García-Rubio, Javier; Ibañez, Sergio José; Kuo, Yong-Hong (2019). Players’ selection for basketball teams, through Performance Index Rating, using multi objective evolutionary algorithms. PLOS ONE, 14(9), e0221258–. doi:10.1371/journal.pone.0221258.
8. Pantuso G. The Football Team composition problem: a stochastic programming approach. Journal of Quantitative Analysis in Sports. 2017: 13(3):113–129. <https://doi.org/10.1515/jqas-2017-0030>.
9. Ojstersek, R., Brezocnik, M., & Buchmeister, B. (2020). Multi-objective optimization of production scheduling with evolutionary computation: A review. International Journal of Industrial Engineering Computations, 359–376. https://doi.org/10.5267/j.ijiec.2020.1.003.
10. Qian Zhao, Lian-ying Zhang, "Research on the Effect of DPSO in Team Selection Optimization under the Background of Big Data", Complexity, vol. 2018, Article ID 1386407, 14 pages, 2018. https://doi.org/10.1155/2018/1386407.
11. Bello, M., Bello, R., Nowé, A., & García-Lorenzo, M. M. (2017). A Method for the Team Selection Problem Between Two Decision-Makers Using the Ant Colony Optimization. Studies in Fuzziness and Soft Computing, 391–410. doi:10.1007/978-3-319-60207-3\_23.
12. Gunantara, N. (2018). A review of multi-objective optimization: Methods and its applications. Cogent Engineering, 5(1). https://doi.org/10.1080/23311916.2018.1502242.
13. Chugh, T., Sindhya, K., Hakanen, J., & Miettinen, K. (2019). A survey on handling computationally expensive multiobjective optimization problems with evolutionary algorithms. Soft Computing, 23(9), 3137–3166. https://doi.org/10.1007/s00500-017-2965-0.
14. Ngo Tung Son, Jafreezal Jaafar, Izzatdin Abdul Aziz, and Nguyen Hoang Giang. 2021. Introduction to A Compromise Programming Based Method for Complex Scheduling and Planning Problems. In 4th International Conference on Information Management and Management Science (IMMS 2021), August 27–29, 2021, Chengdu, China. ACM, New York, NY, USA, 10 pages. NOTE: This block will be automatically generated when manuscripts are processed after acceptance.
15. J. L. Ringuest, “Compromise Programming,” in Multiobjective Optimization: Behavioral and Computational Considerations, Boston, MA: Springer US, 1992.
16. S. Tung Ngo, J. Jafreezal, G. Hoang Nguyen, and A. Ngoc Bui, “A Genetic Algorithm for Multi-Objective Optimization in Complex Course Timetabling,” in 2021 10th International Conference on Software and Computer Applications, Feb. 2021, pp. 229–237, doi: 10.1145/3457784.3457821.
17. S. T. Ngo, J. B. Jaafar, I. A. Aziz, G. H. Nguyen, and A. N. Bui, “Genetic Algorithm for Solving Multi-Objective Optimization in Examination Timetabling Problem,” Int. J. Emerg. Technol. Learn., vol. 16, no. 11, p. 4, Jun. 2021, doi: 10.3991/ijet.v16i11.21017.
18. S. T. Ngo, J. Jaafar, I. A. Aziz, and B. N. Anh, “A Compromise Programming for Multi-Objective Task Assignment Problem,” Computers, vol. 10, no. 2, Jan. 2021, doi: 10.3390/computers10020015.
19. N. T. Son, J. Jaafar, I. A. Aziz, and B. N. Anh, “Meta-Heuristic Algorithms for Learning Path Recommender at MOOC,” IEEE Access, pp. 1–1, 2021, doi: 10.1109/ACCESS.2021.3072222.
20. Son, N. T., Jaafar, J., Aziz, I. A., Anh, B. N., Binh, H. D. et al. (2021). A Compromise Programming to Task Assignment Problem in Software Development Project. CMC-Computers, Materials & Continua, 69(3), 3429–3444.
21. Ngo Tung Son, Jafreezal Jaafar, Doan Van Thang, Phan Lac Duong, and Bui Ngoc Anh. 2021. The Effectiveness of Reference Point Selection Methods for Compromise Programming in Multi-Criteria Learning Path Search Algorithm. 2021 4th International Conference on Information Management and Management Science (IMMS 2021), August 27–29, 2021, Chengdu, China. ACM, New York, NY, USA, 10 pages. NOTE: This block will be automatically generated when manuscripts are processed after acceptance.
22. Katoch, S., Chauhan, S.S. & Kumar, V. A review on genetic algorithm: past, present, and future. Multimed Tools Appl 80, 8091–8126 (2021). https://doi.org/10.1007/s11042-020-10139-6.
23. M. T. M. Emmerich and A. H. Deutz, “A tutorial on multiobjective optimization: fundamentals and evolutionary methods,” Nat. Comput., vol. 17, no. 3, pp. 585–609, Sep. 2018, doi: 10.1007/s11047-018-9685-y.