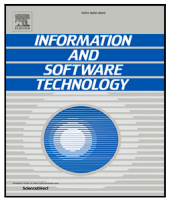
Information and Software Technology 151 (2022) 107023

Contents lists available at ScienceDirect

Information and Software Technology

journal homepage: www.elsevier.com/locate/infsof



A multi-objective agile project planning model and a comparative meta-heuristic approach

Nilay Ozcelikkan [a,b,1], Gulfem Tuzkaya [a,*,1], Cigdem Alabas-Uslu [a,1], Bahar Sennaroglu [a,1]

[a] Department of Industrial Engineering, Marmara University, Istanbul, Turkiye
[b] Softtech Software Technologies Research Development and Marketing Trade Inc., Istanbul, Turkiye

ARTICLE INFO

MSC:
00-01
99-00

Keywords:
Agile software development
Sprint planning
Scrum
Multi-objective model
NSGA-II
SPEA2

ABSTRACT

Agile software development methodologies are used to meet the changing needs in the market. The most popular framework among these methodologies is the Scrum framework. In Scrum planning, the assignment of user stories to sprints requires the consideration of multiple objectives to use the limited resources more effectively. In this paper, a multi-objective mixed-integer programming model is developed which considers three objectives: maximizing the sprint capacity usage, maximizing the assignment of user stories with high priority to primary sprints, and maximizing the assignment of affine user stories to the same sprint. The aim is to contribute to both theory and practice of Scrum planning considering multiple objectives. Additionally, different from the existing literature of Scrum planning, alternative user stories are also taken into account. The proposed model is applied to the small, medium, and big-sized instances of the problem taken from a real-life system. Non-dominated Sorting Genetic Algorithm (NSGA-II) and Strong Pareto Evolutionary Algorithm (SPEA2) are used as heuristic approaches since big-sized instances of the problem could not be solved using optimization approaches. To analyze the performances of these algorithms, Hypervolume (HV), Epsilon (ϵ), Generational Distance (GD), Inverted Generational Distance (IGD), Inverted Generational Distance Plus (IGD+), and Spread (Δ) indicators are used. Results showed that NSGA-II performs better than SPEA2 according to ϵ indicator for big-sized instance. On the other hand, SPEA2 performs better than NSGA-II according to HV, GD, IGD, IGD+, and Δ indicators. However, the results are very close to each other for HV, ϵ, IGD, and IGD+ indicators. In conclusion, both algorithms can be used to deal with the multi-objective Scrum planning problem.

1. Introduction

Companies use various software development models in their Information Technology (IT) Departments. There are seven models of the software development process according to Galin [1]: software development life cycle (SDLC) model, prototyping model, spiral model, object-oriented methodology, incremental delivery model, staged models, and agile methodology models. Since the 1990s, modern agile methods have been evolved. The agile philosophy provides continuous delivery in short periods. In this way, the adoption of agile management leads to high-quality products with fast delivery. Companies that need flexibility in their IT Departments can also apply agile methodology which is an iterative and incremental model. Liker [2] stated that Agile thinking started with Toyota Production System (TPS) which is mostly known as Lean. Then, Rapid Application Development (RAD) stemmed from Tom Gilb's Evolutionary Project Management (EVO) [3] and Barry Boehm's Spiral [4] approaches in the 1990s. Another key agile framework was created in the late 1990s as eXtreme Programming (XP) [5]. In 2001 Agile Manifesto is announced. According to this manifesto, software developers and consultants stated that they give value to [6]:

- Individuals and interactions over processes and tools;
- Working software over comprehensive documentation;
- Customer collaboration over contract negotiation;
- Responding to change over following a plan.

Following the agile manifesto, agile methodologies have become so popular and as stated by Dingsøyr et al. [7] an important number of countries contributed to the research of agile. Of course, it is not easy to transform a company to agile especially if the company is a large-scale one. Dikert et al. [8] proposed a systematic literature review study of industrial large-scale agile transformations. According to their study,

* Corresponding author.
E-mail addresses: nilay.ozcelikkan@gmail.com (N. Ozcelikkan), gulfem.tuzkaya@marmara.edu.tr (G. Tuzkaya), cigdem.uslu@marmara.edu.tr (C. Alabas-Uslu), bsennar@marmara.edu.tr (B. Sennaroglu).
[1] We claim that the authors equally contributed to the article.

https://doi.org/10.1016/j.infsof.2022.107023
Received 19 November 2021; Received in revised form 20 July 2022; Accepted 21 July 2022
Available online 26 July 2022

the most salient success factor categories were management support, choosing and customizing the agile model, training and coaching, and mindset and alignment. One of the difficulties that companies faced is planning while customizing the selected agile model.

Choosing the agile software development method is very important and Dynamic System Development Method, Crystal, Feature Driven Development, Lean and Test Driven Development are some of them. Besides these methods, Scrum is one of the most popular frameworks in agile software development because of its simplicity and high performance [9]. Companies use Scrum to deliver products by completing the projects creatively and productively. According to Scrum Guide which is prepared by Schwaber and Sutherland [10], Scrum is founded on empiricism, so the building blocks of Scrum Theory are transparency, inspection, and adaptation. The Scrum framework consists of Scrum Team, values, events, and artifacts.

Schwaber and Sutherland [10] stated that the heart of the Scrum framework is a sprint which is a fixed time period of about 1 month or less. Sprint planning is a timeboxed Scrum event that initiates the Sprint in Scrum framework. At the beginning of every Sprint, planning is done by the Scrum Team. In sprint planning, the work to be performed through sprint is arranged. The Product Owner is accountable for managing planning events to discuss the most important Product Backlog Items (user stories) to hit the product goal. To help this planning session, in this study, we created a mixed-integer multi-objective mathematical model to assign user stories to sprints considering priority, AND precedence, OR precedence, affinity, alternative usage, the effort of user stories, and capacity of sprints. By solving the proposed model, it is aimed at finding a proper schedule for the project. The solution obtained from the model can be used by Scrum Team to plan the next sprint. Therefore, this model can be used in every Sprint Planning session as the precedence of the user stories can be changed in time. Also, some of the user stories could be completed according to the Definition of Done (DoD) and some of them could not be completed by the Scrum Team at the end of the Sprint, so a new plan is needed for the rest of the user stories which are in the backlog to respond the changes.

The proposed model is used to solve the Scrum planning problem in the IT department of one of the biggest banks in Turkey. The bank decided to use the Scrum framework in its IT department and they needed a systematic approach for the planning of projects in addition to the empirical approach. Therefore, the proposed model in this study is employed to satisfy their requirements to provide a systematic Scrum planning. We applied our multi-objective model to the small and medium-sized instances of the problem taken from this real life application first, and we could solve these instances using optimization approaches. However, the big-sized instances of the problem could not be solved using optimization approach because of the NP-hard structure of the problem [11]. The combinatorial structure of this problem corresponds to the Generalized Assignment Problem (GAP) which is known as an NP-hard problem [12]. Since the GAP is an NP-hard problem even for a single objective function, the multi-objective problem handled in this study is also NP-hard. Therefore, we used heuristic approaches to deal with the big instances.

Heuristic approaches to multi-objective optimization problems are classified, in general, as Pareto dominance-based, indicator-based, and reference-based [13]. Pareto dominance-based approaches treat each objective function separately to find a set of non-dominated solutions. Indicator-based methods permit decision makers' preferences to be implicitly incorporated into the search. These algorithms employ quality indicators to direct the selection process by assigning individuals to an objective function. Reference-based methods, on the other hand, allow decision makers' preferences in the objective space to direct search process. For details, interested readers can be referred to the comprehensive study of Taha [13].

The aforementioned classes have certain advantages and disadvantages over each other. As the Pareto dominance-based category generates high-qualified results for the convergence metric [13], among the approaches in this category, two genetic algorithms that are able to find near-Pareto optimum solutions for multi-objective problems, NSGA-II and SPEA2, were selected to solve big-sized instances of the Scrum planning problem considered in this study. NSGA, which is a multi-objective optimization algorithm (MOOA) developed by Srinivas and Deb in 1994 [14] evolved to NSGA-II in 2002. In Deb et al.'s [15] study, it is explained as an elitist multi-objective evolutionary algorithm, which means that an already found Pareto optimal solution would not be deleted. There are two key concepts in NSGA-II: a fast non-dominated sorting of the population of solutions and a crowding distance calculation mechanism. Also, crowded-comparison approach in this algorithm is used to preserve diversity. According to this algortihm, the user does not need to define any parameter to maintain diversity among population members. SPEA is also a MOOA developed by Zitzler and Thiele in 1999 [16] which is evolved to SPEA2 in 2001. Strength value, raw fitness value, density value, fitness assignment, and environmental selection are major concepts of SPEA2. According to Zitzler et al. [17], SPEA2 has advantages over NSGA-II in higher dimensional objective spaces.

In this study, the research question is "How can we form a project plan for IT departments which use Scrum framework?". For this purpose, it is aimed to develop a multi-objective mathematical model to form a project plan regarding the dependencies and constraints in IT departments that use the Scrum framework. Additionally, we aimed to contribute to both theory and practice of Scrum planning since the studies in the literature of Scrum planning is limited. Only one study prepared by Al-Zubaidi et al. [18] considers bi-objectives in Scrum planning, which are different from the objectives focused on the presented study. Also, the proposed model in this study takes triple objectives into account in addition to the constraint of alternative user stories.

The paper continues as follows. In the second section, previous work on Scrum planning is explained. In the third section, the Scrum concepts that we use in our model are described. The model is described in detail in the fourth section. In the fifth section, NSGA-II and SPEA2 algorithms are explained. In the sixth section, the application is described. Also, the data collection and comparison of results are discussed in this section. In the last section, the concluding remarks are given.

2. Previous work on Scrum planning

In literature, there is a limited number of articles in which mathematical models are proposed and optimization techniques are used to solve the Scrum Planning Problem. Golfarelli et al. [19] presented an article to optimize sprint planning in agile data warehouse design where the single objective is to maximize cumulative utility. In the objective function formulation, early placement of critical stories, which has a strong impact on the other stories, and affinity of stories are taken into account. Golfarelli et al. [20] also proposed a model on multi-sprint planning and smooth replanning. In this article, as in Golfarelli et al. [19], the objective function aims to maximize cumulative utility. Also, the criticality risk is considered in the model in addition to AND/OR precedences and affinity concepts to show relationships of user stories. They used complexity to distinguish complex user stories from easy ones and sprint capacity to obtain the maximum capacity of sprints. Boschetti et al. [21] applied the Lagrangian heuristic for sprint planning in agile software development. Similar to the other two articles, maximization of cumulative utility is used as the single objective function under the consideration of criticality risk of user stories and affinity aspects. Al-Zubaidi et al. [18] studied multi-objective iteration planning in agile software development with two objectives and one constraint. The first objective is to maximize the business value created at the end of each sprint and the second is to maximize collective contribution towards each sprint's goal by selecting proper issues. They

also used priority concept in their fitness function and capacity concept in their constraint.

In addition to mentioned studies, several release planning studies, which are indirectly related to our study, exist in the literature of agile software development. In Scrum, a sprint plan is executed at the beginning of each sprint. If there is only one team that is responsible for one application or project, the releases to form a new version of the software can be done independently. If many teams work on the same application or project, a release plan is needed to avoid confusion in the releases. These release plans can be rescheduled according to the disruptive events during the project. Iqbal and Alam [22] prepared a systematic literature review for the release planning problem. Zhang et al. [23] used meta-heuristics and hyper-heuristics for optimizing software release planning. Escandon-Bailon et al. [24] studied the release plan rescheduling problem and solve it using evolutionary algorithms. Singh et al. [25] studied release planning problem and used a weighted sum approach via the Jaya algorithm. Zapotecas-Martínez et al. [26] proposed a three-objective optimization model for the release plan rescheduling problem. Gademann and Schutten studied capacity planning for projects where the objective function is to minimize the total cost of required nonregular capacity [27]. In their model, important practical issues such as capacity flexibility, precedence relations, and maximum work content per period are taken into account. They used heuristic algorithms to solve the problem. Li et al. [28] studied requirement selection regarding the requirement dependencies in software release planning. Szoke [29] claimed that there is an analogy between agile release scheduling and multiple knapsack problems. In his study, a model to maximize the deliverable value of releases is developed and the branch and bound algorithm is used to solve it. Van Valkenhoef et al. [30] developed an optimization model that generates a release plan taking into account story size, business value, possible precedence relations, themes, and uncertainty in velocity prediction for extreme programming which is one of the methodologies in agile software development.

Scrum framework gives an empirical approach to planning. In this study, we approached the planning process using optimization tools instead of the empirical approach. We developed a mixed-integer multi-objective model to solve instances of Scrum planning problem by assigning user stories to sprints. The objective functions of the model are as follows: Minimizing the unused sprint capacity, maximizing the assignment of affine user stories to the same sprint, and maximizing the assignment of higher priority user stories to prior sprints to minimize the risks of the project. If higher priority user stories are assigned to later sprints, the project may not be completed as desired [19], thus, there may be customer dissatisfaction or even worse customer loss. Considering these potential risks, Golfarelli et al. [19] introduced critical stories and uncertain stories. On the other hand, in Griffiths [31], it is stated that the prioritization is based on the working on stories that results in the highest value to the customer as soon as possible, and thus, it is essential for the team to organize the plan considering budget and timeline. Therefore, in this study, prioritization of the user stories is taken into account by involving the objective function which encourages the assignment of higher priority user stories to early sprints. In real life, usage of alternative user stories can be necessary. Hence, we involved alternative user story constraints into our mathematical model to reflect this real life situation. Among the aforementioned studies, Golfarelli et al. [19], Golfarelli et al. [20], and Boschetti et al. [21] try to assign each user story to a single sprint. However, in our study, user stories, except the alternative ones, must be assigned to a sprint. The maximum number of sprints is given as a parameter to the model and the model computes how many sprints are needed. The model also decides which sprints are necessary to complete the project. Similar to the mentioned articles, coupling groups of user stories such as AND and OR dependencies and affinity between user stories are considered in our study in addition to the effort of user stories in terms of story points and capacities of sprints. Al-Zubaidi et al. [18] focused on the sprint level which is selecting issues for only one sprint, however, we focused on the project level (all sprints). They focused on selecting appropriate user stories considering both priority and complexity to achieve the sprint's goal and to gain more business value. They have only one constraint as the capacity of the team which is also taken into account in our study. Additionally, dependencies between user stories are not taken into consideration by Al-Zubaidi et al. [18]. As a result, the proposed multi-objective model in this study contributes to the literature since it differentiates from the mentioned studies with the consideration of unique objective functions and the alternative user story constraints.

3. Problem definition: Scrum planning

IT departments of companies that arise in different areas adopt the Scrum framework for software product development. Some basic concepts of Scrum are described below. The given definitions are taken from Scrum Guide [10].

User story. User Story or Product Backlog Item (PBI) is a requirement of the product written as a story.

Sprint. The heart of Scrum is a Sprint, which is a fixed period of about one month or less.

Scrum team. The Scrum Team includes one Product Owner, one Scrum Master, and Developers.

Product owner. The Product Owner is responsible for improving the value of the product resulting from the work of the Scrum Team.

Scrum master. The Scrum Master is accountable for applying Scrum according to Scrum Guide.

Developers. Developers commit to creating a usable increment for the product in each Sprint.

Sprint planning. Sprint Planning is a timeboxed Scrum event that initiates the Sprint. The Scrum Team creates the plan. The Product Owner is accountable for directing planning events to discuss the most valuable Product Backlog Items (PBI) to hit the Product Goal. Sprint Planning aims to find answers to the following questions:

- Why is this Sprint valuable? Increasing the product's value and utility makes Sprint more valuable. The Product Owner offers ideas on how to add value to the product. The Scrum Team defines The Sprint Goal through the end of the Planning session.
- Which PBIs will be chosen in this Sprint? Product Owner and Developers choose PBIs to work on the Sprint.
- How will the chosen PBIs get done? For each selected PBI, Developers plan the work to complete them.

Sprint Planning event can be a maximum of eight hours for a one-month Sprint. For shorter Sprints, it can take less time. Planning is needed to complete the project on time within a budget. A poor plan can result in an uncompleted project or a completed project over budget. In this study, we formulated a mixed-integer multi-objective mathematical model for Scrum planning. By using this model, the companies can create better plans to finish the project on time. The model assigns every user story, except the alternative ones, to a sprint until completing all user stories. So, they can forecast the completion time of the project after each planning session. Some basic concepts that we use in our mathematical model presented in the next section are described below.

AND precedence. Some user stories cannot be handled before the completion of some other user stories which are in the AND relation set of this user story. If this relation exists, there are two options. The first one is the assignment of all the user stories into the same sprint. The second one is the assignment of this user story in a sprint after the completion of all other user stories with an AND relation with it.

OR precedence. Some user stories cannot be handled before at least one of the user stories that have OR relation with it is completed. So, if a user story is coupled with other user stories with OR precedence, all of these user stories can be assigned to the same sprint or this user story can be assigned in a sprint after at least one of the other user stories are completed.

Affinity. If user stories are strongly coupled to each other, we call these user stories affine user stories and it will be better if all of them are assigned to the same sprint. There is also a degree of correlation among these user stories. In this study, affinity degree is determined by using a scale between 0.1 and 0.9. 0.1 indicates the least affinity degree and 0.9 indicates the most affinity degree.

Effort. A number assigned to each user story in terms of story points is defined as an effort. In our sample, as recommended by experienced companies, Fibonacci numbers, which is a number sequence, are used for story point designation. The sequence goes like 1, 1, 2, 3, 5, 8, 13, 21, 34, and 55 [32]. Story point is a relative concept dependent on the team members. Story point estimation considers the volume, risk, uncertainty, and complexity of the work.

Sprint capacity. Each sprint has a maximum capacity in terms of story points. Capacity is calculated according to user stories which can be completed in one sprint by the team. For each sprint, maximum sprint capacity may slightly vary due to annual leaves or vacations of the team members, i.e. there is no new team formation for any sprint.

Priority. Each user story has a priority that specifies which user story is needed to be done before others.

Alternative user stories. Each user story must be assigned to a sprint, however depending on the content of the project, the Product Owner allows using alternative user stories. The selection of one of the alternatives is enough for reaching the project goal. At this point, only one of these alternative user stories has to be assigned to a sprint. Managing the campaign process with push notification or SMS, serving a report to an executive manager via a dashboard or e-mail can be examples to alternative user stories.

4. A multi-objective Scrum planning model

Details of the proposed multi-objective Scrum Planning model are given as follows. Nomenclature is presented in Table 1.

The objective functions are formulated and explained as follows:

The first objective function tries to minimize total unused sprint capacity. Here, the difference between available and used sprint capacity is tried to be minimized (Eq. (1)).

$$\text{Min} \sum_{j\in J} c_j M_j - \sum_{i\in I}\sum_{j\in J} z_i X_{ij} \tag{1}$$

The second objective function tries to minimize the total weighted assignments of higher priority user stories to later sprints (Eq. (2)). Here, the weights are priority degrees.

$$\text{Min} \sum_{i\in I}\sum_{j\in J} j p_i X_{ij} \tag{2}$$

The third objective function tries to maximize the assignments of affine user stories to the same sprints (Eq. (3)).

$$\text{Max} \sum_{i\in I}\sum_{j\in J} Y_{ij} \tag{3}$$

The constraints of the model are as follows:

Eq. (4) constraints the model to guarantee that the user story i can be assigned at most one sprint j.

$$\sum_{j\in J} X_{ij} \le 1 \qquad \forall i \in I \tag{4}$$

Table 1
Nomenclature.

Indices	Description
i	Index for user stories $i \in I$
j	Index for sprints $j \in J$
Parameters	**Description**
A_i	The set of user stories that user story i has a dependency type AND
O_i	The set of user stories that user story i has a dependency type OR
B_i	The set of user stories affine to story i
D_i	The set of alternative user stories to story i
N_i	The set of user stories that has no alternative story i
c_j	Maximum available capacity of sprint j in terms of story points
z_i	Effort of user story i in terms of story points
p_i	Priority of user story i
a_{ik}	The degree of correlation between user story i and user story k from story i's affinity set B_i, $a_{ik} \in [0, 1]$
Decision variables	**Description**
X_{ij}	$\begin{cases} 1, \text{if user story is assigned to sprint } j, \\ 0, \text{otherwise} \end{cases}$
Y_{ij}	An accessory variable related to the story i and its affinity set B_i included in sprint j
M_j	$\begin{cases} 1, \text{if sprint } j \text{ is established}, \\ 0, \text{otherwise} \end{cases}$

Eq. (5) constraints the model to guarantee that only one of two alternative user stories is assigned to sprint j.

$$\sum_{l\in D_i}\sum_{j\in J}(X_{lj} + X_{ij}) \le 1 \qquad \forall i \in I \tag{5}$$

Eq. (6) constraints the model to guarantee that each user story i is assigned to a sprint j if it has no alternative.

$$\sum_{j\in J} X_{ij} = 1 \qquad \forall i \in N \tag{6}$$

Eq. (7) constraints the model to guarantee that if a sprint j is not established, a user story i cannot be assigned to that sprint.

$$X_{ij} \le M_j \qquad \forall i \in I, \forall j \in J \tag{7}$$

Eq. (8) constraints the model to guarantee that total story points of assigned user stories to a sprint j cannot exceed its capacity.

$$\sum_{i\in I} z_i X_{ij} \le c_j M_j \qquad \forall j \in J \tag{8}$$

Eq. (9) constraints the model to guarantee that at least one of the user stories in the set of dependency type OR of story i, is assigned in the same sprint or prior sprints of this story.

$$\sum_{l\in O_i}\sum_{k\in J} X_{lk} \ge X_{ij} \quad \forall i \in I^{OR}, \forall j \in J, I^{OR} \subset I \tag{9}$$

Eq. (10) constraints the model to guarantee that each user story in the set of dependency type AND of story i is assigned in the same sprint or prior sprints of this story. Here, $|A_i|$ states the number of elements in set A_i.

$$\sum_{l\in A_i}\sum_{k\in J} X_{lk} \ge X_{ij}|A_i| \quad \forall i \in I^{AND}, \forall j \in J, I^{AND} \subset I \tag{10}$$

Eq. (11) constraints the model to guarantee that the value of accessory variable Y_{ij} cannot exceed the total weighted assignment to that sprint j. Here, story i's affinity set member k's assignment to that sprint j is weighted by affinity degree a_{ik}. a_{ik} can be between 0.1 and 0.9,

Table 2
Assignment.

Sprints	User stories
1	1, 4, 6, 9, 10
2	2, 8
3	3, 5
4	–

inclusively. 0.1 indicates the least affinity degree, 0.9 indicates the most affinity degree.

$$Y_{ij} \leq \sum_{k \in B_i} a_{ik} X_{kj} \qquad \forall i \in I, \forall j \in J \tag{11}$$

Eq. (12) constraints the model to guarantee that if a user story i is not assigned to a certain sprint j, the value of the accessory variable Y_{ij} will be zero regardless of the assignment of some of the members of its affinity set to that sprint j. Here, $|B_i|$ states number of elements in set B_i.

$$Y_{ij} \leq |B_i| X_{ij} \qquad \forall i \in I, \forall j \in J \tag{12}$$

Eq. (13) represents binary variables.

$$X_{ij} \in \{0, 1\} \qquad \forall i \in I, \forall j \in J \tag{13}$$

Eq. (14) represents non-negativity constraints.

$$Y_{ij} \geq 0 \qquad \forall i \in I, \forall j \in J \tag{14}$$

The union of the alternative set and the set without alternatives of user stories is equal to whole user stories (Eq. (15))

$$N_i \cup D_i = I \tag{15}$$

The union of OR and AND sets of user stories is equal to total user stories (Eq. (16)).

$$I^{OR} \cup I^{AND} = I \tag{16}$$

5. Multi-objective algorithms

In the development of heuristic algorithms, solution representation has a significant impact on the solution quality and computational time of the algorithm. In this study, solutions are represented as strings in both NSGA-II and SPEA2. The length of the strings is equal to the maximum number of stories. The position of each cell in a string represents the number of the story. The number in the cell corresponds to the number of the sprint to which the story is assigned. The solution representation, as an example, is string [1 2 3 1 3 1 0 2 1 1] for a 10-user story and 4-sprint instance. Table 2 gives the decoding for the example string. For this example, even 4 sprints are allowed, the solution assigns 10 user stories to 3 sprints out of 4.

Crossover and mutation are significant factors in the performance of the Genetic Algorithm (GA) because they provide population diversity [33]. We used Simulated Binary Crossover (SBX) proposed by Deb and Agrawal [34] for the crossover function, and Polynomial Mutation (PLM) proposed by Deb and Goyal [35] for the mutation function in both algorithms. SBX simulates the single-point crossover on binary strings, and PLM is based on the polynomial distribution [36].

5.1. NSGA-II

In the literature, NSGA-II has been used in various areas such as hybrid hydro-PV power systems [37], compact accelerator-driven neutron sources [38], and multimedia data analysis [39]. NSGA-II has been used for various applications such as benchmarking algorithms in Quantum-Assisted Routing Optimization for Self-Organizing Networks [40], simultaneous determination of optimal capacities of active and reactive power reserve after administration and settlement of active and reactive power markets separately [41] and optimization of pipe node positions [42]. NSGA-II achieved successful results in these applications.

NSGA-II has the advantage of being an elitist multi-objective evolutionary algorithm which means that an already found Pareto optimal solution would not be deleted. It also has a fast non-dominated sorting of the population of solutions and a crowding distance calculation mechanism. Diversity is also preserved with the crowded-comparison approach. Deb et al. explained the major concepts of NSGA-II as follows [15]:

Fast non-dominated sorting. The first solution in the population is taken in a set. Other solutions are compared with the solutions in the set before being added. If a solution dominates any solution in the set, it takes place of that solution in the set. If the solution is not dominated by any other solutions in the set, it is added to the set. When all solutions are evaluated, the set becomes a non-dominated set.

Crowding distance. Crowding Distance gives the estimation of several solutions around a solution. Computations are done according to fitness function and all the results are arranged from the smallest to the largest number. The smallest and the largest numbers are set to infinity, and for the other members, the calculation is done as the sum of the difference between the member and its closest neighbor. This calculation is done for all objective functions. The crowding distance is the sum of individual distance values regarding each objective function. The crowding distance is calculated as below [43]:

n is the number of individuals for each front F_i. First, the distance of every front is assigned to zero as $F_i(d_j) = 0$, where j is jth individual in front F_i.

For each objective function m;

- The individuals of F_i are sorted as $I = sort(F_i, m)$.
- Infinite distance is assigned to boundary values as $I(d_1) = \infty$ and $I(d_n) = \infty$.
- For $k = 2$ to $n - 1$

$$I(d_k) = I(d_k) + \frac{I(k+1).m - I(k-1).m}{f_m^{max} - f_m^{min}} \tag{17}$$

In Eq. (17) $I(k).m$ is the value of the m^{th} objective function of the k^{th} individual in I.

NSGA-II forms a new population using the parent and offspring populations which have the same population size. Then, a non-dominated sorting is applied to classify the entire population. The new population is created and filled with solutions of different non-dominated fronts once. The filling procedure of this new population starts with the best non-dominated front and continues with solutions of the other non-dominated fronts one by one. The rest of the solutions are deleted when the new population reaches the population size. Crowded tournament selection operator is used as a niching strategy to choose the solutions in the least allowed front. One of the main advantages of this algorithm is that no extra niching parameter is required while providing diversity. An already found Pareto-optimal solution is not deleted due to its elitist mechanism [44].

A flowchart for the NSGA-II algorithm is presented in Fig. 1. Detailed information about the NSGA-II algorithm can be found in Deb et al. [15].

5.2. SPEA2

In the literature, SPEA2 has been used for various applications such as Solar Distributed Generations optimization problems [45], constrained multi-objective optimization problems in vehicle passive suspension system [46], multi-objective single-period multi-item inventory problem [47]. Maheta and Dabhi developed an improved SPEA2 multi-objective algorithm with nondominated elitism and generational crossover [48]. Zaenudin and Kistijantoro improved SPEA2

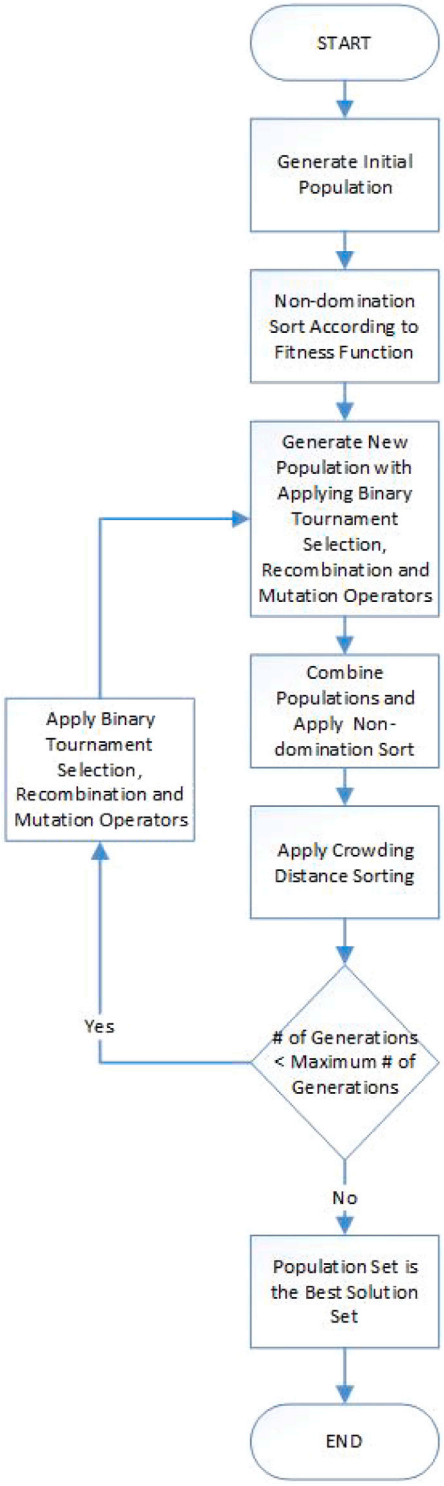


Fig. 1. Flowchart of NSGA-II.

performance to process a population using parallelism in the graphics processing unit (GPU) by optimization fitness and distance calculations [49]. Their results show that NSGA-II generates a better Pareto front in terms of minimizing the objective function but SPEA2 is better in diversification. Zitzler et al. [17] explained the major concepts of SPEA2 as follows:

Strength value. $S(i)$ represents the number of solutions which the solution (string) i dominates. Here, $\succ$ corresponds to the Pareto dominance relation (Eq. (18)).

$$S(i) = \mid \{j \mid j \in P_t + \overline{P}_t \bigwedge i \succ j\} \mid \tag{18}$$

Raw fitness value. Raw fitness value of solution i is calculated by the addition of Strength Values of j that dominates i (Eq. (19)).

$$R(i) = \sum_{j \in P_t + \overline{P}_t, j \succ i} S_j \tag{19}$$

Density value. First, the distances (in objective space) to all individuals in the archive and population are calculated. Then, these distances are sorted in increasing order. Here, σ_i^k is k^{th} element for each individual i after sorting the list in increasing order (Eq. (20)).

$$D(i) = \frac{1}{\sigma_i^k + 2} \tag{20}$$

Fitness assignment. It is calculated by adding Raw Fitness Value and Density Value (Eq. (21)).

$$F(i) = R(i) + D(i) \tag{21}$$

Environmental selection. It is realized by copying all non-dominated solutions of population and archive set to the archive set of the next generation (Eq. (22)).

$$\overline{P}_{t+1} = \{i | i \in P_t + \overline{P}_t \bigwedge F(i) < 1\} \tag{22}$$

If the size of the new archive set exceeds the size of the archive set, truncation operation is applied. If there is still space in the new archive set, the best-dominated solutions of the union of population and archive set are selected to fulfill the archive set.

SPEA2 begins with a randomly created population and an empty archive set. The population size and the size of the archive set must be the same. SPEA2 assigns a strength value $S(i)$ to each member of the both population sets once. Then the raw fitness value $R(i)$ is calculated for each individual. A solution with smaller fitness is better. To preserve elite solutions, archive set is used. Environmental Selection is used to maintain the size of the archive of the elite solutions and to provide diversity among these elite solutions [44].

A flowchart for the SPEA2 algorithm is presented in Fig. 2. Detailed information about the SPEA2 algorithm can be found in Zitzler et al. [17].

6. Application

In the model implementation, Lingo, Eclipse IDE with the version Oxygen.3a, Java SE 1.8 language, and JMetal library are used. The computer used for solution process has Intel(R) Core(TM) i7-7700HQ 2,80 GHz CPU, 16 GB RAM, x64 Windows 10 Pro for Operating System. The proposed multi-objective model is applied to a team's work plan which uses the Scrum framework. This team is a part of the IT department of a private bank. The data infrastructure and problem-solution process are explained in the first and second parts of this section, respectively. The experimental results and analysis of these results are discussed, and the comparison of results according to the algorithms used is explained in the third part.

6.1. Data collection

Real data is gathered from the IT department of a private bank. The model is tested with excerpted small-sized and medium-sized instances from real data. Small-sized problem instance has 10 stories and 4 sprints. A medium-sized problem instance has 60 stories and 10 sprints. 7 small-sized, 1 medium-sized problem instance are used for testing. Real data has 150 stories and 15 sprints. Some of the OR, AND, and affinity dependencies, which are not available in the documents of the bank, are formed based on the expertise and experiences of the team members. Indicator results of small-sized and medium-sized instances and descriptive statistics regarding the dataset are available at https://github.com/NilayOzcelikkan/Data.

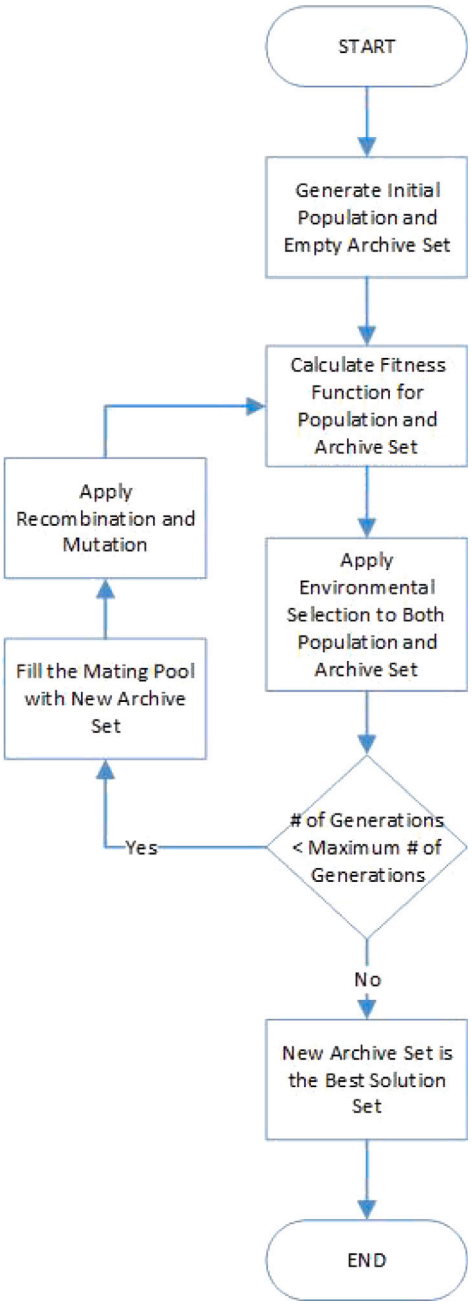


Fig. 2. Flowchart of SPEA2.

6.2. Solution process

Once the Scrum planning problem focused in this study is formulated as a mathematical model and the necessary data is collected to show the applicability of the model, the instances of the problem are solved using the preemptive goal programming approach. In preemptive goal programming, objective functions are ordered considering their priority levels. Then, the problem is solved for the objective function which has the highest priority first by ignoring the remaining objective functions. At the second step, the problem is solved for the second objective function, under the constraint which does not allow the deterioration in the value of the first objective function obtained in the previous step. The procedure continues in this manner until all objective functions are solved considering the constraints of higher-level objective functions. In this study, preemptive goal programming is applied to small and medium-sized problem instances for all possible objective function priority orders. Each step of the preemptive goal programming, that is solving a single objective model under the constraint of higher priority objectives in addition to other constraints which are specific to the problem, is executed using the LINGO solver. Additionally, the best and worst objective values obtained during the execution of the preemptive approach are used to specify the reference points of the performance indicators which are explained in Section 6.3.1.

For the small-sized instance, the compromise solutions can be found in a reasonable time for all possible orders of the multiple objectives. However, the compromise solutions could not be found in a day for the medium-sized instance. It is related to the existence of user story relations like “and, or, affinity” and “alternative user stories” in the medium-sized instance in a more intense way. The solution could not be found for the real data and the program is ended manually. Hence, we decided to use meta-heuristics for the solution procedure. Due to the advantages given in the introduction section, for the defined multi-objective problem, NSGA-II and SPEA2 were applied to the instances of the problem.

Small and medium-sized instances are also used to set the parameters of NSGA-II and SPEA2 and the compromise solutions obtained by the preemptive goal programming are used to validate these algorithms. The minimum and maximum values of objective functions obtained in small and medium-sized instances are used to find bounds for quality indicators of multi-objective algorithms as explained in the next section. Finally, by using the quality indicator values, the parameters setting procedure is completed.

6.3. Experimental study

The experimental study section consists of three subsections as performance indicators, parameter setting, and comparative results. Performance indicators and the approach used in the calculation of the reference points for the required performance indicators are explained in Section 6.3.1. The details of the parameter setting are described in Section 6.3.2. First, small-sized and medium-sized instances are tested using NSGA-II and SPEA2. Then, multivariate tests and nonparametric tests are applied to the instances selected from small and medium-sized instances. According to test results, the best combination of parameters is determined. Finally, in Section 6.3.3, comparative results are reported.

6.3.1. Performance indicators

To compare the performance of multi-objective algorithms, the quality of Pareto-optimal sets is measured in terms of various performance indicators [50]. These indicators create unique values, and in this way, the algorithms can be compared. According to Okabe et al. [50], performance indicators consider mainly three aspects of a solution set:

- The convergence, relative goodness of the solution set;
- The diversity, distribution as well as spread; and
- The number of solutions.

According to Riquelme et al. [51], the Hypervolume (HV) is the most used metric, followed by the Generational Distance (GD), the Epsilon indicator (ϵ), and the Inverted Generational Distance (IGD). To compare the results of NSGA-II and SPEA2 algorithms, HV Indicator, GD metric, Epsilon indicator (ϵ), the IGD, Inverted Generational Distance plus (IGD+) and Spread (Δ) values are used in our study. HV, also known as S metric, hyper-area, or Lebesgue measure, is the volume of an approximation set relative to some reference points [51]. HV takes accuracy, diversity, and cardinality into account during calculations. HV is the only unary metric that considers all of them. A set with a larger hypervolume value is likely to present a better solution than sets with a lower hypervolume value. GD calculates the difference between

Table 3
Algorithm parameters.

Algorithm type	Crossover probability	Max Evaluations/ Max Iterations[a]	Population Size
NSGA-II	0.7, 0.8, 0.9	15 000, 20 000, 25 000	50, 100, 200
SPEA2	0.7, 0.8, 0.9	75, 100, 125, 150, 200, 250, 300, 400, 500	50, 100, 200

[a]MaxIteration is calculated by dividing MaxEvaluations to Population Size.

the point in the approximation set and its closest reference point, then takes the average of these results [51]. GD considers the average Euclidean distance during calculations. GD is also a unary metric and takes accuracy into account. A set with a lower GD value is better than a set with a higher GD value. ϵ is used to find out if one approximation set is dominating another approximation set or not [51]. ϵ uses the worst-case distance during calculations which means one has to improve all points in the approximation set for all objectives to dominate the other approximation set. ϵ is a binary metric. IGD is the inverted version of the generational distance metric [51]. The IGD measure also uses Euclidean distance during calculations however it takes the minimum result instead of the average of the results distinct from the GD. IGD is a unary metric that can measure the diversity and the convergence together with sufficient Pareto front or approximation set data. IGD+ is a variant of the IGD where convergence of the IGD+ to Pareto front is better compared to IGD [52]. In this way, IGD+ overcomes IGD's salient drawbacks. Small IGD and IGD+ values are preferable [52]. Δ is a unary metric that measures the diversity of the approximation set [51]. A set with a larger spread value is likely to present a better solution than sets with a lower spread value.

The quality indicators (QI), which are used to compare multi-objective heuristic algorithms can be classified considering the Pareto-compliant concept as follows: Suppose that there are two non-dominated set A and B obtained from two different multi-objective heuristics, a Pareto compliant QI must satisfy that if A dominates B, QI of A must be strictly better than that of B. If A dominates B and QI of A is the same or better than QI of B, then the QI is called weakly Pareto-compliant [53]. Based on the related literature [54–56], HV is classified as Pareto-compliant, IGD+ is weakly Pareto-compliant and the others (GD, ϵ, IGD, and Δ) are not Pareto-compliant quality indicators.

6.3.2. Parameter setting

The parameter settings of NSGA-II and SPEA2 are determined on the small and medium-sized instances and then applied to the real data. In both algorithms, crossover distribution index and mutation distribution index parameters are used as constant value 20. Mutation probability is calculated according to the size of the problem as 1/(number of stories). An experimental study is performed to designate the best combination of the parameters used in NSGA-II and SPEA2. The parameters are shown in Table 3. For both algorithms, for crossover probability, 0.7, 0.8, and 0.9 are applied to find out which one performs better for each algorithm for each problem size. Max Evaluations and Population Size are both significant parameters for GA to get satisfactory solutions. 15 000, 20 000, and 25 000 for the Max Evaluations parameter, and 50, 100, and 200 values for the Population Size parameter are applied. For SPEA2, the Max Evaluations parameter is converted to Max Iterations to make a fair comparison. Each combination was run 31 times.

Firstly, seven small-sized instances are tested. In these tests, the results are calculated with changing user story values, the priority of user stories, the capacity of sprints, and the dependency of user stories. These results can be seen in Table 4.

One of the small-sized instances (7th problem in Table 4) was used to analyze parameters. We used Pillai's Trace, Wilks' Lambda, Hotelling's Trace, Roy's Largest Root for multivariate tests, and K

Table 4
Indicator means of small-sized problems.

Prb. No	Algorithm type	HV	ϵ	GD	IGD	IGD+	Δ
1	NSGA-II	.886	.103	.013	.260	.066	1.202
	SPEA2	.907	.095	.010	.260	.059	1.205
2	NSGA-II	.931	.060	.009	.842	.032	1.015
	SPEA2	.945	.048	.007	.841	.026	1.021
3	NSGA-II	.893	.106	.065	.441	.067	1.299
	SPEA2	.905	.107	.062	.429	.063	1.346
4	NSGA-II	.903	.111	.149	.704	.062	1.125
	SPEA2	.910	.112	.154	.698	.061	1.133
5	NSGA-II	.898	.122	.120	.504	.050	1.269
	SPEA2	.906	.117	.114	.501	.046	1.279
6	NSGA-II	.921	.097	.093	.583	.056	1.198
	SPEA2	.934	.086	.087	.578	.048	1.215
7	NSGA-II	.941	.059	.079	.624	.038	1.219
	SPEA2	.958	.041	.085	.596	.026	1.274

Table 5
Multivariate tests for NSGA-II for small-sized instance.

Parameter	Test type	*p*-value
Crossover	Pillai's Trace	.107
	Wilks' Lambda	.107
	Hotelling's Trace	.108
	Roy's Largest Root	.061
Max Evaluations	Pillai's Trace	.201
	Wilks' Lambda	.201
	Hotelling's Trace	.202
	Roy's Largest Root	.080
Population Size	Pillai's Trace	.000
	Wilks' Lambda	.000
	Hotelling's Trace	.000
	Roy's Largest Root	.000

Table 6
Multivariate tests for SPEA2 for small-sized instance.

Parameter	Test type	*p*-value
Crossover	Pillai's Trace	.274
	Wilks' Lambda	.274
	Hotelling's Trace	.274
	Roy's Largest Root	.082
Max Evaluations	Pillai's Trace	.081
	Wilks' Lambda	.081
	Hotelling's Trace	.081
	Roy's Largest Root	.055
Population Size	Pillai's Trace	.000
	Wilks' Lambda	.000
	Hotelling's Trace	.000
	Roy's Largest Root	.000

Independent Samples Median Test for the nonparametric tests. The significance level was set to 0.05 and it was concluded that the parameter effect was significant if the *p*-value was less than or equal to 0.05.

For the small-sized instance, multivariate test results for parameters, applied test types and *p*-values for NSGA-II and SPEA2 are shown in Tables 5 and 6, respectively. According to multivariate test results, it is concluded that the only significant parameter is Population Size for both algorithms on the small-sized instance since *p*-values are less than the significant level only for Population Size.

Following the multivariate tests, nonparametric tests are applied to NSGA-II and SPEA2 for all parameters. The null hypotheses used are as follows:

- The medians of HV are the same across categories of the parameter,
- The medians of ϵ are the same across categories of the parameter,

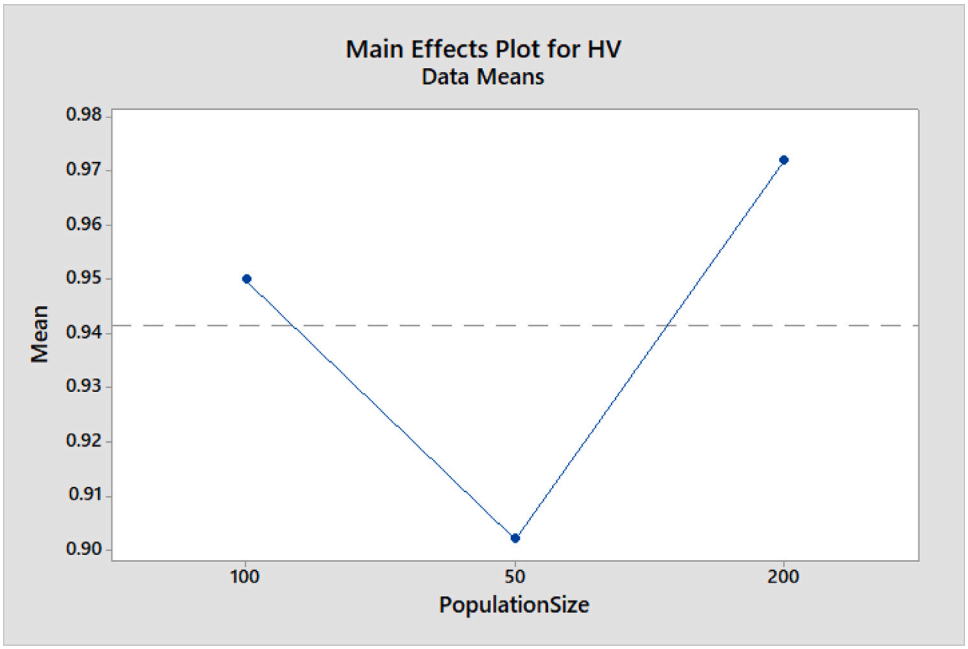


Fig. 3. Main effect graph of population size for HV for NSGA-II.

Table 7
Nonparametric test results for small-sized instance.

Algorithm	Parameter	HV	ϵ	GD	IGD	IGD+	Δ
NSGA-II	PopulationSize	✓	✓	✓	×	✓	✓
	Max Evaluations	✓	×	×	×	×	×
	Crossover	×	×	✓	×	×	×
SPEA2	PopulationSize	✓	✓	✓	×	✓	✓
	Max Iterations	×	×	×	×	×	×
	Crossover	×	×	×	×	×	×

Table 8
Algorithm parameters.

Algorithm type	Crossover probability	Max Evaluations/ Max Iterations[a]	Population Size
NSGA-II	0.9	25000	200
SPEA2	0.9	125	200

[a]MaxIterations is calculated by dividing MaxEvaluations to PopulationSize.

- The medians of GD are the same across categories of the parameter,
- The medians of IGD are the same across categories of the parameter,
- The medians of IGD+ are the same across categories of the parameter,
- The medians of Δ are the same across categories of the parameter.

The results of these nonparametric tests for the small-sized instance are shown in Table 7. The symbol check (✓) represents that the related null hypothesis is rejected, whereas, the symbol cross (×) shows that the null hypothesis is not rejected.

Main effect graphs are used to distinguish which Population Size value is better according to HV, ϵ, GD, IGD+, and Δ indicators. All results with main effect graphs are stored on https://github.com/NilayOzcelikkan/Data page. The main effect graph of Population Size for the HV indicator for NSGA-II can be seen in Fig. 3. The main effect graph of Population Size for the HV indicator for SPEA2 can be seen in Fig. 4. The main effect graphs indicate that Population Size 200 has better results than 50 and 100 according to HV, ϵ, GD, IGD+, and Δ indicators for both algorithms.

All of the statistical tests on the small-sized instance explained above are also applied to the medium-sized instance and the results obtained from the statistical analyses are available at https://github.com/NilayOzcelikkan/Data. Finally, according to the results obtained from both small and medium-sized instances, the set of the parameters selected is given in Table 8.

6.3.3. Comparative results

NSGA-II and SPEA2 are run for the seven small-sized instances using the parameter sets provided in Table 8. The results obtained by the runs are given in Table 9. Bold values indicate which algorithm generates the best result for the given instance. According to indicator results in Table 9, for five instances, SPEA2 has better performance than NSGA-II. Therefore, it can be concluded that for small-sized problems, SPEA2 gives better results than NSGA-II.

In Tables 10 and 11, decoding of the best solutions obtained by NSGA-II and SPEA2 according to HV indicator are given as examples, respectively.

After the analysis of small-sized instances, we applied the two heuristics to the medium-sized instance using the selected parameters. For the medium-sized problem instance, average values of indicator results are shown in Table 12. According to Table 12, as the number of indicators showing better performance is the same for NSGA-II and SPEA2, it can be concluded that no algorithm is better than the other for the medium-sized problem instance.

The solutions of medium-sized instance are shown in Tables 13 and 14, respectively.

As a result, the algorithms are applied with the selected parameters to our real data and the average values of indicator results are shown in Table 15. According to Table 15, the number of indicators showing better performance for SPEA2 is more than the number of indicators for NSGA-II. However, the results are very close to each other for HV, ϵ, IGD, and IGD+ indicators.

For comparison using the best HV indicator value, NSGA-II solution with OF1 = 4691.0 OF2 = 3.3999 OF3 = 175.0 is shown in Table 16.

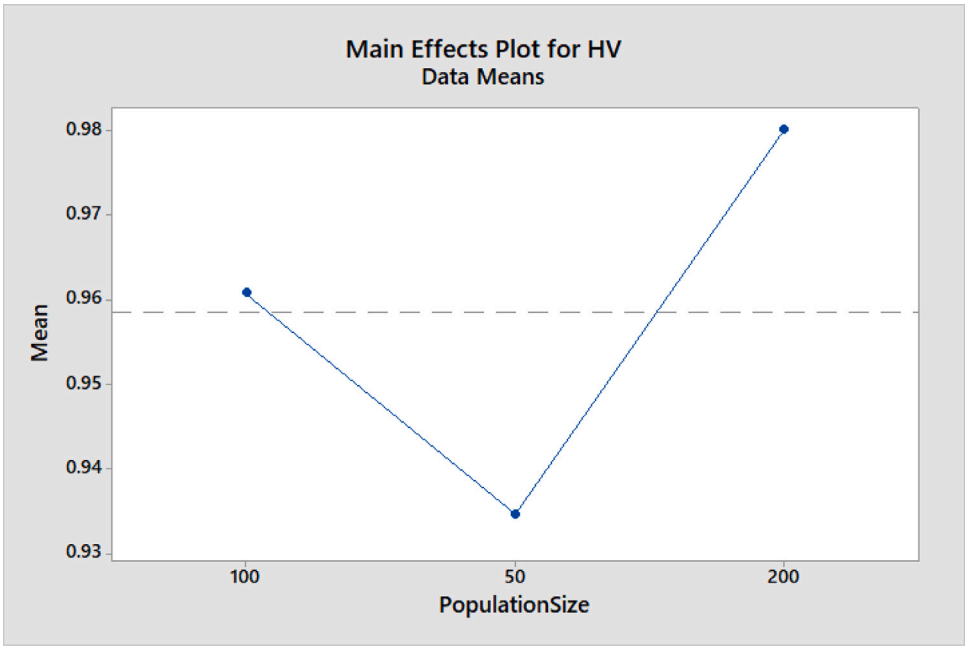


Fig. 4. Main effect graph of population size for HV for SPEA2.

Table 9
Indicator results of small-sized problems.

Prb. No	Algorithm type	HV	ϵ	GD	IGD	IGD+	Δ
1	NSGA-II	.949	**.066**	.005	**.265**	.035	1.191
	SPEA2	**.959**	**.066**	**.002**	.269	**.033**	**1.202**
2	NSGA-II	**.988**	**.012**	**.001**	.863	**.006**	1.000
	SPEA2	.987	.013	**.001**	**.859**	.007	**1.005**
3	NSGA-II	.954	.060	.047	.415	.033	1.381
	SPEA2	**.967**	**.058**	**.041**	**.408**	**.028**	**1.396**
4	NSGA-II	.955	.065	**.104**	.727	.031	1.137
	SPEA2	**.988**	**.032**	.106	**.678**	**.011**	**1.164**
5	NSGA-II	.965	.082	.084	**.497**	.022	**1.380**
	SPEA2	**.974**	**.069**	**.077**	.514	**.016**	1.361
6	NSGA-II	**.980**	**.058**	.067	**.557**	**.023**	1.239
	SPEA2	.964	.067	**.065**	.566	.031	**1.240**
7	NSGA-II	.976	.024	**.052**	.585	.014	1.262
	SPEA2	**.985**	**.014**	.058	**.547**	**.007**	**1.349**

Bold value indicates the best result.

Table 10
Solution to the instance 7 found by NSGA-II according to best HV result.

Sprints	User stories
1	3, 5, 9, 10
2	1, 4, 6
3	2, 8

Table 11
Solution to the instance 7 found by SPEA2 according to best HV result.

Sprints	User stories
1	1, 5, 6, 9, 10
2	2, 3
3	4, 8

Table 12
Indicator results of medium-sized problem instance.

Algorithm type	HV	ϵ	GD	IGD	IGD+	Δ
NSGA-II	**.787**	**.146**	.137	.420	**.103**	1.055
SPEA2	.684	.219	**.030**	**.398**	.163	**1.353**

Bold value indicates the best result.

Table 13
Solution of medium-sized instance obtained by NSGA-II according to best HV result.

Sprints	User stories
1	10, 16, 18, 26, 36, 37, 41, 47, 53, 56, 57, 61
2	2, 19, 21, 24, 25, 30, 35, 44, 52, 60
3	6, 11,17, 20, 28, 31, 51, 54
4	7, 12, 15, 39, 40, 42, 43, 49
5	4, 5, 9, 29, 45
6	3, 8, 48

Table 14
Solution of medium-sized instance obtained by SPEA2 according to best HV result.

Sprints	User stories
1	10, 14, 18, 19, 24, 35, 40, 41, 44, 47, 52, 58, 59, 61
2	4, 6, 7, 13, 20, 28, 31, 33, 39, 57, 60
3	11, 25, 30, 36, 37, 53, 54
4	15, 21, 45, 46, 51
5	5, 8, 26, 43, 49
6	9, 12, 27, 48
7	3

Table 15
Indicator results of real data.

Algorithm type	HV	ϵ	GD	IGD	IGD+	Δ
NSGA-II	.044	**.814**	.436	.637	.699	0.945
SPEA2	**.047**	.820	**.049**	**.595**	**.669**	**1.251**

Bold value indicates the best result.

Each user story is assigned to a sprint except for some of the stories which have alternatives as seen in Table 16. For example, the first user story is assigned to sprint two and the third user story is not assigned

Table 16
Solution of real data obtained by NSGA-II according to best HV result.

Sprints	User stories
1	4, 16, 17, 21, 35, 40, 46, 62, 65, 83, 106, 108, 109, 119, 130, 134, 135, 146
2	1, 10, 13, 27, 32, 81, 82, 90, 96, 104, 123, 133, 140, 142
3	34, 36, 50, 55, 57, 59, 61, 87, 105, 111, 120, 121
4	14, 15, 23, 41, 44, 49, 86, 103, 110, 122, 126, 127, 136, 139, 143
5	7, 8, 26, 52, 66, 97, 100, 113, 115, 116, 137, 145
6	9, 24, 68, 114, 128, 131
7	19, 38, 39, 48, 56, 73, 95, 124, 125, 129, 141
8	2, 12, 47, 51, 64, 67, 85, 88, 98, 107, 138
9	5, 31, 45, 69, 75, 78, 144, 148
10	11, 28, 29, 42, 43, 76, 77, 84, 101, 102, 118, 150
11	18, 30, 53, 70, 74, 94
12	33, 58, 71, 72, 80, 149
13	6, 20, 22, 60, 63, 92, 93, 99, 117
14	54, 79, 91, 112, 132, 147

Table 17
Solution of real data obtained by SPEA2 according to best HV result.

Sprints	User stories
1	3, 21, 27, 36, 42, 48, 55, 61, 79, 82, 84, 90, 99, 134, 135, 141
2	7, 10, 13, 15, 41, 50, 88, 101, 109, 111, 130, 139
3	1, 26, 37, 43, 44, 46, 59, 70, 81, 83, 103, 144, 145
4	16, 33, 40, 52, 62, 63, 98, 105, 112, 126, 136, 138, 140, 150
5	6, 18, 23, 24, 29, 39, 65, 104, 107, 110, 117, 121, 123, 124, 128, 137
6	14, 51, 74, 87, 93, 95, 142, 147
7	22, 35, 49, 56, 57, 73, 96, 100, 113, 116, 143, 148
8	8, 11, 25, 28, 38, 68, 75, 97, 122, 132
9	4, 30, 32, 53, 58, 64, 89, 91, 125, 127, 146
10	9, 54, 60, 72, 80, 94, 131
11	5, 19, 47, 71, 76, 106, 133
12	2, 108, 114, 118, 120
13	17, 31, 34, 66, 67, 69, 77, 78, 102, 115, 119
14	12, 86, 92, 149

to any sprint because it has an alternative story as 129. Similarly, user story 25 is not assigned because it is an alternative story to user story 20.

According to best HV result, one SPEA2 solution with OF1 = 4692.0 OF2 = 0.0 OF3 = 151.0 is shown in Table 17. In this solution, user story 3 is assigned to the first sprint, and user story 129 is not assigned because it is an alternative story to user story 3. Similarly, user story 20 is not assigned because it is an alternative to user story 25. Also, user stories 45, 85 are not assigned. As the NSGA-II result, 14 sprints are used.

Both heuristics produced feasible results and we get totally 31 feasible solutions from each heuristic in less than 10 min. When we analyze the solutions, we saw that both heuristics planned the project in 14 sprints. Also, some of the indicator results are very similar for NSGA-II and SPEA2. As a result, to find a real solution to this company's problem, both algorithms can be used.

The quality indicators reported in the previous tables are computed using the reference set which consists of non-dominated solutions obtained by the preemptive goal programming. For the real data, additionally, NSGA-II and SPEA2 were re-run and the quality indicators were re-computed using all non-dominated solutions obtained by all replications of the two heuristics as a new reference set to see the effect of the reference set. The new results given in Table 18 show that SPEA2 performs better than NSGA-II for most of the quality indicators. However, the two heuristics still exhibit very close indicator results, except GD and Spread, and therefore, both NSGA-II and SPEA2 can be suggested for solving of Scrum planning problem handled in this study.

Table 18
Indicator results of real data using all non-dominated solutions as reference set.

Algorithm type	HV	ϵ	GD	IGD	IGD+	Δ
NSGA-II	**.000**	1.216	.488	.256	1.260	0.959
SPEA2	**.000**	**1.186**	**.059**	**.249**	**1.224**	**1.220**

Bold value indicates the best result.

7. Conclusion

To be successful in the projects, companies need to choose a convenient software development methodology and prepare proper plans. To meet the market's changing needs, a significant number of companies use agile software development methodologies. The most popular framework for agile software development is the Scrum framework. In Scrum, before starting the project, the companies can prepare plans with the assignment of user stories to sprints. This study proposes a multi-objective mixed-integer programming model for Scrum planning. The proposed model assists companies during the Scrum planning process and provides a project plan regarding the dependencies and constraints specific to the problem. It also tends to minimize the project's risks, such as customer dissatisfaction, project cancellation, and customer loss. The model is verified on small and medium-sized instances of the problem scaled from a real case. The multi-objective model is first tried to be solved using the preemptive goal programming approach. Since for the large-sized instance, it becomes impossible to find a solution using the preemptive goal programming approach, we used meta-heuristics for the solution procedure. For the large-sized instance, we used NSGA-II and SPEA2 to find approximated Pareto fronts. These metaheuristics have the advantage of creating more efficient solutions than goal programming because goal programming can only give a unique solution for each different priority order of the three objectives. We compared the results to determine which algorithm is more suitable for the Scrum planning problem. According to the performance indicator results, both algorithms can be used for the problem. Therefore, we can suggest using the multi-objective model and the preemptive goal programming approach to deal with the small-sized instances of the problem. On the other side, the proposed model and NSGA-II and SPEA2 can be employed for larger problem instances. Adding a user-friendly frontend to the proposed multi-objective model can turn it into a product in the market.

This study deals with the feasible assignments of user stories to sprints considering multiple objectives assuming sprint numbers and affinity values are given parameters. In future studies, instead of taking these parameters from the user, they can be tuned through the algorithm's execution. Additionally, different approaches to multi-objective problems, such as decomposition-based or indicator-based, are planning to apply to the multi-objective Scrum planning problem to investigate their effects on solving this problem. Also, the proposed Scrum model can be modified to be applicable for different areas such as the Kanban environment.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The data that has been used is confidential.

References

[1] D. Galin, From SDLC to agile – Processes and quality assurance activities, in: Software Quality: Concepts and Practice, 2018, pp. 635–666, http://dx.doi.org/10.1002/9781119134527.app4.

[2] J.K. Liker, The Toyota Way: 14 Management Principles from the World's Greatest Manufacturer, McGraw-Hill, New York, 2004.

[3] T. Gilb, Evolutionary development, ACM SIGSOFT Softw. Eng. Notes 6 (2) (1981) 17, http://dx.doi.org/10.1145/1010865.1010868.

[4] B.W. Boehm, A spiral model of software development and enhancement, Computer 21 (5) (1988) 61–72, http://dx.doi.org/10.1109/2.59.

[5] K. Beck, Extreme Programming Explained: Embrace Change, Addison-Wesley Professional, 2000.

[6] K. Beck, M. Beedle, A. van Bennekum, A. Cockburn, W. Cunningham, M. Fowler, J. Grenning, J. Highsmith, A. Hunt, R. Jeffries, J. Kern, B. Marick, R.C. Martin, S. Mellor, K. Schwaber, J. Sutherland, D. Thomas, Agile manifesto, 2001, URL http://agilemanifesto.org/.

[7] T. Dingsoyr, S. Nerur, V. Balijepally, N.B. Moe, A decade of agile methodologies: Towards explaining agile software development, J. Syst. Softw. 85 (6) (2012) 1213–1221, http://dx.doi.org/10.1016/j.jss.2012.02.033.

[8] K. Dikert, M. Paasivaara, C. Lassenius, Challenges and success factors for large-scale agile transformations: A systematic literature review, J. Syst. Softw. 119 (2016) 87–108, http://dx.doi.org/10.1016/j.jss.2016.06.013.

[9] E. Johnson, Why is scrum so popular? Why is scrum so successful? 2021, URL https://content.intland.com/blog/agile/scrum/why-is-scrum-so-popular-why-is-scrum-so-successful.

[10] K. Schwaber, J. Sutherland, The 2020 scrum guide, 2020, URL https://scrumguides.org/scrum-guide.html.

[11] R.M. Nauss, Solving the generalized assignment problem: An optimizing and heuristic approach, INFORMS J. Comput. 15 (3) (2003) 249–266, http://dx.doi.org/10.1287/ijoc.15.3.249.16075.

[12] M. Savelsbergh, A branch-and-price algorithm for the generalized assignment problem, Oper. Res. 45 (6) (1997) 831–841, http://dx.doi.org/10.1287/opre.45.6.831.

[13] K. Taha, Methods that optimize multi-objective problems: A survey and experimental evaluation, IEEE Access 8 (2020) 80855–80878, http://dx.doi.org/10.1109/ACCESS.2020.2989219.

[14] N. Srinivas, K. Deb, Multi-objective function optimization using non-dominated sorting genetic algorithms, Evol. Comput. 2 (3) (1994) 221–248, http://dx.doi.org/10.1162/evco.1994.2.3.221.

[15] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multi-objective genetic algorithm: NSGA-II, IEEE Trans. Evol. Comput. 6 (2) (2002) 182–197, http://dx.doi.org/10.1109/4235.996017.

[16] E. Zitzler, L. Thiele, Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach, IEEE Trans. Evol. Comput. 3 (4) (1999) 257–271, http://dx.doi.org/10.1109/4235.797969.

[17] E. Zitzler, M. Laumanns, L. Thiele, SPEA2: Improving the Strength Pareto Evolutionary Algorithm, TIK-Report 103, 2001, http://dx.doi.org/10.3929/ethz-a-004284029.

[18] W.H.A. Al-Zubaidi, H.K. Dam, M. Choetkiertikul, A. Ghose, Multi-objective iteration planning in agile development, in: 2018 25th Asia-Pacific Software Engineering Conference (APSEC), 2018, pp. 484–493, http://dx.doi.org/10.1109/APSEC.2018.00063.

[19] M. Golfarelli, S. Rizzi, E. Turricchia, Sprint planning optimization in agile data warehouse design, DaWaK (2012) 30–41, http://dx.doi.org/10.1007/978-3-642-32584-7_3.

[20] M. Golfarelli, S. Rizzi, E. Turricchia, Multi-sprint planning and smooth re-planning: An optimization model, J. Syst. Softw. 86 (9) (2013) 2357–2370, http://dx.doi.org/10.1016/j.jss.2013.04.028.

[21] M.A. Boschetti, M. Golfarelli, S. Rizzi, E. Turricchia, A Lagrangian heuristic for sprint planning in agile software development, Comput. Oper. Res. 43 (2014) 116–128, http://dx.doi.org/10.1016/j.cor.2013.09.007.

[22] U. Iqbal, K.A. Alam, Next release problem: A systematic literature review, KIET J. Comput. Inf. Sci. 3 (1) (2020) 65–78.

[23] Y. Zhang, M. Harman, G. Ochoa, G. Ruhe, S. Brinkkemper, An empirical study of meta- and hyper-heuristic search for multi-objective release planning, ACM Trans. Softw. Eng. Methodol. 27 (1) (2018) 1–32, http://dx.doi.org/10.1145/3196831.

[24] V. Escandon-Bailon, H. Cervantes, A. García-Nájera, S. Zapotecas-Martínez, Analysis of the multi-objective release plan rescheduling problem, Knowl.-Based Syst. 220 (2021) 106922, http://dx.doi.org/10.1016/j.knosys.2021.106922.

[25] P. Singh, M.S. Bhamrah, J. Kaur, Jaya algorithm using weighted sum approach for the multi-objective next release problem, in: 2018 First International Conference on Secure Cyber Computing and Communication (ICSCCC), 2018, pp. 273–277, http://dx.doi.org/10.1109/ICSCCC.2018.8703315.

[26] S. Zapotecas-Martínez, A. García-Nájera, H. Cervantes, Multi-objective optimization in the agile software project scheduling using decomposition, in: GECCO '20: Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion, Association for Computing Machinery, 2020, pp. 1495–1502, http://dx.doi.org/10.1145/3377929.3398146.

[27] N. Gademann, M. Schutten, Linear-programming-based heuristics for project capacity planning, IIE Trans. 37 (2) (2005) 153–165, http://dx.doi.org/10.1080/07408170590885611.

[28] C. Li, J.M. Van den Akker, S. Brinkkemper, G. Diepen, An integrated approach for requirement selection and scheduling in software release planning, Requir. Eng. 15 (4) (2010) 375–396, http://dx.doi.org/10.1007/s00766-010-0104-x.

[29] A. Szoke, Conceptual scheduling model and optimized release scheduling for agile environments, Inf. Softw. Technol. 53 (6) (2011) 574–591, http://dx.doi.org/10.1016/j.infsof.2011.01.008.

[30] G. van Valkenhoef, T. Tervonen, B. de Brock, D. Postmus, Quantitative release planning in extreme programming, Inf. Softw. Technol. 53 (11) (2011) 1227–1235, http://dx.doi.org/10.1016/j.infsof.2011.05.007.

[31] M. Griffiths, Pmi-Acp Exam Prep, RMC Publications Inc., 2020.

[32] N.K. Paul, H.K.P. Saikia, A generalization of fibonacci sequence, Proyecciones (Antofagasta Line) 39 (6) (2020) 1393–1405, http://dx.doi.org/10.22199/issn.0717-6279-2020-06-0085.

[33] J. Carson, Genetic Algorithms: Advances in Research and Applications, Nova Science Publishers, 2017.

[34] K. Deb, R.B. Agrawal, Simulated binary crossover for continuous search space, Complex Syst. 9 (2) (1995) 115–148.

[35] K. Deb, M. Goyal, A combined genetic adaptive search (GeneAS) for engineering design, Comput. Sci. Inform. 26 (1996) 30–45.

[36] S.M. Lim, A.B. Md. Sultan, M.N. bin Sulaiman, A. Mustapha, K.Y. Leong, Crossover and mutation operators of genetic algorithms, Int. J. Mach. Learn. Comput. 7 (1) (2017) 9–12, http://dx.doi.org/10.18178/ijmlc.2017.7.1.611.

[37] J. Yang, J. Liu, S. Zhang, Optimization for short-term operation of hybrid hydro-PV power system based on NSGA-II, in: 2020 IEEE 4th Conference on Energy Internet and Energy System Integration (EI2), 2020, pp. 2267–2271, http://dx.doi.org/10.1109/EI250167.2020.9346601.

[38] B. Ma, L. Song, M. Yan, Y. Ikeda, Y. Otake, S. Wang, Multiobjective optimization shielding design for compact accelerator-driven neutron sources by application of NSGA-II and MCNP, IEEE Trans. Nucl. Sci. 68 (2) (2021) 110–117, http://dx.doi.org/10.1109/TNS.2020.3040500.

[39] M. Orouskhani, D. Shi, X. Cheng, A fuzzy adaptive dynamic NSGA-II with fuzzy-based borda ranking method and its application to multimedia data analysis, IEEE Trans. Fuzzy Syst. 29 (1) (2021) 118–128, http://dx.doi.org/10.1109/TFUZZ.2020.2979119.

[40] D. Alanis, P. Botsinis, S.X. Ng, L. Hanzo, Quantum-assisted routing optimization for self-organizing networks, IEEE Access 2 (2014) 614–632, http://dx.doi.org/10.1109/ACCESS.2014.2327596.

[41] H. Ahmadi, A.A. Foroud, Design of joint active and reactive power reserve market: a multi-objective approach using NSGA II, IET Gener. Transm. Distrib. 10 (1) (2016) 31–40, http://dx.doi.org/10.1049/iet-gtd.2014.1226.

[42] Q. Liu, G. Jiao, A pipe routing method considering vibration for aero-engine using kriging model and NSGA-II, IEEE Access 6 (2018) 6286–6292, http://dx.doi.org/10.1109/ACCESS.2018.2789361.

[43] A. Seshadri, A fast elitist multiobjective genetic algorithm: NSGA-II, 2000, URL https://web.njit.edu/~horacio/Math451H/download/Seshadri_NSGA-II.pdf.

[44] A. Ghosh, M.K. Das, Non-dominated rank based sorting genetic algorithms, Fund. Inform. 83 (3) (2008) 231–252.

[45] I. Ben Hamida, S.B. Salah, F. Mshali, M.F. Mimouni, Optimal integration of solar distributed generations in distribution network using SPEA2, in: 2016 17th International Conference on Sciences and Techniques of Automatic Control and Computer Engineering (STA), 2016, pp. 368–373, http://dx.doi.org/10.1109/STA.2016.7952079.

[46] B. Gadhvi, V. Savsani, V. Patel, Multi-objective optimization of vehicle passive suspension system using NSGA-II, SPEA2 and PESA-II, Proc. Technol. 23 (2016) 361–368, http://dx.doi.org/10.1016/j.protcy.2016.03.038.

[47] I. Huseyinov, A. Bayrakdar, Performance evaluation of NSGA-III and SPEA2 in solving a multi-objective single-period multi-item inventory problem, in: 2019 4th International Conference on Computer Science and Engineering (UBMK), 2019, pp. 531–535, http://dx.doi.org/10.1109/UBMK.2019.8907139.

[48] H. Maheta, V. Dabhi, An improved SPEA2 multi-objective algorithm with non-dominated elitism and generational crossover, in: 2014 International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT), 2014, pp. 75–82, http://dx.doi.org/10.1109/ICICICT.2014.6781256.

[49] E. Zaenudin, A.I. Kistijantoro, pSPEA2: Optimization fitness and distance calculations for improving strength Pareto evolutionary algorithm 2 (SPEA2), in: 2016 International Conference on Information Technology Systems and Innovation (ICITSI), 2016, pp. 1–5, http://dx.doi.org/10.1109/ICITSI.2016.7858224.

[50] T. Okabe, Y. Jin, B. Sendhoff, A critical survey of performance indices for multi-objective optimisation, in: The 2003 Congress on Evolutionary Computation, 2003. CEC '03, Vol. 2, 2003, pp. 878–885, http://dx.doi.org/10.1109/CEC.2003.1299759.

[51] N. Riquelme, C. Von Lücken, B. Baran, Performance metrics in multi-objective optimization, in: 2015 Latin American Computing Conference (CLEI), 2015, pp. 1–11, http://dx.doi.org/10.1109/CLEI.2015.7360024.

[52] R. Tanabe, H. Ishibuchi, An analysis of quality indicators using approximated optimal distributions in a 3-D objective space, IEEE Trans. Evol. Comput. 24 (5) (2020) 853–867, http://dx.doi.org/10.1109/TEVC.2020.2966014.

[53] E. Dilettoso, S.A. Rizzo, N. Salerno, A weakly Pareto compliant quality indicator, Math. Comput. Appl. 22 (2017) 25, http://dx.doi.org/10.3390/MCA22010025.

[54] J.G. Falcón-Cardona, M. Emmerich, C.A. Coello Coello, On the construction of pareto-compliant quality indicators, in: Computer Science, Proceedings of the Genetic and Evolutionary Computation Conference Companion, 2019, http://dx.doi.org/10.1145/3319619.3326902.

[55] J.G. Falcón-Cardona, S.Z. Martínez, A. García-Nájera, Pareto compliance from a practical point of view, in: Economics, Proceedings of the Genetic and Evolutionary Computation Conference, 2021, http://dx.doi.org/10.1145/3449639.3459276.

[56] H. Ishibuchi, R. Imada, N. Masuyama, Y. Nojima, Comparison of hypervolume, IGD and IGD+ from the viewpoint of optimal distributions of solutions, in: Evolutionary Multi-Criterion Optimization, Springer International Publishing, 2019, pp. 332–345, http://dx.doi.org/10.1007/978-3-030-12598-1_27.