A Multi-Objective E-learning Recommender System at Mandarine Academy[*]

MOUNIR HAFSA, Mandarine Academy, Univ. Lille, CNRS, Centrale Lille, UMR 9189 - CRIStAL, France

PAMELA WATTEBLED, Mandarine Academy, France

JULIE JACQUES, Lille Catholic University, FGES, France

LAETITIA JOURDAN, Univ. Lille, CNRS, Centrale Lille, UMR 9189 - CRIStAL, France

Recommender systems are quickly becoming a part of our daily digital life. Mainly found in applications such as e-commerce, social media, and online entertainment services. They help users overcome the information overload problem by improving the browsing and consumption experience. Mandarine Academy is an Ed-Tech company that operates more than a hundred online e-learning platforms. They creates online pedagogical content (videos, quizzes, documents, etc.) on daily basis to support the digitization of work environments and to keep up with current trends. Suggesting items that are relevant to both users and visitors is challenging, the company is looking for ways to improve the learning experience by providing content that adheres to specific conflicting requirements. These requirements include similarity with user profile, the novelty of proposed content, and diversity of recommendations. Mandarine Academy is looking to implement an approach that can handle multiple conflicting goals with the possibility to adjust which to use in each browsing scenario. In this article, we propose a solution for Mandarine Academy Recommendation System (*MARS*) problem by using Evolutionary Algorithms based on the concept of Pareto Ranking. After modeling objectives (Similarity, Diversity, Novelty, RMSE, and nDCG@5) as an optimization problem, we compared different algorithms (NSGA II, NSGAIII, IBEA, SPEA2, and MOEAD) to study their performance under different test settings. Extended data analysis of real-world user interactions showed drawbacks of many graphical issues that prevented users from learning efficiently and we proposed enhancements to the overall user experience and interface. We discuss initial findings under various objectives which show promising results considering production mode scenarios. A proposed custom mutation operator was able to outperform the classical swap mutation. A multi-Criteria Decision-Making phase that uses by default pseudo weight is responsible for providing results for end users after training our model.

CCS Concepts: • **Applied computing → Multi-criterion optimization and decision-making**; **E-learning**; • **Information systems → Retrieval models and ranking**.

Additional Key Words and Phrases: Recommender Systems, Multi-Objective Optimization, Evolutionary Algorithms, E-Learning, MOOC, Corporate

1 INTRODUCTION

Mandarine Academy is an Ed-tech company that supports the digital transformation of work environments by facilitating the handling and use of new technologies by all employees. It offers a new way of training more effectively in terms of skills, capacity, time, and budget due to an exclusive approach that combines both digital learning and personalized support. With over half million users and more than 100 platforms, the company operates several products for multiple partners in varying sectors. Mandarine Academy saw how Massive Open Online Course (MOOC) impacted the traditional higher education market as well as the increasing rate of industry digitization and proposed custom MOOCs focused on the corporate sector to save employees from obsolescence.

Authors' addresses: Mounir Hafsa, mounir.hafsa@mandarine.academy, Mandarine Academy, Univ. Lille, CNRS, Centrale Lille, UMR 9189 - CRIStAL, Lille, France, 59000; Pamela Wattebled, pamela.wattebled@mandarine.academy, Mandarine Academy, Lille, France, 59000; Julie Jacques, julie.jacques@univ-catholille.fr, Lille Catholic University, FGES, Lille, France, 59000; Laetitia Jourdan, laetitia.jourdan@univ-lille.fr, Univ. Lille, CNRS, Centrale Lille, UMR 9189 - CRIStAL, Lille, France, 59000.

One of their most popular products is the Mooc-office365-training [1], a public bilingual (French & English) MOOC destined for learning Microsoft office 365 tools and soft skills related to workplaces. Having around four thousand monthly active users and more than hundred and thirty thousand registered users, the platform includes different types of learning materials such as:

- Resources: Tutorials & use cases (short format videos), quizzes, documents, recorded live conferences, SCORM (Shareable Content Object Reference Model).
- Courses: A collection of unordered learning resources.
- Learning Paths: A predefined set of courses to master certain skill/job/specialization.

In this work, we focus on video resources (tutorials & use cases) because they make up the majority of the Mooc-office365-training's catalog. The company is providing up-to-date content that matches changes in work environments and current trends. Unfortunately, this impacts users as they have to spend more time selecting the appropriate learning material which can lead to known problems such as information overload, distraction, disorientation, lack of motivation [11], among other identified issues:

- New subscribers/visitors may have difficulty selecting the appropriate content, to begin with, depending on their needs (learn a new skill or career change).
- Watch next: After finishing viewing a video, users are not given a playlist of what to watch next. This can cause frustration and an increase in dropout rates.
- One size fits all: A lack of personalization as the same content is displayed for all users and not tailored to their specific interests.

To reduce the time spent searching for relevant information, scientific literature proposed multiple approaches that matches users with relevant content through the use of recommender systems. Unfortunately, most of the works deal with academic recommender systems which is different from corporate objectives.

In this paper, we investigate and solve problems encountered in a public e-learning platform operated by Mandarine Academy. After reviewing the literature and analysing user behavior by collecting real-world interactions (explicit and implicit), we proceed to mathematically represent company objectives and constraints into an optimization problem. In addition, we perform a critical analysis of the user interface and experience to understand its impact on a user's learning journey. Our approach combines both traditional recommender system techniques with metaheuristics in order to recommend relevant content to users. The established experimental protocol aims to compare the performance of many evolutionary algorithms and provides an in-depth analysis of their results. Finally we give insights about production mode findings and future research directions. The rest of the paper is organized as follows: Section 2 provides an overview of related works. In section 3, we discuss the findings of data analysis done on Mooc-office365-training and list different graphical issues. In section 4, we showcase our proposed approach. Section 5 provides experimentation design and results analysis. Section 6 concludes the paper and gives directions for future work.

2   STATE OF THE ART: RECOMMENDER SYSTEMS AND E-LEARNING

In this section we give a brief description of Multi-Objective Optimization (MOO) and common recommendation system techniques before diving into related scientific studies.

Real-world optimization problems are rarely mono objective, instead we end up with many conflicting goals. This can be defined as optimizing $F(x) = (f_1(x), f_2(x), ..., f_n(x))$ with $x \in F_{sol}$, n is number of objectives ($n \geq 2$), x being a

---

[1]https://mooc.office365-training.com/

vector of decision variables, F_{sol} a set of feasible solutions and $f_i(x)$ represents each objective that we want to minimize or maximize [6]. Unlike mono-objective optimization, results are not a single solution, but a Pareto set [26] of optimal solutions where no improvement can be found for an objective without degrading another objective value.

Metaheuristics represent a family of approximate optimization strategies that offer acceptable solutions to complex problems in reasonable time. Unlike exact optimization algorithms, metaheuristics do not guarantee that the results are optimal [17]. Evolutionary techniques such as Genetic Algorithms (GAs) have been extensively used to solve complex optimization problems GAs were developed by J. Holland in the 1970s [18] to mimic the adaptive processes of natural systems.

Before we can assess "how good" any single run of a Multi-Objective Evolutionary Algorithm (MOEA) is, we must first grasp two concepts. The first is convergence, which indicates how "close" we come to find the best solution. The second metric is diversity which measures if solutions are fully spread throughout the set or are clustered together. The following quantitative metrics can provide an evaluation mechanism for both convergence and diversity [39]: Hypervolume (HV) (Maximize), Generational Distance (GD) (Minimize), Inverse Generational Distance (IGD) (Minimize), Epsilon-Indicator (ϵ) (Minimize).

Recommender Systems (RS) are algorithms aimed at suggesting relevant items to users. This is made possible by filtering massive amounts of data that can be obtained from users, content, or other sources (e.g. context).

Recommender systems are becoming a part of our daily digital life. Mainly found in online entertainment services, e-commerce, and social networks [12, 13, 33], but other sectors are adopting this technology. From the user's perspective, recommenders reduce the time it takes to find appropriate items, increasing brand satisfaction, loyalty, and familiarity. From the standpoint of a business owner, this provides information about what users like without requiring additional marketing/support efforts.

Generally, the most used types of recommender systems are: Collaborative Filtering (CF) and Content-Based filtering (CBF). The main difference between these two techniques is the type of data employed. The CBF approach requires items metadata to recommend content having similar attributes with the user profile [27]. CF approaches on the side, leverages user ratings (explicit/implicit) to predict the likelihood of a user's liking an item. Two different approaches can be used, Model-Based (Machine/Deep learning models) and Memory Based (User/Item-based).

We found that existing recommendation algorithms (CF and CBF) do not perform well when evaluated in terms of accuracy, novelty, and diversity. Approaches that exploit the combination of such recommendation algorithms are known as Hybrid Recommenders [3]. The advantage of this approach is that it limits the cons of each method used in the hybridization process all while inheriting their advantages. Other approaches that exploit the Pareto efficiency concept in order to combine such recommendation algorithms in a way that a particular objective is maximized without significantly hurting the other objectives. Recommender systems and the use of metaheuristics have been the focus of many academic researchers [4, 15, 32, 34]. We critique a few works that take a similar approach in the following section.

Xie et al. [35] integrated a personalized approximate Pareto-efficient recommendation on the WeChat Top Stories section for millions of users. Their approach used reinforcement learning to find objective weights for the target user using list representation. Five online metrics (click through rate, dwell-time scores for both system and item, has-click rate and diversity) were used to evaluate models. Fortes et al. [14] also adopted a similar technique, relying on user preferences concerning objectives weights during both the decision-making and optimization phases.

Zuo et al. [40] proposed a multi-objective personalized recommendations using clustering to improve the computational efficiency. Their approach optimises both accuracy and diversity using NSGA-II algorithm [9]. Another work [22] that collects the tendencies of users, based on their past behavior, to provide a personalized recommendations list

that adhere to the defined goals. Using a greedy re-ranking technique to match items with user profiles. The use of multiple recommendations engines is developed in the work of Ribeiro et al. [28]. A Pareto-efficient recommendation approach optimize weights of associated engines to provide items that are accurate, novel, and diverse.

Our work addresses different aspects that are missing or under-exploited in the aforementioned works:

- The use of multiple recommendation engines to initialise our solution population and provide more diversity.
- Working with real-world implicit ratings to train our model.
- We propose a customized mutation operator to improve diversity of recommended items.
- Performance comparison of various Multi-Objective Evolutionary Algorithms (MOEA).
- Optimizing five conflicting objectives in the context of a real-world problem.
- The use of parameter tuning to optimize algorithms depending on user behaviour and selected objectives.
- Work is being integrated in production-ready environment.

3 UNDERSTANDING USER BEHAVIOR

The more you know about your users, the better equipped you'll be to make informed decisions about your service. In order to gain a richer understanding of how users interact with content, events are used to independently track users journeys. A typical method of providing feedback is in the form of rating methods that captures users preferences in explicit ways (like button, social sharing, course/learning path registration and bookmarks). The disadvantage is that users tend to avoid the burden of explicitly stating their preferences. To overcome the shortage of explicit ratings, platforms tend to collect users behavior through multiple ways (page views, percentage of videos watched, etc). This is called implicit feedback. The advantage is that users can trigger a lot of actions when using a service. This generates a lot of data that can be significant in some cases but shows a major inconvenient, which is not having a ground truth. When running short on ratings (explicit or implicit), content descriptors like (subtitles, title, description, number of views, duration, etc.) are used as additional input to recommender systems.

We conduct our data analysis using the Mooc-office365-training platform (French version) with the following catalog:

- 41 Learning Paths.
- 142 Courses.
- 1294 Tutorials and 113 Use Cases.

Collected data was captured from early 2018 to late 2020. Both Table 1 and Table 2 shows available user events (explicit and implicit). Starting with column **% of users** which indicates the percentage of users that used the feature at-least once. The difference between explicit and implicit ratings is distinguishable. Only about 1% of users have explicitly indicated their feedback, with social sharing being the most used. When we look at implicit interactions, we see a different story, as more significant users are interacting with content.

Same behaviour applies with column **% of content** as explicit interactions have a smaller number of involved content compared to implicit ones. Furthermore, investigating implicit ratings reveals that approximately 7% of pages have never been visited and approximately 9% of videos have never been watched. These findings are alarming, specially when a part of the catalog is hidden from public view.

Finally, column **sparsity score** defines the ratio of unspecified ratings to the total number of entries in the user-item matrix and calculated as follows $sparsity = 1 - \frac{|R|}{|U| x |I|}$.

Observations taken from Table 1 and 2 not only show a high sparsity score which is normal for real-world data but also a usage gap between both implicit and explicit interactions. Since explicit ratings are visible to users, we suppose

Table 1. Explicit interactions captured from Mooc-Office365 (French) starting 2018 to late 2020.

Interaction	% of users	% of content	Number of entries	Sparsity
Likes/Dislikes	0.02%	0.9%	28	0.830%
Social Shares	0.66%	58.11%	2179	0.997%
Learning Path/ Course Registration	0.439%	40%	1202	0.996%
Bookmarks	-	-	-	-

Table 2. Implicit interactions captured from Mooc-Office365 (French) starting 2018 to late 2020.

Interaction	% of users	% of content	Number of entries	Sparsity
Page View	21.86%	93.08%	610,956	0.985%
View Portion (Resources)	8.26%	91.57%	68,894	0.993%

that other reasons beside avoiding to express their opinion might be possible. To confirm our hypothesis we observed the graphical interface available for both registered users and visitors. We list below our findings per page.

(1) Home page: The current homepage offers a list of newest courses and tutorials. Users / visitors are limited if they are looking to learn about certain tools, required skills for specific jobs or certification. What we propose for **visitors** is a list of items (courses and resources) with options to select popular or newer items. Furthermore, categories (skills, jobs, certificates) should be shown in top of the page to guide visitors efficiently. For **registered users**, a multiple personalized lists of recommended items provided by our approach and others (CF, CBF) will help users find relevant content easier.

(2) Content page: Learning paths, courses and videos (tutorials and use cases) are presented to both users and visitors without similar items, visible interactions or feedback options. The like and share buttons are provided without text, only a small icon. In case a video doesn't correspond to a user's needs, they must go back to the previous page and spend additional time looking for another one. We propose for both users and visitors a more appealing interface with visible interactions (like, dislike, social share), addition of a "save to watch later (Bookmark)" and "feedback" options. Multiple recommended lists (CF, CBF, Popularity) of similar items to minimise the burden of content search and provide guidance.

Initial propositions address the way content as well as interactions are displayed and insists on improving both visibility and readability for users. One of the proposed features (Bookmark) has already been integrated into the platform and its shown in Table. 1, since it's relatively new, additional data need to be collected before assessing its significance. Furthermore, the search process is upgraded with more filters to empower users looking for specific information. Finally, we highlight recommended content by improving its graphical positioning for users [5]. Overall, the process aims to make the platform more accessible and easy to use by reassembling certain graphical elements (video player, item sliders, and search bar) to match common online services (entertainment and e-commerce websites). The above propositions aim to reduce cognitive overload caused by clumsy and unfamiliar browsing experiences, which users may encounter on occasion [31]. Not only that but to make the exploration process easier and more productive. Further AB testing campaigns are planned across these pages to measure the effects on user/visitor behavior and satisfaction levels.

4   MARS: MANDARINE ACADEMY RECOMMENDER SYSTEM

The performance evaluation of our approach is highly dependent on the data it processes and the task it has to perform. In our particular context, this task is complex since it must satisfy different goals that are far from being complementary. We had to find a compromise satisfying both the need to match user taste, highlight diverse content and also focus on unpopular items. These goals represent what the company aspires to achieve. These requirements were updated with two additional objectives widely used in the literature and considered as a standard in recommender systems evaluation metrics.

(Objective 1) Maximize similarity with user profile. This is done by calculating the overall cosine score between items in user's profile and items in the proposed recommendation.

$$Psim = \frac{\sum_L^{i=0} \sum_N^{j=0} csim(r_i, u_j)}{S_L} \tag{1}$$

Where L is the recommended list (solution) and r_i is the item number i from L. The user profile is expressed as N where u_j is the item number j from N. $csim$ is the item-item cosine distance score. S_L is the length of the solution.

(Objective 2) Maximize diversity which is responsible for how dissimilar items are in the solution. This can be achieved by using the Intra-List Similarity metric (ILS) [30]. It uses the same logic behind Objective 1 by calculating the average cosine similarity of all items in a list of recommendations. Note that this objective is conflicting with the first objective. As the first looks for similar items to the user profile, the second looks for more diversified items within the proposed list itself.

$$Rdiv = \frac{\sum_{L-1}^{i=0} \sum_L^{j=1} csim(r_i, u_j)}{tc} \tag{2}$$

Where L is the recommended list (solution) and r_i is the item number i from L. $csim$ is the item-item cosine distance matrix. tc is the item pairs count.

(Objective 3) Maximize novelty. In this objective, we aim to recommend less popular items and focus on content having both low number of views and recently added to the catalog. A scoring function that sums the number of views and number of days since release, returns the median. The smaller the median, the more novel the items are.

$$Rnov = \frac{\sum_L^{i=0} ns(r_i)}{L} \tag{3}$$

Where L is the recommended list (solution) and r_i is the item number i from L. ns is novelty scoring function.

To add more flexibility to the proposed recommender system, two additional objectives were added. Root Mean Square Error $RMSE$ and the Normalized Discounted Cumulative Gain $nDCG$. Both are well known metric and widely used in recommender systems research field.

(Objective 4) Minimize $RMSE$. What this metric does essentially is finding difference between a predicted rating and a real rating. Having lower error value, means our model is predicting ratings similar to what the user gave.

$$rmse = \sqrt{(\frac{1}{n}) \sum_{i=1}^{n} (y_i - x_i)^2} \tag{4}$$

Where y_i is the actual rating, x_i is the predicted rating and n is the number of ratings.

(Objective 5) Maximize $nDCG$. This is a measure of ranking quality where highly relevant items are more useful when ranked first. Also this metrics follow the assumption that highly relevant items are more useful than marginally

relevant items, which are in turn more useful than non-relevant items. We will be using **nDCG@5** which corresponds to the number of relevant results among the first 5 recommended items.

$$nDCG_p = \frac{DCG_p}{IDCG_p} \tag{5}$$

With $IDCG_p = \sum_{i=1}^{|REL_p|} \frac{rel_i}{\log_2(i+1)}$ and $DCG_p = \sum_{i=1}^{p} \frac{2^{rel_i}-1}{\log_2(i+1)}$. Where REL_p is a list of top p relevant items (ordered by relevance). rel_i is the graded relevance of the result at position i.

Even though both metrics reflect relevance, $nDCG$ is crucial for ranking problems while $RMSE$ isn't relevant for that case. The intuition behind adding these objectives is to provide the ability for platform managers more options to tune from selecting only a single objective to combining many. For example, a platform wants to highlight on content similarity to users profile can use both **(Objective 1)** and **(Objective 4)**.

Note that initial objectives don't capture learning objectives of users. This is due to available events (explicit and implicit). Work is being conducted to incorporate learning performance tracking events to better suggest items for users with specific learning goals (job requirements or mastery of certain tools).

Since we are working on an optimization problem we must define our constraints in order to determine if a solution is feasible or not. Constraints can be defined as conditions that a solution must satisfy in order to be feasible. Two constraints are considered in this work:

- Recommended list L must be unique and contain no duplicates.
- Recommended items must not exceed the fixed length K.

To define our problem, objectives and constraints, we chose JMetalPy [2], a python framework for solving multi and many-objective optimization problems. JMetalPy offers both parallel computing capabilities and a rich set of features such as real-time interactive visualization of the pareto front. We will be using popular multi-objective genetic algorithms found in the literature due to their proven efficiency handling complex problems [16, 21]. They offer a better exploration of the search space and diversified solutions:

- Non-dominated Sorting Genetic Algorithm 2 (NSGA-II) [9],
- Non-dominated Sorting Genetic Algorithm 3 (NSGA-III) [10],
- Indicator-based evolutionary algorithm (IBEA) [37],
- Multi Objective Evolutionary Algorithms by Decomposition (MOEA/D) [36],
- Strength Pareto evolutionary algorithm (SPEA 2). [38]

4.1 Solution encoding

Genetic algorithms begins with the choice of the chromosome encoding (solution representation). This depends on the problem in-hand. Following the works of [1, 4, 25] we define a solution as a list of unique item identifiers. The list will have a **fixed length** K, contain **unique elements** specific and relevant for each user profile.

4.2 Initial Population

After defining a structure that represents our solutions, we proceed to generate an initial set or population of solutions as input for our approach. Generally initial populations can be produced through many approaches (random, heuristics, etc.) [29]. In order to generate good initial solutions that contain relevant results for each user we took advantage

of available data (interactions and content descriptors) to create multiple recommendation engines. The following approaches are used:

- Random.
- Content Based Filtering (CBF).
- Collaborative Filtering (CF) - Item Based.
- Collaborative Filtering (CF) - User Based.
- Collaborative Filtering (CF) - Model Based (SVD++).
- Collaborative Filtering (CF) - Model Based (ALS).
- Association Rules (FP-Growth).

We try to get a recommendation from each approach listed above for each user we test. In case an algorithm is unable to provide personalized recommendations, we have defined a fallback method, which is the "Random" approach that provides a list of randomly selected items. To implement most of these algorithms, the python library "Surprise" [20] was used. This library gives access to baseline algorithms, neighborhood methods, matrix factorization-based approaches and various similarity scores (Cosine, Mean Squared Difference, Pearson, etc) and evaluation metrics ($RMSE$, Fraction of Concordant Pairs (FCP), etc).

4.3 Crossover

Genetic algorithms usually applies a crossover operation to two solutions in order to produce new chromosomes (solutions) called "children chromosomes". Operators like One-Point and Two-Point crossover are popular in the literature [1, 4, 19, 25]. The idea behind such operators is simple, chose a random one or two items from parent 1 and interchange their places with parent 2. These operators are responsible for the order of which elements are shown. Despite its simplicity, such operators can render a solution invalid. Consider that items from parent 2 were placed in parent 1. These unique items in parent 2 can be redundant in parent 1, thus applying a repairing mechanism is a must. This repair function changes the redundant items in one of the parent solutions by using the random approach and validate the solution.

4.4 Mutation

Mutation is a genetic operation used to maintain genetic diversity. It introduces changes inside solutions in an attempt to avoid local minima. Random mutation is frequently found in the literature [25], [4] along with 1-point mutation [1], 2-point mutation [32] and Uniform mutation [1]. We propose a custom mutation operator named $MARS_{mo}$ to be compared to classical operators. The concept behind $MARS_{mo}$ is to choose N , with ($1 \leq N \leq k/2$), elements from a candidate solution. The selected elements are then randomly swapped with either (1) Similar items (using Content-Based or Item-Based approaches), (2) Random (3) Novel (Recently added to the catalog) items. The full pseudo code is demonstrated in 1.

To make use of initial population generators, platform managers will be able to create these models using an administration dashboard. We identified a total of three model types each with different associated algorithms:

- Popularity: Provide popular content based on either view portions (default) or number of page visits.
- Personalized: Based upon one method (User Based CF, Item-Based CF, Content-Based (default), Model-Based CF, Metaheuristic, FP-Growth).
- Similar Items: Based upon one method (Item-Based CF, Content Based (default), FP-Growth).

Algorithm 1 Pseudo-code of *MARS* custom mutation operator $MARS_{mo}$

Require: $Defined\ Probability, L$
  $Replacement\ Method \leftarrow random\ method \in (Similarity,\ Random,\ Novelty)$
  $Replaced\ Items \leftarrow random\ items \in L$
  $Mutation\ Probability \leftarrow random \in [0.0, 1.0]$
  if $Mutation\ Probability \leq Defined\ Probability$ **then**
      for each Item $\in$ Replaced Items **do**
         Item $\leftarrow Replacement\ Method(Item)$
  end if

Table 3. Implicit Interactions (View Portions) from Mooc-office-365 (French) starting 2018 to late 2020.

User ID	Item ID	View Portion	State	Date
792	995	7%	Not considered	2018-01-21
3475	516	30%	In progress	2018-05-25
...	...	...	...	...
687	520	80%	Finished	2020-08-14
542	498	45%	In progress	2020-09-16

Additional advanced settings can be fine tuned for example, selecting objectives associated to each model and what interactions to use. This makes it possible to experiment with different parameters across different interface placements. This dashboard is still in work and will also provide monitoring features to track performance of each model in real-time by using metrics such as click-rate (*CR*), watch time and click through rate (*CTR*).

5 EXPERIMENTS AND TESTS

Along with the implementation of the metaheuristic and different solution generators, we conducted a series of tests in a more experimental setting in order to compare different approaches for solving the recommendation problem with different objectives combinations.

Technically, platform managers can select any combination of objectives they desire. However, in our test settings we tested a subset of possible combinations to observe the impact of having multiple objectives on performance.

Each test setting has a parameter tuning phase to ensure each algorithm is using the best configuration for the task. The data in this experiment was collected from real-world users and were adapted to our approach.

5.1 Dataset

Moving on to initial experiments where the first step was selecting the right data to work with. We have seen that explicit ratings in Table 1 suffer from low user engagement compared to implicit ratings in Table.2. Unfortunately for page views, there isn't a ground truth that indicates if viewing a content page multiple times leads to increased user satisfaction. However, view portions (Watch time) might be a good fit for our approach as it can be used to measure user interest. Table. 3 details the different attributes and values present in view portions.

The dataset has a sparsity of 99.312%. With a total of 822 users, 776 items, and 3699 ratings. The company uses the following scale to describe viewing events:

- (1) Not considered: viewings from 0% to 10% of the video.
- (2) In progress: viewings from 11% to 69% are considered equal.

<div align="center">(a) The current rating scale.　　　　　　　　　　　　　(b) The proposed rating scale.</div>
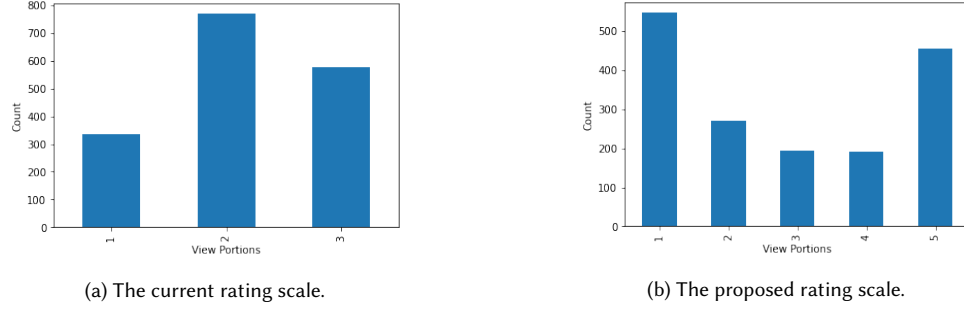
<div align="center">Fig. 1. Count of implicit interactions (View Portions) per user.</div>

- (3) Finished: viewings from 70% to 100% consider the user has finished watching the item.

We believe that the degree of viewing has an impact on the overall impression and that the previous scoring function does not reflect user interest. When plotting the old scoring scale shown in Fig. 1, the majority of users are in the "In progress" state followed by "Finished". This implies that users are still learning or have completed their videos. A new scoring system is proposed. It builds on the previous approach by incorporating more levels of appreciation. The assumption is that longer viewing times indicate a higher user interest and satisfaction. Considering the following scale:

- (1) No interest: viewings from 0% to 20% of the video.
- (2) Small interest: viewings from 21% to 40% of the video.
- (3) Medium interest: viewings from 41% to 60% of the video.
- (4) High interest: viewings from 61% to 80% of the video.
- (5) Finished: viewings from 81% to 100% of the video.

This introduces 5 different levels that have varying degrees of importance and reassembles the classical 5-star rating system. When plotting the new scoring system, a different narrative is found. The majority of users fall into the "No interest" category, followed by "Finished" and "Small interest". Different assumptions are made here, starting with the highest group count "No interest" which indicates that users watched a maximum of 2 seconds on a scale of 10. This can be interpreted as users stumbling into wrong content and returning to search for something more suitable. Perhaps the title wasn't clean enough since descriptions aren't always provided, or the video content was advanced for user's skills. Additional insights are gathered after further dataset analysis. When grouping seen elements per user we obtained an average of 5 items. With 80% of users have seen less than or equal to the average. This might indicate that most users have abandoned their learning path or are unable to locate appropriate content. We only consider videos with over 50% of watch time in users profile. The assumption is that longer watch times indicate a higher user interest and satisfaction.

5.2   Parameter Tuning

Instead of choosing fixed parameters empirically and applying them to all algorithms indifferently, another protocol is used to provide a fair performance comparison. It uses the irace package [24] which implements an iterated racing approach to automatically find optimal settings. The library focus on improving optimization algorithms and machine learning models. For each parameter a list with possible values is shown in Table 4. A fixed computing time limit of one hour is defined as a stopping criteria for our experiment. Concerning results, K was set to 10 items (max number of recommended items).

Table 4. Parameters settings considered for tuning phase.

Parameter	Description	Possible Values
Pop	Population size at each generation.	10, 50, 100, 200, 500, 1000
Cx	Crossover Genetic Operator	1-Point, 2-Point
Cx_p	Crossover Probability	0.1 - 1.0
Mx	Mutation Genetic Operator	Random, $MARS_{mo}$
Mx_p	Mutation Probability	0.1 - 1.0
Kp	Kappa (IBEA)	0.1 - 1.0
NSP	Neighbourhood Selection Probability (MOEAD)	0.1 - 1.0
$MNRS$	Max Number of Replaced Solutions (MOEAD)	10, 50, 100, 200, 500, 1000
NS	Neighbor Size (MOEAD)	10, 50, 100, 200, 500, 1000

Elite configurations are returned based on their average best Hypervolume (HV) metric [39] across different test instances. The HV metric is capable of measuring both convergence and diversity of our solutions. The higher HV value, the better our solutions are. Note that we will be comparing the proposed custom mutation operator $MARS_{mo}$ with the classical $Swap$ mutation.

Our first test experiment will focus on the three initial objectives ($Psim$, $Rdiv$, $Rnov$) which were proposed initially by the company. The second test experiment will add both $RMSE$ and $nDCG@5$ to the other three objectives. This will make it a many-objective optimization task and will test how the parameters and performance adapts. Note that both $RMSE$ and $nDCG@5$ requires a portion of user history to validate predictions. Since 80% of users have less than or equal to 5 items in their history we will be using the remaining 20% of users as they have more watched items. Both test experiments are samples of the possible choices that platform managers can select. For example another setting can be focused on relevance by having three different objectives ($Psim$, $RMSE$ and $nDCG@K$).

Starting with Table 5 which shows the elites configurations provided by irace for each algorithm (NSGA2, NSGA3, SPEA2, MOEA/D, and IBEA) using implicit interactions for three objectives ($Psim$, $Rdiv$, $Rnov$). Observations indicate that $1 - Point$ crossover operator has been chosen over the $2 - Point$ crossover operator by the majority of algorithms. This can be explained by the fact that the crossover operator in this test setting does not impact objectives performance, so selecting a simpler operator could be the reason. Only $MOEA/D$ chose $MARS_{mo}$ as the mutation operator, while the rest of the algorithms used the random mutation operator. This can be attributed to a variety of factors, including the allowed computing time and length of solutions K aside from the number of objectives. When looking at Table 6 for the many-objectives irace runs, most elite configurations have chosen the $2 - Point$ crossover operator over $1 - Point$. This confirms our previous assumption, that crossover operators, are chosen depending on their role in improving the objectives. Since the additional objectives in this experiment have an interest in item ordering, a change in elite configuration was anticipated. Similarly, all algorithms selected $MARS_{mo}$ as a mutation operator, indicating that this operator has superior performance, particularly in complex settings.

5.3 Results and Performance Analysis

The following experiments are based on the results provided by irace. For each algorithm, 30 independent executions are launched using the same settings as seen in parameter tuning: stopping criteria of one hour, objectives ($3OBJ\&5OBJ$) and $K = 10$ recommended items). We used these metrics discussed in Section 2 to measure the quality of our solutions: HV, GD, IGD and ϵ. However, the latter metrics do not guarantee better recommendation results for the end user, for

Table 5. Elites configurations provided by i-race using 3 objectives on implicit dataset.

Parameter	NSGA2	NSGA3	SPEA2	MOEAD	IBEA
Pop	10	10	10	500	10
Cx	1-Point	1-Point	1-Point	2-Point	1-Point
Cx_p	0.3	1.0	0.9	0.7	0.1
Mx	Random	Random	Random	$MARS_{mo}$	Random
Mx_p	1.0	0.6	1.0	0.9	0.9
Kp	-	-	-	-	0.2
NSP	-	-	-	1.0	-
$MNRS$	-	-	-	1000	-
NS	-	-	-	500	-

Table 6. Elites configurations provided by i-race using 5 objectives on implicit dataset.

Parameter	NSGA2	NSGA3	SPEA2	MOEAD	IBEA
Pop	10	10	10	100	100
Cx	1-Point	2-Point	2-Point	2-Point	2-Point
Cx_p	0.1	0.1	0.6	0.3	0.6
Mx	$MARS_{mo}$	$MARS_{mo}$	$MARS_{mo}$	$MARS_{mo}$	$MARS_{mo}$
Mx_p	1.0	0.8	1.0	1.0	0.9
Kp	-	-	-	-	1.0
NSP	-	-	-	0.8	-
$MNRS$	-	-	-	500	-
NS	-	-	-	100	-

this we will be comparing the recommended items after the Multi-Criteria Decision Making (MCDM) phase but as we seen in Section 4, metrics like the click-rate (CR), watch time and click through rate (CTR) will be implemented to measure how users are handling recommended items.

Starting with HV_{3OBJ} column found in Table 7, results show that $NSGA3$ has a maximum score of (**0.91**). In second place *IBEA* followed by *SPEA2* with a score of (**0.85**) and (**0.80**) respectfully. The previous findings are compared with the results of 5 Objectives experiments. Since objectives *Psim*, *Rdiv* and *Rnov* are already included, we aggregate their values. This is indicated by HV_{3OBJ} column in Table 8. Both *SPEA2* (**0.85**) and *NSGA3* (**0.84**) kept a robust performance with *NSGA2* (**0.84**) outperforming its previous HV score. *IBEA* didn't perform well compared to the HV_{3OBJ} experiment. Shifting our focus to 5 Objectives results shown in HV_{5OBJ} column. *SPEA2* and *NSGA3* achieved good scores of (**0.81**) and (**0.79**) respectively.

Taking into account other performance indicators (GD, IGD, ϵ) which must be minimized and starting with GD column shown in Table 7. Most algorithms obtained similar scores with *IBEA* achieving the lowest score (**1.22**). However, looking at IGD column, *SPEA2* was able to obtain a value of (**1.002**) and creating a gap with the rest of algorithms. Note however that both GD and IGD are easier metrics to meet compared to ϵ. For which, *IBEA* was able to achieve the lowest ϵ score with (**0.093**) followed by both *NSGA3* and *SPEA2*. Discussing the same metrics for $5OBJ$ experiments shown in Table 8. *NSGA2* obtained the lowest GD score (**1.41**) not far from other algorithms. Surprisingly thought, *NSGA2* was also able to obtain the lowest IGD score of (**1.06**) followed by *NSGA3*. When considering the ϵ column, *SPEA2* achieved a performance similar to *NSGA2* with scores of (**0.06**) and (**0.07**) respectfully. Setting aside the HV_{5OBJ}

results, *NSGA*2 have shown good results considering the many-objective problem setting. But, *SPEA*2 and *NSGA*3 continued to perform marginally better in which make them more fit for our future experiments.

Moving on to discuss the evolution of the hypervolume *HV* indicator for each algorithm. We start with 3*OBJ* performance charts shown in Fig. 3a. Each algorithm *IBEA*, *MOEAD*, *NSGA*3, *NSGA*2, and *SPEA* has its respective colors (Red, Grey, Blue, Black, Green). Same as our findings in Table 7, *NSGA*3, *IBEA*, and *SPEA*2 are in the lead when looking at the end of the graph. *NSGA*3 was able to maintain its superiority from the beginning, while both *IBEA* and *SPEA*2 lacked behind in the first third of the experiments. This behavior changes when looking at 5*OBJ* graph in Fig. 3b. *SPEA*2 keeps the lead from the start compared to other algorithms. From both experiments, we can see that most algorithms are not improving at the same level as the beginning of the experiments. This indicates a stagnation state and algorithms aren't likely to improve considerably.

Considering production scenarios and after examining Fig. 3 findings show that around five minutes of computing time, good results are obtained. While indeed most approaches continue to improve after that period of time but it doesn't justify the additional computing. The company can quickly update the recommendations for users. However, one major drawback of our approach is the fact that modeling solutions as lists of recommendations for each user does not guarantee that our model will always converge in five minutes with the increase of users or items in the catalog. This scalability issue can be improved by clustering users and providing recommendations per groups instead of individual users, this is considered in future work. At last, these initial findings indicate that for both experiments (3*OBJ* & 5*OBJ*), two algorithms *NSGA*3 and *SPEA*2 possess a robust performance and are the best fit for this task compared to the rest.

Selecting the best solution can be complicated specially since these objectives are conflicting with each other. Multi-Criteria Decision Making (MCDM) deals with such decision problems. Among many methods we selected Pseudo Weights (PS) which calculates the normalized distance to the worst solution regarding each objective *i* [8]. The following equation provides the pseudo weight w_i for the *i*-th objective.

$$w_i = \frac{(f_i^{max} - f_i(x))/(f_i^{max} - f_i^{min})}{\sum_{m=1}^{M}(f_m^{max} - f_m(x))/(f_m^{max} - f_m^{min})} \tag{6}$$

The steps are rather simple, first we get nadir (ideal) points from Pareto front. We proceed to calculate the normalized distance to the worst solution for each objective w_i. Finally we find the closest solution to the normalized distance. Other methods aside from Pseudo Weights can be used, such as the high trade-off method [7] and Compromise Programming [23].

Platform managers are provided a graphical interface to either selected already defined profiles (balanced objectives, high relevance, etc) or tune the weights assigned to each selected objective using a simple slider. The end-user will receive the first solution returned after the MCDM phase.

6 CONCLUSION AND FUTURE WORKS

In this article, we solve a many-objective recommendation problem at Mandarine Academy. The approach is applied on e-learning platforms and took advantage of real-world interactions to better understand user behavior and identify key points for user interface/experience improvement.

Studying related works and following company guidelines, we mathematically formulated our objectives into a Multi-Objective Combinatorial Optimization Problem (MOCOP). With five objectives (*Similarity*, *Diversity*, *Novelty*, *RMSE*, and *nDCG*@5). Our proposed approach focuses on personalization of recommendations by providing each user

Table 7. Performance comparison (**average best value** and the <u>standard deviation</u>) using implicit interactions and 3 Objectives for over 30 independent runs.

Algorithm	HV_{3OBJ}	GD	IGD	EP
NSGA2	0.77 <u>0.05</u>	1.23 <u>0.002</u>	1.019 <u>0.016</u>	0.147 <u>0.024</u>
NSGA3	**0.91** <u>0.04</u>	1.24 <u>0.001</u>	1.018 <u>0.018</u>	0.101 <u>0.025</u>
SPEA2	0.80 <u>0.03</u>	1.24 <u>0.0007</u>	**1.002** <u>0.035</u>	0.146 <u>0.025</u>
MOEAD	0.68 <u>0.07</u>	1.23 <u>0.013</u>	1.076 <u>0.059</u>	0.233 <u>0.042</u>
IBEA	0.85 <u>0.06</u>	**1.22** <u>0.014</u>	1.070 <u>0.021</u>	**0.093** <u>0.037</u>

Table 8. Performance comparison (**average best value** and the <u>standard deviation</u>) using implicit interactions and 5 Objectives for over 30 independent runs.

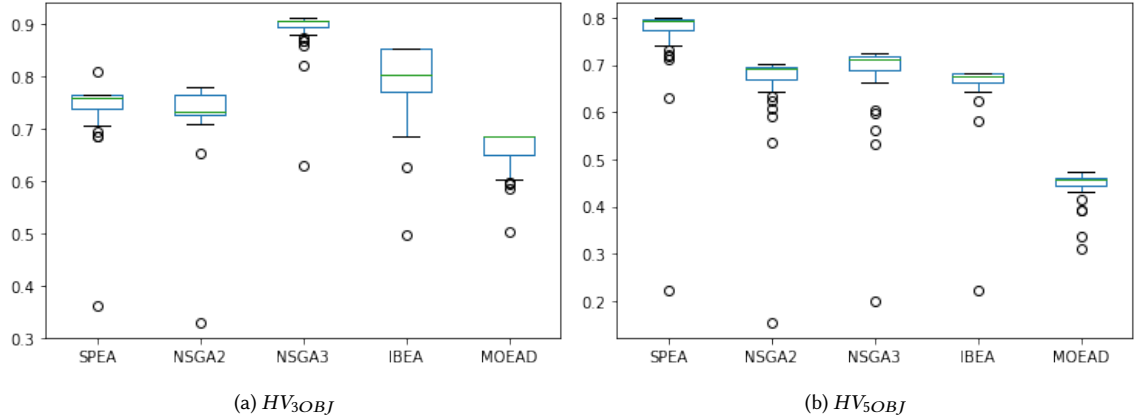| Algorithm	HV_{5OBJ}	HV_{3OBJ}	GD	IGD	EP
NSGA2	0.74 <u>0.05</u>	0.84 <u>0.034</u>	**1.41** <u>0.003</u>	**1.06** <u>0.03</u>	0.07 <u>0.04</u>
NSGA3	0.79 <u>0.03</u>	0.84 <u>0.037</u>	1.44 <u>0.0006</u>	1.07 <u>0.03</u>	0.10 <u>0.08</u>
SPEA2	**0.81** <u>0.06</u>	**0.85** <u>0.026</u>	1.43 <u>0.004</u>	1.12 <u>0.02</u>	**0.06** <u>0.01</u>
MOEAD	0.47 <u>0.02</u>	0.57 <u>0.029</u>	1.45 <u>0.03</u>	1.31 <u>0.07</u>	0.29 <u>0.02</u>
IBEA	0.68 <u>0.04</u>	0.74 <u>0.018</u>	1.46 <u>0.01</u>	1.17 <u>0.01</u>	0.15 <u>0.0008</u>

Fig. 2. Box-plot of HV indicator for both $3OBJ$ and $5OBJ$ experiments using all algorithms (30 Executions).

the items that match their profile and ratings while emphasizing on novelty and diversity. By evaluating different evolutionary algorithms using real-world user data we were able to find best performing approaches considering different test settings.

Existing users may benefit from models that generates diversified or novel items to further explore the catalog, while new users may receive recommendations created by emphasizing ratings and ranking. The freedom of selecting what goals to prioritize is provided for platform managers in a user-friendly interface. Future graphical improvements insists on the same principles of readability, ease of use and availability of interactions.

The most time-consuming part of our approach is performed entirely offline, where chosen objectives are trained on user data and served online through Application Programming Interfaces (APIs). One major drawback is the scalability
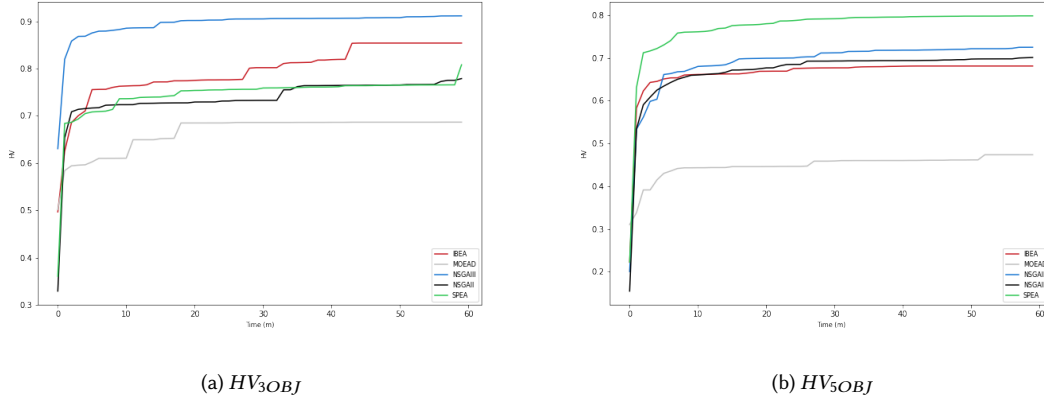
(a) HV_{3OBJ} (b) HV_{5OBJ}

Fig. 3. Evolution of the HV indicator (Y-axis) for both $3OBJ$ and $5OBJ$ over 1 hour of computing time (X-axis) using all algorithms (30 Executions).

of the proposed system when the user base and catalog are expanding, training times can be significantly affected. Also, exploring the possibility for users to indicate their objective preferences which will be taken into account when updating the model is considered in future work.

The integration of Mandarine Academy Recommender System *MARS* is currently underway and will include an administration dashboard specific to each platform owner for managing *MARS* and monitoring performance.

REFERENCES

[1] Bushra Alhijawi and Yousef Kilani. 2020. A collaborative filtering recommender system using genetic algorithm. *Information Processing & Management* 57, 6 (2020), 102310.

[2] Antonio Benitez-Hidalgo, Antonio J Nebro, Jose Garcia-Nieto, Izaskun Oregi, and Javier Del Ser. 2019. jMetalPy: A Python framework for multi-objective optimization with metaheuristics. *Swarm and Evolutionary Computation* 51 (2019), 100598.

[3] Robin Burke. 2002. Hybrid recommender systems: Survey and experiments. *User modeling and user-adapted interaction* 12, 4 (2002), 331–370.

[4] Zheng-Yi Chai, Ya-Lun Li, Ya-Min Han, and Si-Feng Zhu. 2018. Recommendation system based on singular value decomposition and multi-objective immune optimization. *IEEE Access* 7 (2018), 6060–6071.

[5] Li Chen and Ho Keung Tsoi. 2011. Users' decision behavior in recommender interfaces: Impact of layout design. In *RecSys' 11 Workshop on Human Decision Making in Recommender Systems*.

[6] Carlos A Coello Coello, Clarisse Dhaenens, and Laetitia Jourdan. 2010. Multi-objective combinatorial optimization: Problematic and context. In *Advances in multi-objective nature inspired computing*. Springer, 1–21.

[7] Olivier L De Weck. 2004. Multiobjective optimization: History and promise. In *Invited Keynote Paper, GL2-2, The Third China-Japan-Korea Joint Symposium on Optimization of Structural and Mechanical Systems, Kanazawa, Japan*, Vol. 2. 34.

[8] Kalyanmoy Deb. 2011. Multi-objective optimisation using evolutionary algorithms: an introduction. In *Multi-objective evolutionary optimisation for product design and manufacturing*. Springer, 3–34.

[9] Kalyanmoy Deb, Samir Agrawal, Amrit Pratap, and Tanaka Meyarivan. 2000. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In *International conference on parallel problem solving from nature*. Springer, 849–858.

[10] Kalyanmoy Deb and Himanshu Jain. 2013. An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: solving problems with box constraints. *IEEE transactions on evolutionary computation* 18, 4 (2013), 577–601.

[11] Amina Debbah and Yamina Mohamed Ben Ali. 2014. Solving the curriculum sequencing problem with dna computing approach. *International Journal of Distance Education Technologies (IJDET)* 12, 4 (2014), 1–18.

[12] Aurora Esteban, Amelia Zafra, and Cristóbal Romero. 2018. A Hybrid Multi-Criteria Approach Using a Genetic Algorithm for Recommending Courses to University Students. *International Educational Data Mining Society* (2018).

[13] Aurora Esteban, Amelia Zafra, and Cristóbal Romero. 2020. Helping university students to choose elective courses by using a hybrid multi-criteria recommendation system with genetic optimization. *Knowledge-Based Systems* 194 (2020), 105385.

[14] Reinaldo Silva Fortes, Daniel Xavier de Sousa, Dayanne G Coelho, Anisio M Lacerda, and Marcos A Gonçalves. 2021. Individualized extreme dominance (IndED): A new preference-based method for multi-objective recommender systems. *Information Sciences* 572 (2021), 558–573.

[15] Ken Goldberg, Theresa Roeder, Dhruv Gupta, and Chris Perkins. 2001. Eigentaste: A constant time collaborative filtering algorithm. *information retrieval* 4, 2 (2001), 133–151.

[16] Mounir Hafsa, Pamela Wattebled, Julie Jacques, and Laetitia Jourdan. 2021. A Multi-Objective Evolutionary Approach to Professional Course Timetabling: A Real-World Case Study. In *2021 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 997–1004.

[17] Alain Hertz and Marino Widmer. 2003. Guidelines for the use of meta-heuristics in combinatorial optimization. *European Journal of Operational Research* 151, 2 (2003), 247–252.

[18] John H Holland. 1973. Genetic algorithms and the optimal allocation of trials. *SIAM J. Comput.* 2, 2 (1973), 88–105.

[19] Li Huang, Yi-feng Yang, and Lei Wang. 2017. Recommender engine for continuous-time quantum Monte Carlo methods. *Physical Review E* 95, 3 (2017), 031301.

[20] Nicolas Hug. 2020. Surprise: A Python library for recommender systems. *Journal of Open Source Software* 5, 52 (2020), 2174. https://doi.org/10.21105/joss.02174

[21] Hisao Ishibuchi, Ryo Imada, Yu Setoguchi, and Yusuke Nojima. 2016. Performance comparison of NSGA-II and NSGA-III on various many-objective test problems. In *2016 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 3045–3052.

[22] Michael Jugovac, Dietmar Jannach, and Lukas Lerche. 2017. Efficient optimization of multiple recommendation quality factors according to individual user tendencies. *Expert Systems with Applications* 81 (2017), 321–331.

[23] Michael Lindahl. 2017. *Strategic, Tactical and Operational University Timetabling*. Ph. D. Dissertation. University of Denmark.

[24] Manuel López-Ibáñez, Jérémie Dubois-Lacoste, Leslie Pérez Cáceres, Mauro Birattari, and Thomas Stützle. 2016. The irace package: Iterated racing for automatic algorithm configuration. *Operations Research Perspectives* 3 (2016), 43–58.

[25] Behzad Soleimani Neysiani, Nasim Soltani, Reza Mofidi, and Mohammad Hossein Nadimi-Shahraki. 2019. Improve performance of association rule-based collaborative filtering recommendation systems using genetic algorithm. *Int. J. Inf. Technol. Comput. Sci* 11, 2 (2019), 48–55.

[26] Vilfredo Pareto. 1896. *Cours d'economie Politique*. Vol. 1. F. Rouge.

[27] Michael J Pazzani and Daniel Billsus. 2007. Content-based recommendation systems. In *The adaptive web*. Springer, 325–341.

[28] Marco Tulio Ribeiro, Nivio Ziviani, Edleno Silva De Moura, Itamar Hata, Anisio Lacerda, and Adriano Veloso. 2014. Multiobjective pareto-efficient approaches for recommender systems. *ACM Transactions on Intelligent Systems and Technology (TIST)* 5, 4 (2014), 1–20.

[29] Tiago Sousa, Hugo Morais, Rui Castro, and Zita Vale. 2016. Evaluation of different initial solution algorithms to be used in the heuristics optimization to solve the energy resource scheduling in smart grids. *Applied Soft Computing* 48 (2016), 491 – 506. https://doi.org/10.1016/j.asoc.2016.07.028

[30] Saúl Vargas. 2014. Novelty and diversity enhancement and evaluation in recommender systems and information retrieval. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*. 1281–1281.

[31] Wesley Waldner and Julita Vassileva. 2014. Emphasize, don't filter! displaying recommendations in twitter timelines. In *Proceedings of the 8th ACM Conference on Recommender systems*. 313–316.

[32] Pan Wang, Xingquan Zuo, Congcong Guo, Ruihong Li, Xinchao Zhao, and Chaomin Luo. 2017. A multiobjective genetic algorithm based hybrid recommendation approach. In *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE, 1–6.

[33] Shanfeng Wang, Maoguo Gong, Haoliang Li, and Junwei Yang. 2016. Multi-objective optimization for long tail recommendation. *Knowledge-Based Systems* 104 (2016), 145–155.

[34] Shanfeng Wang, Maoguo Gong, Lijia Ma, Qing Cai, and Licheng Jiao. 2014. Decomposition based multiobjective evolutionary algorithm for collaborative filtering recommender systems. In *2014 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 672–679.

[35] Ruobing Xie, Yanlei Liu, Shaoliang Zhang, Rui Wang, Feng Xia, and Leyu Lin. 2021. Personalized approximate pareto-efficient recommendation. In *Proceedings of the Web Conference 2021*. 3839–3849.

[36] Qingfu Zhang and Hui Li. 2007. MOEA/D: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on evolutionary computation* 11, 6 (2007), 712–731.

[37] Eckart Zitzler and Simon Künzli. 2004. Indicator-based selection in multiobjective search. In *International conference on parallel problem solving from nature*. Springer, 832–842.

[38] Eckart Zitzler, Marco Laumanns, and Lothar Thiele. 2001. SPEA2: Improving the strength Pareto evolutionary algorithm. *TIK-report* 103 (2001).

[39] Eckart Zitzler, Lothar Thiele, Marco Laumanns, Carlos M Fonseca, and Viviane Grunert Da Fonseca. 2003. Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Transactions on evolutionary computation* 7, 2 (2003), 117–132.

[40] Yi Zuo, Maoguo Gong, Jiulin Zeng, Lijia Ma, and Licheng Jiao. 2015. Personalized recommendation based on evolutionary multi-objective optimization [research frontier]. *IEEE Computational Intelligence Magazine* 10, 1 (2015), 52–62.