Soft Computing (2021) 25:11019–11034
https://doi.org/10.1007/s00500-021-05836-9

APPLICATION OF SOFT COMPUTING

# A comparative analysis of metaheuristics applied to adaptive curriculum sequencing

André Ferreira Martins[1] · Marcelo Machado[2] · Heder Soares Bernardino[3] · Jairo Francisco de Souza[4]

Accepted: 20 April 2021 / Published online: 6 May 2021


## Abstract

The effective adoption of online learning depends on user satisfaction as distance education approaches suffer from a lack of commitment that may lead to failures and dropouts. The adaptive learning literature argues that an alternative to achieve student satisfaction is to treat them individually, delivering the educational content in a personalized manner. In addition, the sequencing of this content—called Adaptive Curriculum Sequencing (ACS)—is important to avoid cognitive overload and disorientation. The search for an optimal sequence from ever-growing databases is an NP-Hard combinatorial optimization problem. Although some approaches have been proposed, it is challenging to assess their contributions due to the lack of benchmark data available. This paper presents a procedure to create synthetic dataset to evaluate ACS approaches and, as a concept proof, analyzes metaheuristics usually used in ACS approaches: Genetic Algorithm, Particle Swarm Optimization (PSO) and Prey–Predator Algorithm using student's learning goals and their extrinsic and intrinsic information. We also propose an approach based on Differential Evolution (DE). The computational experiments include synthetic datasets with a varied amount of learning materials and real-world datasets for comparison. The results show that DE performed better than the other methods when less than 500 learning materials are used while PSO performed better for larger problems.

**Keywords** Learning path · Adaptive learning · Intelligent tutoring system · Evolutionary computing · Curriculum sequencing · Soft computing

## 1 Introduction

In the last century, we are experiencing a great movement toward the use of online learning systems to meet the most diverse information needs in an ever-growing demanding society. This phenomenon is driven by technological development, which has enabled access to quality information through increasingly powerful and affordable devices. With computer and communication technologies, teachers and students become spatially and temporally dispersed. The time and physical boundaries of the traditional classroom are stretched to a learning space (Khalifa and Lam 2002). Thus, several universities and private companies already provide online courses for their students or employees.

Particularly in 2020, coronavirus emergency is rebuilding social structures and has speed up the shift from a physical space of social interaction to a digital space. It has raised the need for a transformation in education, which is intensively supported by educational technology (Williamson et al. 2020). In addition to enhancing traditional teaching-learning methods, information technology can also enable

✉ Jairo Francisco de Souza
jairo.souza@ice.ufjf.br

André Ferreira Martins
andre.martins@ice.ufjf.br

Marcelo Machado
marcelo.machado@edu.unirio.br

Heder Soares Bernardino
heder@ice.ufjf.br

[1] Postgraduate Program in Computer Science, Federal University of Juiz de Fora, Juiz de Fora, Minas Gerais 36036-900, Brazil

[2] Postgraduate Program in Informatics, Federal University of the State of Rio de Janeiro, Rio de Janeiro 22290-255, Brazil

[3] Department of Computer Science, Federal University of Juiz de Fora, Juiz de Fora, Minas Gerais 36036-900, Brazil

[4] LApIC Research Group, Federal University of Juiz de Fora, Juiz de Fora, Minas Gerais 36036-900, Brazil

new ways of content delivery and innovative pedagogic strategies. Therefore, more and more, the effective use of e-learning systems depends on solutions that guarantee students' satisfaction and learning outcome since online education modality suffer from problems, such as, distraction, disorientation, cognitive overload and lack of commitment that may lead to failures and drop-outs (Davis et al. 2016).

The Adaptive Learning and Intelligent Tutoring System (ITS) literature shows that learning environment must be aware of learners' attributes, such as background, needs, intents and preferences in order to provide easy and effective understanding (Machado et al. 2020; Rathore and Arjaria 2020). However, most e-learning systems provide "one size fits all" environments where all the learners are treated the same way in terms of learning materials, and are self-guided with limited instructor support (Kardan et al. 2015). Besides, with digitization, a rapid growth is seen in educational technology and different formal and informal learning contents are available on the internet. This huge amount of information can lead student and instructors to cognitive overload and disorientation (Debbah and Ali 2014).

A well-known problem in adaptive learning is the Adaptive Curriculum Sequencing (ACS) (Machado et al. 2020; Al-Muhaideb and Menai 2011). It is considered a crucial issue in personalized learning as its purpose is to find the best sequence of learning materials that meets the student profile (Niknam and Thulasiraman 2020; Machado et al. 2019; Muhammad et al. 2016). The challenge of ACS lies in automating the process, since unsuitable sequences may not offer learning materials that meet the student' learning goals and profile (Xie et al. 2017). Previous works showed that the automatic search for an optimal ACS from ever-growing databases is a combinatorial NP-Hard problem (Pushpa 2012; Chang and Ke 2013; de Marcos et al. 2008b). Moreover, the process of adapting at the curriculum sequencing level relies heavily on the parameters used—recently, several papers have explored solutions using a variety of those parameters (Machado et al. 2020). This adds even more complexity to the search for an optimal solution as these parameters define the quality of the relationship between the student and the learning materials in a sequence. Since solutions to large ACS instances can only be approximated, soft computing methods can be considered to approximate its solutions. Even though ACS problem is longstanding, researchers still find it an important problem to be approached in different ways seeking to infer even better results.

Al-Muhaideb and Menai (2011) reviewed publications, from the 2002 to 2009, who used evolutionary computation approaches to the ACS problem. That work revealed the increasing use of metaheuristics for the problem, highlighting Genetic Algorithm (GA), Particle Swarm Optimization (PSO) and Ant Colony Optimization (ACO) according to different problem modeling. Many of the works only present the validity of using a metaheuristic considering that their proposed method converges. For instance, a Prey–Predator Algorithm (PPA) was proposed in (Machado et al. 2019) to address the ACS problem assuming that such metaheuristics outperform GA and PSO (two commonly metaheuristics used for ACS problem) in benchmark problems. However, the authors focused on pedagogical results and no comparison among the approaches were performed using the problem depicted here. Others studies, based on common formulations, seek to compare their approach against others, for instance: PSO vs GA (Li et al. 2012), PSO vs GA vs ACO vs Immune Algorithm (Wan and Niu 2016) and different implementations of PSO and/or ACO (Menai et al. 2018; Chandar et al. 2010). In either case, the studies consider its own dataset for experimentation and in most works use synthetic data without presenting implementation details, as well as not providing the datasets—compromising reproducibility and making it difficult to compare different works.

Given the previously mentioned problems and considering that new metaheuristics have been introduced in recent years, the contributions of this paper are as follows. We proposed a method to create synthetic datasets to evaluate the ACS proposals. Also, the datasets were made available for further research. In addition, we present an experimental comparative analysis of four approaches to address the ACS problem: GA, PSO, PPA, and our proposed DE implementation. The computational experiments were carried out using synthetic datasets with different amounts of learning materials and a real-world dataset. These problem sizes allows for a performance analysis facing different situations. The results show that the algorithms have similar results with both real and synthetic datasets. Also, DE has the best performance in instances with less than 500 learning materials. On the other hand, PSO is a better option for larger instances.

The remaining of this paper is organized as follows: Sect. 2 describes the problem addressed by this work; Sect. 3 reviews related works; Sect. 4 presents the modeling that we adopted; Sect. 5 presents the proposed procedure for generating datasets with learning materials; Sect. 6 describes the implementation of the metaheuristics; Sect. 7 shows the experiments carried out and the results obtained; finally, Sect. 8 concludes the paper.

## 2 Adaptive curriculum sequencing

The research and development of ITS seeks to combine techniques of Artificial Intelligence, Cognitive Psychology and Educational Learning Theories toward learning environment systems able to know what to teach, to whom to teach and how to teach (Nwana 1990; Silva et al. 2018). The problem of selecting the optimal sequence of learning materials which considers the student's individuality is one of the most inter-

esting and crucial goals in Adaptive Learning (Muhammad et al. 2016). There is not yet a consensus related to the term that defines this sequence (Hnida et al. 2016) and, although other terminologies might be used, we adopted Adaptive Curriculum Sequencing.

The goal of ACS is to provide the student with the most suitable ordering of knowledge units and learning tasks (examples, questions, problems, etc.) to work with (Brusilovsky 2003). The intention is to help the student find an "optimal learning path" within the knowledge domain (Hafidi and Bensebaa 2015); therefore, maximizing understanding as well as learning efficiency. According to Ahmad A Kardan et al. (Kardan et al. 2015) the ACS should take into account the student characteristics and the learning materials information.

The concept map is an important attribute for ACS. It can be considered a representation of the programmatic content containing the interconnections between the concepts addressed in a course (de Marcos et al. 2011; Sharma et al. 2012). Three main approaches were used for concept map construction in literature: (1) approximated from mathematical methods (2) pre-defined by experts and (3) based on ontologies. The mathematical methods are used to approximate the concept map automatically and decentrally (Seki et al. 2005; Chen 2008; Huang et al. 2007; Guo and Zhang 2009). However, these methods ignore the relationships between concepts, making it necessary to evaluate their use in the ACS problem since illogical sequences can be produced (Chen 2008). The concept map construction based on expert experience is common and well accepted, but it still has some disadvantages since it is a costly labor, depends on the experience of those involved and it is not flexible for students. Finally, the ontology based approach, seeks to associate semantics with the structure of concepts. It is worth mentioning that the automatic construction of the concept map is also a relevant issue in adaptive learning field (Gutiérrez and Pardo 2007).

The ACS problem can be formulated as either a constraint satisfaction problem (CSP) (e.g., (de Marcos et al. 2008a, 2009)) or a multi-objective optimization problem (e.g., (Chu et al. 2009; Gao et al. 2015)) (Al-Muhaideb and Menai 2011). In this paper, we define the ACS problem as a function $f(u, l, c) \rightarrow S$ that receives as parameters the user model (student representation) $u$, the learning material information $l$ and the concept map structure $c$. This function returns a sequence $s \in S$ that best approximates the student's model among various sequence possibilities contained in $S$.

In Fig. 1, an example of a concept map with a set of concepts and materials is illustrated. Each concept is represented by a green circle, and each learning object is represented by a blue rectangle. Concepts can have prerequisites, which is indicated by an arrow connecting these concepts. For example, the concept C2 depends on concept C1 and C2 is a prerequisite for concepts C4 and C5. The materials can cover one or more concepts. For example, LO2 cover concepts C2 and C5. One of the objectives of ACS is to select a set of materials that not only cover the desired concepts but also fits the learning structure described by the concept map. For example, if one aims at covering the concept C3, it is necessary to select materials that cover the concepts C1 and C5, which in turn requires C2.

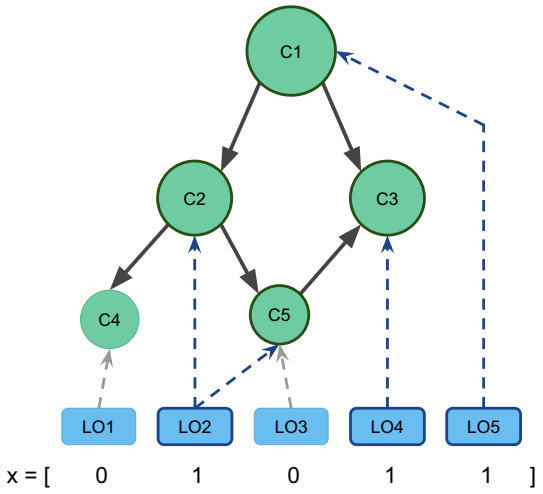



**Fig. 1** Example of a concept map

After defining what are the concepts that need to be taught, the ACS process consists of searching for the best sequence of materials that meet the students' need. An example is to select materials LO2, LO4, and LO5. Note that considering only the objective of concept coverage, multiple solutions can be found. Depending on the student's needs and duration requirements, LO3 material can also be included in the sequence to reinforce the content of the concept C5.

Selecting a proper sequence of learning materials is a challenge as unsuitable selection can bring unexpected results, increasing the dropout and failure rates (Xie et al. 2017). This automatic selection is a difficult task, especially because the variety of learning materials in online repositories is rather expansive. Moreover, several constraints related to these learning materials and student's model are involved in the adaptation process, making the decision process even more complex. Finding an optimal curriculum sequence is a combinatorial problem falling in the NP-Hard class of problems (Pushpa 2012; Acampora et al. 2011). In a course with various constraints like prerequisite relations, fixed-order sequence for some itinerary concepts, etc., a feasible sequence consists of the $C$ concepts arranged in a way satisfying all constraints, and the total number of possible (valid and invalid) sequences (permutations) approaches $C!$ (de Marcos et al. 2008b). This problem becomes even harder with a solution space that is much larger in a realistic e-Learning situation where we consider a student's background, his/her

learning style and similar student-related factors. Thus, several researchers were motivated to use artificial intelligence techniques, especially metaheuristics, to deal with the problem of automatically selecting an optimal Adaptive Curriculum Sequencing (from now on ACS problem), after all in a classical manner, they are employed to solve similar problems (Khamparia and Pandey 2015; Pushpa 2012; Al-Muhaideb and Menai 2011).

## 3 Related works

ACS has been addressed in several papers (Wong and Looi 2010; Pushpa 2012; Al-Muhaideb and Menai 2011; Khamparia and Pandey 2015; Muhammad et al. 2016). In an investigative way, several parameters were used, either in relation to the student or to the knowledge domain (i.e, learning materials).

ACS approaches can be divided in two groups: Individual Sequencing and Social Sequencing. Individual Sequencing approaches consider only the student's own parameter information, i.e., the ACS selection ignores any other users information and interactions among the student. In contrast, Social Sequencing approaches consider experiences of previous students to benefit current students. Evolutionary approaches have been largely used to solve the ACS problem, and among all, Ant Colony Optimization (ACO) and Genetic Algorithms (GA) stand out (Al-Muhaideb and Menai 2011). It is possible to perceive a relation with the sequencing type and evolutionary approach, where in most papers, ACO is related to Social Sequencing, just as GA is related to Individual Sequencing. This behavior is related to the basis of these metaheuristics. GA usually finds the best population of learning materials for a student and in ACO the behavior of the students (ants) interfere in the way of the others. This relationship shows a trend when one approach is used over another.

Several works evaluate their proposed solution by intrinsic evaluations, i.e., checking if the chosen metaheuristic can converge to coherent solutions, observing fitness values (Machado et al. 2019; de Marcos et al. 2011; Chu et al. 2009; de Marcos et al. 2009; Seki et al. 2005; Huang et al. 2007). Other common method is to compare the execution time according to the number of learning materials in a repository. For instance, an experiment comparing PSO and GA in a multi-objective formulation of ACS problem is presented in (Li et al. 2012). They showed that the fitness values of PSO are close to those of GA concerning the average fitness value of 100 independent runs, but GA has more user-defined parameters than PSO. They also presented a comparison of the number of generations and execution time according to the increasing amount of learning materials. When the number of learning materials is less than 300, the number of generations and execution time of PSO are less than those of GA. However, when the number of learning materials is larger than 300, the GA approach performs better than the PSO implementation. The same behavior was reported in (Christudas et al. 2018) when comparing implementations of GAs and PSO.

However, it is challenging to evaluate ACS approaches due to the lack of available datasets or benchmarks. Few studies indicate or make available the data used in the evaluation process. In (Menai et al. 2018), the performance of the solutions found was evaluated in a real e-learning environment on real data from an information technology diploma program and 2,000 learners selected randomly at Buraydah College of Technology[1]. The data were gathered from the Learner Affairs System. A dataset of 10,000 learners from the anonymized Open University Learning Analytics Dataset (OULAD)[2] was used in (Agarwal et al. 2016), in which contains data of courses, students and their interactions with Virtual Learning Environment for seven selected courses (Kuzilek et al. 2017). In many cases, studies have created their own dataset and provide information about them in the body of the paper. For instance, a list of 6 learning materials and its prerequisites and outcomes was described in (Shmelev et al. 2015). However, in most cases the studies do not provide sufficient data to reproduce its experiments.

We present an individual sequencing approach for ACS that takes into account different variables, such as the information of the students (previous knowledge, time availability, learning preferences), learning materials (difficulty, content, and style), and the course (target concepts). Unlike other works, the evaluation is performed here using a larger number of search techniques: four metaheuristics. Also, all data and methods used in the comparative analysis process are freely available for further research.

## 4 Modeling the problem

As a single objective may not adequately represent the ACS problem, another approach is to model it with multiple objectives (Muhammad et al. 2016). These objectives depend on student representation, learning material information and the concept map structure. In this work, the objective function $f(x)$ is defined as the weighted sum of five objectives $O_i(x)$ which are: $O_1$, concept coverage; $O_2$, students ability; $O_3$, expected course duration; $O_4$, materials balancing between concepts; and $O_5$, adequacy to the student's preference. They

[1] www.tvtc.gov.sa.

[2] https://analyse.kmi.open.ac.uk/open_dataset.

are defined as

$$f(\mathbf{x}) = \min \sum_{i=1}^{5} \omega_i O_i(\mathbf{x}) \tag{1}$$

$$O_1(\mathbf{x}) = (1-\rho)(|R(\mathbf{x})| - |R(\mathbf{x}) \cap E|) + \rho(|E| - |R(\mathbf{x}) \cap E|) \tag{2}$$

$$O_2(\mathbf{x}) = \sum_{i=1}^{|M|} x_i \left| D^{m_i} - \frac{\sum_{j=1}^{|C|} R_{c_j}^{m_i} E_{c_j} H_{c_j}}{\sum_{j=i}^{|C|} R_{c_j}^{m_i} E_{c_j}} \right| \frac{1}{\sum_{i=1}^{|M|} x_i} \tag{3}$$

$$O_3(\mathbf{x}) = \max\left(T^{\downarrow} - \sum_{i=1}^{|M|} T^{m_i} x_i, 0\right) + \max\left(0, \sum_{i=1}^{|M|} T^{m_i} x_i - T^{\uparrow}\right) \tag{4}$$

$$O_4(\mathbf{x}) = \sum_{j=1}^{|C|} E_{c_j} \left| \sum_{i=1}^{|M|} x_i R_{c_j}^{m_i} - \frac{\sum_{i'=1}^{|M|} \sum_{j'=1}^{|C|} x_{i'} R_{c_{j'}}^{m_{i'}} E_{c_{j'}}}{\sum_{j'=1}^{|C|} E_{c_{j'}}} \right| \tag{5}$$

$$O_5(\mathbf{x}) = \sum_{k=1}^{4} \frac{\sum_{i=1}^{|M|} x_i \left| 3sgn(\theta_k^{m_i}) - L_k \right|}{4 \sum_{i=1}^{|M|} x_i} \tag{6}$$

where $\omega_i$ are the weights associated to the objective $O_i$, $\mathbf{x} = (x_1, x_2, \ldots, x_{|M|})$ is the vector of the binary design variables, $M$ is the set of learning materials, $m_i \in M$ represents the $i$-th learning material, $|M|$ the number of materials, and $c_j \in C$ represents the $j$-th concept of a course. The objectives and their components are detailed in the following paragraphs.

Objective $O_1(\mathbf{x})$ checks whether the course concepts covered by the selected sequence meet the student's learning goals, where $R(\mathbf{x})$ represents the set of concepts covered by all learning materials present in vector $\mathbf{x}$, $E$ represents a set of learning goals expected by the student, and $\rho \in [0, 1]$. This function associates penalties to the quantity of spare concepts (first part of the equation) and the number of missing concepts (second part of the equation) according to student's learning goals. Let $R^{m_i}$ be the concepts covered by a learning material $m_i$, function $R(\mathbf{x})$ is defined as:

$$R(\mathbf{x}) = \bigcup_{i=1}^{|M|} \{R^{m_i} | x_i = 1\} \tag{7}$$

The learning material difficulty is compared in $O_2(\mathbf{x})$ to the average of the student's ability in the concepts addressed by such learning material and that are included in the student's learning goals, where $D^{m_i}$ represents the difficulty associated to a learning material $m_i$, $R_{c_j}^{m_i}$ indicates whether the learning material cover a concept $c_j$. Thus, $R_{c_j}^{m_i} = 1$ if the learning material cover the concept $c_j$ and $R_{m_i}^{c_j} = 0$ otherwise. $E_{c_j}$ indicates whether a concept $c_j$ is in accordance with the student's learning goals. Thus, $E_{c_j} = 1$ if the concept $c_j$ is expected by the student, and $E_{c_j} = 0$, otherwise. Finally, $H_{c_j}$ represents the current ability level of the student with respect to the concept $c_j$.

Objective $O_3(\mathbf{x})$ represents the deviation of the time estimated and that expected by the student for the course. Thus, $O_3(\mathbf{x})$ checks whether the total time of the course is between the lower and upper limits, where $T^{\downarrow}$ represents the lower and $T^{\uparrow}$ the upper bounds times expected by the student. Besides, $T^{m_i}$ represents the estimated learning time of a learning material $m_i$ (in hours).

The balancing of the selected learning materials is evaluated in $O_4(\mathbf{x})$. This function returns a low value as more concepts are covered by a similar amount of learning materials.

Finally, $O_5(\mathbf{x})$ measures the preference of the user for the material type where the preference is modeled as a 4-dimensional vector. The preference vectors can be calculated using a number of techniques such as questionnaires, explicit or implicit feedback methods. In $O_5(\mathbf{x})$, for each dimension $k = 1, \ldots, 4$, $\theta_k^{m_i}$ indicates which preference is attended by the $i$-th learning material, and $L_k \in [-3, 3]$ represents the student intensity for this dimension (Machado et al. 2019). In this work, the student's preferences are defined by the student's learning style and it is linked to the characteristics of the learning materials according to the Felder and Silverman Learning Style Model (FSLSM) (Felder et al. 1988). The learning style model was used here as a proof of concept. In this model, the student is characterized in four dimensions representing different aspects of how the student learns best, with values representing the intensity of the characteristics as follows: in the dimension Processing, the student may prefer materials that are more reflexive or more active; in the dimension Perception, it describes if the student is more intuitive or more sensing; in the dimension Input, the values represent if the student is more verbal or more visual; and, finally, the dimension Understanding specify if the students prefer to be explained using a global approach or a more sequential approach.

## 5 Proposed procedure for generating learning materials

Performing tests with real data presents some issues. It is necessary to organize a course with several students hav-

ing a large amount of properly classified learning materials. In addition, problems such as student dropout and uneven distribution of students and learning materials among learning style profiles require even larger amounts of data. To work around these problems, this work proposes a method for generating synthetic data based on features extracted from a real-world dataset.

The real-world dataset was obtained as part of a course on Computer Science Fundamentals held in 2017 remotely on the Moodle platform. The concepts taught were distributed over a six week period. Each week the students received a sequence of learning materials covering the expected concepts and were evaluated at the end of the week. The course consisted of 284 learning materials covering a total of 30 separate concepts across eight subjects. Material sequencing was performed as part of an ITS application and was applied to a group of 61 students. However, as the assessments were performed weekly, the number of students for each week might be smaller. The data used were the learning style profile, the student's skill level in each of the concepts and the time available for the course.

For the assessment of students learning style, it was applied a questionnaire based on the Index of Learning Style (ILS) (Solomon and Felder 1999) where each dimension of the learning style model were mapped to the interval [-3, 3]. Students prior knowledge was determined by applying a student self-assessment. Then three questions were delivered: one lower level, one equal level and one higher level than the student statement.

The data collected from this course were used to create a synthetic dataset. The purpose of the synthetic dataset is to address the lack in the literature of comparative assessments of the performance of metaheuristics using a large amount of learning materials. For the creation, we considered characteristics as distribution of material's difficulty, average learning time of materials, amount of concepts covered by each material and the most appropriate learning style profile. The learning materials were divided according to their level of difficulty, each group was analyzed, and the observed characteristics were used to create the new dataset.

The main characteristics obtained from this data collection were: most learning materials cover only few concepts (Fig. 2a), and there is a correlation between the difficulty and the learning time of materials (Fig. 2c). The learning time follows an exponential distribution for all difficulties, but the easiest materials are generally shorter than the most difficult. The other features follow a normal distribution, as shown in Fig. 2b and d .

We propose here the following steps for generating each new learning material. First, the number of concepts the material covers is chosen. This value is randomly obtained according to the same distribution of the real dataset (Fig. 2a), and the probabilities are: 78.17% for one concept, 13.73% for two concepts, 5.63% for three concepts, 1.41% for four concepts and, finally, 1.05% for five concepts. A randomly selected set of concepts is than assigned to the material according to the real-world dataset. The first concept is chosen based on the occurrence rate of each concept. Each of the remaining concepts are selected according to the probability of co-occurrence between each new concept and the concepts already selected. This procedure avoids the creation of materials with a set of unrelated concepts. After the selection of the concepts for the material, the difficulty is randomly selected (Fig. 2b) from a scale from 1 (easy) to 5 (hard). The learning time of the material is chosen according to its difficulty. The learning time is generated from a exponential distribution (Fig. 2c) in which the coefficients depend on the difficulty: the more difficult the material, the longer its learning time tends to be.

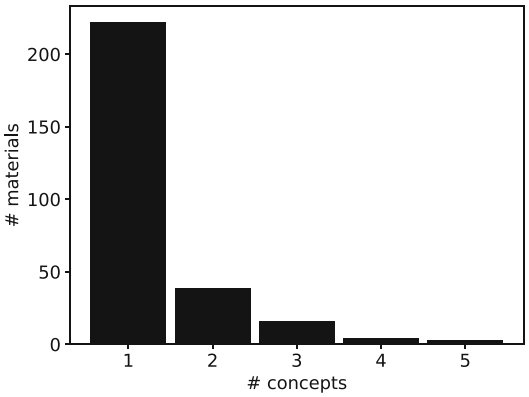
Finally, the most appropriate learning style profile is determined following normal distributions for each of the four FSLSM dimensions: perception, input, processing, and understanding. The LOM (Learning Object Metadata) from real-world learning materials were collected and mapped to FSLSM dimensions. Each dimension was modeled on a scale with values representing the intensity of the characteristics of the learning material in each dimension.

The generation procedure is presented in Fig. 3. All data, metaheuristics implementation and scripts used in the data creation are freely available[3] on the Web for further research. Notice that the proposed method only generates the learning material data of the instance.

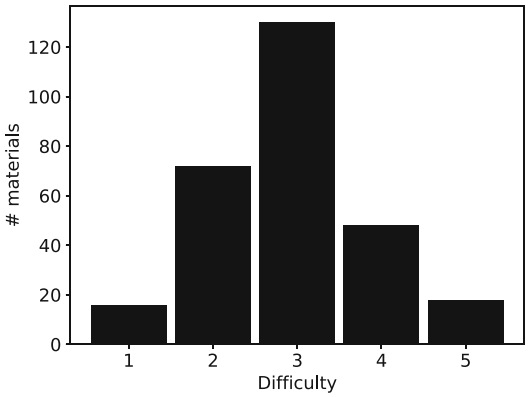
## 6 Metaheuristics for adaptive curriculum sequencing

The main evolutionary computation algorithms used to solve the ACS problem are analyzed here. Two algorithms widely used for Individual Sequencing are Genetic Algorithm and Particle Swarm Optimization. Machado et al. (2019) presented the Prey–Predator Algorithm as a recent and non-fully explored metaheuristic. Finally, we propose a Differential Evolution technique for the context of ACS, as DE obtained good results in other optimization problems from the literature. It is worth noting that we are dealing with the individual sequencing approach (Al-Muhaideb and Menai 2011) modeled as a multi-objective problem (using only student's learning goals and their extrinsic and intrinsic information, without other students' information). Therefore, although other metaheuristics have been investigated, we have selected the most used ones as a baseline.

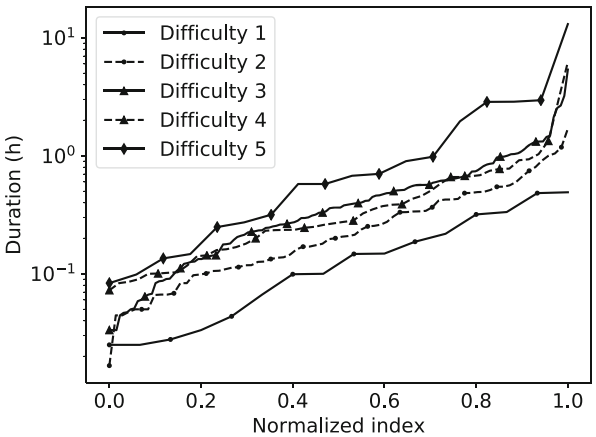
[3] https://github.com/lapic-ufjf/evolutionary-ACS-benchmark.



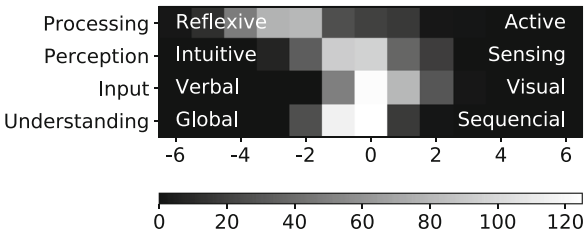

**(a)** Number of learning materials by concept.




**(b)** Number of learning materials by difficulty.




**(c)** Time distribution per difficulty (log scale)




**(d)** Learning styles distribution. The color represents the number of learning materials.

**Fig. 2** Characteristics of the real-world dataset

### 6.1 Genetic algorithm

Genetic Algorithm (GA) was proposed by J. H. Holland in 1975 (Holland et al. 1992) inspired by the Darwin's Theory of Evolution (Pires and Cota 2016). The evolutionary concepts are used to define a search technique for solving optimization problems. In GA, a population of candidate solutions (normally, randomly initialized) evolves by means of selection of the fittest individuals (best candidate solutions), crossover and mutation.

More specifically, the process begins with a set of **Individuals** (**Population**), where each individual is a candidate solution to the problem. The population size is defined as $PS$. In the parental **Selection**, a set of individuals (parents) are chosen. Roulette wheel (fitness proportionate selection), ranking and tournament are commonly procedures to select parents in GA. The individuals selected in the previous phase are recombined by **Crossover** in order to generate new individuals. One of the most simple approaches is the **Point crossover**, where one or more crossover points are chosen at random and the offspring is created by exchanging the parent's genes. Another common method is the **Uniform crossover** where each gene of the parents is swapped following a uniform distribution. **Mutation** is applied to the generated offspring. Mutation is a small modification in the generated candidate solutions. Here, one of the bits in the bit string is flipped according to a **Mutation Probability** (MP). Mutation is important as it maintains the diversity of the population and prevents premature convergence.

The candidate solutions created (children) by crossover and mutation are evaluated with respect to the objective function and the current population is replaced. It is common to keep the best individuals in the population. Also, 10% of the worst candidate solutions can be kept in the population (labeled as "permissive" option in Table 3). To do so, the number of new individuals is given by the population size minus the number of best and worst individuals kept in the population. TS defines the proportion of the best individuals which is not replaced by the offspring (an elitism). The algorithm ends when a stopping criterion is met. Here, a max-

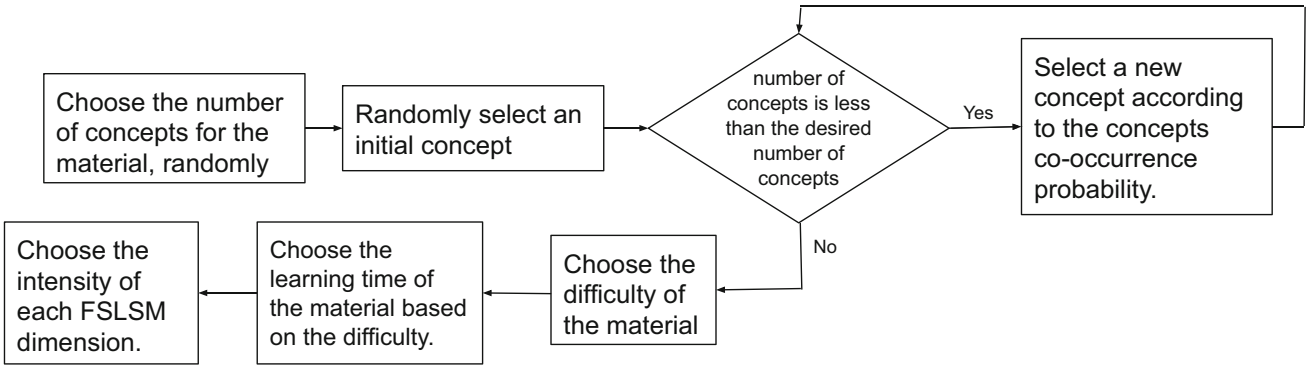



**Fig. 3** Procedure used to create the evaluation datasets

imum number of objective function evaluations is allowed. A pseudo-code of a Genetic Algorithm is presented in Algorithm 1.

**Algorithm 1:** Pseudo-code of a Genetic Algorithm.

```
Result: Sequence of learning materials
Population ← InitializePopulation(PS, |M|);
Objective ← CalculateObjective(Population);
S_best ← GetBestSolution(Objective);
while ¬StopCondition() do
    Parents ← Replacement(Population, TS);
    Children ← Parents;
    while |Children| < PS do
        NewParents ← Selection(Population)
        NewChild ← Crossover(NewParents);
        Children ← Mutation(NewChild, MP);
    Objective ← CalculateObjective(Children);
    S_best ← GetBestSolution(Objective);
    Population ← Children;
```

## 6.2 Particle swarm optimization

**Particle Swarm Optimization (PSO)** was proposed by Russell Eberhart and James Kennedy, and is inspired by the flocking and schooling patterns of birds and fish (Eberhart and Kennedy 1995). The algorithm uses **local** and **global** best information to move and improve the quality of the particles. Each particle consists of: its **velocity** $v_i$, which is a continuous-valued vector that represents how likely the particle is to stay in the current position; its **position** $p_i$, a binary vector representing a candidate solution; and **particle best** $pb$, representing the best position of the current particle. At iteration $t$, the velocities of the particles are updated as

$$\begin{aligned}\mathbf{v_i^{(t+1)}}(x) = c_1 \cdot \mathbf{v_i^{(t)}}(x) \\ + c_2 \cdot \text{rand}() \cdot (\mathbf{pb}(x)_i - x_i) \\ + c_3 \cdot \text{rand}() \cdot (\mathbf{gb}_i - x_i)\end{aligned} \tag{8}$$

where $c_1$, $c_2$ and $c_3$ are constants, $gb$ is the global best and rand() returns a random value uniformly distributed in [0, 1[.

PSO is usually used to solve problems in continuous search space. As the ACS problem solved here was modeled as a binary problem, when the objective function is called the design variables are discretized. Thus, the search occurs in continuous search space and the variables are converted to a binary vector for the evaluation using what we called Evaluation Method (EM).

Similarly to GA, PSO is initialized here with a randomly generated population and the search technique stops when the maximum number of objective function evaluations is reached. Algorithm 2 presents the PSO pseudo-code.

**Algorithm 2:** Pseudo-code of a Particle Swarm Optimization.

```
Result: Sequence of learning materials
Population ← InitializeParticles(PS, |M|);
while ¬StopCondition() do
    foreach x ∈ Population do
        v_i(x) ← UpdateVelocity(v_i(x), pb(x), gb, c1, c2, c3);
        p_i(x) ← Evaluate(v_i(x), EM);
        pb(x) ← Best(pb(x), p_i(x));
        gb ← Best(gb, pb(x));
```

Two Evaluation Methods were considered: fixed and random. In the fixed strategy, the continuous values are in $[-1, 1]$, the variables are discretized to the nearest bound ($-1$ or 1), and the learning material is considered selected when the particle position is positive. The random strategy is based on that presented in (Kennedy and Eberhart 1997), where the $i$-th binary component $x_i^{(b)}$ is defined in a probabilistic manner as

$$x_i^{(b)} = \begin{cases} 0, & \text{if rand}() \leq S(v_i). \\ 1, & \text{otherwise} \end{cases} \tag{9}$$

where $S$ is the sigmoid function. It is important to realize that, unlike standard PSO, here higher velocity values indicate that the position changes less often.

### 6.3 Prey–predator algorithm

the prey–predator algorithm was recently introduced by tilahun and ong (Tilahun and Ong 2015). it is inspired by the ecological interaction among individuals classified as prey and predator. there are different kind of prey–predator interactions based on the kind of prey and how the predator consumes this prey. however, we will stick to the carnivorous ones.

predators actively seek out and pursue their prey, which in turn try to escape by running, hiding or even fighting. the prey often tries to join the stronger ones, after all as faster and stronger prey tend to be more likely to escape. On the other hand, the predator tends to chase the weaker and closer prey. naturally, there are animals that are more likely not to survive, e.g., they are slower or smaller. thus, it can be said that each animal has associated with it a *survival value*. this value indicates how difficult it is to capture that prey.

The so called prey–predator interaction consists in the movement of animals at the time that a predator is hunting. Two basic factors are taken into account in the movement of animals: (1) the direction in which the movement will occur and (2) the step length, which will determine how much the prey or predator will walk in that direction.

Animals are labeled based on the survival value. First, the animal with the worst survival value is labeled as *predator*. Equation 10 describes the predator's movement to chase the worst prey and then move in a random direction $\mathbf{y_r}$.

$$\begin{aligned}\mathbf{x_{predator}} = \mathbf{x_{predator}} &+ \lambda_{\max}\epsilon_5\frac{\mathbf{y_r}}{\|\mathbf{y_r}\|} \\ &+ \lambda_{\min}\epsilon_6\frac{\mathbf{x_i'} - \mathbf{x_{predator}}}{\|\mathbf{x_i'} - \mathbf{x_{predator}}\|}\end{aligned} \tag{10}$$

Second, the animal with the best survival value is labeled as *Best prey*. On the prey–predator interaction, the *Best prey* is considered the one that has found a hiding place and is not affected by the predator. Thus, it has no other prey to follow, so it does a local search trying to find a better position using Eq. 11.

$$\mathbf{x_{best}} = \mathbf{x}_{best} + \lambda_{\min}\epsilon_4\frac{\mathbf{y_l}}{\|\mathbf{y_l}\|} \tag{11}$$

Finally, the remaining animals are called *Ordinary prey*. The ordinary prey moves in direction of other preys that have better survival values $\mathbf{y_i}$ and then moves in a random direction $\mathbf{y_r}$, in the orientation that is farthest from the predator, as in Eq. 12. Algorithm 3 presents a pseudo-code of PPA.

$$\mathbf{x_i} = \mathbf{x_i} + l_{\max}(i)\epsilon_2\frac{\mathbf{y_i}}{\|\mathbf{y_i}\|} + \epsilon_2\frac{\mathbf{y_r}}{\|\mathbf{y_r}\|} \tag{12}$$

We considered a version of the algorithm proposed by Machado et al. (2019) that adapts the movement to work in binary space. In this approach the distance between two individuals is calculated using the Hamming Distance (HD). The movement direction is done step by step, choosing one of the better preys using a roulette and moving toward it. Equation 13 describes how to calculate the chance for the prey $x_i$ to follow prey $x_j$. Here, moving toward means changing one of its binary values to be equal to the other individual.

$$P_{f_{\mathbf{x_i}}}(\mathbf{x_j}) = 1 - \frac{\tau\frac{HD(\mathbf{x_j},\mathbf{x_i})}{m} + \eta\frac{f(\mathbf{x_j})}{f(\mathbf{x_i})}}{2} \tag{13}$$

The distance traveled by the prey is calculated using Eq. 14 and increases the closer the prey is to the predator. Equations 15 and 16 show how to calculate the random direction a prey will follow. $\tau$, $\eta$, and $\beta$ are all constants in [0, 1].

$$l_{max}(i) = \frac{\lambda_{max}\epsilon}{\exp(\beta\frac{HD(\mathbf{x_i},\mathbf{x_{predator}})}{m})} \tag{14}$$

$$d_1 = HD(\mathbf{x_{predator}}, \mathbf{y_r}) \tag{15}$$

$$d_2 = HD(\mathbf{x_{predator}}, -\mathbf{y_r}) \tag{16}$$

**Algorithm 3:** Pseudo-code of Prey–Predator Algorithm.

**Result**: Sequence of learning materials

$Population \leftarrow$ InitializePopulation($PS$, $|M|$);
$Objective \leftarrow$ CalculateObjective($Population$);
$x_{best} \leftarrow$ GetBestSolution($Objective$);
$x_{predator} \leftarrow$ GetWorseSolution($Objective$);
**while** $\neg$*StopCondition*() **do**
  $Population \leftarrow$ UpdatePreyFollow($Population$, $\tau$, $\eta$, $\lambda_{max}$, $\lambda_{min}$, $\beta$, $FC$);
  $Population \leftarrow$ UpdatePreyRun($Population$, $x_{predator}$, $\lambda_{max}$);
  $Population \leftarrow$ UpdateBestPrey($x_{best}$, $\lambda_{min}$);
  $Population \leftarrow$ UpdatePredator($x_{predator}$, $\lambda_{max}$, $\lambda_{min}$);
  $Objective \leftarrow$ CalculateObcjetive($Population$);
  $x_{best} \leftarrow$ GetBestSolution($Objective$);
  $x_{predator} \leftarrow$ GetWorseSolution($Objective$);

### 6.4 Differential evolution

Differential Evolution (DE) (Storn and Price 1997) is a metaheuristic where the candidate solutions are vectors in $\mathbb{R}^n$ and new candidate solutions are generated by mutation (based on differences between vectors) and crossover. Normally, the initial population is randomly created. The new generated individuals are evaluated and the selection for survival

is local: each new individual is compared to a base individual and the best one between composes the next population. For each **Individual** in the **Population**, a mutant vector is generated as:

$$\mathbf{x}_i^{(m)} = \mathbf{x}_i^{(1)} + F \times (\mathbf{x}_i^{(2)} - \mathbf{x}_i^{(3)}) \tag{17}$$

where $\mathbf{x}_i^{(p)}$, $p = 1, 2, 3$, are individuals randomly selected from the population with $\mathbf{x}_i^{(1)} \neq \mathbf{x}_i^{(2)} \neq \mathbf{x}_i^{(3)}$, F is the **Mutation Scale**, a user-defined parameter which controls the step size of the difference vector $(\mathbf{x}_i^{(2)} - \mathbf{x}_i^{(3)})$.

In order to increase the diversity of the population, **Crossover** is introduced (Storn and Price 1997). The crossover is performed as

$$\mathbf{x}_i^{(a)} = \begin{cases} \mathbf{x}_i^{(m)}, & \text{if } cp < \text{CR OR } i == j \\ \mathbf{x}_i, & \text{otherwise,} \end{cases} \tag{18}$$

where $cp$ is randomly generated for each component $i$, and $j$ is an randomly drawn component (this assures that the new individual receives at least one value from the mutant vector). Thus, a new individual, named **applicant**, is created.

The new population replaces the current one according to: the applicant individual replaces the corresponding target vector when the objective function value of the applicant is better. This iterative procedure continues while the stopping criteria isn't met. A pseudo-code of DE is presented in Algorithm 4.

**Algorithm 4:** Pseudo-code of Differential Evolution.

**Result**: Sequence of learning materials
$Population \leftarrow$ InitializePopulation$(PS, |M|)$;
$Objective \leftarrow$ CalculateObjective$(Population)$;
$S_{best} \leftarrow$ GetBestSolution$(Objective)$;
**while** $\neg StopCondition()$ **do**
  $NewPopulation \leftarrow \emptyset$;
  **foreach** $x \in Population$ **do**
    $x^{(1)}, x^{(2)}, x^{(3)} \leftarrow$ SelectIndividuals$(Population)$;
    $x^{(m)} \leftarrow$ Mutation$(x, x^{(1)}, x^{(2)}, x^{(3)}, F)$;
    $x^{(a)} \leftarrow$ Crossover$(x, x^{(m)}, CR)$;
    $X^{(a)} \leftarrow$ Evaluate$(x^{(a)}, EM)$;
    $NewPopulation \leftarrow$ Best$(x, X^{(a)})$;
  $Objective \leftarrow$ CalculateObcjetive$(NewPopulation)$;
  $S_{best} \leftarrow$ GetBestSolution$(Objective)$;
  $Population \leftarrow NewPopulation$;

Similarly to PSO, DE is also designed to solve optimization problems with continuous search space. Thus, the same strategies adopted for PSO were used when evaluating candidate solutions in DE.

## 7 Computational experiments

To compare the metaheuristics, a real dataset was used, as well as a synthetic dataset generated from characteristics extracted from this real data. The stop criterion for all experiments was defined as a maximum number of objective function evaluations. Tests were carried out comparing the quality of the solutions for instances with different sizes (number of learning materials). Here, an 'instance' is a given set of materials and the term 'problem' refers to the sequencing of a set of materials for a single student. Also, the objectives which composes the objective function (defined in Sect. 4) were analyzed.

### 7.1 Conduction and parameter settings

The comparison of the results of each algorithm was performed with the real-world dataset. Due to the limited amount of learning materials, tests were also carried out with the synthetic dataset. The tests with the synthetic data were carried out with different amount of materials to evaluate the performance of the algorithms when the instance size grows. All methods were compared using 5 executions in datasets with different number of learning materials, from 50 to 1000 learning materials. Each dataset was used to find solutions for 24 student profiles with different characteristics. The stop criterion was defined as the maximum number of objective function evaluations equals to 100,000. Also, the convergence of the techniques is analyzed in terms of the normalized value of the objective function value. The normalization is done by dividing each result by the best solution found for that problem. Thus, the performance of the search techniques can be evaluated for different budgets[4]. The majority of SCA approaches from literature use equal weights for the objective functions (Machado et al. 2020). This is a strategy to transform multiple objectives in a single one without any priority among these objectives. Thus, a search technique for single objective optimization can be used. Here, $\omega_i = 1$ for all the objective functions.

The student data were defined as a group of profiles, exploring the different combinations of characteristics. These profiles were defined as all combinations of the characteristics described in Table 1 for a total of 24 profiles. These profiles were defined to test the meta-heuristics with different problems.

For the learner profile, group 1 is composed of students that tend toward the axes active, sensory, visual, sequential from FSLSM. Group 2 consists of students that tend toward the axes reflexive, intuitive, verbal, global. The students were separated into two skill levels, one group with students beginning to learn the concepts and the other with more advanced

[4] Number of objective function evaluations allowed.

**Table 1** Possible values for each characteristic of the students

| Characteristic | Possible values |
|---|---|
| $T^{\uparrow}$ | {0, 10} |
| Learner profile | {Group 1, Group 2} |
| Skill level | {Low, High} |
| Concepts | {All, Most, Few} |

**Table 2** Possible values for each parameter selected by irace

| Method | Parameter name | Possible values |
|---|---|---|
| GA | $PS$ | {10, 20, 30} |
| | $TS$ | {0.01, 0.05, 0.1, 0.15, 0.2} |
| | Replacement | {Elitism, Permissive} |
| | Selection | {Random, Roulette} |
| | Crossover | {Single point, Two point, Three parents, Uniform} |
| | Mutation | {Single bit, Multi bit} |
| PSO | $PS$ | {10, 20, 30} |
| | $c_1$ | [0, 5] |
| | $c_2$ | [0, 5] |
| | $c_3$ | [0, 5] |
| | EM | {Random, Fixed} |
| PPA | $PS$ | {10, 20, 30} |
| | $\tau$ | [0, 1] |
| | $\eta$ | [0, 1] |
| | $\lambda_{min}$ | {2, 4, 10, 20} |
| | $\lambda_{max}$ | {15, 25, 50, 75} |
| | $\beta$ | [0, 1] |
| | $FC$ | [0, 1] |
| DE | $PS$ | {10, 20, 30} |
| | $F$ | {0.01, 0.05, 0.1, 0.2, 0.5, 0.7, 0.9, 1} |
| | $CR$ | {0.01, 0.05, 0.1, 0.2, 0.5, 0.7, 0.9, 1} |
| | EM | {Random, Fixed} |

students. Three groups of objectives were defined: all concepts, which includes all concepts present in the dataset; most concepts, which include 21 out of the 30 concepts; and few concepts, which include only 9 concepts.

The performance of the metaheuristics depends on a correct parameter setting. In order to conduct the comparisons with a good performance of the metaheristics tested here, their parameters have been optimized using the irace package (López-Ibáñez et al. 2016). The possible values for each parameter are presented in Table 2. For the continuous constants, it was given the option to choose numbers with one decimal place.

The best parameters found by irace are presented in Table 3. Only the real-world data were used in the selection of the parameters. Thus, the performance observed in the analysis of the results also considers the generalization capacity of the methods.

**Table 3** Best parameter values selected by irace

| Method | Parameter values |
|---|---|
| GA | $PS$: 10 $TS$: 0.15; Replacement: permissive; Selection: roulette; Crossover: uniform; Mutation: single bit |
| PSO | $PS$: 20; $c_1$: 1.3; $c_2$: 4.1; $c_3$: 2.5; EM: random |
| PPA | $PS$: 10; $\tau$: 0.6; $\eta$: 0.9; $\lambda_{min}$: 10; $\lambda_{max}$: 15; $\beta$: 0.5; $FC$: 0.6 |
| DE | $PS$: 20; $F$: 0.2; $CR$: 0.05; EM: fixed |

## 7.2 Results

The performance of the algorithms was compared concerning final values and convergence. Considering the use of multiple objectives, an analysis of the convergence for each objective function was also carried out.

### 7.2.1 Performance analysis

The performance of GA, PSO, PPA, and the proposed DE is analyzed here when solving ACS problems with 50-1000 learning materials—real-world and synthetic data are considered. The search methods are compared with respect to (i) the final value of the objective function (normalized using the best value found for each student and number of materials), (ii) boxplots of the results considering different number of learning materials, and (iii) convergence. Also, the value of each objective in the objective function (given by a weighted sum of these objectives) is analyzed during the search.

Figure 4 compares the algorithms behavior regarding the increasing number of learning materials using the synthetic data. To remove the variation between the range of objective values for each instance and student profile, the results were normalized based on the best result in each dataset for each student profile. It serves as a good approximation as the best solution for each problem is not known in advance.

The best results were always found by DE for instances with less than 500 materials or by PSO for the other instances. Besides, the worst results are those found by PPA.

Figure 5 presents the results of each metaheuristic. All methods achieve good solutions in most situations, but larger variations in the objective functions can be observed for larger datasets, due to the increased difficulty of finding good solutions. It is worth mentioning that all tests were executed using the same budget. DE and PSO obtained the most consistent results according to the variation presented. This means that these methods are able to find similar values in most runs

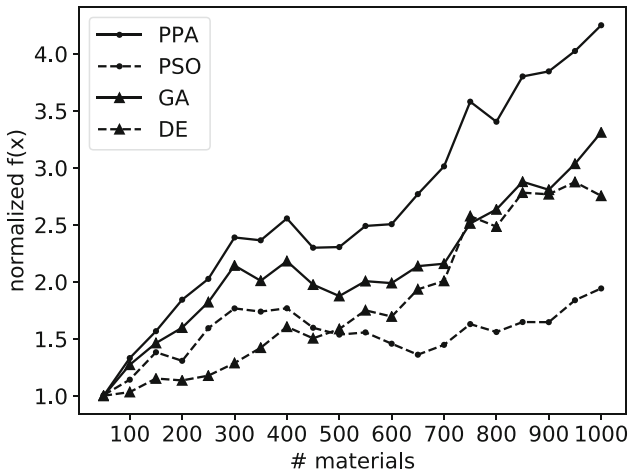
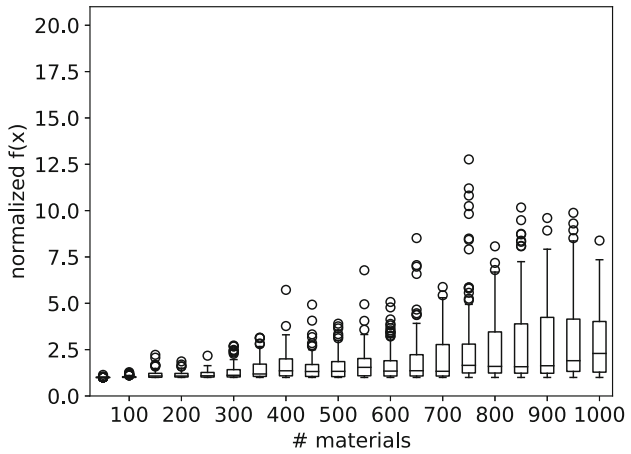


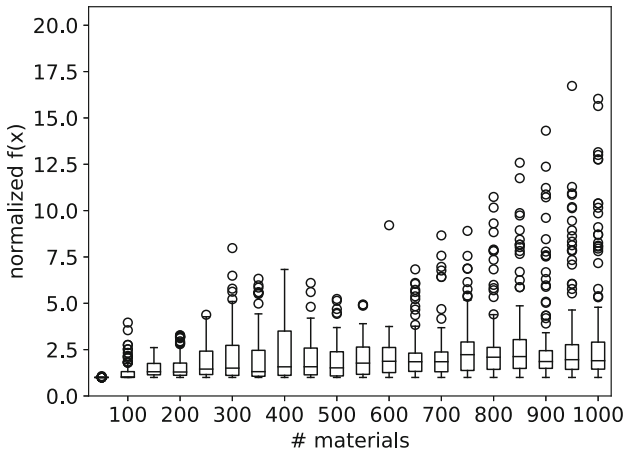

**Fig. 4** Methods comparison for each synthetic dataset

when compared to other methods. PPA presented the largest variation.

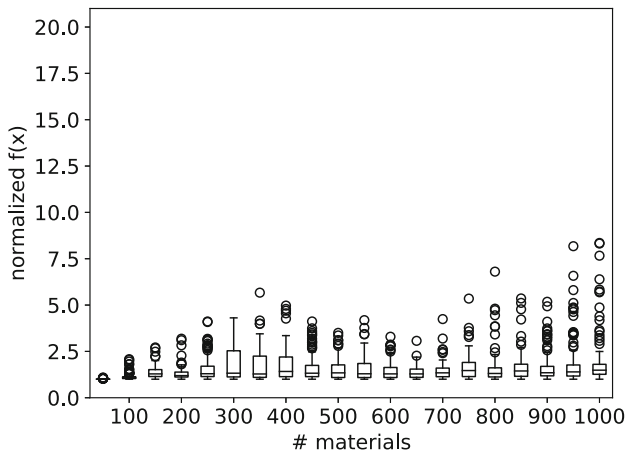
The tests were carried out with a high budget to compare the best results that each algorithm was able to achieve. For these tests, 100,000 evaluations were enough for all the algorithms to stabilize their results. However, in a real scenario, it would be important to find good results with low computational cost. Figure 6 shows the convergence of each metaheuristic during the search. Figure 6a presents the convergence in the real dataset and Fig. 6b presents the mean convergence of all synthetic datasets. For the real-world dataset, which contains 284 learning materials, PPA achieves the worst results while DE achieves the best results of all four algorithms. Also, PSO gets results that are only slightly better than GA. In the synthetic datasets, however, PSO achieves better objective values than DE, although it takes about 75,000 evaluations to do so. This is likely because



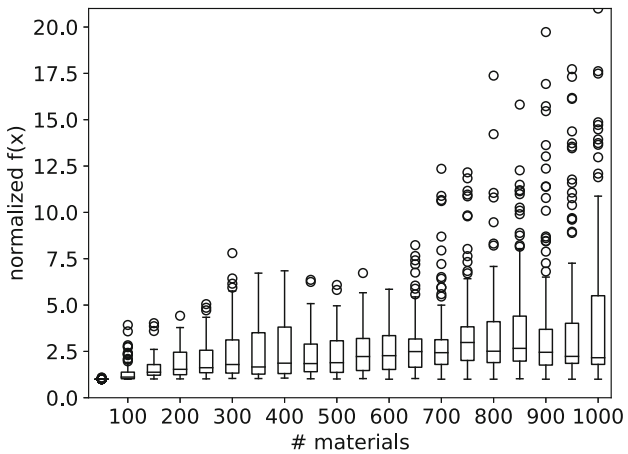

**(a)** Results obtained by DE.




**(b)** Results obtained by GA.

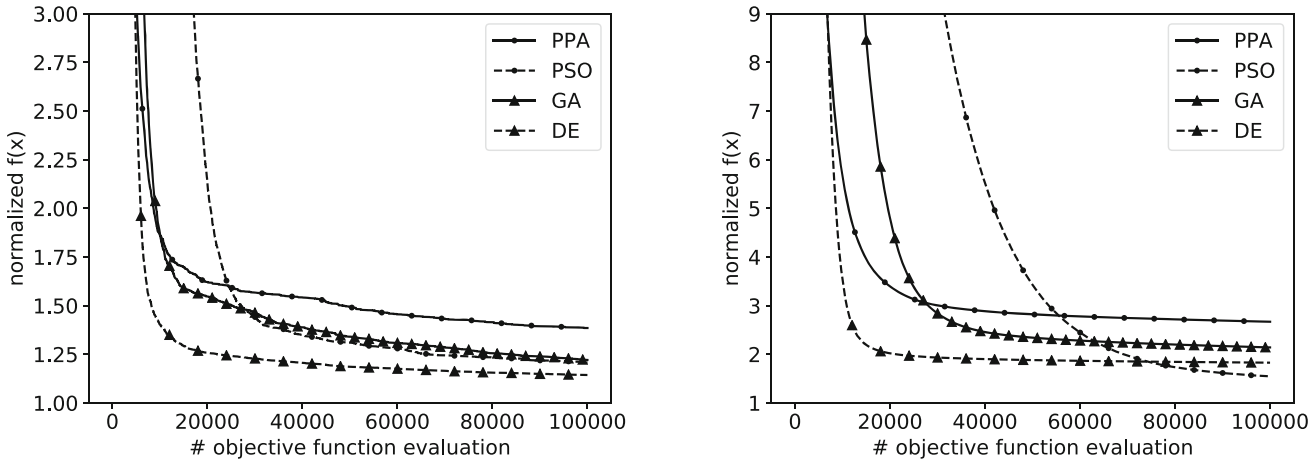



**(c)** Results obtained by PSO.




**(d)** Results obtained by PPA.

**Fig. 5** Boxplots of the results found by DE, GA, PSO, and PPA




**(a)** Real-world dataset.

**(b)** Mean of all synthetic dataset.

**Fig. 6** Methods convergence comparison

the results of PSO are better for larger datasets. From 7,000 to 75,000 evaluations, DE achieved the best results.

#### 7.2.2 Analysis of the multiple objectives

The problem modeling considers different characteristics of each solution with multiple objective functions, but the optimization only considers the weighed sum of them. However, it is important to consider not only the final result but which objectives the methods were capable to solve. Figure 7 shows the convergence of each objective function for DE. The other methods performed similarly. In Figure 7a, it is shown the objective value for the real-world dataset and Figs. 7b–d shows the objective value for the synthetic dataset with 50, 300 and 700 learning materials, respectively. The main difference between instances is in the function $O_3$ that measures the total learning time of the selected learning materials. In large datasets the algorithms are not able to remove enough material from the solution to attend the desired learning time. Another difference happens with very small datasets where there is not enough learning materials to solve the problem, making the values from functions $O_1$ and $O_4$ high. Although functions $O_2$ and $O_5$ are the functions that model the students' preferences and abilities, their objective values have remained constant throughout the iterations. This is a major concern as these are objectives that consider the individual characteristics of each student. None of the algorithms compared in this work were able to find good solutions for these two objectives.

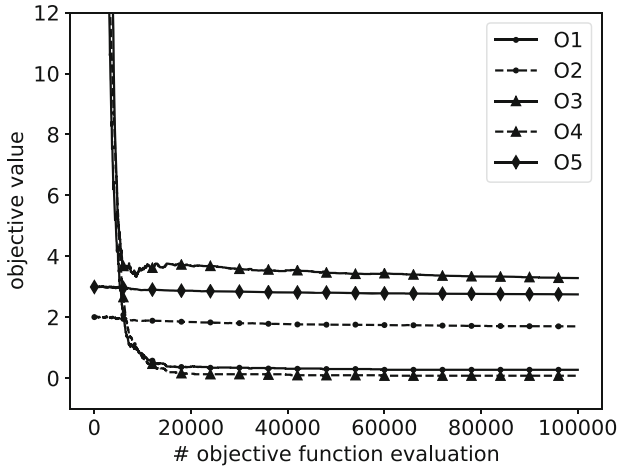
### 7.3 Discussion

The ACS is a longstanding problem that still raises challenges to be overcome as data increase, technology evolves, and people chooses more and more to use online learning systems. As the scenarios continue to transform, the solutions need to be revisited as there is no "silver bullet solution," especially when dealing with NP-Hard problems involving many objectives. The proposed method to generate synthetic data from a real-world dataset makes possible to create increasingly robust solutions to the problem as it presents a way of comparing the proposals.

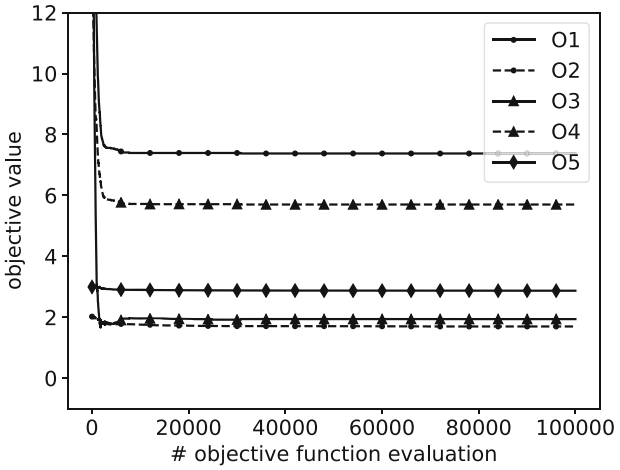
We have used the proposed method to generate datasets and to compare some proposals. One of the points raised by Machado et al. (2019) was that PPA could have better results than GA or PSO and further comparison was needed. Despite PPA having outperformed GA and PSO in domain-independent problems—as shown in (Tilahun and Ong 2015)—the results presented here showed that the only situation where PPA obtained better results in approximating ACS instances was with less than 7,000 function evaluations. For high budget, PPA was not able to find solutions as good as the other algorithms. Also, the results of PPA present high variations.

The preference in the literature for GA and PSO is somewhat justified specially for bigger datasets. PSO shows the best results for datasets with more than 500 learning materials and, although GA is worse than DE in most scenarios, for some datasets it is able to outperform DE. The behavior of GA and PSO is similar to the one reported by Chu et al. (2009) and Li et al. (2012), that is, PSO requires more budget to find good solutions in terms of objective function evaluation.

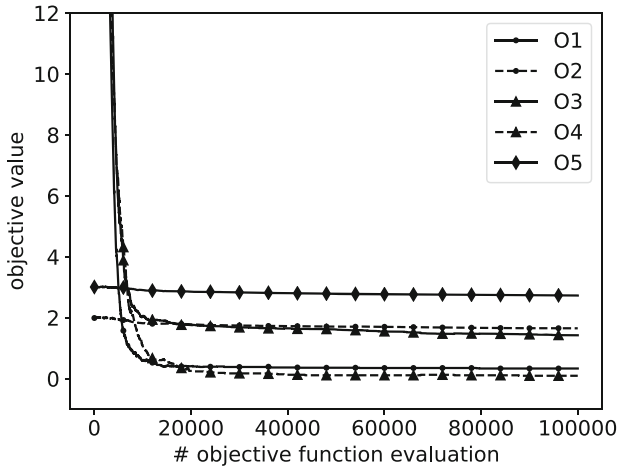
As soft computing solutions are vast some approaches have not yet been explored. Here we have presented an adaptation of DE to address the ACS problem. The choice for DE was due to good performance in similar problems. Although little explored in ACS literature, DE obtained good solu-



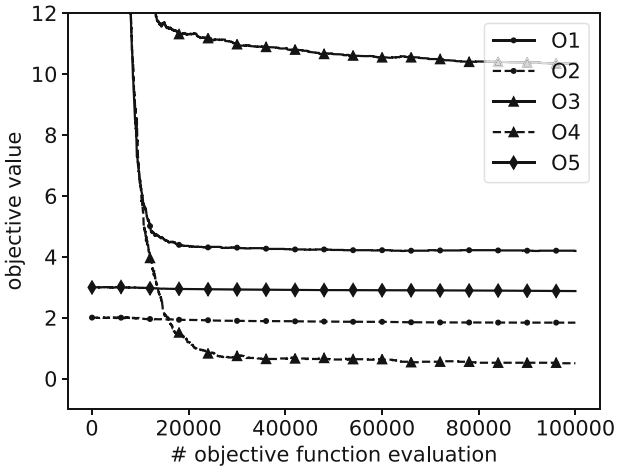

**(a)** Real-world dataset with 284 learning materials.




**(b)** Synthetic dataset with 50 learning materials.




**(c)** Synthetic dataset with 300 learning materials.




**(d)** Synthetic dataset with 700 learning materials.

**Fig. 7** Average of the functions ($O_i(x)$) that compose the objective function ($f(x)$) calculated during the optimization process considering every student for DE

tions using a small number of objective function evaluations. DE outperforms the other techniques in this situation. However, the quality of the solutions obtained varies by a wide range for larger datasets, increasing the average value. With a high enough budget, PSO has demonstrated that it scale better, being able to achieve better results, especially for larger datasets.

Although the intent here was not to attest the ACS effectiveness, the results raise an important insight about the design of personalized courses. The offer of even more assertive sequences depends on the presence of learning materials that meet the most diverse student models. This implies authoring more specialized content that includes different levels of difficulty, forms of presentation, and learning time. In addition, authorship should consider distribution and reuse as it is still a hard task to find annotated bases for effective use and reuse. In other words, even though our method can support in generating synthetic data that meet diverse attributes, this work turns on the light on discussions about Learning Objects and Open Educational Resources to deal with real-world learning scenarios.

## 8 Concluding remarks

Several recent works on adaptive curricular sequencing propose different ways to solve this problem. One of these approaches that have been gaining a lot of popularity is the use of evolutionary computing algorithms. Despite being a longstanding problem, there are few comparisons analyzing which algorithms are able to obtain best results. Given the diversity of modeling for the problem and the lack of available dataset for use, this comparison becomes a hard task.

To deal with the lack of dataset and the difficulty of carrying out experiments with real-world data, a method for generating synthetic data has been proposed here. This method uses data collected from a real-world dataset to generate learning materials that simulate characteristics present in the real ones. The proposed procedure for creating synthetic datasets to evaluate ACS algorithms allows for researchers to fairly compare different methods in specific scenarios (large amount of materials, different distribution of materials and concepts).

The dataset generated using this procedure was applied to compare four search techniques: GA, PSO, PPA, and the proposed DE. Although DE is not easily found in the literature applied to the problem solved here, the proposal has reached the best results among the tested algorithms for datasets smaller than 500 materials. On the other hand, PSO has obtained the best results for the larger ones. In general, it is important to highlight that DE converged faster than the other techniques tested here.

The influence of different objective functions used to describe the quality of the solution was also analyzed. Although a solution to the problem needs to meet several targets, it is possible to see that none of the algorithms were able to find solutions that suited the students' learning style or skill level.

The objective function used here is a linear combination of targets of interest. Although the problem involves multiple objectives, their relationship was not analyzed. Thus, the study of the objectives of the ACS problem is an important research avenue. Also, we intend to investigate the optimization via multiobjective metaheuristics assisted by a multicriteria decision-making approach.

**Acknowledgements** The authors would like to thank CAPES for their financial support.

**Funding** This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Finance Code 001

## Declarations

**Conflict of interest** The authors declare that they have no conflict of interest.

**Ethical approval** The authors declared that this article does not contain any studies with human participants or animals performed by any of the authors.

**Code availability** The source code is available on Github: https://github.com/lapic-ufjf/evolutionary-ACS-benchmark.

## References

Acampora G, Gaeta M, Loia V (2011) Hierarchical optimization of personalized experiences for e-learning systems through evolutionary models. Neural Comput Appl 20(5):641–657

Agarwal S, Goyal M, Kumar A, Rajalakshmi K (2016) Intuitionistic fuzzy ant colony optimization for course sequencing in e-learning. In: *2016 Ninth International Conference on Contemporary Computing (IC3)*, IEEE, pp 1–6

Al-Muhaideb S, Menai MEB (2011) Evolutionary computation approaches to the curriculum sequencing problem. Natural Comput 10(2):891–920

Brusilovsky P (2003) Adaptive and intelligent technologies for web-based education. Int J Artif Intell Educ (IJAIED) 13(4):159–172

Chandar SA, Dheeban S, Deepak V, Elias S (2010) Personalized e-course composition approach using digital pheromones in improved particle swarm optimization. In: *2010 Sixth International Conference on Natural Computation*, IEEE, vol 5, pp 2677–2681

Chang TY, Ke YR (2013) A personalized e-course composition based on a genetic algorithm with forcing legality in an adaptive learning system. J Netw Comput Appl 36(1):533–542

Chen CM (2008) Intelligent web-based learning system with personalized learning path guidance. Comput Educ 51(2):787–814

Christudas BCL, Kirubakaran E, Thangaiah PRJ (2018) An evolutionary approach for personalization of content delivery in e-learning systems based on learner behavior forcing compatibility of learning materials. Telemat Inf 35(3):520–533

Chu CP, Chang YC, Tsai CC (2009) Pc2pso: personalized e-course composition based on particle swarm optimization. App Intell 34(1):141–154

Davis D, Chen G, Hauff C, Houben GJ (2016) Gauging mooc learners' adherence to the designed learning path. In: *International Educational Data Mining Society*

Debbah A, Ali YMB (2014) Solving the curriculum sequencing problem with dna computing approach. Int J Distance Educ Technol (IJDET) 12(4):1–18

Eberhart R, Kennedy J (1995) A new optimizer using particle swarm theory. In: *MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, Ieee, pp 39–43

Felder RM, Silverman LK et al (1988) Learning and teaching styles in engineering education. Eng Educ 78(7):674–681

Gao Y, Peng L, Li F, Li W, et al. (2015) A multi-objective pso with pareto archive for personalized e-course composition in moodle learning system. In: *2015 8th International Symposium on Computational Intelligence and Design (ISCID)*, IEEE, vol 2, pp 21–24

Guo Q, Zhang M (2009) Implement web learning environment based on data mining. Knowl-Based Syst 22(6):439–442

Gutiérrez S, Pardo B (2007) Sequencing in web-based education: approaches, standards and future trends. In: *Evolution of Teaching and Learning Paradigms in Intelligent Environment*, Springer, pp 83–117

Hafidi M, Bensebaa T (2015) Architecture for an adaptive and intelligent tutoring system that considers the learner's multiple intelligences. Int J Distance Educ Technol (IJDET) 13(1):1–21

Hnida M, Idrissi MK, Bennani S (2016) Adaptive teaching learning sequence based on instructional design and evolutionary computation. In: *2016 15th International Conference on Information Technology Based Higher Education and Training (ITHET)*, IEEE, pp 1–6

Holland JH et al (1992) Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence. MIT Press, Cambridge

Huang MJ, Huang HS, Chen MY (2007) Constructing a personalized e-learning system based on genetic algorithm and case-based reasoning approach. Expert Syst Appl 33(3):551–564

Kardan AA, Aziz M, Shahpasand M (2015) Adaptive systems: a content analysis on technical side for e-learning environments. Artif Intell Rev 44(3):365–391

Kennedy J, Eberhart RC (1997) A discrete binary version of the particle swarm algorithm. In: *1997 IEEE International conference on systems, man, and cybernetics. Computational cybernetics and simulation*, IEEE, vol 5, pp 4104–4108

Khalifa M, Lam R (2002) Web-based learning: effects on learning process and outcome. IEEE Trans Educ 45(4):350–356

Khamparia A, Pandey B (2015) Knowledge and intelligent computing methods in e-learning. Int J Technol Enhanc Learn 7(3):221–242

Kuzilek J, Hlosta M, Zdrahal Z (2017) Open university learning analytics dataset. Sci Data 4:170–171

Li JW, Chang YC, Chu CP, Tsai CC (2012) A self-adjusting e-course generation process for personalized learning. Expert Syst Appl 39(3):3223–3232

López-Ibáñez M, Dubois-Lacoste J, Cáceres LP, Birattari M, Stützle T (2016) The irace package: iterated racing for automatic algorithm configuration. Oper Res Perspect 3:43–58

Machado MdOC, Barrére E, Souza J (2019) Solving the adaptive curriculum sequencing problem with prey-predator algorithm. Int J Distance Educ Technol (IJDET) 17(4):71–93

Machado MdOC, Bravo NFS, Martins AF, Bernardino HS, Barrere E, de Souza JF (2020) Metaheuristic-based adaptive curriculum sequencing approaches: a systematic review and mapping of the literature. Artif Intell Rev 54:1–44

de Marcos L, Martínez JJ, Gutiérrez JA (2008a) Swarm intelligence in e-learning: a learning object sequencing agent based on competencies. In: *Proceedings of the 10th annual conference on Genetic and evolutionary computation*, ACM, pp 17–24

de Marcos L, Martínez JJ, Gutiérrez JA, Barchino R, Gutiérrez JM (2008b) An evolutionary approach for domain independent learning object sequencing. In: Lytras MD (ed) World summit on knowledge society. Springer, Berlin, pp 192–197

de Marcos L, Barchino R, Martínez J, Gutiérrez J (2009) A new method for domain independent curriculum sequencing: a case study in a web engineering master program. Int J Eng Educ 25(4):632

de Marcos L, Martinez JJ, Gutiérrez JA, Barchino R, Hilera JR, Oton S, Gutiérrez JM (2011) Genetic algorithms for courseware engineering. Int J Innov Comput Inf Control 7(7):1–27

Menai ME, Alhunitah H, Al-Salman H (2018) Swarm intelligence to solve the curriculum sequencing problem. Comput Appl Eng Educ 26(5):1393–1404

Muhammad A, Zhou Q, Beydoun G, Xu D, Shen J (2016) Learning path adaptation in online learning systems. In: *2016 IEEE 20th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, IEEE, pp 421–426

Niknam M, Thulasiraman P (2020) Lpr: a bio-inspired intelligent learning path recommendation system based on meaningful learning theory. Educ Inf Technol 25(1):3797–3819

Nwana HS (1990) Intelligent tutoring systems: an overview. Artif Intell Rev 4(4):251–277

Pires JM, Cota MP (2016) "intelligent" adaptive learning objects applied to special education needs: extending the elearning paradigm to the ulearning environment. In: *2016 11th Iberian Conference on Information Systems and Technologies (CISTI)*, IEEE, pp 1–6

Pushpa M (2012) Aco in e-learning: towards an adaptive learning path. Int J Comput Sci Eng 4(3):458

Rathore AS, Arjaria SK (2020) Intelligent tutoring system. In: Utilizing educational data mining techniques for improved learning: emerging research and opportunities, IGI Global, pp 121–144

Seki K, Matsui T, Okamoto T (2005) An adaptive sequencing method of the learning objects for the e-learning environment. Electr Commun Japan (Part III Fundam Electr Sci) 88(3):54–71

Sharma R, Banati H, Bedi P (2012) Adaptive content sequencing for e-learning courses using ant colony optimization. In: *Proceedings of the International Conference on Soft Computing for Problem Solving (SocProS 2011)* December 20-22, 2011, Springer, pp 579–590

Shmelev V, Karpova M, Dukhanov A (2015) An approach of learning path sequencing based on revised bloom's taxonomy and domain ontologies with the use of genetic algorithms. Procedia Comput Sci 66:711–719

Silva RC, Direne AI, Marczal D, Borille AC, Guimarães PRB, da Silva Cabral A, Camargo BF (2018) Adaptability of learning objects using calibration and adaptive sequencing of exercises. Braz J Comput Educ 26(01):70

Solomon BA, Felder RM (1999) Index of learning styles. North Carolina State University Available online, Raleigh, NC

Storn R, Price K (1997) Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces. J Glob Optim 11(4):341–359

Tilahun SL, Ong HC (2015) Prey-predator algorithm: a new metaheuristic algorithm for optimization problems. Int J Inf Technol Decis Mak 14(06):1331–1352

Wan S, Niu Z (2016) A learner oriented learning recommendation approach based on mixed concept mapping and immune algorithm. Knowl-Based Syst 103:28–40

Williamson B, Eynon R, Potter J (2020) Pandemic politics, pedagogies and practices: digital technologies and distance education during the coronavirus emergency. Learn Media Technol 45(2):107–114. https://doi.org/10.1080/17439884.2020.1761641

Wong LH, Looi CK (2010) A survey of optimized learning pathway planning and assessment paper generation with swarm intelligence. Intelligent tutoring systems in E-learning environments: design, implementation and evaluation. IGI Global, Hershey, pp 285–302

Xie H, Zou D, Wang FL, Wong TL, Rao Y, Wang SH (2017) Discover learning path for group users: a profile-based approach. Neurocomputing 254:59–70