Universidade Federal de Juiz de Fora
Instituto de Ciências Exatas
Bacharelado em Ciência da Computação

A Comparative Analysis of Metaheuristics Applied to Adaptive Curriculum Sequencing

André Ferreira Martins

JUIZ DE FORA
AGOSTO, 2020

A Comparative Analysis of Metaheuristics Applied to Adaptive Curriculum Sequencing

André Ferreira Martins

Universidade Federal de Juiz de Fora

Instituto de Ciências Exatas

Departamento de Ciência da Computação

Bacharelado em Ciência da Computação

Orientador: Jairo Francisco de Souza

Coorientador: Heder Soares Bernardino

JUIZ DE FORA

AGOSTO, 2020

A Comparative Analysis of Metaheuristics Applied to Adaptive Curriculum Sequencing

André Ferreira Martins

MONOGRAFIA SUBMETIDA AO CORPO DOCENTE DO INSTITUTO DE CIÊNCIAS EXATAS DA UNIVERSIDADE FEDERAL DE JUIZ DE FORA, COMO PARTE INTEGRANTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE BACHAREL EM CIÊNCIA DA COMPUTAÇÃO.

Aprovada por:

Jairo Francisco de Souza
D.Sc. em Informática - PUC-Rio

Heder Soares Bernardino
D.Sc. em Modelagem Computacional

Luciana Brugiolo Gonçalves
D.Sc. em Computação - UFF

Lorenza Leão Oliveira Moreno
D.Sc. em Informática - PUC-Rio

JUIZ DE FORA
10 DE AGOSTO, 2020

Aos meus amigos e irmãos.

Aos pais, pelo apoio e sustento.

Resumo

A adoção efetiva do aprendizado on-line depende da satisfação do usuário, pois as abordagens de educação a distância sofrem com a falta de comprometimento que pode levar a falhar e desistências. A literatura de aprendizagem adaptativa argumenta que uma alternativa para alcançar a satisfação dos alunos é tratá-los individualmente, fornecendo o conteúdo educacional de maneira personalizada. Além disso, o sequenciamento deste conteúdo é importante para evitar a sobrecarga cognitiva e desorientação – esse problema é chamado de Sequenciamento Curricular Adaptativo (SCA). A busca de uma sequência ótima de bancos de dados que não param de crescer é um problema de otimização combinatória NP-Difícil. Embora algumas abordagens tenham sido propostas, é desafiador avaliar suas contribuições devido à falta de dados de *benchmark* disponíveis. Este trabalho apresenta um procedimento para criar conjuntos de dados sintéticos para avaliar abordagens SCA e, como prova de conceito, analisa metaheurísticas normalmente usadas em abordagens SCA: Algoritmo Genético, Otimização por Enxame de Partículas (OEP) e Algoritmo Presa-Predador usando os objetivos de aprendizagem dos alunos e suas características extrínsecas e intrínsecas. Também propomos uma abordagem baseada na Evolução Diferencial (ED). Os experimentos computacionais incluem conjuntos de dados sintéticos com uma quantidade variada de materiais de aprendizado e conjuntos de dados do mundo real para comparação. Os resultados mostram que o DE teve um desempenho melhor do que os outros métodos quando menos de 500 materiais de aprendizado são usados, enquanto o PSO teve um desempenho melhor em problemas maiores.

Palavras-chave: Caminho de Aprendizagem, Aprendizagem Adaptativa, Sistema Tutorial Inteligente, Computação Evolutiva, Sequenciamento Curricular.

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 is important to avoid cognitive overload and disorientation – this problem is called Adaptive Curriculum Sequencing (ACS). 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.

Agradecimentos

A todos os meus parentes, pelo encorajamento e apoio.

Aos professores Jairo e Heder pela orientação, amizade e principalmente, pela paciência, sem a qual este trabalho não se realizaria.

Aos professores do Departamento de Ciência da Computação pelos seus ensinamentos e aos funcionários do curso, que durante esses anos, contribuíram de algum modo para o nosso enriquecimento pessoal e profissional.

“The past is past, but that’s okay! It’s never really gone completely. The future is always built on the past, even if we won’t get to see it.“

Riebeck (Outer Wilds)

Contents

List of Figures **7**

List of Tables **8**

List of Abbreviations **9**

1 Introduction **10**

2 Adaptive Curriculum Sequencing **13**

3 Related Works **16**

4 Modeling the problem **19**

5 Proposed Procedure for Generating Learning Materials **22**

6 Metaheuristics for Adaptive Curriculum Sequencing **25**
6.1 Genetic Algorithm 25
6.2 Particle Swarm Optimization 27
6.3 Prey-Predator Algorithm 28
6.4 Differential Evolution 30

7 Computational Experiments **32**
7.1 Conduction and parameter settings 32
7.2 Results 33
7.3 Discussion 36

8 Concluding remarks **38**

Bibliography **40**

List of Figures

5.1 Characteristics of the real-world dataset. 23
5.2 Procedure used to create the evaluation datasets. 24

7.1 Methods comparison for each instance. 34
7.2 Boxplots of the results found by DE, GA, PSO, and PPA. 35
7.3 Methods convergence comparison. 35
7.4 Average of the functions ($O_i(x)$) that compose the objective function ($f(x)$) calculated during the optimization process considering every student. . . . 36

List of Tables

7.1 Best parameter values selected by irace. 33

List of Abbreviations

ACO	Ant Colony Optimization
ACS	Adaptive Curriculum Sequencing
CSP	Constraint Satisfaction Problem
DE	Differential Evolution
FSLSM	Felder and Silverman Learning Style Model
GA	Genetic Algorithm
ILS	Index of Learning Style
ITS	Intelligent Tutoring System
LOM	Learning Object Metadata
PPA	Prey-Predator Algorithm
PSO	Particle Swarm Optimization

1 Introduction

The distance learning has begun widely adopted making it important to design techniques to support this type of education (KARADENIZ, 2009). The effective use of e-learning systems depends on solutions that aims the students' satisfaction, as distance education modality suffers from the lack of commitment that may causes 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 (RATHORE; ARJARIA, 2020; SILVA et al., 2018; PHOBUN; VICHEANPANYA, 2010). 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; AZIZ; SHAHPASAND, 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; ALI, 2014).

One well known problem in adaptive learning is the Adaptive Curriculum Sequencing (ACS) (WEBER; SPECHT, 1997; CHEN, 2008). It is considered a crucial issue in personalized learning as its purpose is to find the best sequence of learning materials that meet the student profile (NIKNAM; THULASIRAMAN, 2020; MACHADO; BARRÉRE; SOUZA, 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, leading to an increase in failure and dropout rates (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; KE, 2013; MARCOS et al., 2008). 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; BARRÉRE;

SOUZA, 2019; WONG; LOOI, 2010; AL-MUHAIDEB; MENAI, 2011; WANG; WU, 2011; PUSHPA, 2012; KHAMPARIA; PANDEY, 2015). 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. Solutions to large ACS instances can only be approximated and, therefore, heuristic and metaheuristic 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.

In Al-Muhaideb e Menai (2011), publications from the 2002 to 2009 whose used evolutionary computation approaches to the ACS problem were reviewed. 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, Barrére e Souza (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; NIU, 2016) and different implementations of PSO and/or ACO (MENAI; ALHUNITAH; AL-SALMAN, 2018; CHANDAR et al., 2010). In either case, the studies considers 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 experi-

mental comparative analysis of four approaches to address the ACS problem: GA, PSO, PPA, and our proposed Differential Evolution (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: Section 2 describes the problem addressed by this work; Section 3 reviews related works; Section 4 presents the modeling that we adopted; Section 5 presents the proposed procedure for generating datasets with learning materials; Section 6 describes the implementation of the metaheuristics; Section 7 shows the experiments carried out and the results obtained; Finally, Section 8 concludes the paper.

2 Adaptive Curriculum Sequencing

The research and development of Intelligent Tutoring System (ITS) seeks to combine techniques of Artificial Intelligence, Cognitive Psychology and Educational Learning Theories towards 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 interesting and crucial goal in Adaptive Learning (MUHAMMAD et al., 2016). There is not yet a consensus related to the term that defines this sequence (HNIDA; IDRISSI; BENNANI, 2016) and, although other terminologies may 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; BENSEBAA, 2015) therefore maximizing understanding as well as learning efficiency. According to Kardan, Aziz e Shahpasand (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 (MARCOS et al., 2011; SHARMA; BANATI; BEDI, 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; MATSUI; OKAMOTO, 2005; CHEN, 2008; HUANG; HUANG; CHEN, 2007; GUO; 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 labour, 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; PARDO, 2007).

The ACS problem can be formulated as either a Constraint Satisfaction Problem (CSP) (MARCOS; MARTÍNEZ; GUTIÉRREZ, 2008; MARCOS et al., 2009) or a multi-objective optimization problem (CHU; CHANG; TSAI, 2009; GAO et al., 2015) (AL-MUHAIDEB; 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 the various sequence possibilities contained in S.

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; GAETA; LOIA, 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 $C!$ (MARCOS et al., 2008). 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; PANDEY, 2015; PUSHPA, 2012; AL-MUHAIDEB; MENAI, 2011).

3 Related Works

The research and development of Intelligent Tutoring Systems (ITS) seeks to combine techniques of Artificial Intelligence, Cognitive Psychology and Educational Learning Theories towards 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 interesting and crucial goal in Adaptive Learning (MUHAMMAD et al., 2016). ACS has been addressed in several papers (WONG; LOOI, 2010; PUSHPA, 2012; AL-MUHAIDEB; MENAI, 2011; KHAMPARIA; 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 GA stand out (AL-MUHAIDEB; 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; BARRÉRE; SOUZA, 2019; MARCOS et al., 2011; CHU; CHANG; TSAI, 2011; MARCOS et al., 2009; SEKI; MATSUI; OKAMOTO, 2005; HUANG; HUANG;

CHEN, 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, Kirubakaran e Thangaiah (2018) when comparing implementations of GAs and PSOs.

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, Alhunitah e Al-Salman (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 anonymised 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; HLOSTA; ZDRAHAL, 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; KARPOVA; DUKHANOV, 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 (difficult, content, and style), and

[1] ⟨www.tvtc.gov.sa⟩

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

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 $O_i(x)$ as

$$f(\mathbf{x}) = \min \sum_{i=1}^{5} \omega_i O_i(\mathbf{x}) \tag{4.1}$$

$$O_1(\mathbf{x}) = (1-\rho)(|R(\mathbf{x})| - |R(\mathbf{x}) \cap E|) + \rho(|E| - |R(\mathbf{x}) \cap E|) \tag{4.2}$$

$$O_2(\mathbf{x}) = \sum_{i=1}^{|M|} x_i \left| D^{m_i} - \frac{\sum_{j=1}^{|C|} R_{c_j}^{m_i} E_{c_j} H_{c_j}}{\sum_{j=i}^{|C|} R_{c_j}^{m_i} E_{c_j}} \right| \frac{1}{\sum_{i=1}^{|M|} x_i} \tag{4.3}$$

$$O_3(\mathbf{x}) = \max\left(T^{\downarrow} - \sum_{i=1}^{|M|} T^{m_i} x_i, 0\right) + \max\left(0, \sum_{i=1}^{|M|} T^{m_i} x_i - T^{\uparrow}\right) \tag{4.4}$$

$$O_4(\mathbf{x}) = \sum_{j=1}^{|C|} E_{c_j} \left| \sum_{i=1}^{|M|} x_i R_{c_j}^{m_i} - \frac{\sum_{i'=1}^{|M|} \sum_{j'=1}^{|C|} x_{i'} R_{c_{j'}}^{m_{i'}} E_{c_{j'}}}{\sum_{j'=1}^{|C|} E_{c_{j'}}} \right| \tag{4.5}$$

$$O_5(\mathbf{x}) = \sum_{k=1}^{4} \frac{\sum_{i=1}^{|M|} x_i \left|3 sgn(\theta_k^{m_i}) - L_k\right|}{4 \sum_{i=1}^{|M|} x_i} \tag{4.6}$$

where ω_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, and $c_j \in C$ represents the j-th concept of a course.

We also defined that $\sum_{i=1}^{5} \omega_i = 1$ and $\omega_i \in [0,1]\ \forall i \in \{1,\ldots,5\}$ in order to constraint the weight values. 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:

$$R(x) = \bigcup_{i=1}^{|M|} \{R^{m_i} | x_i = 1\} \tag{4.7}$$

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 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 higher bounds times expected by the student. Besides, T^{m_i} represents the estimated learning time of a learning material m_i.

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, $O_5(\mathbf{x})$ relates the student's learning style to the characteristics of the

learning material according to the Felder and Silverman Learning Style Model (FSLSM) (FELDER; SILVERMAN et al., 1988). In $O_5(\mathbf{x})$, $\theta_k^{m_i}$ indicates which learning style the i-th learning material tends to contemplate the k-th dimension of FSLSM, $k = 1, \ldots, 4$, and $L_k \in [-3, 3]$ represents the student intensity in a learning style of the k-th FSLSM dimension (MACHADO; BARRÉRE; SOUZA, 2019).

5 Proposed Procedure for Generating Learning Materials

Performing tests with real data presents some difficulties. It is necessary to organize a course with several students having 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.

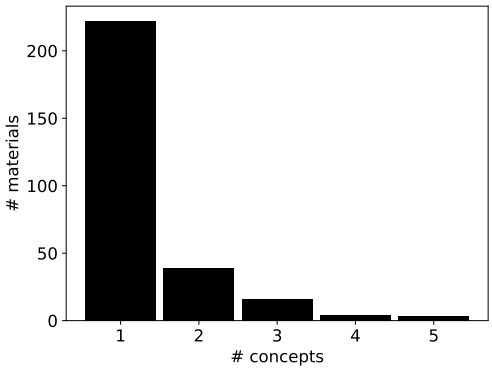
The real-world dataset was obtained as part of a course on Computer Science Fundamentals held in 2017 remotely on the Moodle platform. The concepts taught were distributed over a six week period. Each week the students received a sequence of learning materials covering the expected concepts and were 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 may 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; 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.

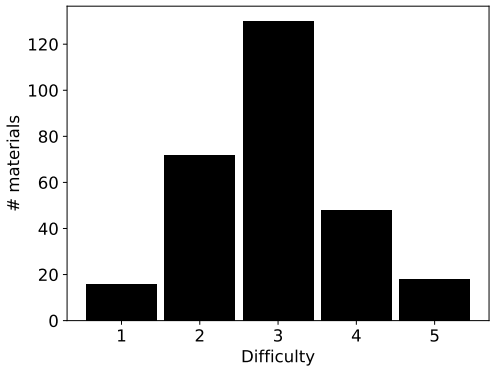
The data collected from this course was used to create a synthetic dataset. The purpose of the synthetic dataset is to address the lack in the literature of comparative assessments of the performance of metaheuristics using a large amount of learning materials. For the creation, we considered characteristics 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 were analyzed, and the observed characteristics were used to create the new dataset.

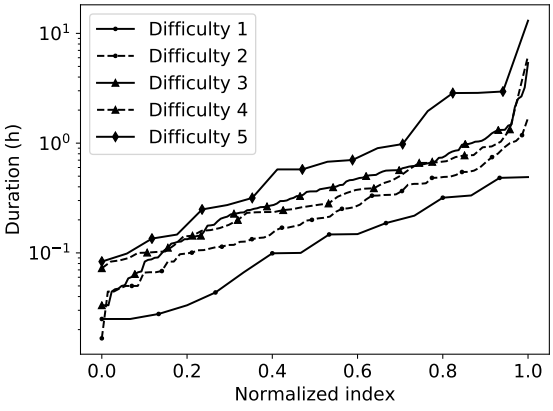
The main characteristics obtained from this data collection were: most learning materials cover only few concepts (Figure 5.1a), and there is a correlation between the difficult and the learning time of materials (Figure 5.1c). The learning time follows an exponential distribution for all difficulties but the easiest materials are generally shorter than the most difficult. The other features follow a normal distribution, as can be seen in Figures 5.1b and 5.1d.



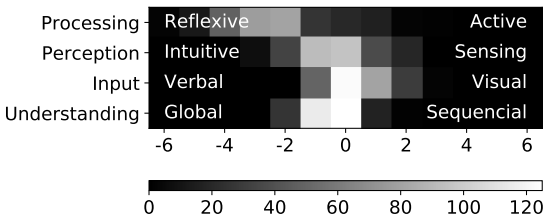
(a) Number of learning materials by concept.



(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.

Figure 5.1: Characteristics of the real-world dataset.

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 (Figure 5.1a), 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 difficult is randomly selected (Figure 5.1b) from a scale from 1 (easy) to 5 (hard). The learning time of the material is chosen according to its difficult. The learning time is generated from a exponential distribution (Figure 5.1c) 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 Learning Object Metadata (LOM) from real-world learning materials were collected and mapped to FSLSM dimensions. Each dimension was modeled in a scale with values representing the intensity of the characteristics of the learning material in each dimension, as follows: in the dimension Processing, the material may be used for students more reflexive or more active; in the dimension Perception, it may be used for students more intuitive or more sensing; in the dimension Input, the values can represent contents more verbal or more visual; and, finally, the dimension Understanding represents a knowledge explained using a global approach or a more sequential approach.

The generation procedure is presented in Figure 5.2. All data and scripts used in the data creation are freely available[3] on the Web for further research.

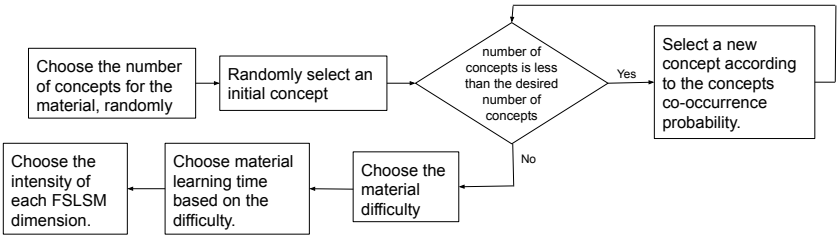


Figure 5.2: Procedure used to create the evaluation datasets.

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

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. In Machado, Barrére e Souza (2019), the Prey-Predator Algorithm is presented 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; 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.

6.1 Genetic Algorithm

Genetic Algorithm (GA) was proposed by J. H. Holland in 1975 inspired by the Darwin's Theory of Evolution (PIRES; 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 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. 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 approach 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 7.1). 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 maximum number of objective function evaluations is allowed. A pseudo-code of a Genetic Algorithm is presented in Algorithm 1.

Algoritmo 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 ← SelectParents(Population, TS);
    Children ← ∅;
    foreach Parent ∈ Parents do
        Children ← Crossover(Parent);
        Children ← Mutation(Children, MP);
    end foreach
    Objective ← CalculateObjective(Children);
    S_best ← GetBestSolution(Objective);
    Population ← Replace(Population, Children);
end while
```

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; 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**, a continuous-valued vector which is how certain it is of the values in the current position; its **position**, a binary vector representing a candidate solution; and **local best**, representing the best position of the current particle. At iteration t, the velocities of the particles are updated as

$$\mathbf{v_i^{(t+1)}}(x) = c_1 \cdot \mathbf{v_i^{(t)}}(x) + c_2 \cdot \text{rand}() \cdot (\mathbf{pb}(x)_i - x_i) + c_3 \cdot \text{rand}() \cdot (\mathbf{gb}_i - x_i) \tag{6.1}$$

where 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, the design variables are discretized when the objective function is called. Thus, the search occurs in continuous search space and the variables are converted to a binary vector for the evaluation.

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.

Algoritmo 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));
    end foreach
end while
```

Two discretization strategies, called here Evaluation Method (EM), were consid-

ered: 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 e Eberhart (1997), where the i-th binary component $x_i^{(b)}$ is defined in a probabilistic manner as

$$x_i^{(b)} = \begin{cases} 0, & \text{if rand}() \leq S(x_i). \\ 1, & \text{otherwise} \end{cases} \tag{6.2}$$

where S is the sigmoid function.

6.3 Prey-Predator Algorithm

The Prey-Predator Algorithm was recently introduced by Tilahun and Ong (TILAHUN; ONG, 2015). It is inspired by the ecological interaction among individuals classified as prey and predator. There are different kind of prey-predator interaction based on the kind of the 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 difficulty it is to capture that prey.

Animals are labeled based on the survival value. First, the animal with the worst survival value is labeled as *Predator*. 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 thus is not affected by the predator. Finally, the remaining animals are called *Ordinary 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.

The ordinary prey moves in direction of other preys that have better survival values and then moves in a random direction, in the orientation that is farthest from the predator, as in Equation 6.3. The best prey does not have any other prey to follow so it does a local search trying to find a better position using Equation 6.4. Equation 6.5 describes the predator's movement to chase the worst prey and then move in a random direction. Algorithm 3 presents a pseudo-code of PPA.

$$\mathbf{x_i} = \mathbf{x_i} + l_{\max}(i)\epsilon_2 \frac{\mathbf{y_i}}{\|\mathbf{y_i}\|} + \epsilon_2 \frac{\mathbf{y_r}}{\|\mathbf{y_r}\|} \tag{6.3}$$

$$\mathbf{x_{best}} = \mathbf{x_{best}} + \lambda_{\min}\epsilon_4 \frac{\mathbf{y_l}}{\|\mathbf{y_l}\|} \tag{6.4}$$

$$\begin{aligned} \mathbf{x_{predator}} = \mathbf{x_{predator}} &+ \lambda_{\max}\epsilon_5 \frac{\mathbf{y_r}}{\|\mathbf{y_r}\|} \\ &+ \lambda_{\min}\epsilon_6 \frac{\mathbf{x'_i} - \mathbf{x_{predator}}}{\|\mathbf{x'_i} - \mathbf{x_{predator}}\|} \end{aligned} \tag{6.5}$$

We considered a version of the algorithm proposed in Machado, Barrére e Souza (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 towards it. Here, moving towards means changing one of its binary values to be equal to the other individual.

$$P_{f_{\mathbf{s_i}}}(\mathbf{s_j}) = 1 - \frac{\tau \frac{HD(\mathbf{s_j},\mathbf{s_i})}{m} + \eta \frac{f(\mathbf{s_j})}{f(\mathbf{s_i})}}{2} \tag{6.6}$$

$$l_{max}(i) = \frac{\lambda_{max}\epsilon}{\exp(\beta \frac{HD(\mathbf{s_i},\mathbf{s_{predator}})}{m})} \tag{6.7}$$

$$d_1 = HD(\mathbf{s_{predator}}, \mathbf{y_r}) \tag{6.8}$$

$$d_2 = HD(\mathbf{s_{predator}}, -\mathbf{y_r}) \tag{6.9}$$

$$l_{predator} = \lambda_{min}\varepsilon \tag{6.10}$$

Algoritmo 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$(Objective)$;
end while

6.4 Differential Evolution

Differential Evolution (DE) (STORN, 1995) 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:

$$\mathbf{x}^{(m)} = \mathbf{x}^{(1)} + F \times (\mathbf{x}^{(2)} - \mathbf{x}^{(3)}) \tag{6.11}$$

where $\mathbf{x}^{(i)}$, $i = 1, 2, 3$, are individuals randomly selected from the population with $\mathbf{x}^{(1)} \neq \mathbf{x}^{(2)} \neq \mathbf{x}^{(3)}$, F is the **Mutation Scale**, a user-defined parameter in $]0, 2]$ which controls the step size of the difference vector $(\mathbf{x}^{(2)} - \mathbf{x}^{(3)})$.

In order to increase the diversity of the population, **Crossover** is introduced (STORN, 1995). The crossover is performed as

$$\mathbf{x}_i^{(a)} = \begin{cases} \mathbf{x}_i^{(m)}, & \text{if } cp < \text{CR} \ \ \text{OR} \ \ i == j \\ \mathbf{x}_i, & \text{otherwise,} \end{cases} \tag{6.12}$$

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.

Algoritmo 4: Pseudo-code of Differential Evolution.

```
Result: Sequence of learning materials
Population ← InitializePopulation(PS, |M|);
Objective ← CalculateObjective(Population);
S_best ← GetBestSolution(Objective);
while ¬StopCondition() do
    NewPopulation ← ∅;
    foreach x ∈ Population do
        x^(1), x^(2), x^(3) ← SelectIndividuals(Population);
        x^(m) ← Mutation(x, x^(1), x^(2), x^(3), F);
        x^(a) ← Crossover(x, x^(m), CR);
        X^(a) ← Evaluate(x^(a), EM);
        NewPopulation ← Best(x, X^(a));
    end foreach
    Objective ← CalculateObcjetive(NewPopulation);
    S_best ← GetBestSolution(Objective);
    Population ← Replace(Population, Children);
end while
```

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). Also, the objectives which composes the objective function were analyzed (Section 4).

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 problem 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. Thus, the performance of the search techniques can be evaluated for different budgets.

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. The best parameters found by irace are presented in Table 7.1. Only the real-world data was 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 7.1: 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; τ: 0.6; η: 0.9; λ_{min}: 10; λ_{max}: 15; β: 0.5; FC: 0.6
DE	PS: 20; F: 0.2; CR: 0.05; EM: fixed

7.2 Results

The performance of GA, PSO, PPA, and the proposed DE are 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 7.1 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.

Figure 7.2 presents the results of each metaheuristic. All methods achieve good solutions most of the cases, but larger variations in the objective functions can be observed for larger datasets, due to the increased difficult 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 when compared to other methods. PPA presented the largest variation.

The tests were carried out with a high budget (i.e. 100,000 objective function evaluation) to allow that all algorithms have their results stabilized. However, in a real scenario, it would be important to find good results with low computational cost. Figure 7.3 shows the convergence of each metaheuristic during the search. Figure 7.3a presents

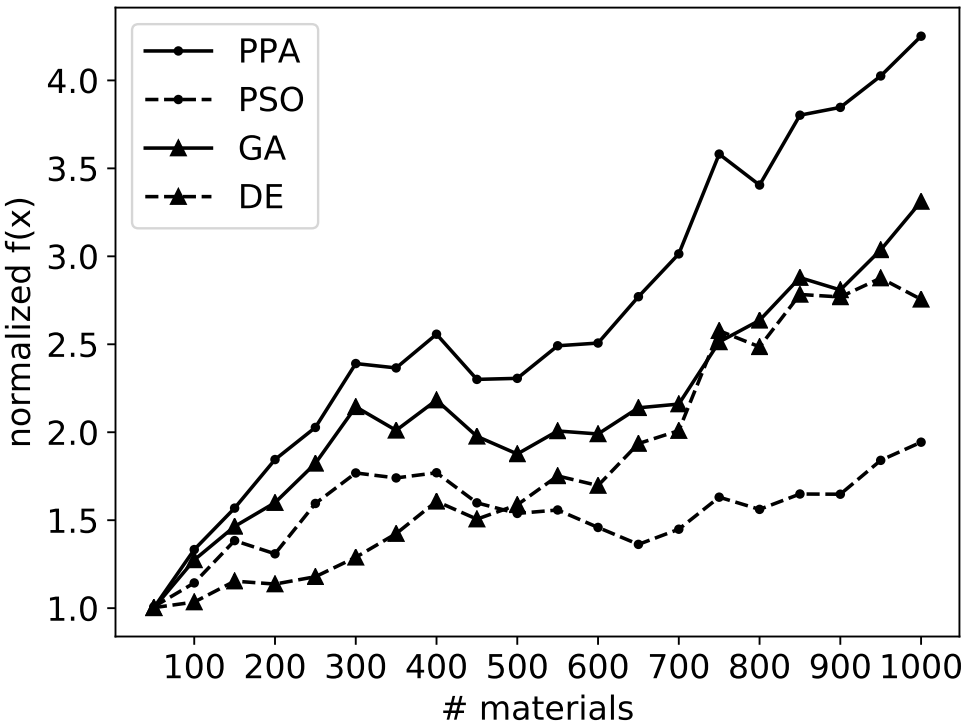
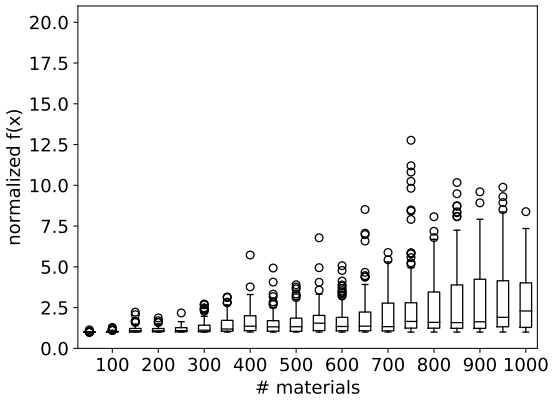


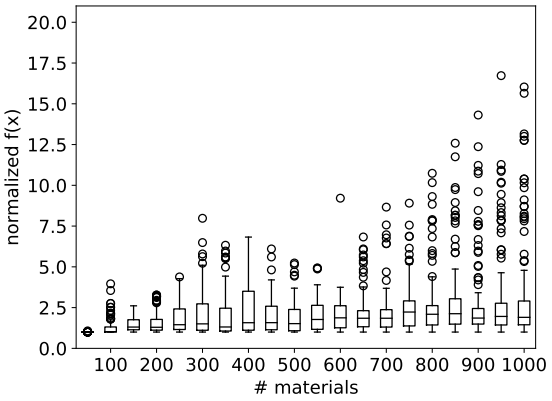
Figure 7.1: Methods comparison for each instance.

the convergence in the real dataset and Figure 7.3b presents the mean convergence of all synthetic datasets. PPA and PSO achieve worse results than DE and GA in the real-world dataset, which contains 284 learning materials. However, PSO reaches the best objective value from 75,000 evaluations in the synthetic dataset, but DE obtained better results from approximately 7,000 to 75,000 objective function evaluations.

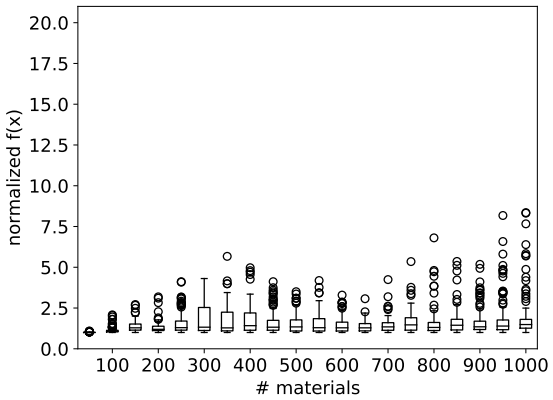
The problem modeling considers different characteristics of each solution with multiple objective functions but the optimization only considers the weighed sum of them. However it is important to consider not only the final result but which objectives the methods were capable to solve. Figure 7.4 shows the convergence of each objective function for DE. The other methods performed similarly. In Figure 7.4a, it is shown the objective value for the real-world dataset and Figures 7.4b, 7.4c, 7.4d 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



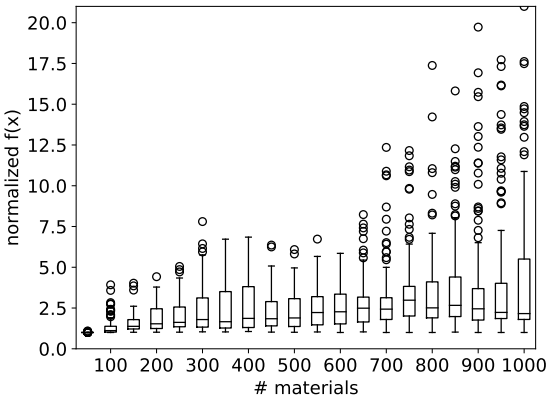
(a) Results obtained by DE.



(b) Results obtained by GA.

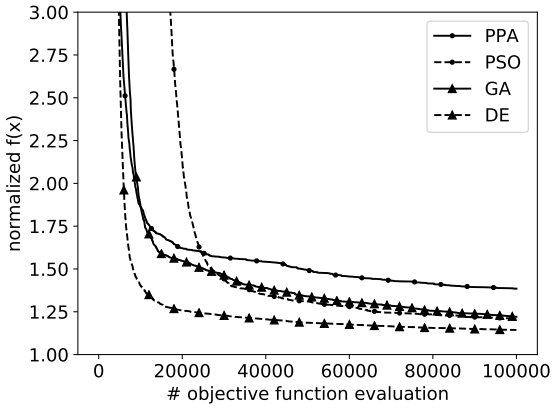


(c) Results obtained by PSO.

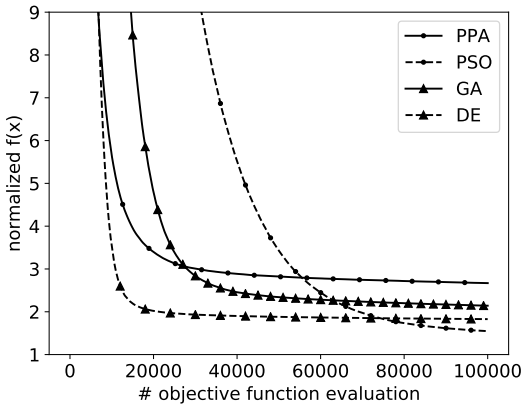


(d) Results obtained by PPA.

Figure 7.2: Boxplots of the results found by DE, GA, PSO, and PPA.



(a) Real-world dataset.

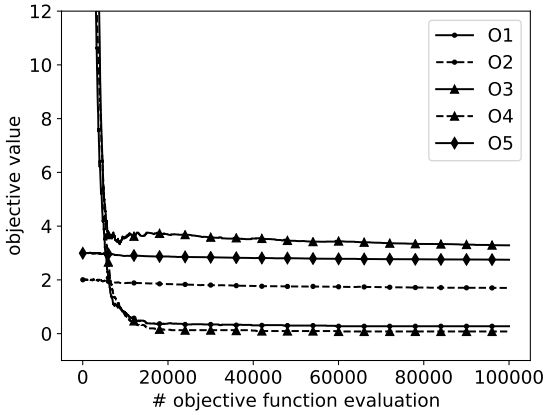


(b) Synthetic dataset.

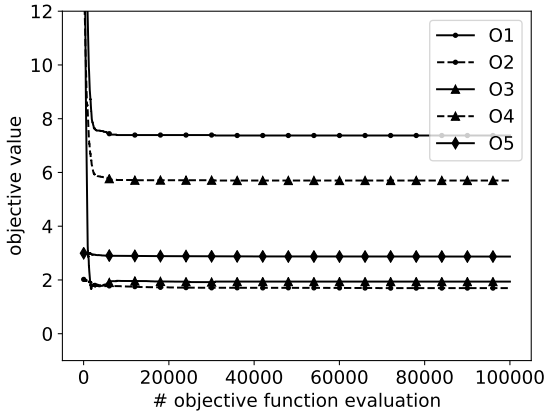
Figure 7.3: Methods convergence comparison.

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

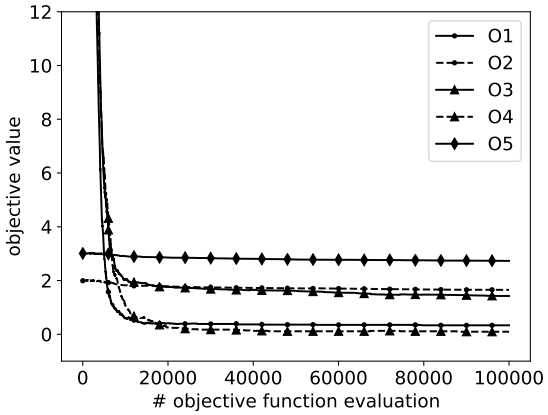
are objectives that consider the individual characteristics of each student. None of the algorithm compared in this work were able to find good solutions for this two objectives.



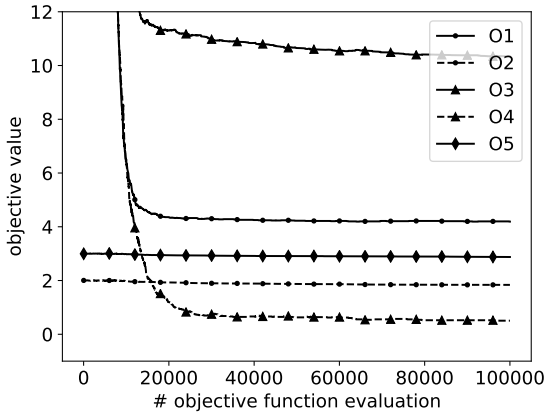
(a) Real-world dataset with 284 learning materials.



(b) Synthetic dataset with 50 learning materials.



(c) Synthetic dataset with 300 learning materials.



(d) Synthetic dataset with 700 learning materials.

Figure 7.4: Average of the functions ($O_i(x)$) that compose the objective function ($f(x)$) calculated during the optimization process considering every student.

7.3 Discussion

One of the points raised in Machado, Barrére e Souza (2019) was that PPA could have better results than GA or PSO and further comparison was needed. The results presented here showed that the only situation where PPA obtained better results was with very low budget. 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 found in Chu, Chang e Tsai (2009) and Li et al. (2012) where PSO requires more budget to find good solutions in terms of objective function evaluation.

Although little explored in literature, DE obtained good solutions 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, decreasing 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.

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 the amount of research on ACS, few comparisons are analyzing which algorithms can obtain the best results. Given the diversity of modeling for the problem and the lack of available databases 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, large number of students, different distribution of materials and student features).

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 small datasets. 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 the 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.

Bibliography

ACAMPORA, G.; GAETA, M.; LOIA, V. Hierarchical optimization of personalized experiences for e-learning systems through evolutionary models. *Neural Computing and Applications*, Springer, v. 20, n. 5, p. 641–657, 2011.

AGARWAL, S. et al. Intuitionistic fuzzy ant colony optimization for course sequencing in e-learning. In: IEEE. *2016 Ninth International Conference on Contemporary Computing (IC3)*. [S.l.], 2016. p. 1–6.

AL-MUHAIDEB, S.; MENAI, M. E. B. Evolutionary computation approaches to the curriculum sequencing problem. *Natural Computing*, Springer, v. 10, n. 2, p. 891–920, 2011.

BRUSILOVSKY, P. Adaptive and intelligent technologies for web-based education. *International Journal of Artificial Intelligence in Education (IJAIED)*, v. 13, n. 4, p. 159–172, 2003.

CHANDAR, S. A. et al. Personalized e-course composition approach using digital pheromones in improved particle swarm optimization. In: IEEE. *2010 Sixth International Conference on Natural Computation*. [S.l.], 2010. v. 5, p. 2677–2681.

CHANG, T.-Y.; KE, Y.-R. A personalized e-course composition based on a genetic algorithm with forcing legality in an adaptive learning system. *Journal of Network and Computer Applications*, Elsevier, v. 36, n. 1, p. 533–542, 2013.

CHEN, C.-M. Intelligent web-based learning system with personalized learning path guidance. *Computers & Education*, Elsevier, v. 51, n. 2, p. 787–814, 2008.

CHRISTUDAS, B. C. L.; KIRUBAKARAN, E.; THANGAIAH, P. R. J. An evolutionary approach for personalization of content delivery in e-learning systems based on learner behavior forcing compatibility of learning materials. *Telematics and Informatics*, Elsevier, v. 35, n. 3, p. 520–533, 2018.

CHU, C.-P.; CHANG, Y.-C.; TSAI, C.-C. Pc2pso: personalized e-course composition based on particle swarm optimization. *Applied Intelligence*, Springer, v. 34, n. 1, p. 141–154, 2009.

CHU, C.-P.; CHANG, Y.-C.; TSAI, C.-C. Pc 2 pso: personalized e-course composition based on particle swarm optimization. *Applied Intelligence*, Springer, v. 34, n. 1, p. 141–154, 2011.

DAVIS, D. et al. Gauging mooc learners' adherence to the designed learning path. In: *EDM '16: 9th International Conference on Educational Data Mining*. [S.l.: s.n.], 2016. p. ...

DEBBAH, A.; ALI, Y. M. B. Solving the curriculum sequencing problem with dna computing approach. *International Journal of Distance Education Technologies (IJDET)*, IGI Global, v. 12, n. 4, p. 1–18, 2014.

EBERHART, R.; KENNEDY, J. A new optimizer using particle swarm theory. In: IEEE. *MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science*. [S.l.], 1995. p. 39–43.

FELDER, R. M.; SILVERMAN, L. K. et al. Learning and teaching styles in engineering education. *Engineering education*, v. 78, n. 7, p. 674–681, 1988.

GAO, Y. et al. A multi-objective pso with pareto archive for personalized e-course composition in moodle learning system. In: IEEE. *Computational Intelligence and Design (ISCID), 2015 8th International Symposium on*. [S.l.], 2015. v. 2, p. 21–24.

GUO, Q.; ZHANG, M. Implement web learning environment based on data mining. *Knowledge-Based Systems*, Elsevier, v. 22, n. 6, p. 439–442, 2009.

GUTIÉRREZ, S.; PARDO, B. Sequencing in web-based education: approaches, standards and future trends. In: *Evolution of Teaching and Learning Paradigms in Intelligent Environment*. [S.l.]: Springer, 2007. p. 83–117.

HAFIDI, M.; BENSEBAA, T. Architecture for an adaptive and intelligent tutoring system that considers the learner's multiple intelligences. *International Journal of Distance Education Technologies (IJDET)*, IGI Global, v. 13, n. 1, p. 1–21, 2015.

HNIDA, M.; IDRISSI, M. K.; BENNANI, S. Adaptive teaching learning sequence based on instructional design and evolutionary computation. In: IEEE. *Information Technology Based Higher Education and Training (ITHET), 2016 15th International Conference on*. [S.l.], 2016. p. 1–6.

HUANG, M.-J.; HUANG, H.-S.; CHEN, M.-Y. Constructing a personalized e-learning system based on genetic algorithm and case-based reasoning approach. *Expert Systems with Applications*, Elsevier, v. 33, n. 3, p. 551–564, 2007.

KARADENIZ Şirin. Flexible design for the future of distance learning. *Procedia - Social and Behavioral Sciences*, v. 1, n. 1, p. 358 – 363, 2009. World Conference on Educational Sciences: New Trends and Issues in Educational Sciences.

KARDAN, A. A.; AZIZ, M.; SHAHPASAND, M. Adaptive systems: a content analysis on technical side for e-learning environments. *Artificial Intelligence Review*, Springer, v. 44, n. 3, p. 365–391, 2015.

KENNEDY, J.; EBERHART, R. C. A discrete binary version of the particle swarm algorithm. In: IEEE. *1997 IEEE International conference on systems, man, and cybernetics. Computational cybernetics and simulation*. [S.l.], 1997. v. 5, p. 4104–4108.

KHAMPARIA, A.; PANDEY, B. Knowledge and intelligent computing methods in e-learning. *International Journal of Technology Enhanced Learning*, Inderscience Publishers (IEL), v. 7, n. 3, p. 221–242, 2015.

KUZILEK, J.; HLOSTA, M.; ZDRAHAL, Z. Open university learning analytics dataset. *Scientific data*, Nature Publishing Group, v. 4, p. 170171, 2017.

LI, J.-W. et al. A self-adjusting e-course generation process for personalized learning. *Expert Systems with Applications*, Elsevier, v. 39, n. 3, p. 3223–3232, 2012.

MACHADO, M. d. O. C.; BARRÉRE, E.; SOUZA, J. Solving the adaptive curriculum sequencing problem with prey-predator algorithm. *International Journal of Distance Education Technologies (IJDET)*, IGI Global, v. 17, n. 4, p. 71–93, 2019.

MARCOS, L. de et al. A new method for domain independent curriculum sequencing: a case study in a web engineering master program. *International Journal of Engineering Education*, v. 25, n. 4, p. 632, 2009.

MARCOS, L. de; MARTÍNEZ, J.-J.; GUTIÉRREZ, J.-A. Swarm intelligence in e-learning: a learning object sequencing agent based on competencies. In: ACM. *Proceedings of the 10th annual conference on Genetic and evolutionary computation*. [S.l.], 2008. p. 17–24.

MARCOS, L. de et al. An evolutionary approach for domain independent learning object sequencing. In: SPRINGER. *World Summit on Knowledge Society*. [S.l.], 2008. p. 192–197.

MARCOS, L. de et al. Genetic algorithms for courseware engineering. *International Journal of Innovative Computing, Information and Control*, v. 7, n. 7, p. 1–27, 2011.

MENAI, M. E.; ALHUNITAH, H.; AL-SALMAN, H. Swarm intelligence to solve the curriculum sequencing problem. *Computer Applications in Engineering Education*, Wiley Online Library, v. 26, n. 5, p. 1393–1404, 2018.

MUHAMMAD, A. et al. Learning path adaptation in online learning systems. In: IEEE. *Computer Supported Cooperative Work in Design (CSCWD), 2016 IEEE 20th International Conference on*. [S.l.], 2016. p. 421–426.

NIKNAM, M.; THULASIRAMAN, P. Lpr: A bio-inspired intelligent learning path recommendation system based on meaningful learning theory. *Education and Information Technologies*, Springer, p. 1–23, 2020.

NWANA, H. S. Intelligent tutoring systems: an overview. *Artificial Intelligence Review*, Springer, v. 4, n. 4, p. 251–277, 1990.

PHOBUN, P.; VICHEANPANYA, J. Adaptive intelligent tutoring systems for e-learning systems. *Procedia-Social and Behavioral Sciences*, Elsevier, v. 2, n. 2, p. 4064–4069, 2010.

PIRES, J. M.; COTA, M. P. "intelligent" adaptive learning objects applied to special education needs: Extending the elearning paradigm to the ulearning environment. In: IEEE. *2016 11th Iberian Conference on Information Systems and Technologies (CISTI)*. [S.l.], 2016. p. 1–6.

PUSHPA, M. Aco in e-learning: Towards an adaptive learning path. *International Journal on Computer Science and Engineering*, Engg Journals Publications, v. 4, n. 3, p. 458, 2012.

RATHORE, A. S.; ARJARIA, S. K. Intelligent tutoring system. In: *Utilizing Educational Data Mining Techniques for Improved Learning: Emerging Research and Opportunities*. [S.l.]: IGI Global, 2020. p. 121–144.

SEKI, K.; MATSUI, T.; OKAMOTO, T. An adaptive sequencing method of the learning objects for the e-learning environment. *Electronics and Communications in Japan (Part III: Fundamental Electronic Science)*, Wiley Online Library, v. 88, n. 3, p. 54–71, 2005.

SHARMA, R.; BANATI, H.; BEDI, P. Adaptive content sequencing for e-learning courses using ant colony optimization. In: SPRINGER. *Proceedings of the International Conference on Soft Computing for Problem Solving (SocProS 2011) December 20-22, 2011*. [S.l.], 2012. p. 579–590.

SHMELEV, V.; KARPOVA, M.; DUKHANOV, A. An approach of learning path sequencing based on revised bloom's taxonomy and domain ontologies with the use of genetic algorithms. *Procedia Computer Science*, Elsevier, v. 66, p. 711–719, 2015.

SILVA, R. C. et al. Adaptability of learning objects using calibration and adaptive sequencing of exercises. *Brazilian Journal of Computers in Education*, v. 26, n. 01, p. 70, 2018.

SOLOMON, B. A.; FELDER, R. M. Index of learning styles. *Raleigh, NC: North Carolina State University. Available online*, 1999.

STORN, R. Differrential evolution-a simple and efficient adaptive scheme for global optimization over continuous spaces. *Technical report, International Computer Science Institute*, v. 11, 1995.

TILAHUN, S. L.; ONG, H. C. Prey-predator algorithm: A new metaheuristic algorithm for optimization problems. *International Journal of Information Technology & Decision Making*, World Scientific, v. 14, n. 06, p. 1331–1352, 2015.

WAN, S.; NIU, Z. A learner oriented learning recommendation approach based on mixed concept mapping and immune algorithm. *Knowledge-Based Systems*, Elsevier, v. 103, p. 28–40, 2016.

WANG, S.-L.; WU, C.-Y. Application of context-aware and personalized recommendation to implement an adaptive ubiquitous learning system. *Expert Systems with applications*, Elsevier, v. 38, n. 9, p. 10831–10838, 2011.

WEBER, G.; SPECHT, M. User modeling and adaptive navigation support in www-based tutoring systems. In: SPRINGER. *User Modeling*. [S.l.], 1997. p. 289–300.

WONG, L.-H.; LOOI, C.-K. 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, Hershey: IGI Global*, p. 285–302, 2010.

XIE, H. et al. Discover learning path for group users: A profile-based approach. *Neurocomputing*, Elsevier, v. 254, p. 59–70, 2017.