
**An Approach to the Personalized Learning
Objects Recommendation Problem as a Set
Covering Problem Using Ontologies and
Metaheuristics**

Clarivando Francisco Belizário Júnior



FEDERAL UNIVERSITY OF UBERLÂNDIA
FACULTY OF COMPUTER SCIENCE
GRADUATE PROGRAM IN COMPUTER SCIENCE

Uberlândia
2024

Clarivando Francisco Belizário Júnior

**An Approach to the Personalized Learning
Objects Recommendation Problem as a Set
Covering Problem Using Ontologies and
Metaheuristics**

Ph.D. Thesis presented to the Graduate Program of the Faculty of Computer Science of the Federal University of Uberlândia as part of the requirements for obtaining the title of Doctor in Computer Science.

Area of concentration: Computer Science

Supervisor: Prof. Dr. Fabiano Azevedo Dorça

Co-supervisor:

Prof. Dr. Alessandro Vivas Andrade
(UFVJM)

Prof. Dr. Luciana Pereira de Assis (UFVJM)

Uberlândia

2024

Ficha Catalográfica Online do Sistema de Bibliotecas da UFU
com dados informados pelo(a) próprio(a) autor(a).

B431
2024

Belizário Júnior, Clarivando Francisco, 1987-
An Approach to the Personalized Learning Objects
Recommendation Problem as a Set Covering Problem Using
Ontologies and Metaheuristics [recurso eletrônico] /
Clarivando Francisco Belizário Júnior. - 2024.

Orientador: Fabiano Azevedo Dorça.

Coorientadora: Luciana Pereira de Assis.

Coorientador: Alessandro Vivas Andrade.

Tese (Doutorado) - Universidade Federal de Uberlândia,
Pós-graduação em Ciência da Computação.

Modo de acesso: Internet.

Disponível em: <http://doi.org/10.14393/ufu.te.2024.554>

Inclui bibliografia.

1. Computação. I. Dorça, Fabiano Azevedo,1979-,
(Orient.). II. Assis, Luciana Pereira de,1981-,
(Coorient.). III. Andrade, Alessandro Vivas,1973-,
(Coorient.). IV. Universidade Federal de Uberlândia.
Pós-graduação em Ciência da Computação. V. Título.

CDU: 681.3

Bibliotecários responsáveis pela estrutura de acordo com o AACR2:

Gizele Cristine Nunes do Couto - CRB6/2091
Nelson Marcos Ferreira - CRB6/3074



UNIVERSIDADE FEDERAL DE UBERLÂNDIA
Coordenação do Programa de Pós-Graduação em Ciência da
Computação

Av. João Naves de Ávila, 2121, Bloco 1A, Sala 243 - Bairro Santa Mônica, Uberlândia-MG,
CEP 38400-902

Telefone: (34) 3239-4470 - www.ppgco.facom.ufu.br - cpgrafacom@ufu.br



ATA DE DEFESA - PÓS-GRADUAÇÃO

Programa de Pós-Graduação em:	Ciência da Computação				
Defesa de:	Tese, 25/2024, PPGCO				
Data:	29 de julho de 2024	Hora de início:	14:00	Hora de encerramento:	17:45
Matrícula do Discente:	11923CCP002				
Nome do Discente:	Clarivando Francisco Belizário Júnior				
Título do Trabalho:	An Approach to the Personalized Learning Objects Recommendation Problem as a Set Covering Problem Using Ontologies and Metaheuristics				
Área de concentração:	Ciência da Computação				
Linha de pesquisa:	Inteligência Artificial				
Projeto de Pesquisa de vinculação:	Chamada CNPq/MCTI/FNDCT Nº 18/2021 (Universal 2021) Processo: 402431/2021-9				

Reuniu-se por videoconferência, a Banca Examinadora, designada pelo Colegiado do Programa de Pós-graduação em Ciência da Computação, assim composta: Professores Doutores: Luciana Pereira de Assis - Departamento de Computação/UFVJM (Coorientadora), Alessandro Vivas Andrade - Sistemas de Informação/UFVJM (Coorientador), Márcia Aparecida Fernandes - FACOM/UFU, Renan Gonçalves Cattelan - FACOM/UFU, José Palazzo Moreira de Oliveira - INF/UFRGS, Evandro de Barros Costa - IC/UFAL e Fabiano Azevedo Dorça - FACOM/UFU, orientador do candidato.

Os examinadores participaram desde as seguintes localidades: José Palazzo Moreira de Oliveira - Porto Alegre/RS, Evandro de Barros Costa - Maceió/AL, Alessandro Vivas Andrade - Diamantina/MG e Luciana Pereira de Assis - Diamantina/MG. Os outros membros da banca e o aluno participaram da cidade de Uberlândia.

Iniciando os trabalhos o presidente da mesa, Prof. Dr. Fabiano Azevedo Dorça, apresentou a Comissão Examinadora e o candidato, agradeceu a presença do público, e concedeu ao Discente a palavra para a exposição do seu trabalho. A duração da apresentação do Discente e o tempo de arguição e resposta foram conforme as normas do Programa.

A seguir o senhor presidente concedeu a palavra, pela ordem sucessivamente, aos examinadores, que passaram a arguir o candidato. Ultimada a arguição, que se desenvolveu dentro dos termos regimentais, a Banca, em sessão secreta, atribuiu o resultado final, considerando o candidato:


Aprovado

Esta defesa faz parte dos requisitos necessários à obtenção do título de Doutor.


O competente diploma será expedido após cumprimento dos demais requisitos, conforme as normas do Programa, a legislação pertinente e a regulamentação interna da UFU.

Nada mais havendo a tratar foram encerrados os trabalhos. Foi lavrada a presente ata que após lida e achada conforme foi assinada pela Banca Examinadora.


Referência: Processo nº 23117.045095/2024-64
SEI nº 5588455

Documento assinado digitalmente
 **FABIANO AZEVEDO DORCA**
Data: 07/08/2024 10:06:54-0300
Verifique em <https://validar.iti.gov.br>


Fabiano Azevedo Dorça - FACOM/UFU (Orientador)

Documento assinado digitalmente
 **LUCIANA PEREIRA DE ASSIS**
Data: 07/08/2024 17:13:36-0300
Verifique em <https://validar.iti.gov.br>


Luciana Pereira de Assis - Departamento de Computação/UFVJM (Coorientadora)

Documento assinado digitalmente
 **ALESSANDRO VIVAS ANDRADE**
Data: 08/08/2024 09:48:54-0300
Verifique em <https://validar.iti.gov.br>


Alessandro Vivas Andrade - Sistemas de Informação/UFVJM (Coorientador)

Documento assinado digitalmente
 **MARCIA APARECIDA FERNANDES**
Data: 08/08/2024 14:06:28-0300
Verifique em <https://validar.iti.gov.br>


Márcia Aparecida Fernandes – FACOM/UFU

Documento assinado digitalmente
 **RENAN GONCALVES CATTELAN**
Data: 08/08/2024 15:43:24-0300
Verifique em <https://validar.iti.gov.br>

Renan Gonçalves Cattelan - FACOM/UFU

Documento assinado digitalmente
 **JOSE PALAZZO MOREIRA DE OLIVEIRA**
Data: 12/08/2024 11:17:27-0300
Verifique em <https://validar.iti.gov.br>

José Palazzo Moreira de Oliveira - INF/UFRGS,

Documento assinado digitalmente
 **EVANDRO DE BARROS COSTA**
Data: 14/08/2024 12:11:07-0300
Verifique em <https://validar.iti.gov.br>

Evandro de Barros Costa - IC/UFAL

FEDERAL UNIVERSITY OF UBERLÂNDIA – UFU
FACULTY OF COMPUTER SCIENCE – FACOM
GRADUATE PROGRAM IN COMPUTER SCIENCE – PPGCO

The undersigned hereby certify they have read and recommend to the PPGCO for acceptance the thesis entitled “**An Approach to the Personalized Learning Objects Recommendation Problem as a Set Covering Problem Using Ontologies and Metaheuristics**” submitted by “**Clarivando Francisco Belizário Júnior**” as part of the requirements for obtaining the **degree of Doctor in Computer Science**.

Uberlândia, July 29, 2024

Supervisor: -----

Prof. Dr. Fabiano Azevedo Dorça
Universidade Federal de Uberlândia (UFU)

Cosupervisor: -----

Profa. Dra. Luciana Pereira de Assis
Universidade Federal dos Vales do Jequitinhonha e Mucuri (UFVJM)

Cosupervisor: -----

Prof. Dr. Alessandro Vivas Andrade
Universidade Federal dos Vales do Jequitinhonha e Mucuri (UFVJM)

Examining Committee Members:

Prof. Dr. Evandro de Barros Costa
Universidade Federal de Alagoas (UFAL)

Prof. Dr. José Palazzo Moreira de Oliveira
Universidade Federal do Rio Grande do Sul (UFRGS)

Profa. Dra. Márcia Aparecida Fernandes
Universidade Federal de Uberlândia (UFU)

Prof. Dr. Renan Gonçalves Cattelan
Universidade Federal de Uberlândia (UFU)

Acknowledgments

I thank God and my blessed family for the strength that supports us. Additionally, I would like to thank my advisors, colleagues, and family for their unwavering support and encouragement.

I thank my supervisor Fabiano and my co-supervisors Alessandro and Luciana for their shared knowledge, contributing to my intellectual and moral progress.

I thank the members of the board for their contribution to the improvement of this work.

I would like to thank Professor Renan Gonçalves Cattelan, who kindly accepted the participation of his procedural programming class in this research, contributing with guidance for the correct development of practical experiments.

I thank my colleague Arthur Henrique de Souza, a Scientific Initiation student, who helped implement the Anya chatbot with gamification.

I thank CAPES for the scholarship granted to carry out this research. This study was financed in part by the Coordination for the Improvement of Higher Education Personnel—Brazil (CAPES)—Finance Code 001, and it was supported by the National Council for Scientific and Technological Development (CNPq; Grant 402431/2021-9).

Finally, I thank all the people who with their qualities inspired me to become a better person.

Abstract

Recommender Systems are extensively utilized in e-commerce platforms, such as sales websites and Netflix, to intelligently suggest products, movies, and series tailored to the user's preferences. In the context of education, the key challenge for these systems is to provide personalized recommendations of educational content that align with students' needs, considering their knowledge levels, learning styles, and cognitive preferences. This work implements a recommender system designed to suggest learning objects across various areas of knowledge, integrating small learning objects, called interventions, such as definitions, examples, and hints. To personalize these recommendations, the Learning Objects Recommendation Problem is formulated as a set-covering problem, which belongs to the class of NP-Hard problems. A heuristic search-based algorithm was proposed and compared with other metaheuristics, resulting in a promising approach to solving this problem, as demonstrated by the results. The proposed solution aims to minimize the challenges of cold-start and rating sparsity, common in traditional recommender systems, by using advanced collaborative filtering techniques and an ontology that models the students' needs, knowledge, learning styles, and search parameters. Additionally, the recommender system was implemented with a chatbot and tested for recommending content on the C programming language for first-year students of the Computer Science course, using gamification to alleviate possible pedagogical difficulties in the teaching-learning process.

Keywords: Learning objects recommendation. Collaborative filtering. Learning styles. Ontology. Set covering. Recommender system. Chatbot. Gamification.

Resumo

Sistemas de Recomendação são amplamente utilizados em plataformas de e-commerce, como sites de vendas e Netflix, para sugerir de forma inteligente produtos, filmes e séries adaptados às preferências do usuário. No contexto educacional, o principal desafio desses sistemas é fornecer recomendações personalizadas de conteúdo educacional que estejam alinhadas com as necessidades dos estudantes, considerando seus níveis de conhecimento, estilos de aprendizagem e preferências cognitivas. Este trabalho implementa um sistema de recomendação projetado para sugerir objetos de aprendizagem em várias áreas do conhecimento, integrando pequenos objetos de aprendizagem, chamados de intervenções, como definições, exemplos e dicas. Para personalizar essas recomendações, o Problema de Recomendação de Objetos de Aprendizagem é formulado como um problema de cobertura de conjunto, que pertence à classe dos problemas NP-Hard. Um algoritmo baseado em busca heurística foi proposto e comparado com outras meta-heurísticas, resultando em uma abordagem promissora para resolver este problema, conforme demonstrado pelos resultados. A solução proposta visa minimizar os problemas cold-start e esparsidade de ratings, comuns em sistemas de recomendação tradicionais, utilizando técnicas avançadas de filtragem colaborativa e uma ontologia que modela as necessidades, conhecimentos, estilos de aprendizagem e parâmetros de busca dos estudantes. Além disso, o sistema de recomendação foi implementado com um chatbot e testado para recomendar conteúdo sobre a linguagem de programação C para estudantes do primeiro ano do curso de Ciência da Computação, utilizando gamificação para diminuir possíveis dificuldades pedagógicas no processo de ensino-aprendizagem.

Palavras-chave: Recomendação de objetos de aprendizagem. Filtragem colaborativa. Estilos de aprendizagem. Ontologia. Cobertura de conjuntos. Sistema de recomendação. Chatbot. Gamificação.

List of Figures

Figure 1 – The Semantic Web Stack	30
Figure 2 – Traditional architecture of an ITS	34
Figure 3 – Rating matrix of CF and KB recommendation	36
Figure 4 – Overview of the proposed recommender system	51
Figure 5 – Ontology classes and subclasses	53
Figure 6 – Domain model in the ontology	55
Figure 7 – Learner model in the ontology	56
Figure 8 – The subclasses of LearningObject class	58
Figure 9 – Relationship (hasLearningObject) between intent and hint-type LOs . .	59
Figure 10 – Ideal LO	60
Figure 11 – Input matrix and cost vector	62
Figure 12 – How heuristic_search procedure works for $top_k = [9, 7, 5, 8, 15]$	67
Figure 13 – Recommendation of LOs in the Anya chatbot	73
Figure 14 – Evaluation of LOs in the Anya chatbot	74
Figure 15 – Learner profile and gamification dashboard	74
Figure 16 – Gamification questions/quizzes	75
Figure 17 – Component diagram of the recommender system	76
Figure 18 – Sequence diagram of the login process	77
Figure 19 – Boxplot to compare the versions of each algorithm with (Yes) and with- out (No) prediction	85
Figure 20 – Boxplot to compare the versions of each algorithm with (Yes) and with- out (No) the I (intervention) variable	87
Figure 21 – Comparison of algorithm runtimes when the less interventions the better	90
Figure 22 – Comparison of algorithm runtimes when the more interventions the better	90
Figure 23 – Class learning style	91
Figure 24 – Average scores for learning styles	92
Figure 25 – Learner satisfaction with the usability of RS and chatbot	92
Figure 26 – Educational content with interventions (Part 1 of 3)	113

Figure 27 – Educational content with interventions (Part 2 of 3)	114
Figure 28 – Educational content with interventions (Part 3 of 3)	115
Figure 29 – Ontosearch API - Swagger documentation (Part 1 of 2)	117
Figure 30 – Ontosearch API - Swagger documentation (Part 2 of 2)	118
Figure 31 – Edulors API - Swagger documentation (Part 1 of 3)	119
Figure 32 – Edulors API - Swagger documentation (Part 2 of 3)	120
Figure 33 – Edulors API - Swagger documentation (Part 3 of 3)	121
Figure 34 – Usability and satisfaction at the end of the term	123
Figure 35 – Processing a Wikipedia summary for concept generation	139

List of Tables

Table 1 – Categories of the IEEE-LOM standard	32
Table 2 – Comparison of related literature with the proposal of this research . . .	44
Table 3 – Resource types recommended for each learner profile	57
Table 4 – Two new SWRL rules	61
Table 5 – Chatbot functions/intents in Gamification	70
Table 6 – Actions and rewards	71
Table 7 – Gamification levels	71
Table 8 – Achievements (badges)	72
Table 9 – The main features of the used benchmark instances	80
Table 10 – Rating matrix used in testing	82
Table 11 – Comparison of the average ratings in the solutions with (Yes) and without (No) the P variable in solving the LORP	84
Table 12 – Estimated difference in average performance between the prediction variables (No/Yes)	84
Table 13 – Comparison of the average number of interventions in the solutions with (Yes) and without (No) the I variable in solving the LORP	86
Table 14 – Estimated difference in average performance between the intervention variables (No/Yes)	86
Table 15 – Comparison between the objective functions of the selected algorithms .	89
Table 16 – Wikipedia quality classes	138

Acronyms list

ACO Ant Colony Optimization

ASCII American Standard Code for Information Interchange

API Application Programming Interface

CBF Content-based Filtering

CF Collaborative Filtering

CGA Compatible Genetic Algorithm

CLEO Customized Learning Experience Online

FSLSM Felder-Silverman Learning Style Model

GA Genetic Algorithm

ITS Intelligent Tutoring System

IRI Internationalized Resource Identifier

ILS Index of Learning Styles

JPSO Jumping Particle Swarm Optimization

KB Knowledge-based

LO Learning Object

LORP Learning Objects Recommendation Problem

NLP Natural Language Processing

OWL Web Ontology Language

PSO Particle Swarm Optimization

RDF Resource Definition Framework

RS Recommender System

SWRL Semantic Web Rule Language

SW Semantic Web

SCP Set Covering Problem

SQL Structured Query Language

SPM Sequential Pattern Mining

SCORM Sharable Content Objects Reference Model

SDT Self-Determination Theory

SQL Structured Query Language

URI Uniform Resource Identifier

UML Unified Modeling Language

XML eXtensible Markup Language

Contents

1	INTRODUCTION	25
1.1	Motivation	26
1.2	Objectives	26
1.3	Hypotheses	27
1.4	Contributions	28
1.5	Thesis organization	28
2	FUNDAMENTALS	29
2.1	Semantic Web	29
2.2	Modeling learning objects	31
2.3	Modeling students	33
2.4	Intelligent systems and pedagogical interventions	34
2.4.1	Intelligent Tutoring System	34
2.4.2	Recommender system	36
2.4.3	Pedagogical interventions in intelligent systems	37
2.5	Chatbot and gamification	38
2.6	Set Covering Problem	39
3	RELATED WORKS	41
4	PROPOSED APPROACH	49
4.1	Ontology	53
4.1.1	Domain model	54
4.1.2	Learner model	55
4.1.3	SWRL rules	56
4.2	Improvements in Ontology for individualized intervention recommendation	57
4.2.1	QualityLOs subclass	58

4.2.2	Hint-type LOs	58
4.2.3	Ideal LO	59
4.2.4	New SWRL rules	60
4.3	Learning Object Recommendation Problem defined as a Set Covering Problem	61
4.4	Improvements in cost calculation	62
4.5	Proposed heuristic algorithm	65
4.6	Anya chatbot and gamification	70
4.7	Modeling the recommender system	76
5	EXPERIMENTAL RESULTS	79
5.1	Computational results and comparison of algorithms	79
5.1.1	Benchmark instances and dataset	79
5.1.2	Selected algorithms and parameters	81
5.1.3	Comparison of the versions of each algorithm with and without prediction	83
5.1.4	Comparison of the versions of each algorithm with and without inter- vention variable	85
5.1.5	Comparison between algorithms	88
5.2	Learner testing results with the recommender system	90
5.2.1	Learning styles of the student class	91
5.2.2	Learner satisfaction measure	92
6	FINAL REMARKS	95
6.1	Main Contributions	96
6.2	Limitations and Future Work	96
6.3	Contributions to Bibliographic Production	98
6.4	Acknowledgments	99
	BIBLIOGRAPHY	101

APPENDIX 111

APPENDIX A	– GROUPED INTERVENTIONS	113
APPENDIX B	– RS DOCUMENTATION	117
APPENDIX C	– QUESTIONNAIRE	123

ANNEX 125

ANNEX A	– INDEX OF LEARNING STYLES QUESTIONNAIRE	127
----------------	---	------------

ANNEX B	– SWRL RULES	133
ANNEX C	– METADATA GENERATION FOR WIKI SECTIONS	137
C.1	Quality assessment of wiki sections	137
C.2	Identification of wiki section concepts	139

Introduction

Recommender Systems are widely used in e-commerce, such as sales sites and Netflix, for the intelligent recommendation of products, movies and series based on the user's preferences. In the teaching and learning environment, the challenge of these systems is the personalized recommendation of educational content that meets students' needs according to their knowledge level, and learning and cognitive styles.

This educational content, technically called Learning Object (LO), can be any content that the student uses to learn, such as texts, images, videos, exercises, animations, wiki pages and slides. Several studies (ABECH *et al.*, 2016; TARUS; NIU; KHADIDJA, 2017; RAMIREZ-ARELLANO; BORY-REYES; HERNÁNDEZ-SIMÓN, 2017; TARUS; NIU; KALUI, 2018; CHRISTUDAS; KIRUBAKARAN; THANGAIAH, 2018) show that students can learn more and in less time through good personalized educational content recommender systems. These systems try to suggest the best LOs considering the student's characteristics, thus reducing possible pedagogical difficulties.

We propose a Recommender System (RS) to recommend refined learning objects from various fields of knowledge, enhancing the system's ability to provide personalized and effective learning support tailored to each student's specific needs. The Learning Objects Recommendation Problem (LORP) is a challenge inherent in these systems. It is treated in the literature by several techniques, the most used being content-based filtering (VANETTI *et al.*, 2010), collaborative filtering (MEDIO *et al.*, 2020) and the combination of two or more techniques (hybrid recommendation) (BURKE, 2007; BARRAGÁNS-MARTÍNEZ *et al.*, 2010; CHOI *et al.*, 2012; TARUS; NIU; YOUSIF, 2017). These techniques suffer from the rating sparsity (ZHAO *et al.*, 2015) and cold-start (ADOMAVICIUS; TUZHILIN, 2005) problems. In the e-learning environment, the rating sparsity problem occurs when few students have evaluated the same LO, and there is no overlap in the classification preferences. The cold-start problem occurs when it is not possible to make reliable recommendations due to the lack of initial assessments for new students or learning resources (ADOMAVICIUS; TUZHILIN, 2005).

We define the LORP as a problem whose goal is to address the previous drawbacks

by finding a coverage of LOs at minimum cost that includes all the concepts a student needs to learn. To solve the LORP, we employed metaheuristic algorithms, such as genetic algorithm and particle swarm optimization. Additionally, we used exact and greedy algorithms to explore potential solutions, ensuring a comprehensive approach to finding the most cost-effective coverage of learning objects.

We propose a RS based on a hybrid recommendation approach that uses an ontology (GRUBER, 1993) to model knowledge about students and learning resources, being able to recommend LOs from all areas of knowledge using fine-grained concepts, contributing to the state of the art. Furthermore, our approach implements a chatbot as a natural communication interface with students and uses gamification, making the teaching-learning process motivating and engaging.

Some works use ontology to model the knowledge about the students and learning resources (FERREIRA *et al.*, 2023; BAJENARU; BOROZAN; SMEUREANU, 2015; SHISHEHCHI *et al.*, 2012; MORENO *et al.*, 2013; RUOTSALO *et al.*, 2013), and in our study, it is also used to model fine-grained LOs (called interventions). In addition, the ontology stores the concepts that each LO covers, providing a fine-grained recommendation of LOs that cover the concepts that the student has not yet mastered, including subjects in which the learner has doubts, for which interventions will be recommended.

1.1 Motivation

The personalized recommendation of LOs is handled by RSs through filtering techniques. Currently, these techniques are combined to improve the recommendation of LOs, however, there are still bottlenecks in these systems, which are the cold-start and rating sparsity problems, in addition to the fact that they do not consider during the recommendation the fine-grained concepts that the student needs to learn.

In this work, we aim to alleviate these problems through a RS that combines ontology-based recommendation and collaborative filtering techniques for the recommendation of LOs based on fine-grained concepts and the reuse of Web content. The ontology models educational resources and student's knowledge level and profile, and it implements inference rules to help the recommendation process. Furthermore, the implemented RS uses a chatbot and gamification to engage and motivate students during the teaching-learning process.

1.2 Objectives

The main objective of this work is to implement a RS that effectively contributes to student learning through personalized recommendation of learning objects by addressing the Personalized Learning Objects Recommendation Problem as a Set Covering Problem

using ontologies and metaheuristics. The specific objectives for achieving this goal are listed below:

- a) Improve an ontology that stores the concepts that each LO covers, providing a fine-grained recommendation of LOs that cover the concepts that students have not yet mastered, including subjects in which they have doubts;
- b) Implement a fine-grained RS that is capable of recommending LOs from different areas of knowledge through the reuse of Web content and that meets the students' knowledge and learning style reducing cold-start and sparsity problems using knowledge-based recommendation;
- c) Implement a chatbot able to effectively dialogue with students to tutor their process of teaching and learning;
- d) Integrate gamification techniques into RS to engage and motivate students in the teaching-learning process;
- e) Validate the chatbot and the RS;
- f) Deploy the RS in Moodle, including its interface (chatbot), and evaluate the results obtained.

1.3 Hypotheses

The main hypotheses of this research and their respective questions are:

1. The use of collaborative filtering (variable P_j) and the incorporation of refined learning objects (interventions) in the calculation of the cost (c_j) of the Set Covering Problem (SCP) objective function significantly improve the quality and accuracy of LOs recommendations for students.
 - a) Does the application of collaborative filtering (variable P_j) facilitate the delivery of LOs with the highest ratings for students?
 - b) How does incorporating I (interventions: refined LOs) in the calculation of c_j within the objective function enhance the quality of LO recommendations concerning the number of interventions expected by students?
2. Recommended LOs meet the students' knowledge and learning style and are useful to assist students in learning and problem solving.
 - a) Were the LOs recommended to the students well evaluated by them?
 - b) Does the RS provide reliable materials?

1.4 Contributions

The main contribution of this work is to facilitate learning in face-to-face and remote teaching through a hybrid RS that combines collaborative filtering and ontology-based recommendation techniques to customize the delivery of LOs to the student according to their needs. In addition, the system will assist the student in solving exercises and in understanding the course content by recommending fine-grained LOs. Other important contributions are:

- a) The improvement of an ontology that stores the concepts covered by each LO, enabling detailed recommendations of LOs that address concepts the student has not yet mastered, including topics where the student has uncertainties;
- b) The formalization of LORP as a SCP that considers the user's search parameters, the collaborative filtering and fine-grained LOs, taking into account the concepts that the student needs to learn, while also mitigating cold-start and sparsity issues through knowledge-based recommendations;
- c) The personalized recommendation of fine-grained LOs (interventions, such as hints) from different areas of knowledge improving the RSs;
- d) Creation of an intelligent communication interface using a chatbot with gamification that will be able to tutor students, their learning and mediate more engaging dialogues, motivating and facilitating student learning and reducing possible pedagogical difficulties.

1.5 Thesis organization

The rest of this work is organized as follows. The background of this research is presented in Chapter 2. In Chapter 3, we discuss the related work relevant to this study. The proposed approach is detailed in Chapter 4. Chapter 5 is dedicated to the experimental results. Finally, the final remarks are outlined in Chapter 6.

Fundamentals

This section presents the most relevant concepts and theories that ground this study. In Section 2.1, we present a summary about the Semantic Web (SW), a technology that semantically represents the vast content of the traditional Web using ontologies. Ontologies structure entities and their relationships allowing the inference of new knowledge. In our approach, the ontology stores information about LOs and learners. Section 2.2 shows the educational standards used to structure information about the LOs, and Section 2.3 presents how students can be computationally modeled.

The main filtering and recommendation techniques that support RSs are presented in Section 2.4. Our approach uses the ontology-based recommendation technique, in which the domain model and the learner model are structured in an ontology. Our RS uses a chatbot with gamification as an interface. The terms chatbot and gamification are explained in Section 2.5. Finally, to understand the LORP as a covering problem, in Section 2.6, we describe the set covering problem and its usefulness in formalizing real-world problems.

2.1 Semantic Web

The Semantic Web (SW) (BERNERS-LEE; HENDLER; LASSILA, 2001), as the name suggests, extends the traditional Web through technologies that promote the modeling and manipulation of knowledge by machines (see Fig. 1). SW aims to give meaning to its content by structuring and assigning semantics to its data, allowing the interpretation, connection and availability of data tailored to the needs and interests of each user. The well-established layers of the SW are explained below:

- **Ontology - Web Ontology Language (OWL):** OWL is a semantic markup language for publishing and sharing ontologies on the Web. OWL is a vocabulary extension of Resource Definition Framework (RDF) and is derived from the DAML+OIL Web

Ontology Language, which is a syntax, built on top of RDF and eXtensible Markup Language (XML), that has been replaced by OWL.

- ❑ RDF and RDFS: is a fundamental language of the SW for expressing objects and their relationships. RDF Schema (RDFS) extends RDF using vocabulary with semantics for describing properties and classes.
- ❑ XML: is a markup language that provides a basic syntax for content structure within documents. It defines a set of rules for encoding documents in a human-readable and machine-readable format.
- ❑ URI/IRI: is a sequence of characters that uniquely identifies a logical or physical resource used by Web technologies. A Uniform Resource Identifier (URI) can be used to identify anything, including concepts, webpages, books and real-world objects, such as people and places. URIs are limited to the American Standard Code for Information Interchange (ASCII) and have few characters, and so the Internationalized Resource Identifier (IRI) extends the set of permitted characters of URIs by using the Universal Character Set (Unicode).

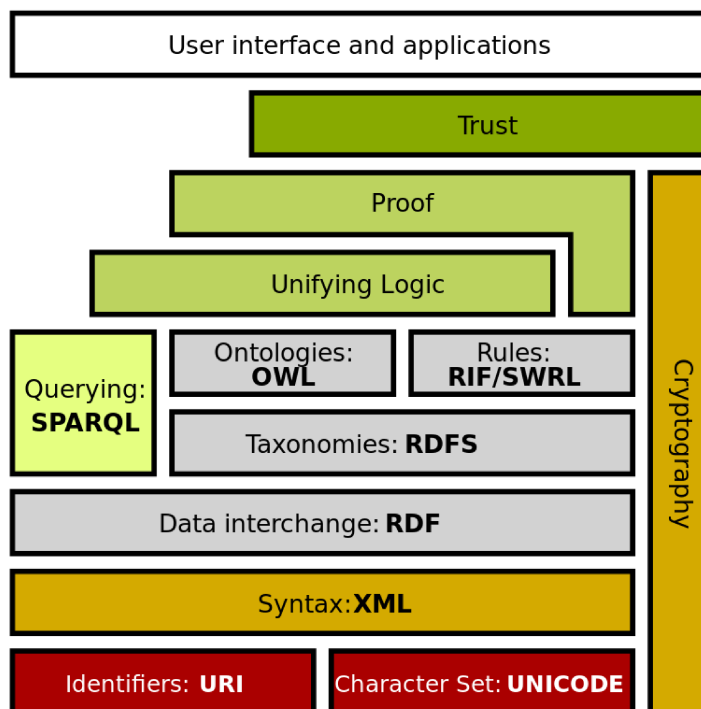


Figure 1 – The Semantic Web Stack

Source: https://en.wikipedia.org/wiki/Semantic_Web

The XML, RDF and OWL are used to represent information in the SW and make it interoperable between applications, with OWL being the language with the highest

knowledge representation potential and XML the language with less power of information representation. OWL elements extend RDF elements, which extend XML elements. Note that SW layers are supported by cryptography techniques to ensure information integrity.

Viewing this layer architecture makes it easier to understand how the SW extends the traditional Web. URI elements and their generalization IRI are used to uniquely identify SW resources. The XML allows the creation of documents formed by structured data, to which SW gives meaning.

The RDF layer has XML triples formed by subject, predicate and object capable of representing the semantics of Web information. The OWL increases the power of this semantic representation by serving to instantiate ontologies on the Web. The ontological layer enhances the understanding of internet information by computers.

Ontologies can be thought of as non-relational databases, which are consulted through queries in SPARQL, a language equivalent to Structured Query Language (SQL). This language is very similar to SQL for relational databases, including clauses (select, from, and where), query modifiers (order by and distinct), and logical and comparison operators. A great advantage of ontologies is the possibility of discovering new knowledge through inference rules described in RIF and Semantic Web Rule Language (SWRL) (HORROCKS *et al.*, 2004). RIF is the W3C Rule Interchange Format and an XML language for expressing Web rules that computers can run.

The SWRL is a standard SW language for implementing rules that can be used to infer new knowledge by the logical layer. The phases followed by reasoners in this inference process are used by the proof layer to validate this inference process. Finally, the trust layer aims to ensure that the inferred knowledge is reliable. The Logic, Proof and Trust layers are not yet fully realized.

2.2 Modeling learning objects

A learning object (LO) is digital content that can be reused in various learning contexts. Each LO is composed of content and its metadata, which follow some pattern to ensure reusability. This metadata is stored in LO repositories, facilitating the reuse of educational content. Some of the standards used to describe metadata are: IEEE-LOM (LTSC, 2002), OBAA metadata standard (VICCARI *et al.*, 2010) and the SCORM standard (ADL, 2001).

The IEEE-LOM standard is recognized worldwide for facilitating the creation and search of LOs. This standard defines nine categories for describing LOs in XML language. These categories are presented in Table 1. Each category is composed of a set of fields relevant to the identification of a given LO. The values assumed by each of these fields are used in the search/retrieval of the desired resource. Among the fields of the General category, the entry field serves to store the LO's link (for example, from a YouTube video

or from a Wikipedia page), and the keyword field can store the concepts that the LO covers.

Table 1 – Categories of the IEEE-LOM standard

Category	Description
1. General	Aggregates general information that describes the LO, such as identifier, title and keywords.
2. Life Cycle	Describes the history of the LO from its creation to its current state using attributes such as version and contribution, which refers to the people or organizations that contributed to creating, editing or publishing the LO.
3. Meta-metadata	It gathers attributes about the metadata, such as contribution, creation date and the language in which it is described.
4. Technical	It includes the technical characteristics of the LO, such as format, size, duration and installation remarks.
5. Educational	It describes the pedagogical characteristics of the LO, such as interactivity type, learning resource type, semantic density, age group, degree of difficulty and learning time.
6. Rights	It aggregates the intellectual property rights and the conditions of use of the LO.
7. Relation	Defines the LO's relationships with other LOs, for example: " <i>is part of</i> ", " <i>is based on</i> " and " <i>is required by</i> ".
8. Annotation	Provides feedback from people who have used the LO.
9. Classification	Classifies the LO within a taxonomy.

The IEEE-LOM standard serves as the basis for other standards, such as the OBAA metadata standard (VICCARI *et al.*, 2010) and the SCORM standard (ADL, 2001). The OBAA standard extends the IEEE-LOM standard ensuring interoperability between platforms in the Brazilian educational context. The Sharable Content Objects Reference Model (SCORM) standard provides interoperability between different learning management systems.

The SCORM standard aims at portability by allowing different types of learning objects to communicate efficiently with the various existing e-learning platforms. The standard enables the encapsulation and sequencing of LOs of different sizes, such as images, exercises or even a complete course, facilitating the reuse of this content between the various platforms compatible with this standard.

The IEEE-LOM standard for describing metadata may be incomplete in some contexts and, on the other hand, describe metadata that is not widely used. To solve the problem of its incompleteness, there are some extensions of the IEEE-LOM standard, such as the extension defined by Customized Learning Experience Online (CLEO). The contributions of CLEO in relation to the IEEE-LOM are:

1. Additional vocabularies to improve the aggregation level (general category);
2. Alternative vocabularies for learning resources types (educational category) and purpose (classification category);
3. New elements for the educational category.

One of the increments proposed by CLEO is related to the learning resource type. The IEEE-LOM defines 15 different resource types, such as exercise, simulation, questionnaire, diagram and figure. CLEO adds 29 more types to them, such as additional resource, assessment, definition, example and introduction.

2.3 Modeling students

In addition to learning content, the learner also needs to be modeled by computer systems. One of the most commonly modeled characteristics in this context is the student's learning style. Learning styles refer to the notion that individuals have varying preferences for how they receive and process information (PASHLER *et al.*, 2008). This concept is fundamental in designing adaptive learning systems that can meet the diverse needs of students, enhancing their engagement and effectiveness in learning.

The Felder-Silverman Learning Style Model (FSLSM), proposed by Felder and Silverman (1988), is widely regarded as one of the most comprehensive frameworks for modeling student learning styles. The FSLSM stands out due to its detailed consideration of psychological aspects, making it a robust tool for understanding learners' preferences (DEBORAH; BASKARAN; KANNAN, 2014). Unlike other models that might focus on a single dimension of learning, FSLSM encompasses four polar dimensions, providing a holistic view of how students interact with information.

These four dimensions are Input, Organization, Perception, and Processing. The Input dimension distinguishes between Visual and Verbal learners, recognizing that some students better absorb information through images and diagrams, while others prefer text and spoken words. The Organization dimension contrasts Sequential learners, who understand information in linear steps, with Global learners, who grasp concepts more holistically. The Perception dimension differentiates Sensitive learners, who are practical and detail-oriented, from Intuitive learners, who are more abstract and innovative. Finally, the Processing dimension compares Active learners, who learn best through interaction and experimentation, with Reflective learners, who prefer to think through information quietly and introspectively.

To effectively assess and identify these learning preferences, the Index of Learning Styles (ILS) questionnaire is commonly used. Developed by Soloman and Felder (2005), this 44-question instrument (see Annex A) evaluates learners across the four dimensions outlined in the FSLSM. By analyzing the responses, educators and adaptive learning systems can gain valuable insights into each student's preferred learning style. This information can then be used to tailor instructional methods and materials to better align with individual learner needs, thereby enhancing the overall educational experience.

The application of FSLSM in modeling students has significant implications for the design of Intelligent Tutoring System (ITS) and other educational technologies. By in-

corporating detailed learner profiles based on the FSLSM, these systems can dynamically adapt content delivery, feedback, and instructional strategies to suit each student's unique preferences. This personalized approach enhances engagement and motivation, addresses diverse learning challenges, and leads to improved learning outcomes.

2.4 Intelligent systems and pedagogical interventions

In this section, we present an overview of Intelligent Tutoring Systems and Recommender Systems, as well as their respective pedagogical interventions. We begin by exploring the architecture and functionality of Intelligent Tutoring Systems, highlighting how these systems utilize models of instructional content and teaching strategies to provide personalized and adaptive learning experiences. This is followed by a detailed discussion on the different types of pedagogical interventions implemented in ITS, such as tips, feedback, explanations, and motivational strategies. Subsequently, we delve into Recommender Systems, examining their role in enhancing pedagogical interventions through personalized content recommendations based on students' learning profiles. We also address the integration of these systems into educational contexts, emphasizing their potential to improve learning outcomes by offering tailored support and resources.

2.4.1 Intelligent Tutoring System

An Intelligent Tutoring System (ITS) is a computer-based instructional system with models of instructional content (what to teach) and teaching strategies (how to teach) (WENGER, 1987). An ITS can be thought of as a computer program that makes inferences about the student using artificial intelligence techniques to dynamically adapt the content or style of instruction (MURRAY, 1999).

The classic architecture of ITSs is composed of four modules, as shown in Fig. 2.

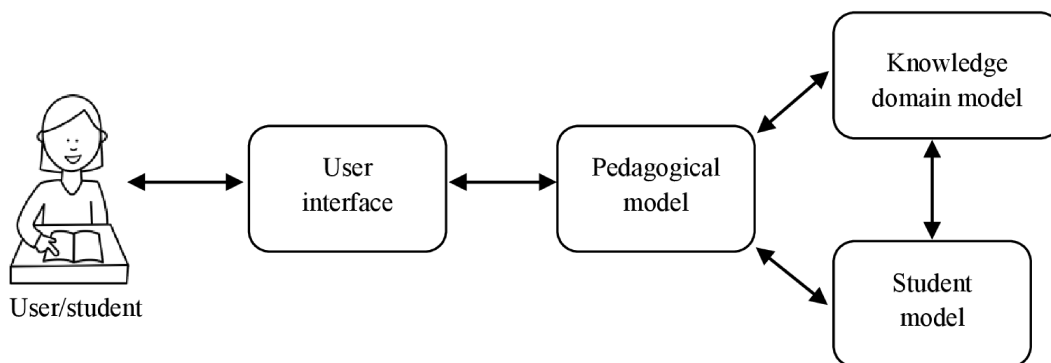


Figure 2 – Traditional architecture of an ITS

Source: Adapted from Nwana (1990)

These modules (HARTLEY; SLEEMAN, 1973) are detailed below:

1. Knowledge domain: aggregates specialist knowledge about the content to be taught to the student.
2. Student model: models the student according to their level of instruction, prior knowledge and learning and cognitive styles.
3. Pedagogical model: contains teaching strategies and tutoring knowledge to guide the student's learning process.
4. Interface: establishes the communication path between the user and the system.

In addition to the modules mentioned above, VanLehn (2006) argues that it is common in ITSs to have two loops of repetition: one loop is responsible for assignments of tasks and materials (outer loop), while the second loop deals with feedbacks, hints for exercises and reviews of submitted solutions (inner loop). Loop-based ITSs typically focus on a limited range of content, such as mathematical problem-solving, that is well-suited to the step-by-step approach of the inner loop.

An important concept in the context of this thesis is “intervention”. In Chapter 5, the concept of intervention is defined by relating it to inner loop actions and recommendations, such as tips, feedback, explanations, examples, and definitions. Below we describe some ITSs based on these interventions.

Mazk's intelligent tutor system (VALERIANO; CORRÊA; POZZEBON, 2019) offers specific hints to aid students in problem-solving and understanding concepts, as well as immediate feedback and detailed explanations when necessary. In parallel, the Andes Physics Tutoring System (VANLEHN *et al.*, 2005) addresses the use of contextual feedback and dynamic adaptation, providing detailed explanations and guided questions to enhance student understanding. Both ITSs underscore the impact of adaptive and contextualized interventions on the teaching-learning process.

Some ITSs relate interventions to students' emotions. In (BAKER *et al.*, 2010), the authors focus on monitoring students' emotions and using motivational interventions to maintain engagement, suggesting that it is preferable to face frustration rather than boredom. It recommends incorporating mechanisms in ITS to detect and respond to students' emotional states. Similarly, the AutoTutor (GRAESSER *et al.*, 2001) explores the use of conversational dialogues for immediate feedback and detailed explanations, employing personalization techniques to cater to individual student needs. These studies emphasize the importance of considering students' emotional states and ensuring continuous engagement for effective education.

These are just a few of the many ITSs in the literature that collectively illustrate the diversity and efficacy of interventions in intelligent tutoring systems, showing how

personalization, immediate feedback, motivation, and performance analysis contribute to more effective and engaging teaching.

2.4.2 Recommender system

Unlike ITSs, RSs do not rely on the learning interface and are categorized by the technique used in the recommendation. The different classes of recommendation techniques include collaborative filtering, content-based, knowledge-based and hybrid recommendation. In Content-based Filtering (CBF) (VANETTI *et al.*, 2010), the recommended objects have content characteristics similar to those objects that the target user liked in the past. The disadvantage of CBF is that the learners can only receive LOs similar to their past experience, whereas in Collaborative Filtering (CF) (MEDIO *et al.*, 2020), the recommendation of new LOs for the target learner considers the recommendation history of other students.

CF uses object evaluations (see Fig. 3) to calculate the similarity of users or objects and make the recommendation. Both techniques suffer from cold-start problem, and CF also has the rating sparsity problem while Knowledge-based (KB) recommendation (TARUS; NIU; YOUSIF, 2017) is able to address these issues. KB recommendation aggregates knowledge about the student and learning materials to apply them in the recommendation process. When the KB recommendation uses ontology to represent this knowledge, it is called an ontology-based recommendation (TARUS; NIU; YOUSIF, 2017).

Learner	O_1	O_2	O_3
L_1	4	3	3
L_2	3	?	4
L_3	5	1	4
L_4	3	3	5
L_5	4	2	4

(a) Rating matrix of LOs for collaborative filtering

Learner	LO	Level	Rating
L_2	O_2	Beginner	2
L_1	O_2	Advanced	3
L_2	O_2	Intermediate	?
L_3	O_2	Beginner	2
L_3	O_2	Intermediate	5

(b) Rating matrix of LOs for knowledge-based recommendation

Figure 3 – Rating matrix of CF and KB recommendation

Notice in Fig. 3 that the rating matrix in CF takes into account only the ratings of the LOs, while KB recommendation considers the student's level, which can be beginner, intermediate or advanced. The target learner is identified by L_2 . In KB recommendation, the prediction of the score that L_2 would give the O_2 learning object will depend on the grade that O_2 received from other students at the same level as L_2 . This level is an example of contextual information or knowledge about the learner. It can be said that the

KB recommendation is thus a type of CF that aggregates contextual information about the student, helping to reduce the rating sparsity and cold-start problems in CF.

KB recommendation addresses the cold-start problem when there is a new target learner, but if students similar to the target learner are also new to RS, then the problem remains, as they have not rated any LOs. The cold-start problem also remains in calculating the rating prediction of new LOs. Contextual information about the student in KB recommendation alleviates the rating sparsity problem in the similarity calculation, but the lack of overlapping ratings for the same LOs affects the ratings prediction calculation.

2.4.3 Pedagogical interventions in intelligent systems

Pedagogical intervention refers to deliberate strategies and actions carried out by educators with the aim of improving the teaching-learning process. These interventions can be directed at specific students or groups of students and aim to address learning difficulties, promote specific skills or reinforce content already covered. The intervention can be preventive, corrective or enriching, depending on the needs identified.

Pedagogical intervention is based on theories from authors such as Lev Vygotsky, Jean Piaget, Benjamin Bloom and David Ausubel. Vygotsky's sociocultural theory emphasizes the importance of social and cultural context in cognitive development. His ideas about the Zone of Proximal Development are fundamental to understanding how pedagogical interventions can be structured to promote learning. Although Piaget focused more on the natural cognitive development of children, his theories about developmental stages provide a basis for understanding how interventions can be adapted to different stages of learning.

Known for his Taxonomy of Educational Objectives, Bloom contributed significantly to the understanding of how to structure pedagogical interventions to achieve different levels of knowledge mastery. David Ausubel's theories about meaningful learning and advance organizers are essential for developing intervention strategies that help students integrate new information with pre-existing knowledge.

RSs represent a technological application that can enhance pedagogical interventions. These systems can adapt content and activities based on each student's learning profile, offering a more individualized approach that respects different learning rhythms and styles. Through continuous analysis of performance data, recommender systems can identify areas where students are struggling and suggest specific interventions to overcome these gaps in knowledge.

By providing materials that are more relevant and appropriate to students' level of knowledge and interest, these systems can increase student engagement and motivation. In addition, RSs can provide real-time feedback, allowing students to adjust their learning strategies and educators to adjust their teaching practices more effectively. Thus, the integration of personalized learning object recommender systems into the context of

pedagogical intervention offers a powerful tool for educators. By combining pedagogical expertise with personalization technology, it is possible to create more effective learning environments that are tailored to the individual needs of students, promoting deeper and more meaningful learning.

In the proposed approach (Chapter 4), we define the concept of intervention by relating it to refined LOs such as hints, definitions, and examples. Defining intervention as a learning object makes it possible to bring ITS interventions into the context of RS, enabling personalized intervention recommendations. Additionally, the proposed recommender system expands the range of knowledge domains, allowing for a more comprehensive coverage of educational content.

2.5 Chatbot and gamification

A chatbot is a program capable of establishing a dialogue using natural language to communicate with a human user. Since the early 1970s, chatbots have been used as pedagogical agents in educational environments. In this work, the terms “chatbot” and “conversational agent” are used synonymously. Conversational pedagogical agents use artificial intelligence techniques to enhance and personalize teaching-learning automation.

In the teaching-learning context, chatbots can simulate human tutoring (GRAESSER, 2016), motivate students to learn about course content (RUAN *et al.*, 2019) and increase students’ willingness to communicate when they are learning a new language (AYEDOUN; HAYASHI; SETA, 2019). In addition, conversational agents can help combat depression in students (PATEL *et al.*, 2019) and read stories to children by interacting with questions and answers (XU *et al.*, 2021).

The use of chatbots is associated with student motivation. Motivation, according to the Self-Determination Theory (SDT) (RYAN; DECI, 2000), can be intrinsic or extrinsic. Intrinsically motivated students undertake an activity to satisfy, have fun, or challenge themselves. On the other hand, when external forces, such as awards, are the cause of the students’ action, then the students are extrinsically motivated.

According to the SDT, intrinsic motivation is based on three psychological needs: autonomy, competence and relatedness. Autonomy refers to actions seen as selective and self-initiated. Competence refers to the perception that the individual has effectively performed a task with confidence. Relatedness is defined as the affective support that an individual receives or gives to others during interactions. Given this theory, chatbots can provide affective support (relatedness) to students to promote their intrinsic motivation in the teaching-learning process (YIN *et al.*, 2021).

In addition to the chatbot, gamification can promote student engagement and motivation. Gamification is an integration of game elements and game thinking in activities that are not games (KIRYAKOVA; ANGELOVA; YORDANOVA, 2014). This concept

has been widely adopted in several areas, including education, business and healthcare, due to its effectiveness in making activities more engaging and challenging.

Core elements of gamification include levels, achievements, scores, challenges, and rewards. Levels provide a clear progression structure, allowing participants to visualize their progress and feel motivated to reach higher levels. Achievements are specific goals that, when achieved, provide rewards and recognition, encouraging persistence and exploration of new activities. Scores and rankings foster healthy competition, while well-designed challenges maintain interest and encourage overcoming obstacles. Rewards, whether tangible or symbolic, positively reinforce desired behaviors and keep participants engaged.

In the educational context, gamification has been shown to be a powerful tool for improving student motivation, increasing knowledge retention, and promoting active learning (MENDES *et al.*, 2019; MOREIRA *et al.*, 2022). By transforming educational tasks into playful activities, students are more likely to participate and engage in the learning process. Furthermore, gamification can personalize the learning experience by adjusting the level of difficulty and providing immediate feedback, which helps students identify their weaknesses and work to overcome them. The use of game elements can also foster social skills such as collaboration and communication, especially in group learning environments.

Integrating gamification with chatbots represents a groundbreaking synergy in the educational field. Gamified chatbots, like the one we created, Anya, can provide a user-friendly and interactive interface that continuously engages students. Through gamification techniques, the chatbot can assign levels and achievements as students interact and complete educational tasks. This not only keeps students motivated but also provides personalized learning by adapting challenges based on individual performance. Additionally, gamification integrated into the chatbot can provide instant feedback and rewards, creating a learning cycle that encourages continuous practice and the development of new skills. With the ability to interact in a natural and accessible way, gamified chatbots make the learning process more dynamic and engaging.

2.6 Set Covering Problem

The SCP is a well-known combinatorial optimization problem that has been applied to a wide range of applications (LAN; DEPUY; WHITEHOUSE, 2007), such as crew scheduling in railway and airlines (HOUSOS; ELMROTH, 1997; CAPRARA; FISCHETTI; TOTH, 1999), facility location problem (VASKO; WILSON, 1984) and industry production planning (VASKO; WOLF; STOTT, 1987).

The mathematical formulation of the SCP is as follows. Given m rows, n columns and an $(m \times n)$ sparse matrix of zero-one elements a_{ij} , where $a_{ij} = 1$ if row i is covered by column j , and $a_{ij} = 0$ otherwise. Each column j covers at least one row from m rows and

has an associated cost $c_j > 0$. The objective is to find a subset from n columns that covers all m rows at a minimal cost. In Section 4.3, we define the LORP as the mathematical programming model of the SCP.

The SCP is NP-Hard (GAREY; JOHNSON, 1979) and exact algorithms (BALAS; CARRERA, 1996; FISHER; KEDIA, 1990) are used to find its optimal solution, but these procedures are capable of solving only very limited size instances and are very time consuming, so exact algorithms are not practical for large scale instances due to the computational complexity of the SCP. For this reason, many researchers make a lot of efforts on developing metaheuristic algorithms (BILAL; GALINIER; GUIBAULT, 2013) based on constructive metaheuristics as Ant Colony Optimization (ACO) (REN *et al.*, 2008; REN *et al.*, 2010), evolutionary algorithms as Genetic Algorithm (GA) (BEASLEY; CHU, 1996; SOLAR; PARADA; URRUTIA, 2002; WANG; OKAZAKI, 2007) and local search (MUSLIU, 2006; YAGIURA; KISHIDA; IBARAKI, 2006).

In the educational context, we noticed that the SCP has small instances that can be solved by exact and greedy algorithms. The greedy algorithm of Chvatal (CHVATAL, 1979) is the simplest approach to solving the SCP. Greedy algorithms are fast, but they have a hard time finding the best solution, while the exact algorithms spend more time to find the optimal solution. In this research, we implemented these two types of algorithms in addition to GA and Particle Swarm Optimization (PSO), and compared them with our heuristic search used to solve LORP.

Related Works

Several works in the literature have addressed the LORP using different techniques for recommending educational resources. In this work, the LORP is modeled as a problem that aims to cover concepts using a set of LOs. This chapter presents works that have this same goal and works that use other LO recommendation techniques, such as CF and Sequential Pattern Mining (SPM), without modeling the LORP as a coverage problem. In addition, we present the works that use chatbots and gamification in e-learning.

There is a problem very similar to SCP that is used in modeling educational resource recommendation problems. In the works by Acampora *et al.* (2008), Acampora, Gaeta and Loia (2010), Acampora *et al.* (2011) and Gaeta *et al.* (2013), the authors model LORP as a facility location problem, whose objective is to allocate m didactic activities (such as LOs) to n concepts of the learning path (the concepts are ordered by their prerequisite relationships). More specifically, the objective is to build the smallest set of activities covering all n concepts with the minimum sum of distances between activities and covered concepts.

This problem can be formalized as follows. Let y_i ($i = 1, \dots, m$) be a binary vector that takes the value 1 if learning resource i is used, and 0 otherwise. Given m rows, n columns and an $(m \times n)$ binary matrix of elements x_{ij} that takes the value 1 if concept j is covered by LO i , and 0 otherwise, the problem is defined as:

$$\text{Minimize } \sum_{i=1}^m p(i)y_i + \sum_{i=1}^m \sum_{j=1}^n d(i, j)x_{ij} \quad (1)$$

$$\text{Subject to } \sum_{j=1}^n x_{ij} = 1 \quad i = 1, \dots, m$$

$$x_{ij} \leq y_i \quad i = 1, \dots, m \quad j = 1, \dots, n$$

$$x_{ij} \in \{0, 1\} \quad i = 1, \dots, m \quad j = 1, \dots, n$$

$$y_i \in \{0, 1\} \quad i = 1, \dots, m$$

where $p(i)$ represents the cost of introducing the i -th LO in the sequence of LOs recommended to the student, and the distances $d(i, j)$ are calculated by a function that compares the student's learning preferences with the metadata values of the i -th LO which covers concept j .

Acampora *et al.* (2008) solved this problem using a memetic algorithm whose main steps are the genetic search, with crossover and mutation operations, and the local search to optimize the solutions found. The system proposed by the authors has the Knowledge Model, the Learner Model and the Didactic Model, which interact through specific processes aiming at personalized recommendation. The Domain Model uses ontologies that model concepts and their relationships, which are fundamental in the construction of the learning path.

Later, Acampora, Gaeta and Loia (2010) used memetic optimization in a multi-agent system. This system is explored by Acampora *et al.* (2011) to solve the recommendation problem of Eq. (1) through a memetic algorithm that uses evolutionary techniques followed by local search strategies. Among the evolutionary optimization techniques, they used GA and the particle swarm method, which are executed in parallel by the Evolutionary Agent. The Local Search Agent controls the parallel execution of local search strategies, such as Tabu Search and the Simulated Annealing metaheuristic. The exchange of solutions by the optimization methods is controlled by a set of fuzzy rules.

This solution proposed by Acampora *et al.* (2011) is best employed in the context of Web 2.0, which is characterized by distributed repositories, a variety of educational activities and subject-filled learning paths. The same problem, Eq. (1), was solved by Gaeta *et al.* (2013) using simply a greedy algorithm. However, the authors cautioned that while this is a quick approach, it does not guarantee very good results.

All these works use a concept-based recommendation approach, presenting a solution to the problem defined in Eq. (1). The concepts are modeled in ontologies of knowledge domains. According to Gruber (1993), "an ontology is an explicit specification of a conceptualization", so it is natural to think that an ontology can be used, mainly, to model concepts and their relationships. The "prerequisite" type relation, for example, is used to order concepts taking into account that the learning of a given concept depends on the mastery of another. The ordered list of concepts that the student is expected to learn is commonly called the "learning path".

Building a learning path often comes at a high cost. Creating knowledge domains is an expensive task that involves expert knowledge. The advantage of our approach to these works is that our RS does not rely on creating a detailed and expensive domain of knowledge, as its objective is to recommend LOs to solve students' doubts rather than recommending learning paths. Furthermore, we model the LORP as the SCP, which is

a simpler formulation than Eq. (1) used by the authors mentioned above. Next, we compare our approach with works that use other LO recommendation techniques that are not explicitly based on the coverage of LOs given a set of concepts.

Much research combines recommendation techniques with ontologies and/or the Web, including Wikipedia, for the recommendation of learning resources as shown in Table 2. For example, Limongelli, Gasparetti and Sciarrone (2015) created a module in a system for collaborative recommendation of wiki pages used by teachers when creating their courses. The target teacher benefits from the recommendation made in the past to other teachers who have a teaching style similar to yours.

Another approach that recommends wiki content is presented in Belizário Júnior and Dorça (2018), but the content is recommended directly to the target learner without using the teacher as an intermediary. This approach selects the best quality wiki pages using the quality classes assigned to them by users. The sections (within these pages) that cover the concepts that the target learner needs to learn are recommended. The approach uses an ontology for modeling students and LOs. The LO recommendation problem is formalized as a set covering problem and is solved by a GA.

A faster algorithm solves this same problem considering a greedy heuristic, as shown in Falci *et al.* (2019). The intuition underlying heuristics is that LOs that meet the student's learning style while covering more concepts tend to deliver better candidates for the final solution. The algorithm that implements this heuristic is faster than GA, mainly for instances with thousands of LOs, for which GA can become impractical given the exponential search space and the high number of calculations of the fitness function.

This LO recommendation problem defined as a covering problem is also solved in Belizário Júnior *et al.* (2020) using CF, SWRL and PSO (KENNEDY; EBERHART, 1995). In previous research (BELIZÁRIO JÚNIOR; DORÇA, 2018; FALCI *et al.*, 2019), the authors considered only the user's search parameters when recommending LOs. In Belizário Júnior *et al.* (2020), the authors also consider the history of rating given to LOs by students with ratings similar to the learner to whom the recommendation is directed.

GAs can be used to personalize the recommendation of LOs in contexts with many learning parameters. Christudas, Kirubakaran and Thangaiah (2018) proposed a Compatible Genetic Algorithm (CGA) to the recommendation of LOs. The CGA forces compatibility of a) the LO type in relation to the learning style of the student, b) the LO complexity level with respect to the knowledge level of the learner and c) the interactivity level of the LO based on the satisfaction level of the student during the learning process.

Birjali, Beni-Hssane and Erritali (2018) created an adaptive e-learning model based on Big Data that use a MapReduce-based Genetic Algorithm to determine the relevant future educational objectives through the adequate learner e-assessment method and an ACO algorithm to generate an adaptive learning path for each learner. After that, a MapReduce-based Social Networks Analysis is performed to determine the learning mo-

tivation and social productivity in order to assign a specific learning rhythm to each learner.

Table 2 – Comparison of related literature with the proposal of this research

Reference	Web content reuse	Ontology or Semantic Web technologies	LOs recommend. technique	LOs coverage using fine-grained concepts from different areas of knowledge	Chatbot
Limongelli, Gasparetti and Sciarrone (2015)	Yes	No	CBF and CF	No	No
Belizário Júnior and Dorça (2018)	Yes	Yes	GA	No	No
Falci <i>et al.</i> (2019)	Yes	Yes	Greedy alg.	No	No
Belizário Júnior <i>et al.</i> (2020)	Yes	Yes	CF, SWRL and PSO	No	No
Christudas, Kirubakaran and Thangaiah (2018)	No	No	CGA	No	No
Birjali, Beni-Hssane and Erritali (2018)	Yes	No	ACO and GA	No	No
Ouf <i>et al.</i> (2017)	No	Yes	SWRL	No	No
Abech <i>et al.</i> (2016)	Yes	Yes	SWRL and SPARQL	No	No
Pereira <i>et al.</i> (2018)	Yes	Yes	SPARQL	No	No
Tarus, Niu and Khadidja (2017)	No	Yes	CF	No	No
Jeevamol and Renumol (2021)	No	Yes	CBF and CF	No	No
Ramirez-Arellano, Bory-Reyes and Hernández-Simón (2017)	Yes	No	Learning style based filtering	No	No
Xiao <i>et al.</i> (2018)	Yes	No	Association rules, CBF and CF	No	No
Klašnja-Milićević, Vesin and Ivanović (2018)	No	Yes	Social tagging and SPM	No	No
Tarus, Niu and Kalui (2018)	No	No	Context awareness, SPM and CF	No	No
Wan and Niu (2020)	No	No	SPM and self-organization	No	No
Yin <i>et al.</i> (2021)	No	No	No	No	Yes
Colace <i>et al.</i> (2018)	Yes	Yes	LDA algorithm	No	Yes
Katz <i>et al.</i> (2021)	No	No	No	No	Yes
Nguyen <i>et al.</i> (2019)	No	No	No	No	Yes
Our proposal	Yes	Yes	CF, SWRL and heuristic algs.	Yes	Yes

The Semantic Web, in addition to ontologies, also has technologies that have been explored by some authors for the recommendation of LOs. Ouf *et al.* (2017) developed a tool for an intelligent learning ecosystem using ontologies and rules in SWRL. Ontologies are used to model students and to tailor components of the learning process to students,

such as LOs, preferred learning activities and relevant teaching-learning methods.

Abech *et al.* (2016) developed a model, called EduAdapt, that, in addition to using an ontology and rules in SWRL, uses SPARQL to recommend the most appropriate LOs for the student's context and learning styles. The model can be coupled to learning environments to be used on mobile devices. It has an ontology that uses inference rules that help adapt the content and that uses classes to model students, LOs, mobile device characteristics and student context (for example, if the student is stationary or moving).

SW technologies, such as ontology and SPARQL, are also used by Pereira *et al.* (2018). They created an infrastructure for the recommendation of learning resources based on information, such as the user's profile and the educational context, extracted from the Facebook social network. Information extraction techniques and SW technologies are used to extract, enrich and define the profiles and interests of users. The recommendation strategy is based on linked data, LO repositories and videos, benefiting from the time the user spends on the Web.

Other approaches in the literature are based on classic recommendation techniques, such as Content-based Filtering (CBF) and Collaborative Filtering (CF). Tarus, Niu and Khadidja (2017), for example, proposed a CF and ontology-based recommendation technique for personalized recommendation of learning materials. The ontology used models student characteristics, such as learning styles and knowledge and skill levels alleviating the cold-start problem.

Jeevamol and Renumol (2021) address the cold-start problem using CBF in addition to an ontology. The ontology-based content recommender system proposed by the authors uses the ontology to model the learner and the LOs with their characteristics. The recommendation model uses collaborative and content-based filtering techniques to generate the top N recommendations based on student ratings.

Another way to address the cold-start problem is through a recommendation based on learning styles. Ramirez-Arellano, Bory-Reyes and Hernández-Simón (2017) developed a system to rank LOs through term-based queries and learning styles. The best ranked LOs are grouped in a SCORM standard package (ADL, 2001), allowing their sharing between learning management systems. LOs are merged into the same package based on search terms and the student's learning style.

More recent research combines recommendation techniques to improve personalization of recommended educational content. Xiao *et al.* (2018) developed a system for recommending educational resources to learners enrolled in formal online courses. The recommender system combines association rules and collaborative and content-based filtering for personalized recommendation of learning materials, taking into account the student's profile, browsing history and the time spent on a given content in the online learning system.

Klašnja-Milićević, Vesin and Ivanović (2018) presented a method of personalized rec-

ommendation of LOs that uses algorithms based on the use of the so-called “most popular tags”. Tags are created collaboratively and are used to improve the search and recommendation of educational resources. Their approach combines social tagging and SPM for generating recommendations of learning resources to learners.

Tarus, Niu and Kalui (2018) proposed a hybrid recommendation approach that combines computational context, SPM and CF algorithms for the recommendation of educational resources. The computational context is used to aggregate contextual information about the learner, such as knowledge level and learning goals. A sequential pattern mining algorithm is used to mine Web logs and discover the student’s sequential access patterns.

Wan and Niu (2020) proposed a hybrid filtering recommendation approach combining learner influence model, self-organization based recommendation strategy and SPM together for recommending LOs to students. The learner influence model is independent of ratings and is used to address the cold-start and data sparsity problems by mining explicit and implicit behaviors and computing the influence that a learner exerts on others.

None of these works mentioned above uses chatbots with gamification to assist in the process of recommending educational resources. In general, the works that use chatbots do not explore the LO recommendation techniques found in the literature.

Yin *et al.* (2021) proposed a chatbot-based micro-learning system, which was tested in an experiment with 99 first-year students of a basic computer course on number systems conversion. The learners were assigned to a traditional learning group or a chatbot based micro-learning group. They concluded that students in the chatbot learning group achieved significantly higher intrinsic motivation than the traditional learning group.

Colace *et al.* (2018) proposed a framework to identify student needs using Natural Language Processing (NLP) and select the best answer thanks to the ontological representation of the knowledge domain. The tool contains a module (Interaction Quality Tracker) that evaluates the conversation between the bot and the user and highlights the critical aspects of this interaction. In addition, it uses the student’s context (profile, location) to direct the dialogue (Context-Aware Information Manager). The authors employ Latent Dirichlet Allocation (LDA) to provide a semantic inference engine that connects the user query and learning object metadata.

State-of-the-art works that use chatbots in ITSs also do not explore LO recommendation techniques. Instead, they focus on the inner loop using the chatbot to assist the student in step-by-step problem solving, such as the papers presented by Katz *et al.* (2021) and Nguyen *et al.* (2019).

Katz *et al.* (2021) proposed an ITS, called Rimac, that engages students in natural language. Rimac combines student modeling with tutorial dialogues about conceptual physics. Rimac dynamically builds a persistent learner model that drives reactive and proactive decision making to deliver adaptive instruction.

Nguyen *et al.* (2019) presented a method to design an intelligent chatbot for solving

mathematical problems in high-school. The chatbot helps in solving problems about determining the value of a parameter of a function by giving some tips to the student and showing the step-by-step solution, simulating a human tutor.

The inner loops implemented in ITSs are responsible for providing feedback at each step of student interaction. However, for its correct functioning, the ITS needs to know the elements involved in the step-by-step resolution process of the activities. Currently, the representation of this knowledge in ITSs is context-dependent or linked to the system interface and the content creation process depends on the teacher, generating an overload of work.

To address this problem associated with the high cost of creating a knowledge domain, our approach does not implement a complete knowledge domain, but implements an ontology that models students and store metadata of LOs associated with the knowledge domain. In addition, our RS delivers fine-grained LOs (interventions, such as hints), which are customized to the student's needs and doubts while studying course content or solving step-by-step exercises.

Research suggests that much remains to be done in building smarter conversational agents. Smutny and Schreiberova (2020) evaluated 47 conversational chatbots using the Facebook Messenger platform, considering attributes of teaching quality, humanity, affection and accessibility. The authors found that such chatbots range from the basic level of sending personalized messages to recommending content. State-of-the-art chatbots are still at the entry level to become AI-powered teaching assistants.

To make the chatbot we created in this work more attractive, we implemented gamification techniques within RS to engage and motivate students in the learning process. Several studies in the literature indicate that gamification enhances student motivation and engagement (MENDES *et al.*, 2019), leading to increased interest and perceived competence among students. This, in turn, promotes a more engaging and effective learning environment (MOREIRA *et al.*, 2022).

In our previous paper (BELIZÁRIO JÚNIOR *et al.*, 2020), we formulate the LORP as a covering problem, and in this work, we take advantage of this idea to define the LORP as the SCP, so the LORP becomes able to consider the concepts that the student needs to learn. It is noted that the related literature uses the Web for the reuse of content (including from LO repositories) and/or use SW technologies, but they do not combine the recommendation of fine-grained concepts typical of the step-by-step approach with the recommendation of content from different areas of knowledge.

This study contributes to the solution of this state-of-the-art challenge by proposing an approach for the recommendation of LOs from different areas of knowledge considering concepts at a higher level of granularity. The results (in Chapter 5) demonstrate that our approach addresses this challenge and the cold-start and rating sparsity problems. Furthermore, the results also show that most of the students, who used our RS through

the gamified chatbot, were satisfied and approved the usability of the system. These are the main advances of our research in relation to the work initially proposed in Belizário Júnior *et al.* (2020). Our proposed approach is detailed in the next chapter.

Proposed Approach

During the learning process, students encounter difficulties that are natural to the teaching-learning process. These difficulties can often delay the student's learning and leave them unmotivated and behind the rest of the class, with a high chance that they will not have a satisfactory performance at the end of the course.

These difficulties arise from specific needs of students that must be met so that they can advance in their learning. To help students overcome these difficulties, in this work, we propose an intelligent RS based on interventions that combines Collaborative Filtering (CF) and ontology-based recommendation technique. Furthermore, we created a chatbot, Anya, combined with gamification techniques to engage and motivate students.

The use of a traditional intent-based chatbot for recommending learning objects is justified by several factors. Firstly, intent-based chatbots are well-established and reliable, providing a stable platform for delivering consistent recommendations. These chatbots are easier to implement and require less computational power compared to generative AI technologies, making them more accessible and cost-effective for educational institutions. Additionally, intent-based chatbots allow for greater control over the recommendations, ensuring that the suggested learning objects align precisely with the educational objectives and curriculum requirements. While generative AI offers advanced capabilities, the simplicity and efficiency of intent-based chatbots make them a practical choice for educational settings where reliability and precise alignment with learning goals are paramount.

Two important concepts in this approach are collaborative filtering and intervention. CF is employed to ensure that the recommended LOs are aligned with the highest ratings given by other students, thereby enhancing the overall quality of the recommendations. Besides CF, another crucial concept is intervention, which tailors the learning experience to meet the specific needs of each student.

An *intervention* is a refined learning object tailored to the student's profile to meet their specific needs. From this definition, we understand that an intervention is a customized, small learning object designed to help students overcome their challenges. The proposed recommender system can be implemented in learning management systems that

utilize student diagnostic metrics to determine the most suitable intervention for each student. These interventions aim to mitigate students' learning difficulties in online education and enhance attention through constant monitoring and feedback. The types of interventions considered in this context include: *definition*, *usefulness*, *attention*, *example* and *hint*.

The *definition* of a concept, for example, can be thought of as a refined learning object, which can be within a video or PDF with text and/or images, depending on the student's learning profile. An *example* can also be considered an intervention. Furthermore, a *definition* can be combined with *examples*, points of *attention* and *usefulness* (what is the concept for) in a coherent manner in the same content, which covers a concept and can be recommended to address the difficulties inherent in the teaching-learning process.

Some interventions, such as *examples* and *hints*, can help students resolve specific doubts (not necessarily exclusive to a student) and common doubts of the class. Thus, such interventions can be recommended individually or grouped to meet the learning context. The disadvantage of grouped recommendations is that they overload the student with unnecessary interventions, but this grouping makes the recommendation easier and has a greater chance of meeting the student's real needs.

Therefore, in the context of this work, we chose to group interventions into didactic and structured educational resources. Each educational resource is a structured set of interventions in a PDF file, which has: a header, a *definition*, what the content is for (*usefulness*), points of *attention*, *examples* and frequently asked questions with *hints* as shown in Figs. 26-28 of Appendix A.

To model students and store metadata of LOs, we implemented an ontology, which does not model a complete knowledge domain, but stores the metadata (according to the IEEE-LOM standard) of the LOs associated with the knowledge domain, i.e., the knowledge domain model is partially implemented by the ontology. The advantage of this modeling is to avoid the costly task of creating a complete knowledge domain.

Our RS reuses Web content, especially Wikipedia pages, to recommend content from different areas of knowledge when the ontology LOs are not enough to cover all the concepts that the student needs to learn. Although, in the context of this work, interventions have been grouped, we show how they can be represented in an ontology to implement the approach that considers the individual recommendation of interventions. Thus, we implement, as an example, the *hint* intervention in the ontology as a possibility for an even more refined recommendation for learners.

The approach of our RS is prepared for the implementation of interfaces that capture the users' search parameters, such as the concepts they need to learn, their preferences and questions. In the context of this thesis, we use a chatbot as interface because it has the ability to understand human language, which can be exploited to extract concepts that the student has to learn or has doubts. These search parameters are stored in an

ontology class called IdealLOs, as these are the ideal characteristics expected to be found in the recommended LOs. The system is ready to work when the ideal LO is filled in, regardless of the interface used to capture user input. While having more information or parameters in the ideal LO is beneficial, the proposed system is capable of functioning using only the concepts.

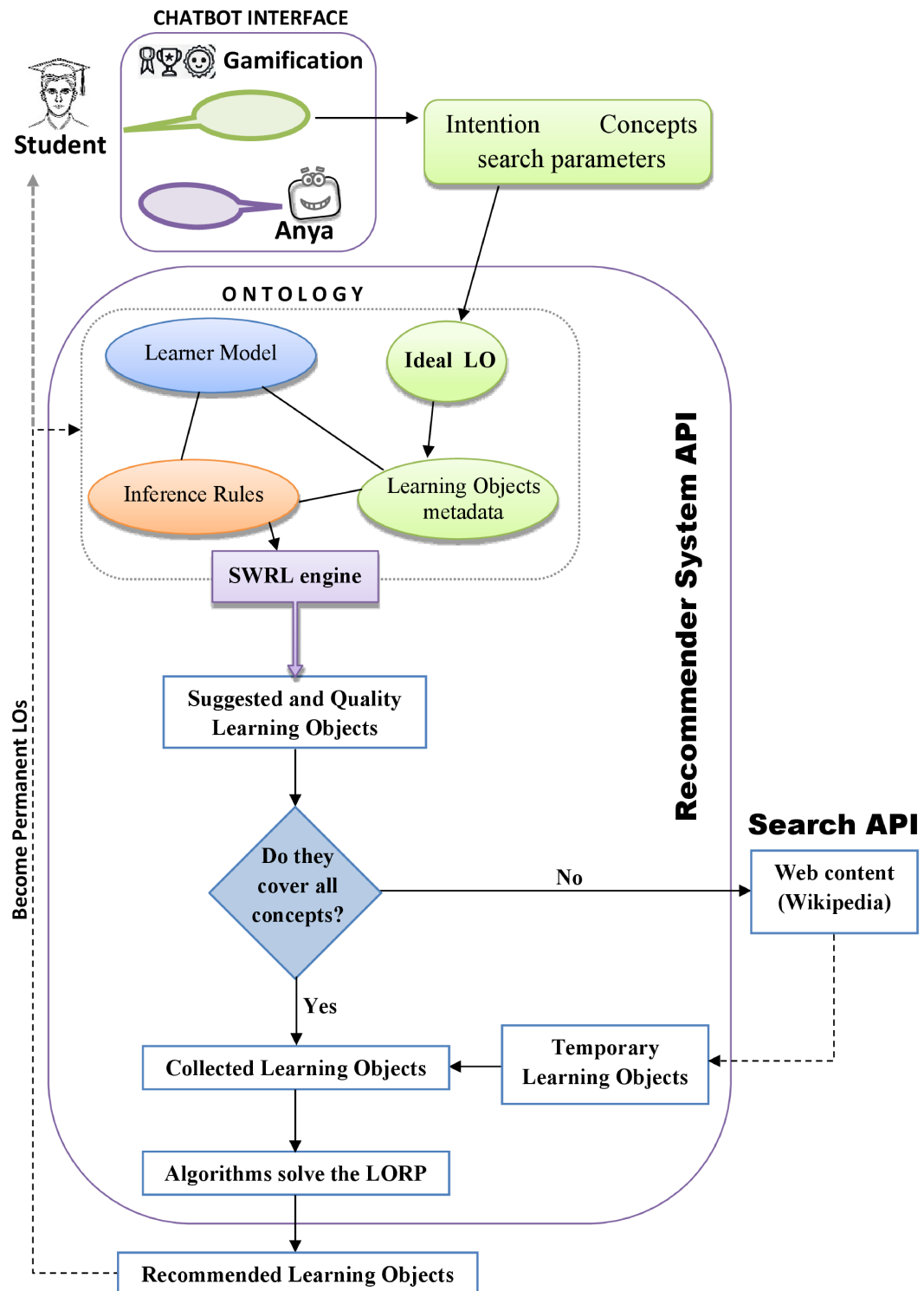


Figure 4 – Overview of the proposed recommender system

The concepts that the student needs to learn are the main search parameter. Such concepts can be filled explicitly in the ideal LO and/or identified in the student's intentions (captured by the chatbot). The intentions recognized by the chatbot are also implemented in the ontology. Each intention in the ontology has concepts associated with it. Inference rules are used to suggest LOs that cover at least one of these concepts.

Some intervention LOs, such as hints, are generally created to answer the student's doubts/intentions during the study of contents or during the resolution of exercises. These contents and exercises deal with concepts that generate doubts. The interventions created especially to solve these doubts are more likely to be recommended (they have higher quality) than the other LOs of the ontology that cover at least one of the concepts that the student has not yet mastered. These hints are associated with the intent for which they were created in the ontology. These quality LOs (hints) are distinguished from the suggested LOs (SuggestedLOs class) through an inference rule that makes them instances of the QualityLOs class.

Fig. 4 shows the recommendation model in which the inference rules implemented in the ontology are used to suggest LOs. If there is at least one uncovered concept, i.e., the set of suggested and quality LOs do not cover all concepts, then Web content including Wikipedia pages dealing with the uncovered concepts are transformed into temporary LOs in the ontology, which are joined with the suggested LOs and the quality LOs to form the set of collected LOs. The collected LOs are supposed to cover all concepts.

Each collected LO has a cost given by the dissimilarity between the LO and the user's search parameters. The parameters title, interactivity type, learning resource type, interactivity level, semantic density and difficulty are learning objects metadata used to calculate the cost of collected LOs. These LOs, as well as their costs and the concepts that the student needs to learn are the input to the algorithms that solve the LORP, which is a cover problem that aims to recommend lower cost LOs that cover the concepts that the learner needs to master. To solve LORP, we implemented metaheuristic algorithms such as GA and PSO, as well as the exact and greedy algorithms, which were compared with our Lorp algorithm. After solving it, the LOs of the best solution found are recommended to the learner. The temporary LOs of this recommendation become permanent LOs in the ontology.

In the following sections, we present the ontology developed during the master's degree in Section 4.1 and the improvements made to it in Section 4.2. Section 4.3 presents the formal definition of LORP based on the SCP. Section 4.4 describes how the cost of LOs is calculated, and Section 4.5 presents the algorithm we created to solve the LORP. Section 4.6 presents Anya chatbot with gamification, and Section 4.7 describes the technical modeling of our recommender system.

4.1 Ontology

The ontology used in this study was initially proposed in (BELIZÁRIO JÚNIOR; DORÇA, 2018). It was built according to the Ontology Development 101 methodology (NOY; MCGUINNESS, 2001), using the Protégé tool and the OWL language. A significant advantage of ontologies is their ability to uncover new knowledge using inference rules specified in RIF and SWRL. Additionally, ontologies facilitate interoperability between different systems, improve data integration and consistency, and enable more effective information retrieval and semantic search capabilities. Fig. 5 shows the ontology classes and subclasses.

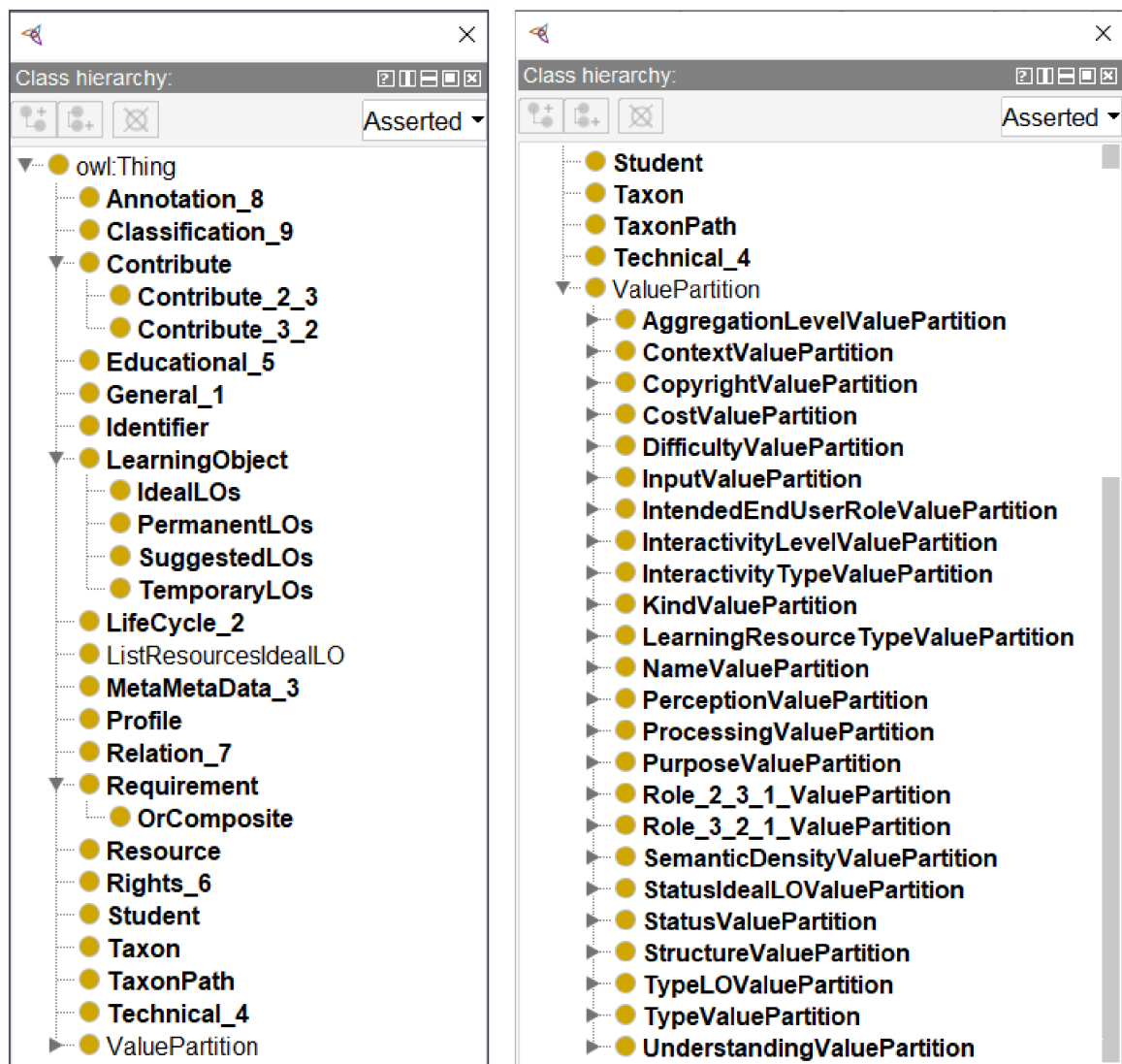


Figure 5 – Ontology classes and subclasses

The `LearningObject` and `Student` classes aggregate instances of learning objects and instances of students, respectively. The student has a profile (instance of the `Profile` class) that corresponds to the FSLSM. Each LO has the nine categories of the IEEE-LOM standard, which are represented in the ontology by the classes `General_1`, `LifeCycle_2`,

MetaMetaData_3, Technical_4, Educational_5, Rights_6, Relation_7, Annotation_8 and Classification_9.

The classes Conbribute, Contribute_2_3, Contribute_3_2, Identifier, Requirement, Or-Composite, Resource, Taxon and TaxonPath, associated with the IEEE-LOM standard, are all (non-primitive type) range of some property.

Properties whose range is a set of fixed values, such as the hasDifficulty property that has the range VeryEasy, Easy, Medium, Difficult and VeryDifficult, were implemented using the Value Partition pattern (RECTOR, 2005). The name of all classes that follow this pattern in the ontology ends with ValuePartition, and their subclasses correspond to fixed values.

The Owlready2¹ library is a Python module that makes it easy to manipulate ontologies by loading them as Python objects. This library allows the programmer to import, edit and save ontologies. The user can create classes, properties and instances, including defining property constraints. It is also possible to create new ontologies and make inferences.

In the next sections, we present the domain model implemented according to the IEEE-LOM standard and its CLEO extension (Section 4.1.1), the learner model that corresponds to the FSLSM (Section 4.1.2) and the SWRL rules used to infer the types of LOs appropriate to the student's learning style and used to perform the selection of LOs that are similar to the user's search parameters (Section 4.1.3).

4.1.1 Domain model

The ontology does not contain the LOs, but their metadata. Each LO has the nine categories of the IEEE-LOM standard implemented as properties of the LearningObject class as shown in Fig. 4.

There are four types of LOs in the original ontology (see Fig. 6):

1. *Ideal LO*: It stores the user's search parameters, such as the concepts that the student is expected to learn. The recommended LOs are expected to equal the ideal LO.
2. *Permanent LO*: It is the class of LOs permanently stored in the ontology. They have been created either by the tutor or previously recommended.
3. *Suggested LO*: It aggregates instances of permanent LOs suggested by inference rules if they have some similarity to the ideal LO.
4. *Temporary LO*: It contains instances of LOs from the Web that are temporarily stored in the ontology if they have some similarity to the ideal LO.

¹ Available at: <https://pythonhosted.org/Owlready2/>

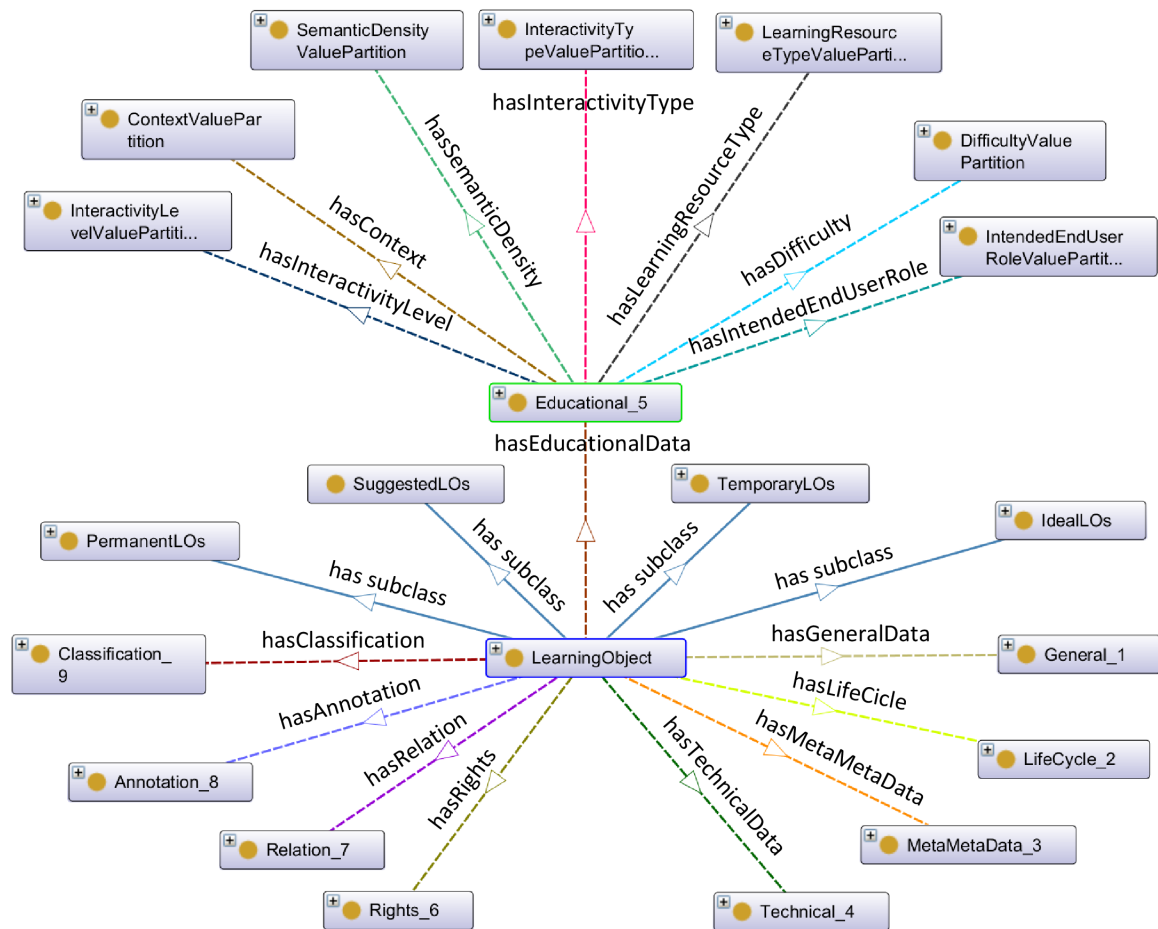


Figure 6 – Domain model in the ontology

Fig. 6 expands the Educational category to show the key educational or pedagogic characteristics in describing the content of LOs. These attributes have a set of fixed values. In addition, the Educational category has 4 attributes characterized by primitive data types: Typical age range, Typical learning time, Description and Language.

4.1.2 Learner model

The ontology is open for the addition of new implementations, including learner characteristics, such as name and knowledge level, according to the educational environment in which it will be used. In this work, we consider the psychological aspects of the students that are structured in the ontology through the FSLSM as shown in Fig. 7. This model helps to recommend the most appropriate types of LOs for each type of learner profile.

The student's learning style is given by the Profile class, which has the four polar dimensions of the FSLSM. For example, in the Understanding dimension, the learner will be either Sequential or Global. The Value Partition pattern is used to properly structure these dimensions into classes and their fixed value subclasses.

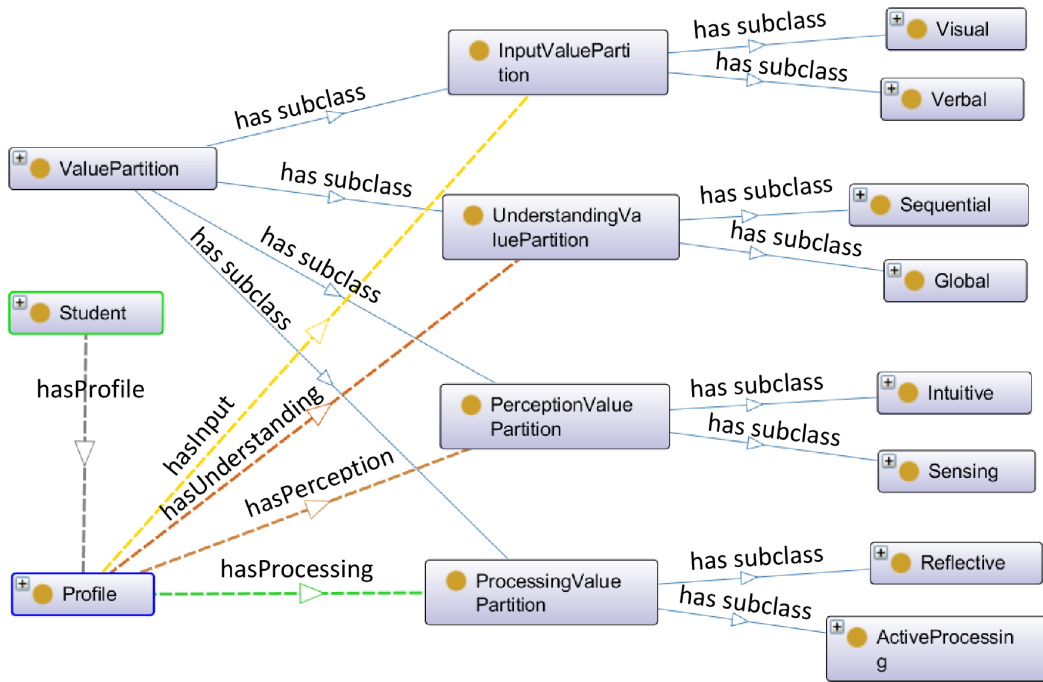


Figure 7 – Learner model in the ontology

4.1.3 SWRL rules

The ontology implements SWRL rules with two different purposes. First, some rules are used to infer the types of LOs appropriate to the student's learning style based on the theory described by Graf, Kinshuk and Ives (2010), who address which types of LOs should be recommended for each type of learner profile associated with the FLSM (see Table 3). Second, other rules are used to perform the selection of LOs that are similar to the user's search parameters. The inference process is run by the HermiT reasoner, which is coupled with the Owlready2 library.

All 32 rules implemented in the original ontology are in Annex B. Rules numbered 1 to 24 are referenced in Table 3. They are used to infer the types of LOs appropriate to the student's learning style. Rules 25 to 31 are used to infer the LOs that have some similarity in relation to the ideal LO in the ontology. These LOs are inferred as instances of the SuggestedLOs class and made available in the recommendation process. These rules are important to filter those LOs that have some chance of being recommended, separating them from the other LOs in the ontology.

During the recommendation process, a permanent LO from the ontology can be the same as a temporary LO found on the Web. Rule 32 suggests the permanent LO and eliminates the redundant temporary LO. It is worth remembering that the recommended LOs must necessarily be instances of the SuggestedLOs or TemporaryLOs class. The recommended LOs can also be instances of the QualityLOs class of the improved ontology presented in the next section.

Table 3 – Resource types recommended for each learner profile

Rule	Learner		LOs
	Dimension	Profile	Learning resource type
1	input	verbal	additionalReading
2	input	verbal	forumActivity
3	input	visual	animation
4	understanding	global	example
5	understanding	global	realLifeApplication
6	understanding	sequential	additionalReading
7	understanding	sequential	animation
8	understanding	sequential	exercise
9	understanding	sequential	reflectionQuiz
10	understanding	sequential	selfAssessment
11	perception	intuitive	additionalReading
12	perception	intuitive	exercise
13	perception	intuitive	reflectionQuiz
14	perception	sensing	animation
15	perception	sensing	example
16	perception	sensing	exercise
17	perception	sensing	realLifeApplication
18	processing	active	animation
19	processing	active	exercise
20	processing	active	forumActivity
21	processing	active	selfAssessment
22	processing	reflective	additionalReading
23	processing	reflective	example
24	processing	reflective	reflectionQuiz

4.2 Improvements in Ontology for individualized intervention recommendation

We incremented the ontology proposed in (BELIZÁRIO JÚNIOR; DORÇA, 2018) to improve the LO recommendation process. In the teaching-learning process, the learner naturally has doubts when studying content or solving exercises. These doubts may be related to concepts and LOs. In (BELIZÁRIO JÚNIOR; DORÇA, 2018), the authors made the recommendation considering only the concepts that the student needs to learn. In this research, beyond concepts, we consider the learner doubts. Thus, it is possible to recommend to the students interventions (e.g., hints) related to the concepts about which they have doubts.

4.2.1 QualityLOs subclass

One of the increments made in the ontology was the creation of the QualityLOs class (see Fig. 8). The instances of the QualityLOs class are LOs specially created to meet the student's intentions/needs. These instances are usually hints created to solve the student's doubts. The quality LOs, as well as the suggested LOs, are inferred LOs that cover at least one of the concepts that the student needs to master, the difference is that the quality LOs receive a bonus so that they are more likely to be recommended to the student given that they were specially created to answer/solve specific doubts (intentions) of the learner.

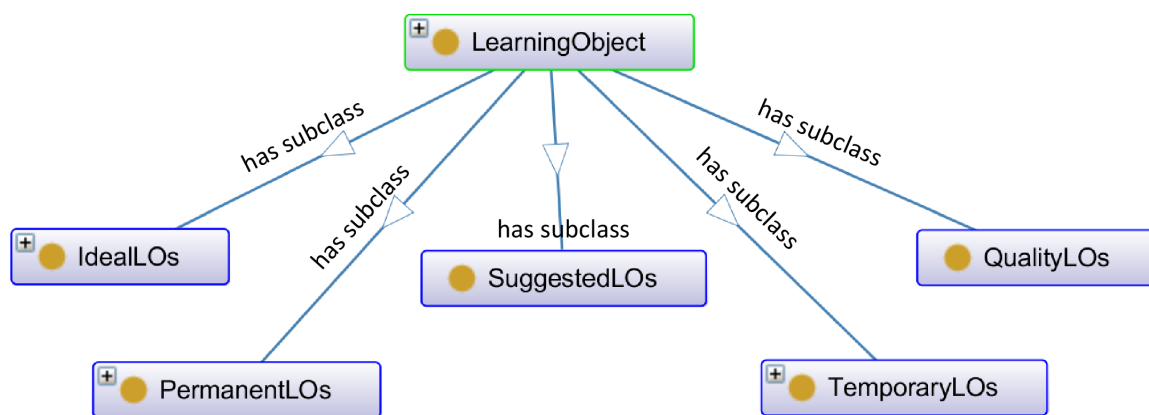


Figure 8 – The subclasses of LearningObject class

The intents (doubts) captured by the chatbot are mapped within the ontology into concepts and LOs of the PermanentLOs class. These LOs can be inferred as instances of the QualityLOs class, and other LOs of the PermanentLOs class can be inferred as instances of the SuggestedLOs class if they cover at least one of these concepts. Also, the uncovered concepts (not covered by any ontology LO), if any, will be covered by Web content, especially wiki pages, which will be treated as LOs of the TemporaryLOs class. Thus, the union of the instances of these three classes results in the set of collected LOs, which will compose the LORP input.

4.2.2 Hint-type LOs

To improve the LO recommendation process, we implement in the ontology the hint type. The hint-type LOs are the aforementioned permanent LOs that can be inferred as instances of the QualityLOs class. We created the `hasLearningObject` property to link the hint-type LOs to intents (students' doubts). The `#intent_001` in Fig. 9 is an intent (recognized by a chatbot) that has two hints (`LO_2` and `LO_3`), which are instances of the PermanentLOs class. Note that `intent_001` relates to hints via the `hasLearningObject` property.

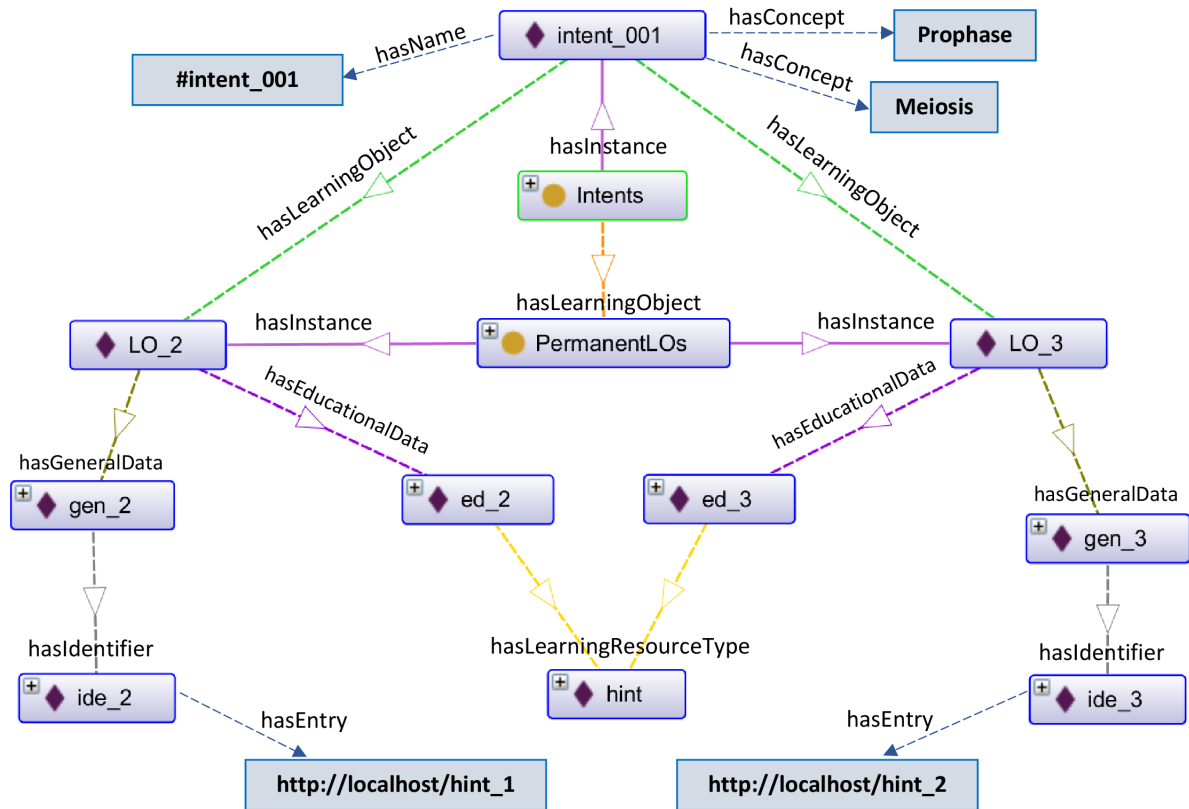


Figure 9 – Relationship (*hasLearningObject*) between intent and hint-type LOs

An intent can have as many hints and concepts (e.g., Prophase and Meiosis in Fig. 9) as needed. Each hint is uniquely identified by an URI. For example, the location of LO₂ and LO₃ is given by the URIs http://localhost/hint_1 and http://localhost/hint_2, respectively.

4.2.3 Ideal LO

The user's search parameters are stored in the ontology's IdealLO class. Considering that an intent has been identified by the chatbot, then the intent name is stored in the ideal LO. The intent name is an important search parameter because it is used to identify the concepts that the student has doubts about and LOs (e.g, hint-type LOs) specially created for the identified intent. Fig. 10 shows an example of this relationship, in which LO_{ideal_00001} has the identification of the intent associated with the student's doubt.

The ideal LO (LO_{ideal_00001}) relates to a resource (res_{ideal_00001}) identified by #intent₀₀₁. This relation (rel_{ideal_00001}) is of the *hasPart* type, so this resource (intent) is part of the ideal LO; i.e., it is part of the user's search parameters. This is how the ideal LO stores the intent (student's doubt) recognized by the chatbot.

Fig. 10 shows that the ideal LO is recommended to the student₀₀₀₀₁, who has a profile characterized by intuitive, global, verbal and active. It is worth remembering that 24 SWRL rules are used to infer the types of LOs appropriate to the student's learning

style and rules 25 to 31 are used to infer the LOs that have some similarity in relation to the ideal LO in the ontology. The new rules created in the improved ontology are presented in the next section.

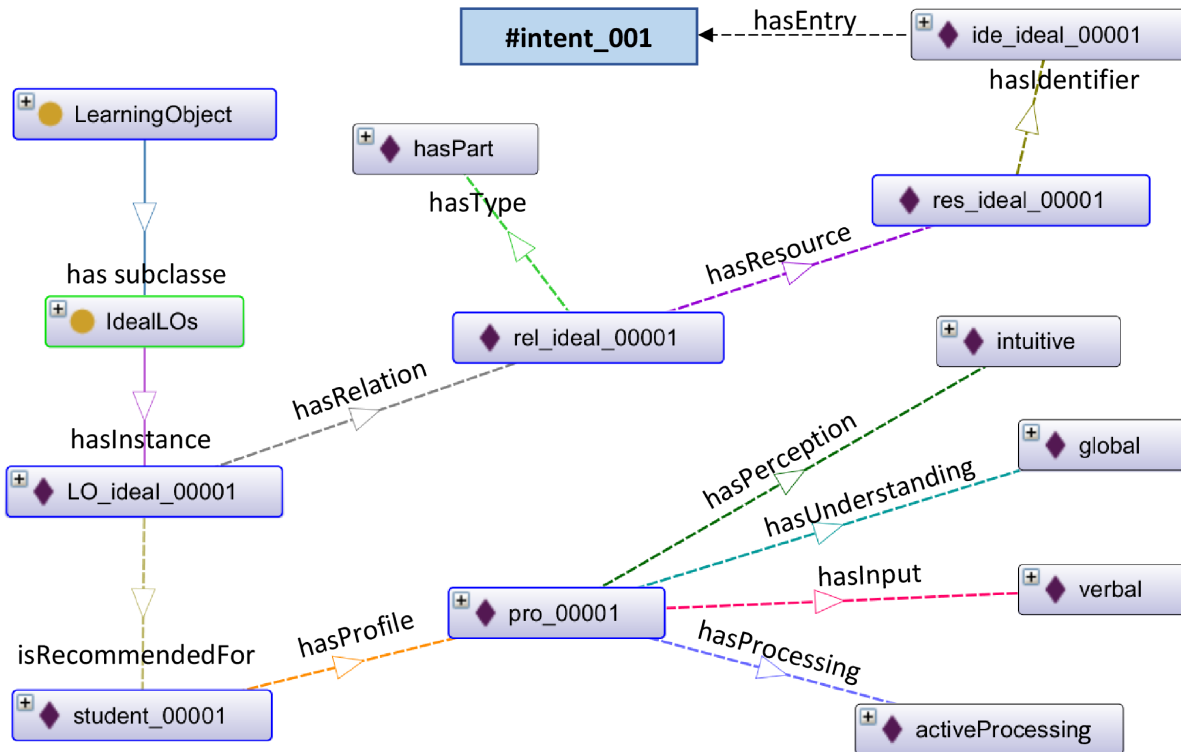


Figure 10 – Ideal LO

4.2.4 New SWRL rules

We increased the number of ontology inference rules to allow convenient LOs (hints) for students to be chosen along with collected LOs. Table 4 shows the two SWRL rules created and their meaning.

The intent name stored in the idealLO identifies the intent associated with the student's doubt. The concepts that are part of the intent (e.g., Prophase and Meiosis in Fig. 9) are used by rule 33, in Table 4, to suggest LOs that cover at least one of these concepts in the ontology, and all the hints associated with this intent become instances of the QualityLOs class through rule 34. The RS is able to infer the suggested and quality LOs through rules 33 and 34, respectively. Temporary LOs are created (with Web content) only if the suggested and quality LOs are not enough to cover all the concepts the student needs to learn.

Table 4 – Two new SWRL rules

Rule	Meaning
Rule 33	
$\begin{aligned} & \text{IdealLOs}(\text{?idealLO}) \wedge \text{hasState}(\text{?idealLO}, \\ & \quad \text{activeIdealLO}) \wedge \text{Relation_7}(\text{?rel}) \wedge \\ & \text{hasRelation}(\text{?idealLO}, \text{?rel}) \wedge \text{Resource}(\text{?res}) \wedge \\ & \text{hasResource}(\text{?rel}, \text{?res}) \wedge \text{Identifier}(\text{?ideideal}) \wedge \\ & \text{hasIdentifier}(\text{?res}, \text{?ideideal}) \wedge \text{hasEntry_}(\text{?ideideal}, \\ & \quad \text{?intentName}) \wedge \text{Intents}(\text{?intent}) \wedge \\ & \text{hasName}(\text{?intent}, \text{?intentName}) \wedge \\ & \text{hasConcept}(\text{?intent}, \text{?keyword}) \wedge \\ & \text{PermanentLOs}(\text{?lo}) \wedge \text{General_1}(\text{?gen}) \wedge \\ & \text{hasGeneralData}(\text{?lo}, \text{?gen}) \wedge \text{hasKeyword}(\text{?gen}, \\ & \quad \text{?keyword}) \rightarrow \text{SuggestedLOs}(\text{?lo}) \end{aligned}$	<p><i>IF</i> there exists an active ?idealLO such that ?idealLO has relation with an intent ?intent <i>AND</i> ?intent has concept ?keyword, which is covered by a permanent LO ?lo <i>THEN</i> ?lo is a suggested LO</p>
Rule 34	
$\begin{aligned} & \text{IdealLOs}(\text{?idealLO}) \wedge \text{hasState}(\text{?idealLO}, \\ & \quad \text{activeIdealLO}) \wedge \text{Relation_7}(\text{?rel}) \wedge \\ & \text{hasRelation}(\text{?idealLO}, \text{?rel}) \wedge \text{Resource}(\text{?res}) \wedge \\ & \text{hasResource}(\text{?rel}, \text{?res}) \wedge \text{Identifier}(\text{?ideideal}) \wedge \\ & \text{hasIdentifier}(\text{?res}, \text{?ideideal}) \wedge \text{hasEntry_}(\text{?ideideal}, \\ & \quad \text{?intentName}) \wedge \text{Intents}(\text{?intent}) \wedge \text{hasName}(\text{?intent}, \\ & \quad \text{?intentName}) \wedge \text{PermanentLOs}(\text{?lo}) \wedge \\ & \text{hasLearningObject}(\text{?intent}, \text{?lo}) \rightarrow \text{QualityLOs}(\text{?lo}) \end{aligned}$	<p><i>IF</i> there exists an active ?idealLO such that ?idealLO has relations with an intent ?intent <i>AND</i> ?intent has learning object ?lo <i>THEN</i> ?lo is a quality LO</p>

4.3 Learning Object Recommendation Problem defined as a Set Covering Problem

In the context of e-learning, imagine a situation in which a student needs to learn four concepts belonging to the finite set $X = \{C_1, C_2, C_3, C_4\}$. Consider a collection of subsets of X given by $F = \{O_1, O_2, O_3, O_4\}$, where $O_1 = \{C_2, C_3\}$, $O_2 = \{C_1, C_3\}$, $O_3 = \{C_1\}$ and $O_4 = \{C_3, C_4\}$. The sets O_1 , O_2 , O_3 and O_4 have costs 2, 5, 2 and 3, respectively. Each element of F is a LO that covers a set of concepts. LO O_1 , e.g., covers the concepts C_2 and C_3 . In this scenario, the objective is:

- Find a set of LOs that together cover all concepts (elements of X) at minimal cost.

This goal is equivalent to the objective of the SCP, and the solution for the previous example is $\{O_1, O_3, O_4\}$ with cost 7.

The formal definition of the SCP is as follows. Let $A = (a_{ij})$ be a binary matrix with m rows and n columns, the goal is to cover all rows using a subset of columns at minimal cost. Let $x_j = 1$ if the column j (with cost $c_j > 0$) is part of the solution, and $x_j = 0$ otherwise, then the SCP is formulated as:

$$\text{Minimize } \sum_{j=1}^n c_j x_j \quad (2)$$

$$\text{Subject to } 1 \leq \sum_{j=1}^n a_{ij} x_j, \quad i = 1, \dots, m$$

$$x_j \in \{0, 1\}, \quad j = 1, \dots, n$$

The LORP, whose objective is to find a coverage of LOs that covers all concepts at minimum cost, corresponds to the SCP formalized by Eq. (2). The value c_j is calculated in Section 4.4.

The input matrix A is filled with the user input concepts and with the LOs collected by the RS (i.e., the collected LOs shown in Fig. 4). Each row i corresponds to an input concept C_i , and each column j is associated with an LO O_j resulting from the set of collected LOs. $a_{ij} = 1$, if O_j covers C_i , and $a_{ij} = 0$ otherwise.

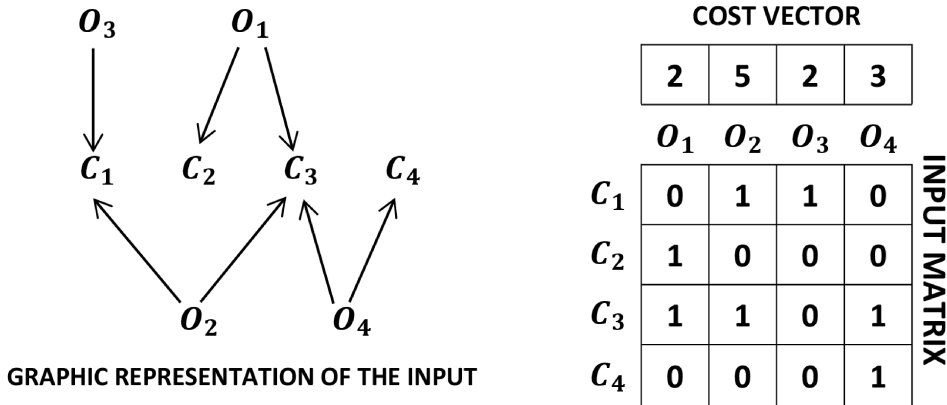


Figure 11 – Input matrix and cost vector

Fig. 11 shows the input matrix, the cost vector and the graphic representation of the previous example. The LOs O_1 , O_2 , O_3 and O_4 have costs 2, 5, 2 and 3, respectively, as shown in Fig. 11. The solution for this example is $\{O_1, O_3, O_4\}$ with cost 7, as stated earlier.

4.4 Improvements in cost calculation

The cost of the column j (c_j) corresponds to the j -th value in the cost vector. The cost c_j is given by the dissimilarity between the user's search parameters (O_{ideal}) and O_j . The search parameters filled in the ideal LO of the ontology, e.g., the type (exercise, video, etc.) and keywords, are compared with the equivalent search parameters of each LO. Belizário Júnior and Dorça (2018) initially proposed the calculation of this cost as:

$$c_j = \text{diss}(O_{ideal}, O_j) \quad (3)$$

The $\text{diss}(O_{ideal}, O_j)$ value is inversely proportional to the degree of similarity between O_{ideal} and O_j . The result of $\text{diss}(O_{ideal}, O_j)$ depends on the proximity between O_{ideal} and O_j . The parameters of O_{ideal} given by the user are compared with the corresponding parameters of O_j , such as degree of difficulty, semantic density and learning resource type.

Formally, let α_i be the value of the i -th parameter. The calculation of the dissimilarity between O_j and the user's search parameters is given by Eq. (4):

$$\text{diss}(O_{ideal}, O_j) = \sum_{i=1}^p (\alpha_{i(ideal)} - \alpha_{i(j)}) \quad (4)$$

where p is the number of parameters, $\alpha_{i(ideal)}$ is the value of the i -th parameter of O_{ideal} , and $\alpha_{i(j)}$ is the value of the i -th parameter of O_j . In this work, we consider six parameters:

- **Title:** the cosine similarity is used to compare the titles.
- **Interactivity type:** each vocabulary term is mapped to a value ($active = 0$, $mixed = 0.5$, $expositive = 1$) that corresponds to the $\alpha_{i(j)}$ of Eq. (4).
- **Learning resource type:** if the O_{ideal} and the O_j are the same resource type, then Eq. (4) results in 0, and it results in 1 otherwise.
- **Interactivity level and semantic density:** each vocabulary term is mapped to a value ($verylow = 0$, $low = 0.25$, $medium = 0.5$, $high = 0.75$, $veryhigh = 1$) that corresponds to the $\alpha_{i(j)}$ of Eq. (4).
- **Difficulty:** each vocabulary term is mapped to a value ($veryeasy = 0$, $easy = 0.25$, $medium = 0.5$, $difficult = 0.75$, $verydifficult = 1$) that corresponds to the $\alpha_{i(j)}$ of Eq. (4).

Later Belizário Júnior *et al.* (2020) reformulated this cost as:

$$c_j = \text{diss}(O_{ideal}, O_j) + (1 - P_j^L) \quad (5)$$

In this case, the calculation of the cost c_j takes into account the prediction P_j^L in addition to the degree of dissimilarity between O_{ideal} and O_j . The relevance that O_j has for the target learner L is given by the prediction P_j^L . This relevance is calculated using collaborative filtering.

In this work, we improve this cost calculation to make LO recommendations at a higher level of granularity using interventions for this. The new cost is formally defined as:

$$c_j = \text{diss}(O_{ideal}, O_j) + (1 - P_j^L + 1 - I_j) * \max_{j \in \{1, \dots, n\}} \text{diss}(O_{ideal}, O_j) \quad (6)$$

where the *max* operator is a weight given to P_j^L and I_j to assign them the same importance as *diss*. The value of I_j depends on the recommendation mode used:

- **Recommendation mode 1 (the less interventions the better):** $I_j = 0$ if O_j is a intervention-type LO, and $I_j = 1$ otherwise.
- **Recommendation mode 2 (the more interventions the better):** $I_j = 1$ if O_j is a intervention-type LO, and $I_j = 0$ otherwise.

The RS has two recommendation modes. If the student has doubts when studying some content or solving an exercise, then **the more interventions the better** to provide a more fine-grained recommendation. On the other hand, if the student has no doubts and needs to learn new concepts, then **the less interventions the better** to recommend. In this case, the RS should recommend other types of LOs, such as lectures and exercises.

In this work, the value P_j^L in Eq. (6) is the prediction of the rating the target learner would give to the new O_j . This value represents the importance that the LO has for the student and is given in a real interval $[0, 1]$, in which the higher its value, the greater the importance that the LO has for the student. This prediction is calculated using the *k*-Nearest Neighbors (*k*NN) (ADOMAVICIUS; TUZHILIN, 2005) approach proposed in (TARUS; NIU; KALUI, 2018). We chose this approach because the *k*NN is a simple algorithm, whose training phase corresponds to the simple storage of instances, and it is the most used algorithm in CF (ADOMAVICIUS; TUZHILIN, 2005). It finds the *k* students, among those who evaluated the resource O_j , more similar to the target learner. The goal is to predict the rating the target learner would give O_j using the ratings that O_j received from other similar students (nearest neighbors).

KB recommendation aggregates knowledge about the student and learning materials to use them in the recommendation process. The similarity calculation, in this case, considers only students contextually similar to the target learner L to predict P_j^L . For example, the similarity calculation takes into account only students who have a similar level of knowledge or learning style as the target learner.

The rating sparsity problem occurs when few students have evaluated the same LO, and the cold-start problem arises when a new student has not rated any LOs. It is not possible to make a reliable calculation of similarity in both cases, so information about students, such as their knowledge level and learning style, can be used in the similarity calculation to predict P_j^L . Thus, the KB recommendation contributes to reducing the rating sparsity and cold-start problems.

It was possible to simplify the experimental tests without compromising them, using only the collaborative filtering proposed in (TARUS; NIU; KALUI, 2018), disregarding the use of KB recommendation, which can be properly used in a real educational context.

4.5 Proposed heuristic algorithm

We create the *lorp_algorithm* (Algorithm 1), which employs a heuristic search procedure (Algorithm 2) to solve the LORP. Unlike the conventional greedy strategy that selects a single column at each step to build the solution, our algorithm selects a list of columns at each step, decreasing the number of steps and finding the solution faster.

Given a column set $C = \{1, \dots, n\}$, the columns chosen in each step are added to the solution S (which starts empty) until S contains a subset of C that covers all the concepts of the $(m \times n)$ input matrix $A = \{a_{ij}\}$. The objective of the *lorp_algorithm* is to find a solution S to the LORP. The list S starts empty (line 2). While there are rows not yet covered by solution S (line 3), one or more columns are chosen to compose the solution. This iterative process (lines 3-25) ends when the objective of the problem is reached, that is, all rows (concepts) are covered.

Algorithm 1 Lorp algorithm

Input: Matrix A , $costs$, k , num_rows

Output: S

```

1: Let  $num\_rows$  be the number of rows in the matrix
2:  $S \leftarrow \emptyset$ 
3: while The number of rows covered by  $S < num\_rows$  do
4:    $H[j] \leftarrow$  calculate the heuristic  $Gain([j])$ ,  $\forall j \in C - S$ , according to Eq. (7)
5:    $j_{max} \leftarrow argmax\{H[j], \forall j \in C - S\}$ 
6:   Let  $uncov_{j_{max}}$  be the set of rows covered by  $j_{max}$  and not covered by  $S$ .
7:    $D \leftarrow \{j \mid \forall j \in C, j \text{ covers at least one row in } uncov_{j_{max}}\}$ 
8:   if  $k > |D|$  then
9:      $k \leftarrow |D|$ 
10:  end if
11:   $top_k \leftarrow []$ 
12:  while  $|top_k|$  is not equal to  $k$  do
13:     $H[j] \leftarrow$  calculate the heuristic  $Gain([j])$ ,  $\forall j \in D - top_k$ , according to Eq. (7)
14:    Let  $j \leftarrow argmax\{H[j], \forall j \in D - top_k\}$ 
15:     $top_k \leftarrow insert(top_k, j)$  ▷ Insert column  $j$  at the end of  $top_k$ 
16:  end while
17:   $candidates \leftarrow heuristic\_search(matrix, costs, top_k, num\_rows, cost_{j_{max}}, uncov_{j_{max}})$ 
18:  Let  $best_i \leftarrow argmax\{Gain(candidates[i]), 1 \leq i \leq |candidates|\}$ 
19:  Let  $branch\_nodes_{best}$  be the mapped columns of the indexes of  $candidates[best_i][0]$ 
20:  if  $Gain(branch\_nodes_{best}) > Gain([j_{max}])$  then
21:     $S.extend(branch\_nodes_{best})$ 
22:  else
23:     $S.extend([j_{max}])$ 
24:  end if
25: end while
26: return The best found solution  $S$ 

```

The set top_k contains the k columns with the highest heuristic gain. The selection of the column with the highest heuristic gain is executed by the *argmax* function in lines

5, 14 and 18. In the event of a tie, the leftmost column in the input matrix is chosen. A conventional greedy approach would choose the j_{max} column (line 5), but in our algorithm, some columns generated from the top_k set are also considered as candidates to replace j_{max} column. The heuristic gain in line 13 is calculated based on the concepts covered by j_{max} and not covered by S . It considers cov as the number of rows covered by both j and j_{max} but not covered by S , and $uncov$ as the number of rows covered by j_{max} but not covered by j .

Definition 1. The *partial solution* S' is a work-in-progress solution of the Lorp algorithm, whose columns do not yet cover all rows of the input instance, i.e., it is an infeasible solution.

Definition 2. A *branch* consists of a list containing three elements: [$branch_nodes$, $cost$, $uncov_rows$]. The $branch_nodes$ corresponds to the indices of the columns that can compose the solution, it is a path from the root of the tree (first element) to a leaf node (the last one). The $cost$ represents the cumulative cost of the columns in $branch_nodes$. The $uncov_rows$ list stores the rows covered by j_{max} but not covered by the $branch$ or the columns in S' .

Definition 3. A *candidate branch* is a branch whose $uncov_rows$ is empty, i.e., its columns cover the rows covered by j_{max} but not covered by the columns in S' .

The $branch_nodes$ with the highest heuristic gain ($candidates[best_i][0]$) is compared to j_{max} and the one with the superior gain is chosen to compose the solution (lines 20-24). After adding either the j_{max} column or the columns associated with the best branch to the solution S , if S covers all rows, then the algorithm returns S ; otherwise, the process continues until all rows are covered.

$Gain[j]$ corresponds to the heuristic gain of the solution S when it receives column $j \in C - S$. The calculation of $Gain$ is given by:

$$Gain[j] = \frac{cov}{costs[j] * (1 + uncov)} \quad (7)$$

where cov is the number of rows covered by j and not covered by S , and $uncov$ is the number of rows not covered by $S \cup \{j\}$.

The variable $candidates$ (line 17) represents a list of branches generated by the procedure $heuristic_search(matrix, costs, top_k, num_roots, cost_{j_{max}})$. The first element of each branch (Def. 2) is a list ($branch_nodes$) containing a unique combination of column indices. The $branch_nodes$ are composed of the column indices of the top_k set, which includes the k non-solution columns with the highest $Gain$.

Each $branch_nodes$ includes at least one column index (a root that derives from top_k) and at most k column indices. Thus, index 1 (associated with column top_1) will be the root of first tree and index 2 (associated with column top_2) will be the root of the second

tree, allowing the creation of up to k trees. The number of trees created corresponds to the number of roots $num_roots \leq k$.

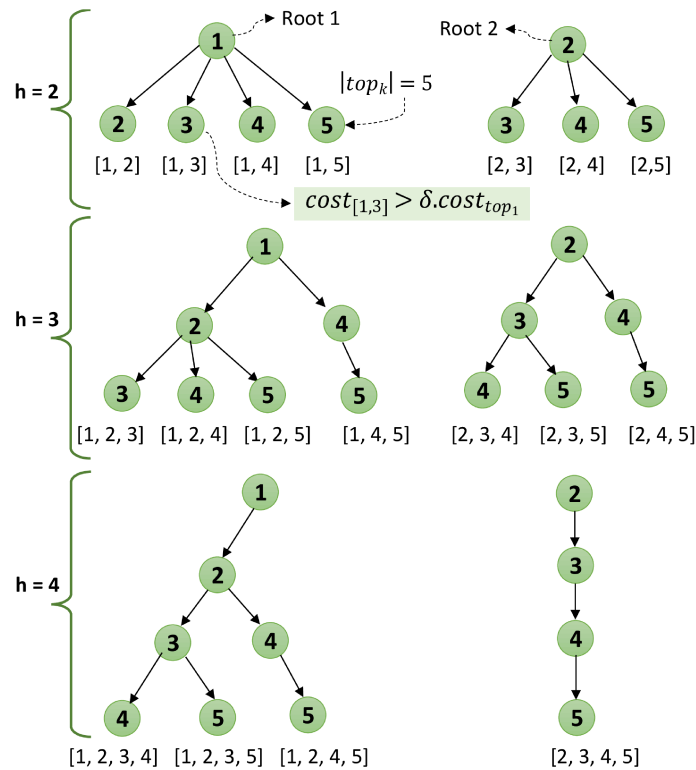


Figure 12 – How heuristic_search procedure works for $top_k = [9, 7, 5, 8, 15]$

Fig. 12 shows an example where twenty *branch_nodes* are created from two trees ($num_roots = 2$). The tree with root equal to 1 has *branch_nodes* [1], [1, 2], [1, 3], [1, 4], [1, 5], [1, 2, 3], [1, 2, 4], [1, 2, 5], [1, 4, 5], [1, 2, 3, 4], [1, 2, 3, 5] and [1, 2, 4, 5], and the tree with root equal to 2 has *branch_nodes* [2], [2, 3], [2, 4], [2, 5], [2, 3, 4], [2, 3, 5], [2, 4, 5], and [2, 3, 4, 5]. Each *branch_nodes* represents one of the possible combinations to be generated with the k indices of top_k (note that $k = |top_k|$). In the example in Fig. 12, $k = 5$ and $top_k = [9, 7, 5, 8, 15]$, then the *branch_nodes* [2, 4] corresponds to columns 7 (index 2) and 8 (index 4).

To limit the number of index combinations, this heuristic procedure has two thresholds: the number of roots (num_roots) and the cost of the branches. Note that with $k = 5$, it is possible to generate $C(5, 1) + C(5, 2) + C(5, 3) + C(5, 4) + C(5, 5) = 31$ *branch_nodes*². However, the number of nodes generated is limited by the number of roots. With two roots, it is possible to generate 24 *branch_nodes* but in the example in Fig. 12, 20 were generated (including the roots [1] and [2]), as the branch cost is another threshold that limits the number of combinations.

² A combination is a selection of items from a set with distinct members, where the order of selection does not matter. For a set with n elements, the number of k -combinations is given by $C(n, k) = \frac{n!}{k!(n-k)!}$

Algorithm 2 heuristic_search procedure**Input:** $matrix, costs, top_k, num_roots, cost_{j_{max}}, uncov_{j_{max}}$ **Output:** $branches$

```

1:  $k \leftarrow |top_k|$ 
2:  $trees \leftarrow []$ 
3:  $size\_trees \leftarrow 0$ 
4:  $candidates \leftarrow []$ 
5:  $size\_candidates \leftarrow 0$ 
6: if  $num\_roots > k$  then
7:    $num\_roots \leftarrow k$ 
8: end if
9: for  $j$  from 1 to  $num\_roots$  do
10:   $new\_cost \leftarrow costs[top\_k[j]]$ 
11:   $uncov_j \leftarrow \{i \mid \forall i \in uncov_{j_{max}}, matrix[i][j] = 0\}$ 
12:   $new\_branch \leftarrow [[j], new\_cost, uncov_j]$ 
13:  if  $uncov_j$  then
14:     $trees \leftarrow insertion\_sort(trees, new\_branch)$ 
15:     $size\_trees \leftarrow size\_trees + 1$ 
16:  else
17:     $candidates \leftarrow insertion\_sort(candidates, new\_branch)$ 
18:     $size\_candidates \leftarrow size\_candidates + 1$ 
19:  end if
20: end for
21: while  $trees \neq \emptyset$  do
22:   $head\_branch \leftarrow tree[0]$   $\triangleright$  Get first branch:  $[branch\_nodes, cost, uncov\_rows]$ 
23:   $trees \leftarrow remove\_head(trees)$   $\triangleright$  Remove first branch
24:   $size\_trees \leftarrow size\_trees - 1$ 
25:   $j \leftarrow head\_branch[0][-1] + 1$   $\triangleright$  Gets last element of  $branch\_nodes$  and add 1
26:  while  $j \leq k$  do
27:     $new\_branch\_nodes \leftarrow head\_branch[0] + [j]$ 
28:     $new\_cost \leftarrow head\_branch[1] + costs[top\_k[j]]$ 
29:    if  $new\_cost > \delta * cost_{j_{max}}$  then
30:       $j \leftarrow j + 1$ 
31:      continue  $\triangleright$  Go to the next iteration of the loop
32:    end if
33:     $uncov_j \leftarrow \{i \mid \forall i \in uncov \text{ of } head\_branch, matrix[i][j] = 0\}$ 
34:    if  $uncov_j$  and  $new\_cost > cost_{j_{max}}$  then
35:       $j \leftarrow j + 1$ 
36:      continue  $\triangleright$  Go to the next iteration of the loop
37:    end if
38:     $new\_branch \leftarrow [new\_branch\_nodes, new\_cost, uncov_j]$ 
39:    if  $uncov_j$  then
40:       $trees \leftarrow insertion\_sort(trees, new\_branch)$ 
41:       $size\_trees \leftarrow size\_trees + 1$ 
42:    else
43:       $candidates \leftarrow insertion\_sort(candidates, new\_branch)$ 
44:       $size\_candidates \leftarrow size\_candidates + 1$ 
45:    end if
46:  end while
47: end while
48: return  $candidates$ 

```

The LORP solution is incrementally constructed in Algorithm 1, which selects the column with the highest gain (j_{max}) to insert into solution S , similar to greedy heuristics. However, before this greedy insertion, it is checked whether there is a combination of columns (candidates) whose gain is better than j_{max} . These candidates are columns that cover all the rows covered by j_{max} and that are not covered by the partial solution. Although the candidates may have a higher cost than j_{max} , they might also cover other rows not covered by j_{max} and not yet covered by the partial solution, thus potentially having a higher gain than j_{max} , since our heuristic (Eq. 7) evaluates candidate branches based on both cost and the number of rows covered and not covered by the branches.

Algorithm 2 is the *heuristic_search* procedure called on line 17 of Algorithm 1 to perform heuristic search. The heuristic procedure starts building *trees* from the roots (lines 9-20), initially generating a number of branches equal to *num_roots*. Each branch contains the index of a column j belonging to top_k , the cost of j and the $uncov_j$. If $uncov_j$ is different from empty, then the branch is inserted into *trees* (lines 13-15), otherwise it is inserted into *candidates* (lines 16-18). This insertion is ordered by the cost of the branch. Thus, the first element of *trees* corresponds to the branch with the lowest cost.

In the loop of lines 21 to 47, the first branch of *trees* is removed (lines 22-23) and expanded in the inner loop (lines 26-46). If the cost of the new branch resulting from the expansion is greater than $\delta * cost_{j_{max}}$, then the branch is ignored (lines 29-31), as it is not considered a suitable candidate to replace j_{max} , otherwise it is either inserted into *trees* sorted by cost (lines 39-41) or inserted into *candidates* sorted by cost (lines 42-44) depending on whether $uncov_j$ is an empty list or has rows not covered by j .

The value δ is a margin that allows a candidate branch to have a cost slightly greater than j_{max} , but non-candidate branches, i.e. with non-empty $uncov$, that exceed this margin are not expanded further (lines 34-36). When a candidate branch is found, it is stored in *candidates* (lines 17 and 43 of Algorithm 2) and is not expanded further, such as the branch [1, 2, 3, 4], whose columns [9, 7, 5, 8] cover all rows covered by j_{max} and not yet covered by the solution.

The Algorithm 1 used to solve the LORP is implemented within our RS. In Chapter 5, we compare the Algorithm 1 with GA, PSO, and the exact and greedy algorithms to evaluate the time and quality of their recommendations. This LO recommendation process starts in the interface model, through the chatbot that establishes a friendly communication channel with the student, and in the knowledge domain model, through the ontology, which uses inference rules and relies on the reuse of Web content to collect the LOs that are likely to be recommended to the student. The LOs that bring the subjects that the student needs to learn and intervention LOs are recommended according to the student's need to learn new concepts and solve doubts during the study, respectively.

4.6 Anya chatbot and gamification

To create a user-friendly and engaging interface for the proposed recommender system, we developed a chatbot named Anya. This chatbot, implemented in Python using the Microsoft Bot Framework, leverages our personalized content recommender system to teach new concepts and address doubts through gamification, thereby enhancing student motivation. In this study, Anya was trained specifically in a Procedural Programming course focused on the C language and has been integrated with Moodle, a widely utilized learning platform in universities.

Table 5 – Chatbot functions/intents in Gamification

Intent	Description	Example
Learn New Concept	Used to learn new concepts about C.	I want to learn if (or any other C topic)
Evaluate	Call it to evaluate your experience using the chatbot.	I want to give you a rating.
Welcome	If you feel like having a quick chat, just use this function.	Good morning.
Capture Profile	This function is called when you want to answer questions from the index of learning styles.	I want to answer questions from the index.
Goodbye	It's never a farewell. After this function, a logoff will be performed.	Bye!
I'm Sad	An attempt to cheer you up.	I feel bad!
Manual	Sends a link to this PDF you are viewing.	What can you do?
Show Profile	Shows your learning style, score, level, and ranking.	I want to know my profile.
Practice	Sends the user links to questions about the C programming language.	I want to practice a bit.
Report Bug	You will receive a link to a form where you can send us a detailed message about any bug you encountered, which will be resolved quickly.	Report bug.
Clear Doubt	Clear all your doubts about C with this function.	How to use for (what you are doubtful about).

The chatbot has 11 intents (see Table 5). Each intent is the purpose or goal behind the user's messages, crucial for effective automated responses. The personalized content recommendation intents are: Learn New Concept and Clear Doubt. To evaluate such LOs, the Evaluate intent must be triggered. Another important intent is the Capture Profile, which is triggered for the student to answer ILS questions; Anya can show the student's profile through Show Profile. In addition, Anya can tell a joke if the student is sad (I'm Sad), present the user manual (Manual), and encourages the student to put into practice the acquired knowledge (Practice).

Table 6 – Actions and rewards

Action	Condition	Reward	Positive Effect
Answer index questions	At least one question	1 pt	
Answer a quiz question (easy)	At least one question	5 pts	After 3 consecutive correct answers, the score for each question increases by 5%.
Answer a quiz question (medium)		10 pts	
Answer a quiz question (hard)		15 pts	
Get a quiz question wrong	Got only one question wrong	1 pt	
Clear a doubt Learn something new		3 pts	If you evaluate the returned learning object, you will receive a 2-point bonus

Table 7 – Gamification levels

Level	Name	Ceiling score
Level 1	Beginner 1	5
Level 2	Beginner 2	15
Level 3	Beginner 3	30
Level 4	Prodigy 1	50
Level 5	Prodigy 2	75
Level 6	Prodigy 3	120
Level 7	Highlight 1	200
Level 8	Highlight 2	280
Level 9	Highlight 3	360
Level 10	Star 1	420
Level 11	Star 2	500
Level 12	Star 3	600
Level 13	Computer Scientist 1	700
Level 14	Computer Scientist 2	800
Level 15	Computer Scientist 3	1000
MAX Level	Turing's Pride	-

The intentions Capture Profile, Practice, Clear Doubt and Learn New Concept are related, respectively, to the following actions: answering ILS questions, getting quiz questions right or wrong, resolving a doubt and learning something new. The way each action is scored is presented in Table 6. The gamification process considers the level of the questions and the maximum number of errors to define the student's reward. In addition, there are positive effects that increase the percentage of points earned if the student gets

more than three correct answers in a row or if the learner evaluates at least one of the recommended LOs.

As students earn points, they move up a level and earn badges. Learners can reach 16 different levels. To leave a level and move on to the next one, it is necessary to surpass the maximum score (ceiling score) of the current level. Table 7 shows all the levels with their respective maximum scores. For example, a student with a score greater than 30 and less than or equal to 50 is at level 4, which corresponds to a Prodigy 1. Regarding achievements, students can earn up to 9 badges as shown in Table 8.

Table 8 – Achievements (badges)

Name	Condition	Score
Knows everything about you	Completed all questionnaire questions	20
Your opinion matters	Gave feedback on the chatbot (positive or negative)	15
Truly a prodigy	Achieved Prodigy Level 1	5
You are my highlight	Achieved Highlight Level 1	10
You are my sunshine	Achieved Star Level 1	20
A new scientist emerges	Achieved the impressive Computer Scientist Level 1	30
Turing would be proud	Reached the maximum level (Achieved >900 points)	100
You are a machine	Answered 10 quiz questions correctly in a row	15
Always right, right, and right	Answered 15 quiz questions correctly in a row	20

Upon registering in the RS through the chatbot, students are prompted to honestly answer at least 8 questions from the ILS questionnaire, 2 questions from each dimension. This helps mitigate the cold-start problem. For each question answered, the student earns 1 point, encouraging them to complete the remaining ILS questions throughout their learning process. Students also accumulate points by correctly answering easy (5 pts), medium (10 pts), or difficult (15 pts) quiz questions; resolving doubts or learning new concepts (3 pts); and evaluating the recommended LOs (2 pts). Additionally, students receive badges, such as “Knows everything about you”, for completing all ILS questions, which aids in tailoring the recommended LOs to each student’s learning style.

After logging into the system, the student is invited to answer more ILS questions so that the system can capture his/her profile more accurately. In Fig. 13, there is an example in which the student chooses not to answer the IDL questions, preferring to clarify a doubt by typing “How to use the while loop”. Subsequently, at least one group of interventions is provided. This group is a standardized educational PDF (1-4 pages) comprising five succinct interventions: definition of the concept, its purpose, key points to note, examples of its usage, and answers to frequently asked questions related to the concept.

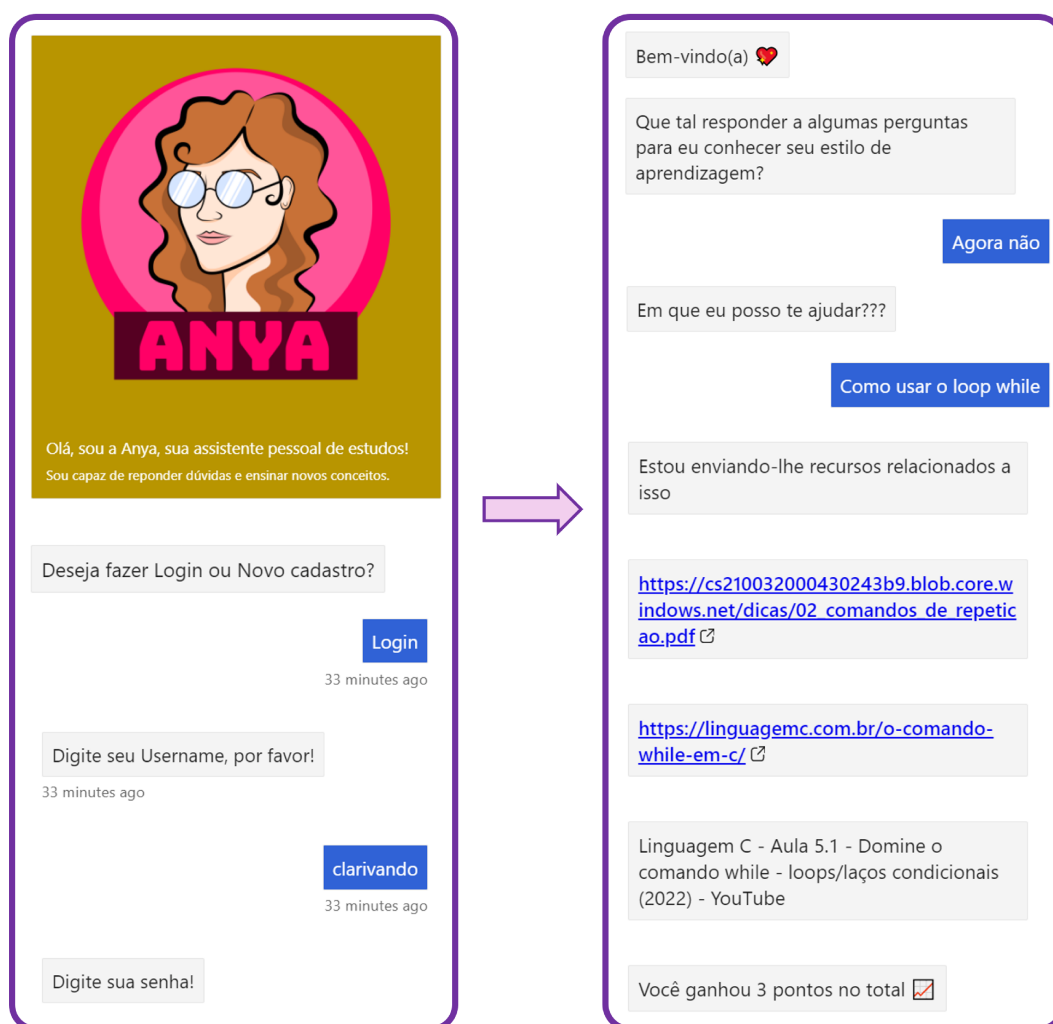


Figure 13 – Recommendation of LOs in the Anya chatbot

Conversely, to learn a new concept (the less interventions the better), students can type, for example, “I want to learn about the for loop” (in Portuguese). They are then recommended three different resources (two PDFs and one video) on the same concept (“for”). After reviewing the recommendations, students can rate each LO on a scale from 1 to 5 (see Fig. 14). These ratings help filter high-quality LOs, especially those sourced from the Web without human intervention, and are used in collaborative recommendation. The ratings also contribute to the calculation of P_j^L in Eq. (6), which personalizes content recommendations for students based on the scores of peers with similar assessment profiles.

To view your profile, the student can type “show my profile” (in Portuguese), upon which Anya displays the student’s level, accumulated score, achievements, the number of ILS questions answered, and their learning styles as shown in Fig. 15.

Students are encouraged to put their acquired knowledge into practice by answering questions related to the subject as show in Fig. 16. In this scenario, they type “I want to practice” and then choose the difficulty level and the number of questions they wish to answer. Each correct response enhances their score and contributes to earning badges.

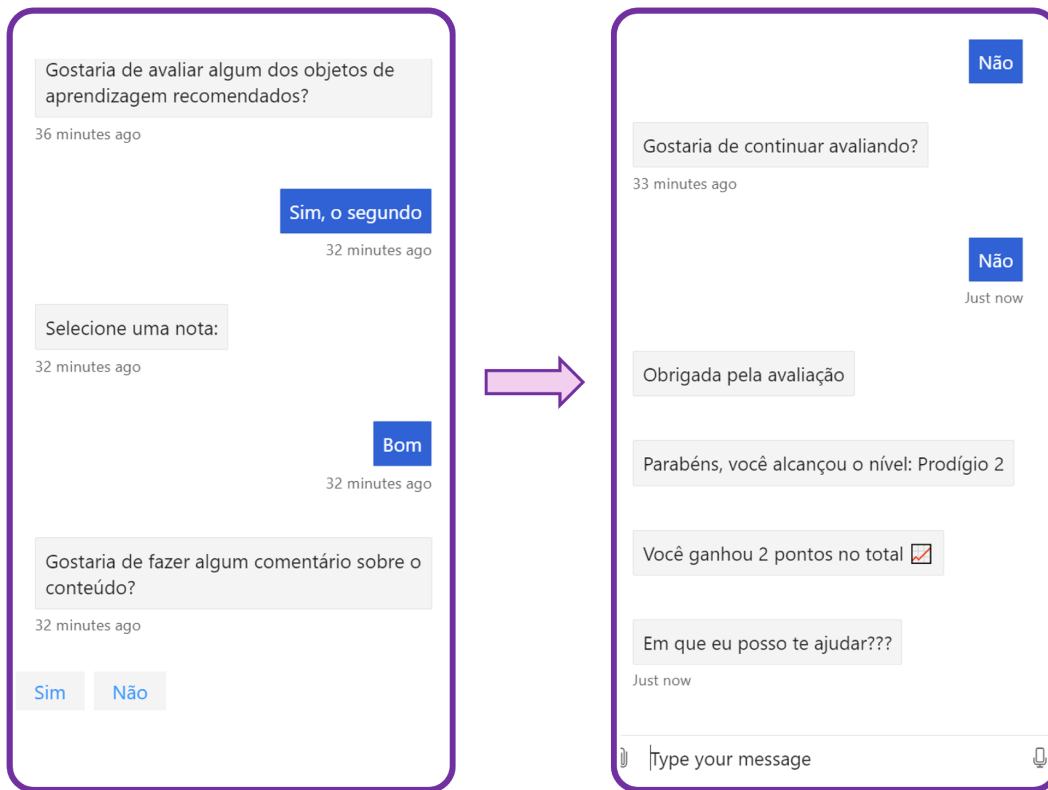


Figure 14 – Evaluation of LOs in the Anya chatbot

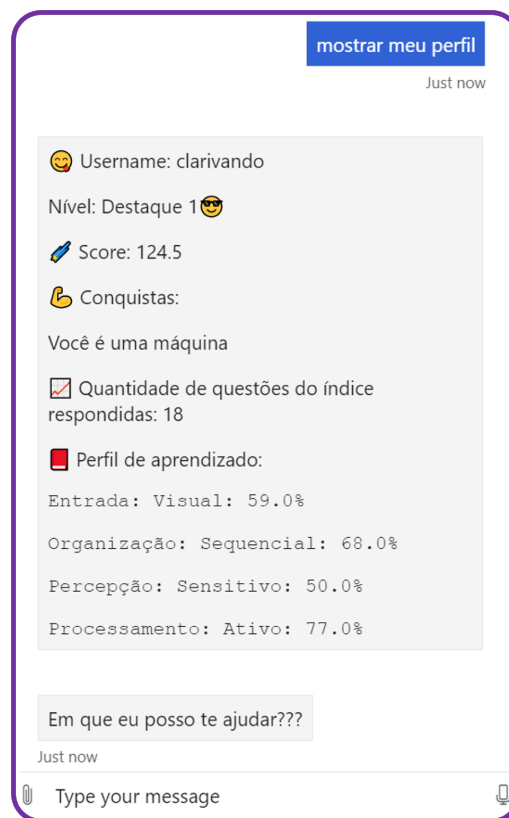


Figure 15 – Learner profile and gamification dashboard

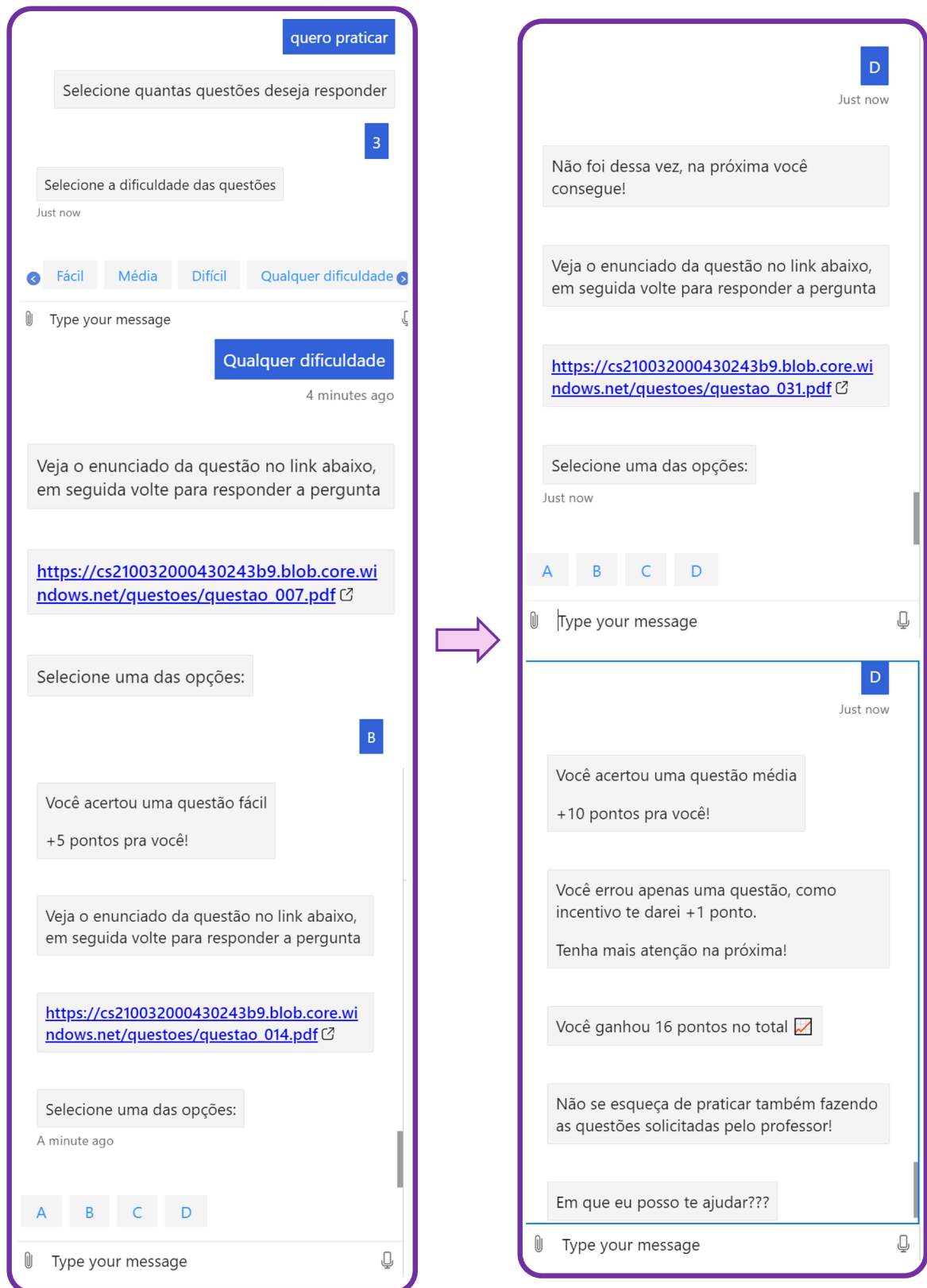


Figure 16 – Gamification questions/quizzes

4.7 Modeling the recommender system

In this section, we show the modeling of the recommender system, highlighting some diagrams that help in understanding the system as a whole. For this modeling, we use the Unified Modeling Language (UML) (BOOCH; RUMBAUGH; JACOBSEN, 1997), which was created to establish a common, semantically and syntactically rich visual modeling language for architecting, designing, and implementing complex software systems, both structurally and behaviorally. For this, there are many UML diagrams.

The sequence diagram is a UML diagram that shows process interactions organized in temporal sequence and facilitates the understanding of the system by showing how the sequence of the business rule occurs in the various components within the system. Because it is intuitive, the sequence diagram facilitates interpretation and aids in communication between the members who develop software.

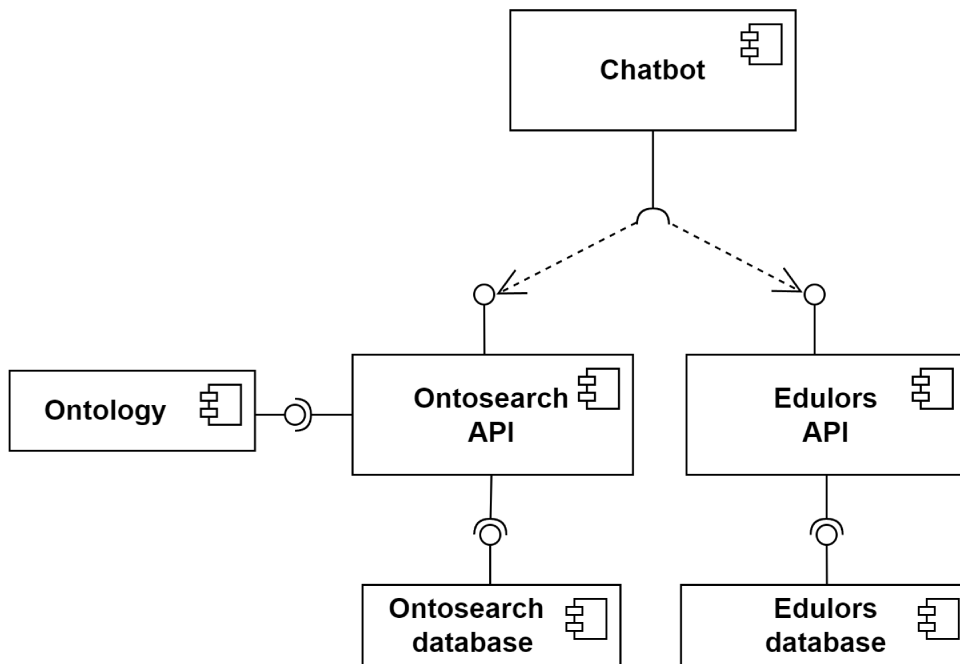


Figure 17 – Component diagram of the recommender system

The proposed RS has six main components as shown in Fig. 17: Ontosearch API, Edulors API, Chatbot, Ontology, Ontosearch database and Edulors database, all deployed on Microsoft Azure. Azure is the cloud computing platform run by Microsoft. It offers management, access, and development of applications and services to individuals, businesses, and governments through its global infrastructure. The first three modules were implemented in Python programming language, the Ontology component corresponds to an ontology described in OWL and the databases were created by the MySQL database management system, which uses the SQL as an interface.

Implementing a computational service as an Application Programming Interface (API) is advantageous because it enables modularity, facilitates integration with other systems,

enhances scalability, supports different client applications, promotes reusability of code, improves security by abstracting the backend, and allows for easier maintenance and updates. The Ontosearch (see Figs. 29 and 30) and Edulors (see Figs. 31-33) API routes are presented in Appendix B. These routes are the API interface and can be thought of as a service contract between two applications. The Swagger toolkit in Python was used for API documentation. This toolkit provides robust capabilities for generating interactive documentation and enabling thorough testing, ensuring that APIs are well-documented and easily accessible for developers.

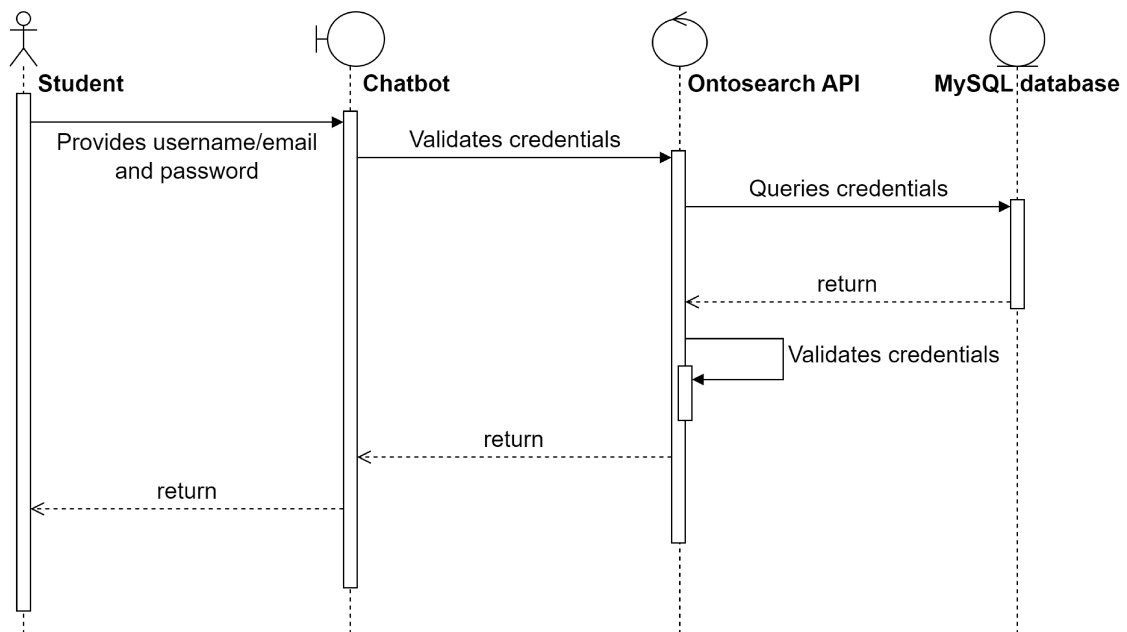


Figure 18 – Sequence diagram of the login process

Fig. 18 shows the login process. In it, the user provides their credentials, username (or email) and password, then the chatbot requests validation of these credentials to the Ontosearch API, which searches for the credentials in the database to perform validation and returns the response to the chatbot, which returns it to the student.

The Ontosearch API contains the routes (services) for manipulating the ontology, allowing: reading information about students and learning objects modeled in the ontology; capturing student profiles; saving ILS questions answered by the student and returning those not yet answered; and performing web searches that return metadata from Wikipedia sections, including the concepts covered, the title, and the URL of the section, as shown in Annex C.

The Ontosearch API serves as a channel for the Chatbot and the Edulors API to access the ontology. The Edulors API provides routes for: recommending LOs; evaluating LOs by students; and creating, reading, updating, and deleting LOs, concepts, and types of learning resources in the Edulors database. Additionally, the Edulors API includes feedback routes related to the chatbot evaluations conducted by students, as well as

gamification routes that allow for capturing and updating the student's score, level, and achievements, and returning question-type LOs used to compose the quizzes.

Although the ontology stores the metadata of the LOs and students, which is important for inferences that assist in the recommendation process, some information about these entities is stored in the database, avoiding unnecessary access to the ontology through routes that do not depend on inference, making the recommender system more efficient in response time for the end user.

Experimental Results

5.1 Computational results and comparison of algorithms

The algorithms were implemented in Python 3 using numpy, math, random and statistics libraries. The experiments were run on a notebook with AMD Quad-Core A10-9600P 2.40 GHz/8G and Windows 10 OS.

We have employed statistical tests designed to detect significant differences and to estimate their magnitude from the tests performed. The experiments were designed as a randomized complete block design (RCBD). By treating the problems as blocks, it was possible to model and remove the effects of different instances on the performance of the algorithm and obtain an overall performance difference across all test instances used (MONTGOMERY, 2012). The null hypotheses of absence of differences among the methods evaluated over all problems were considered against two-sided alternatives. To avoid the assumptions of normality, Wilcoxon test (non-parametric) was employed.

After testing for significance, least squares estimators of the block (instance) effects were obtained and subtracted from the samples, thus allowing a problem-independent estimation of the effect size for each algorithm (MONTGOMERY, 2012). The estimations of effect size were calculated by Tukey’s test for post-hoc analysis (TUKEY, 1949).

5.1.1 Benchmark instances and dataset

The dataset has 24 instances, as shown in Table 9. Note that there are 6 different values (2, 6, 10, 25, 40, 55) for the number of rows (concepts) and 4 values (100, 500, 2000, 10000) for the number of columns (LOs). Each instance was solved 10 times to find the average runtime, the average prediction and the average number of interventions. The tests were executed in two recommendation modes, the less/more interventions the better, which were explained in Section 4.4.

Table 9 shows features of a small dataset that we created to test solutions to the LORP. The benchmark instances simulate the educational context. The main characteristics of the instances are the name of instances (Inst), the number of rows (m), the number of columns (n) and the density. Instances were created to have densities equal to either 10%, 20% or 50%.

Table 9 – The main features of the used benchmark instances

Inst	m	n	Density (%)	Inst	m	n	Density (%)
1	2	100	50	13	25	100	10
2	2	500	50	14	25	500	10
3	2	2000	50	15	25	2000	10
4	2	10000	50	16	25	10000	10
5	6	100	50	17	40	100	10
6	6	500	50	18	40	500	10
7	6	2000	50	19	40	2000	10
8	6	10000	50	20	40	10000	10
9	10	100	20	21	55	100	10
10	10	500	20	22	55	500	10
11	10	2000	20	23	55	2000	10
12	10	10000	20	24	55	10000	10

Each instance is composed of the input matrix and a cost vector (see Fig. 11), but in our dataset, this vector is interpreted as a dissimilarity vector that is used to calculate the cost vector. The number of columns covering each row of the input matrix is defined by the density of the instance. For example, if the density is 10%, then for each row of the input matrix, 10% of the total number of columns are randomly chosen to cover it.

The input matrix and the dissimilarity vector of the instances were created to simulate a real-world scenario. Dissimilarity ($diss$ in Eq. (6)) is an integer value ranging from 1 to 10% of the number of columns in the instance. For example, in 500-column instances, the columns from 1 to 10 have $diss = 1$, columns from 11 to 20 have $diss = 2$, and so on. Note that the last 10 columns have $diss = 50$ (10% of 500 columns).

In addition to $diss$, it is necessary to determine P_j^L and I_j to calculate the cost using Eq. (6). To determine the variable P_j^L , we use the rating matrix with the ratings the students gave for LOs they evaluated (see Table 10). It has 30 students (each one in a row) and 50 LOs, each in a column; they correspond to the first 50 LOs of the instances. The value of row i and column j corresponds to the grade, in an integer interval $[1, 5]$, that student i gave to LO j . If this grade is 0, then student i has not rated LO j . The first row of the rating matrix is used to identify the type of each LO. The value of the first row and column j is 1 if the LO j is of the intervention type. Otherwise, the value is 0. Half of the LOs (25) were randomly chosen to be of the intervention type. The other LOs of each instance ($O_{51}, O_{52}, \dots, O_n$) have $I_j = 0.5$ in both recommendation modes (the less/more interventions the better).

The variable I_j is related to interventions, which include definitions, usefulness, attention points, examples, and hints. These interventions are grouped into didactic and

structured educational resources, each presented as a PDF file containing a header, definition, usefulness, attention points, examples, and frequently asked questions with hints, as illustrated in Figures 26-28 of Appendix A

For the first 50 LOs of each instance it is possible to predict P_j^L . The other LOs of each instance ($O_{51}, O_{52}, \dots, O_n$) were not rated by any student (cold-start problem), so the prediction value assigned to them is given by the arithmetic mean of the ratings of the LOs evaluated by the target student L .

For the execution of the tests, the target student $L11$ was chosen in Table 10. The k -NN algorithm used to calculate the prediction is set to $k = 3$. It finds the three students, among those who evaluated the resource O_j , more similar to the learner $L11$.

5.1.2 Selected algorithms and parameters

In this work, GA, PSO, Exact, Greedy and Lorp algorithms were considered for the LORP solution. We utilized the genetic algorithm proposed by (BELIZÁRIO JÚNIOR; DORÇA, 2018). In this algorithm, each individual is represented by a vector of integers with m positions, corresponding to the number of rows. The integer value at position i represents the column that covers row i . Tournament selection is used to choose four individuals (forming two pairs). In each iteration, two new individuals are generated by applying a fitness-based crossover operator to each pair. A mutation is applied with a probability of 10%, where a randomly selected LO (integer value) is replaced with another. Two individuals with below-average fitness are randomly selected to be replaced by the two newly generated individuals.

The second algorithm employed to solve the LORP is an adaptation of PSO, known as Jumping Particle Swarm Optimization (JPSO) (BALAJI; REVATHI, 2016). In this algorithm, each particle is represented as a binary vector, where each element is 1 if the corresponding column belongs to the particle and 0 otherwise. Each particle retains the best position it has encountered (p_best) and, with a certain probability, moves towards either its p_best or the global best position (g_best) through a merge procedure. For the tests conducted, ten particles were used. The JPSO typically converges within the initial iterations, demonstrating its efficiency in quickly finding optimal or near-optimal solutions.

The third algorithm is the Exact, which belongs to the Pulp library (MITCHELL; OSULLIVAN; DUNNING, 2011), which is an open source package written in Python to express linear programming models in a way similar to the conventional mathematical notations. The Greedy algorithm (GOLAB *et al.*, 2015) is the fourth algorithm used in testing and was implemented as described in (BELIZÁRIO JÚNIOR *et al.*, 2024); the goal is to cover all concepts, using a number of columns at most equal to the number of rows (m) of the input matrix. Therefore, for solving the LORP, the Greedy algorithm is configured with $s = 1$ and $k = m$.

Table 10 – Rating matrix used in testing

	O1	O2	O3	O4	O5	O6	O7	O8	O9	O10	O11	O12	O13	O14	O15	O16	O17	O18	O19	O20	O21	O22	O23	O24	O25	O26	O27	O28	O29	O30	O31	O32	O33	O34	O35	O36	O37	O38	O39	O40	O41	O42	O43	O44	O45	O46	O47	O48	O49	O50								
type	1	0	0	1	1	0	1	0	1	0	1	1	0	0	0	0	1	0	1	1	1	0	1	1	0	0	1	1	1	0	0	0	1	1	1	0	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0					
L1	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0					
L2	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0					
L3	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0					
L4	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0					
L5	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0					
L6	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0					
L7	0	2	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0					
L8	0	2	4	0	0	0	0	0	0	0	0	0	0	0	0	0	2	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0					
L9	0	2	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0					
L10	0	1	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0					
L11	0	2	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0					
L12	0	3	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
L13	0	0	0	1	3	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	4	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
L14	0	0	0	1	3	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	4	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
L15	0	0	0	1	3	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
L16	0	0	0	2	3	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
L17	0	0	0	1	4	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
L18	0	0	0	1	5	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
L19	0	0	0	0	0	0	1	2	4	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
L20	0	0	0	0	0	0	1	2	4	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
L21	0	0	0	0	0	0	1	2	4	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
L22	0	0	0	0	0	0	1	2	3	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
L23	0	0	0	0	0	0	1	3	3	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0			
L24	0	0	0	0	0	0	2	3	4	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0			
L25	0	0	0	0	0	0	0	0	0	0	1	2	3	4	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
L26	0	0	0	0	0	0	0	0	0	0	1	2	3	4	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
L27	0	0	0	0	0	0	0	0	0	0	1	2	3	4	5	2	3	1	5	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
L28	0	0	0	0	0	0	0	0	0	0	2	2	3	5	5	1	2	2	4	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
L29	0	0	0	0	0	0	0	0	0	0	3	2	3	3	4	0	0	0	0	0	0	5	1	4	5	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
L30	0	0	0	0	0	0	0	0	0	0	1	1	2	2	4	0	0	0	0	0	0	3	2	5	5	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The *lorp_algorithm* (Algorithm 1) proposed in this work was used in numerical experiments to determine the values of the parameters k , num_roots and δ . Various combinations of values for these parameters were tested until the best configuration was found. The optimal configuration identified was $k = 153$, $num_roots = 38$ and $\delta = 1.3$, which was used to run the tests.

5.1.3 Comparison of the versions of each algorithm with and without prediction

The validation of the variables used in the cost calculation does not depend on the algorithm used to solve the LORP. In fact, we have shown this in a previous work (BELIZÁRIO JÚNIOR *et al.*, 2023), in which the cost variables were validated using the GA, JPSO, Exact and Greedy algorithms. Therefore, to evaluate the collaborative filtering (prediction) implemented in the proposed approach, we use only the Lorp algorithm (Algorithm 1) and the Exact algorithm to solve the LORP.

Two versions of each algorithm were implemented. The difference between them relates to how the c_j cost is calculated. In the first, the cost is calculated by Eq. (8), which does not use the variable P (prediction), while in the second, the cost is calculated by Eq. (6), which uses the prediction.

$$c_j = diss(O_{ideal}, O_j) + (1 - I_j) * \max_{j \in \{1, \dots, n\}} diss(O_{ideal}, O_j) \quad (8)$$

The two versions of each algorithm are compared in Table 11. In the No columns, c_j is calculated by Eq. (8), and in the Yes columns, c_j is calculated by Eq. (6). Each LO in the solution has a rating that is either given in Table 10 or predicted by calculating P_j . The results obtained for the experimental comparison are summarized in Table 11, which reports the mean values of the 10 runs (replications). These values represent the importance that the LO has for the student. The closer to 1, the greater the importance that the LO has for the student.

From Table 11, it can be seen that the Yes variables have a higher median value than No variables in the two recommendation modes (the less/more interventions the better). This means that the use of collaborative filtering (variable P_j) contributes to the recommendation of the LOs with the best rating for the student.

For each algorithm analyzed, the differences between the Yes and No variables (using or not using the predictive variable in the cost function) are statistically significant (p -value < 0.05). Table 12 summarizes the results of the statistical analysis (p -value) and the magnitude of the statistically significant differences (*magnitude diff*). The gain in using the CF is greater when the more interventions the better, with the magnitudes of the differences reaching the maximum value of 0.11, approximately. This difference demonstrates that the use of P (collaborative filtering) in the calculation of the objective

Table 11 – Comparison of the average ratings in the solutions with (Yes) and without (No) the P variable in solving the LORP

Instance	The less interventions the better				The more interventions the better			
	Exact		Lorp alg.		Exact		Lorp alg.	
	No	Yes	No	Yes	No	Yes	No	Yes
1	0.697	0.730	0.730	0.730	0.697	1.000	0.697	1.000
2	0.697	0.809	0.730	0.809	0.697	0.995	0.697	0.995
3	0.596	0.809	0.596	0.809	0.697	0.995	0.697	0.995
4	0.697	0.809	0.697	0.809	0.697	0.929	0.697	0.929
5	0.734	0.753	0.703	0.703	0.697	0.929	0.697	0.929
6	0.697	0.753	0.697	0.753	0.601	0.929	0.686	0.929
7	0.663	0.753	0.663	0.714	0.697	1.000	0.697	1.000
8	0.697	0.787	0.674	0.787	0.843	0.856	0.843	0.900
9	0.734	0.734	0.734	0.734	0.625	0.723	0.625	0.723
10	0.717	0.753	0.745	0.756	0.697	0.810	0.697	0.813
11	0.746	0.746	0.672	0.746	0.810	0.813	0.751	0.813
12	0.697	0.701	0.705	0.697	0.697	0.798	0.697	0.846
13	0.709	0.735	0.698	0.731	0.729	0.731	0.741	0.761
14	0.698	0.698	0.699	0.698	0.725	0.734	0.729	0.750
15	0.722	0.715	0.706	0.715	0.781	0.783	0.766	0.781
16	0.706	0.713	0.646	0.699	0.728	0.765	0.722	0.773
17	0.692	0.709	0.710	0.750	0.721	0.721	0.733	0.733
18	0.703	0.726	0.704	0.713	0.720	0.764	0.717	0.765
19	0.639	0.709	0.661	0.704	0.749	0.734	0.731	0.761
20	0.695	0.719	0.712	0.734	0.731	0.774	0.714	0.774
21	0.713	0.714	0.711	0.712	0.744	0.744	0.732	0.737
22	0.658	0.706	0.658	0.720	0.711	0.735	0.710	0.742
23	0.658	0.715	0.668	0.726	0.734	0.734	0.711	0.764
24	0.719	0.711	0.700	0.711	0.665	0.763	0.670	0.767
Median	0.697	0.728	0.700	0.728	0.715	0.778	0.710	0.777

Table 12 – Estimated difference in average performance between the prediction variables (No/Yes)

	The less interventions the better		The more interventions the better	
	Exact	Lorp alg.	Exact	Lorp alg.
p-value	< .001	< .001	< .001	< .001
Magnitude diff	0.0428	0.0433	0.1069	0.1177

Note: If p -value < 0.05, then there is a statistically significant difference between the variables.

function (specifically c_j) increases on average 0.11 the average rate of the LORP solution, improving the quality of the LOs recommended to the learners. When the less interventions the better, the increase was on average 0.04 in the average rate of the LORP solutions, relatively more modest gains.

Figure 19 shows a boxplot of the average ratings of the solutions of the LORP. The red boxes represent the average rating without using the P (prediction) variable; i.e., the cost c_j is calculated by Eq. (8). On the other hand, green boxes represent the average rating using the P variable. In this case, the cost is calculated by Eq. (6). The graphs corroborate the data presented in Table 12, demonstrating more significant gains when the more interventions the better. Therefore, improvements can be seen in both recommendation modes and in the two algorithms evaluated with the use of CF. We can see that the algorithms implemented to solve the LORP using the prediction find solutions composed of LOs with better ratings, while not using the prediction decreases the quality of the solutions in relation to the rating.

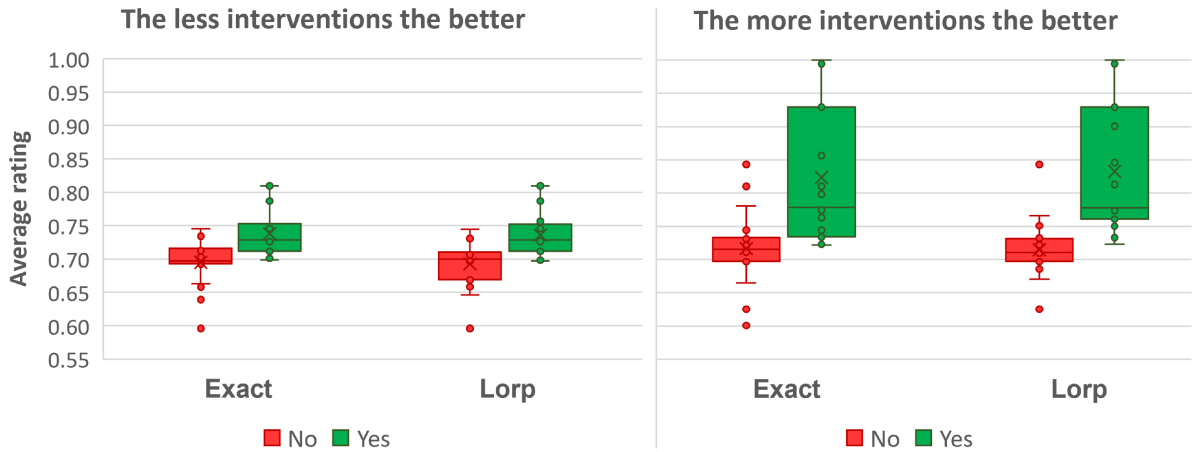


Figure 19 – Boxplot to compare the versions of each algorithm with (Yes) and without (No) prediction

5.1.4 Comparison of the versions of each algorithm with and without intervention variable

To evaluate the I (intervention) variable implemented in the proposed approach, we use the same strategy presented in Section 5.1.3. Two versions of each algorithm were implemented, and the difference between them relates to how the c_j cost is calculated. In the first, the cost is calculated by Eq. (9), which does not use the I variable, while in the second, the cost is calculated by Eq. (6), which uses the intervention variable.

$$c_j = \text{diss}(O_{ideal}, O_j) + (1 - P_j^L) * \max_{j \in \{1, \dots, n\}} \text{diss}(O_{ideal}, O_j) \quad (9)$$

The two versions of each algorithm are compared in Table 13. In the No columns, c_j is calculated by Eq. (9), and in the Yes columns, c_j is calculated by Eq. (6). In Table 13, each value corresponds to an average number of interventions of the 10 runs (replications).

Table 13 – Comparison of the average number of interventions in the solutions with (Yes) and without (No) the I variable in solving the LORP

Instance	The less interventions the better				The more interventions the better			
	Exact		Lorp		Exact		Lorp	
	No	Yes	No	Yes	No	Yes	No	Yes
1	1	0	1	0	1	1	1	1
2	2	0	2	0	2	2	2	2
3	2	0	2	0	2	2	2	2
4	2	0	2	0	2	2	2	2
5	2	0	2	0	2	2	2	2
6	2	0	2	0	2	2	2	2
7	1	0	1	0	1	1	1	1
8	2	0	2	0	2	2	2	3
9	3	0	3	0	3	3	3	3
10	2	0	1	0	2	4	1	4
11	1	0	1	0	1	4	1	4
12	3	0	2	0	3	3	2	4
13	2	1	3	1	2	8	3	7
14	2	0	2	0	2	6	2	7
15	5	0	4	0	5	8	4	8
16	2	0	3	0	2	8	3	6
17	6	0	6	2	6	10	6	11
18	7	0	7	0	7	10	7	10
19	3	0	2	0	3	5	2	5
20	3	0	3	0	3	9	3	9
21	5	2	5	2	5	9	5	11
22	3	1	6	0	3	9	6	6
23	4	0	4	0	4	11	4	8
24	3	0	3	0	3	5	3	7
Median	2.0	0.0	2.0	0.0	2.0	4.5	2.0	4.5

Table 14 – Estimated difference in average performance between the intervention variables (No/Yes)

	The less interventions the better		The more interventions the better	
	Exact	Lorp	Exact	Lorp
p-value	< .001	< .001	0.001	< .001
Magnitude diff	-2.6667	-2.6667	2.4167	2.3333

Note: If p -value < 0.05, then there is a statistically significant difference between the variables.

From Table 13, it can be seen that the Yes variables have a lower median value than No variables in the two algorithms when the less interventions the better. On the other hand, Yes variables have a higher median value than No variables in both algorithms when the more interventions the better. These differences are statistically significant, as shown in Table 14. This table shows the p -value and magnitude of the differences between the analyzed samples.

When the less interventions the better, the magnitudes are negative, demonstrating that the number of interventions returned when I is applied to the objective function is smaller (which is expected in this modality). In the two algorithms, on average, approximately 2.66 less interventions are presented to the learner when I is applied in the objective function.

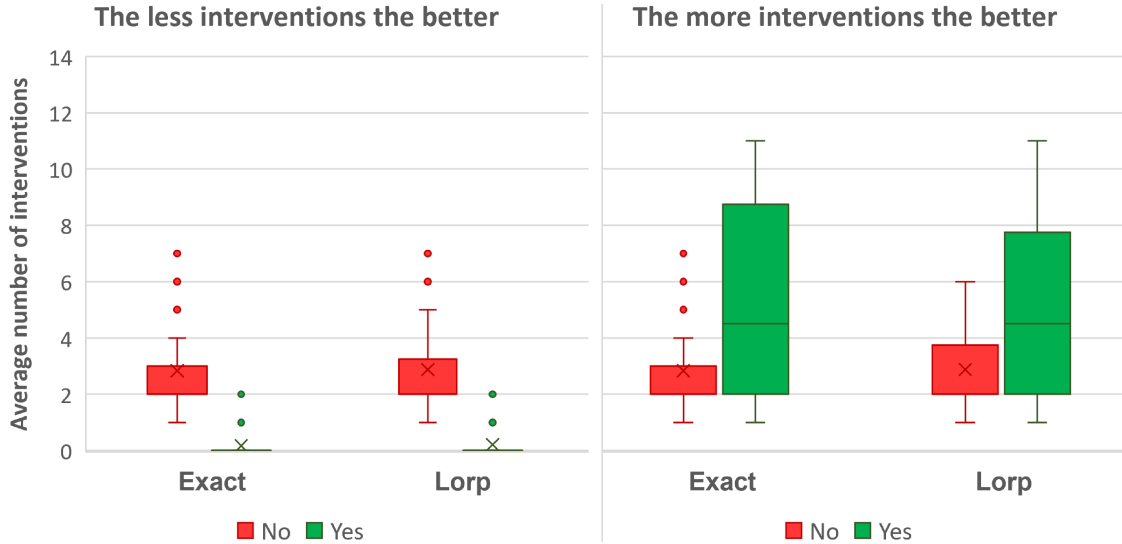


Figure 20 – Boxplot to compare the versions of each algorithm with (Yes) and without (No) the I (intervention) variable

When the more interventions the better, the magnitudes are positive, demonstrating that the number of interventions returned when I is applied to the objective function is higher, as expected in this modality. In the two algorithms, on average, approximately 2.37 more interventions are presented to the learner when I is applied in the objective function.

The use of I (intervention: fine-grained LOs) in the calculation of c_j in the objective function improves the quality of LOs recommendation in relation to the expected amount of interventions. Thus, the results demonstrate that the RS will adapt better to the learners' needs, given the modality to be used at that moment, by returning more or less interventions.

Figure 20 shows a boxplot of the average number of interventions in the solution of the LORP. The red boxes represent the average number of interventions without using the I (intervention) variable; i.e., the cost c_j is calculated by Eq. (9). On the other hand, green boxes represent the average number of interventions using the I variable. In this case, the cost is calculated by Eq. (6).

From boxplot in Figure 20, we can see that in the less interventions the better mode, the algorithms tend to present no interventions. The algorithms implemented to solve the LORP using the I variable find solutions with more intervention-type LOs when the more interventions the better, and they find solutions with less interventions when the

less interventions the better, while not using the I variable decreases the quality of the solutions in relation to the expected amount of interventions.

Therefore, the variables P and I used in Eq. (6) contribute significantly to the recommendation of solutions with the best-rated LOs and with an appropriate number of interventions. Specifically, the P variable leverages CF to ensure that the recommended LOs align with the highest ratings given by students, enhancing the overall quality of the recommendations. On the other hand, the I variable integrates the concept of interventions from ITS, such as hints, feedback, and examples, to personalize the learning experience further. This dual approach ensures that the system not only suggests high-quality LOs but also tailors the recommendations to the student's specific learning needs and preferences. Moreover, when the recommended LOs do not cover all the concepts the student needs to learn, the system seeks additional educational resources on the Web to fulfill the student's requirements, thereby ensuring comprehensive coverage of the necessary knowledge.

5.1.5 Comparison between algorithms

The algorithms are compared based on the objective function in Eq. (2), and the c_j is calculated in terms of the variables I and P , according to Eq. 6.

Data under analysis from Table 15 do not have a normal distribution, so we used a median value and the *magnitude diff* to compare the algorithms. The bottom of Table 15 shows that the objective function values of the Exact, Greedy and Lorp algorithms are almost equal when the less interventions the better. However, the difference between the algorithms is statistically significant. This table also summarizes the results of the statistical analysis (*p-value*) and the magnitude of the statistically significant differences (*magnitude diff*). According to the *magnitude diff*, the Lorp algorithm has the best performance in both recommendation modes.

When the less interventions the better, the Greedy, JPSO, GA and Lorp algorithms find the exact solution for 8, 16, 5, and 10 out of 24 instances, respectively; and when the more interventions the better, they find the exact solution for 9, 17, 8, and 10 out of 24 instances. Thus, JPSO is the algorithm that finds the largest number of exact solutions and, on the other hand, finds bad solutions for some instances. In both recommendation modes, the second best algorithm for finding exact solutions is Lorp and the worst is GA.

In both recommendation modes (Figs. 21 and 22) the Greedy and Lorp algorithms have better average runtime than the Exact algorithm in almost all instances. The downside of the Exact algorithm is that it is very time consuming for larger instances. Therefore, the Exact algorithm would be the best option for solving small instances of the LORP, but if the shortest time is the priority, then the Greedy and Lorp algorithms are the best. In this case, GA and PSO are not good candidates for solving LORP as they are very time consuming.

Table 15 – Comparison between the objective functions of the selected algorithms

Instance	The less interventions the better					The more interventions the better				
	Objective function					Objective function				
	Exact	Greedy	JPSO	GA	Lorp	Exact	Greedy	JPSO	GA	Lorp
1	3.70	3.70	3.70	3.70	3.70	3.00	3.00	3.00	3.00	3.00
2	11.54	11.54	11.54	11.54	11.54	5.54	5.54	5.54	5.54	5.54
3	40.14	40.14	40.14	40.14	40.14	7.16	7.16	7.16	61.64	7.16
4	192.72	192.72	192.72	306.69	192.72	146.96	146.96	146.96	305.69	146.96
5	8.94	9.94	8.94	8.94	9.94	6.42	6.42	6.42	6.42	6.42
6	27.70	27.70	27.70	27.70	27.70	12.10	12.10	12.10	12.10	12.10
7	104.78	119.54	135.19	117.54	119.54	3.00	3.00	3.00	3.00	3.00
8	430.62	430.62	430.62	845.69	430.62	293.79	306.59	293.79	716.29	306.59
9	16.97	16.97	16.97	18.53	16.97	14.32	15.59	14.32	14.32	14.32
10	59.39	59.76	59.39	71.36	59.76	45.96	46.42	45.96	92.40	46.42
11	158.69	158.69	189.25	485.17	158.69	158.67	158.67	158.67	413.42	158.67
12	905.04	917.58	1108.31	2189.99	917.58	613.39	768.15	624.19	2685.58	624.19
13	48.22	53.25	48.22	50.76	53.25	55.21	64.50	55.21	57.72	62.47
14	192.55	222.87	192.55	254.56	206.71	191.46	215.22	191.46	236.13	214.45
15	625.47	717.39	625.55	1257.47	625.47	515.31	585.02	538.31	1224.17	529.39
16	2660.89	2660.89	3444.26	5727.27	3019.76	2698.76	3146.53	3656.06	5555.17	2910.57
17	82.16	98.88	82.16	85.62	87.51	64.65	68.07	64.65	64.65	68.07
18	226.99	252.46	226.99	330.86	230.77	186.03	217.58	186.03	223.23	200.26
19	916.26	1176.03	1047.46	1317.33	1068.93	835.85	923.48	931.93	1239.10	955.09
20	3590.66	4134.67	5959.25	6974.66	4155.28	2895.72	2895.72	3507.50	7084.56	2895.72
21	96.07	101.14	96.07	97.74	103.14	90.74	112.86	90.74	90.74	103.80
22	263.29	300.30	263.29	328.35	305.83	298.24	335.71	298.24	376.19	306.92
23	926.44	1021.48	977.89	1552.46	1054.84	775.24	825.40	776.32	1445.43	869.12
24	4145.79	4341.13	6234.57	8427.45	4145.79	4349.17	5224.31	6591.95	9483.64	4916.92
Median	175.62	175.70	190.90	280.63	175.70	152.81	152.81	152.81	229.68	152.81
<i>p</i> – value		< 0.001	0.009	< 0.001	< 0.001		< 0.001	0.022	< 0.001	< 0.001
Magnitude diff		55.5991	236.9883	616.5216	54.6315		76.1392	164.2840	713.8942	45.8527

Note: If *p*-value < 0.05, then there is a statistically significant difference between the variables.

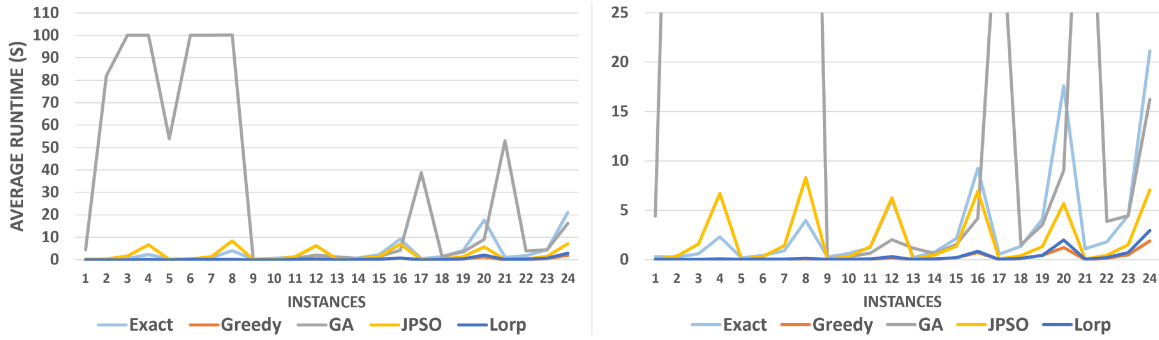


Figure 21 – Comparison of algorithm runtimes when the less interventions the better

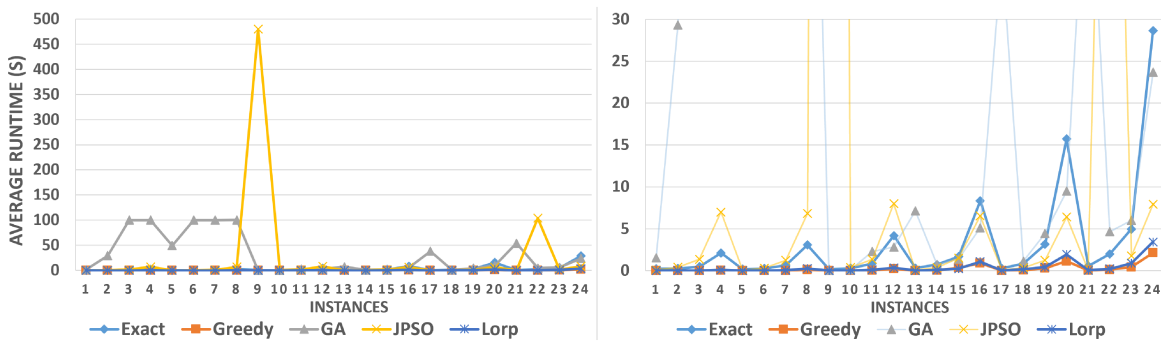


Figure 22 – Comparison of algorithm runtimes when the more interventions the better

These results demonstrate that modeling LORP as a coverage problem that aims to meet student needs (such as concepts to be learned and doubts to be resolved) and solving it using the Lorp algorithm (in addition to the exact alternative) is promising. Furthermore, the results found confirm the first hypothesis: “The use of collaborative filtering (variable P_j) and the incorporation of refined learning objects (interventions) in the calculation of the cost (c_j) of the SCP objective function significantly improve the quality and accuracy of LOs recommendations for students.”

5.2 Learner testing results with the recommender system

The proposed recommender system, integrated with the Anya chatbot, was evaluated in a Procedural Programming class of the Computer Science course at the Federal University of Uberlândia. Thirty-two students consented to participate in the study¹ and used the LO recommender system to learn new concepts and clarify doubts by receiving interventions.

Creating interventions is a costly task that requires human expertise to tailor the content of the interventions to the specific needs of students. To alleviate this difficulty,

¹ Research project approved by the Ethics Committee: Protocol number 57343822.0.0000.5152

interventions were created only for the content related to the first exam of the Procedural Programming course. However, students used the RS throughout the semester, as metadata was created for a set of learning objects that cover all concepts related to the C Programming Language.

The evaluation is divided into two subsections, each addressing different aspects of the students' interaction with the system. First, we explore the learning styles of the student class, providing insights into how their individual preferences align with the system's recommendations. Finally, we assess learner satisfaction through a detailed survey, measuring the system's usability and the students' overall experience. This comprehensive analysis aims to validate the effectiveness of the recommender system in a real educational setting, highlighting its strengths and identifying areas for improvement.

5.2.1 Learning styles of the student class

Twenty-five students answered ILS questions and the average number of questions answered is 16. Fig. 23 shows the distribution of the number of students in each dimension. The class of 25 students has a more visual, sensitive and reflective profile, in the Input, Perception and Processing dimensions, respectively. In the Understanding dimension, the class profile is both global and sequential. The middle column in each dimension represents the number of students with a style that encompasses both poles of the dimension.

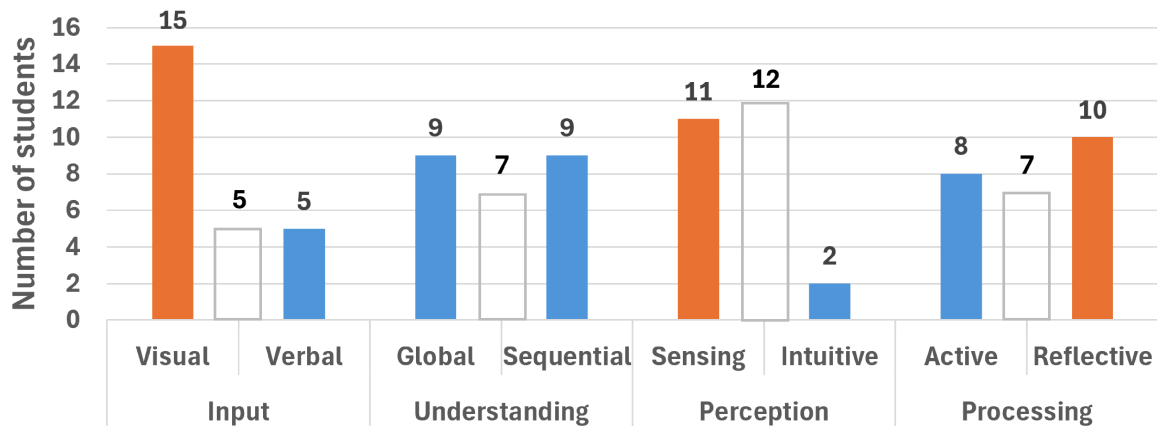


Figure 23 – Class learning style

Figure 24 shows the average score (magnitude) of the class's learning style in each dimension. In this case, the class of students also has a more visual, sensitive and reflective profile, in the Input, Perception and Processing dimensions, respectively. In the Understanding dimension, the class profile is slightly more sequential than global.

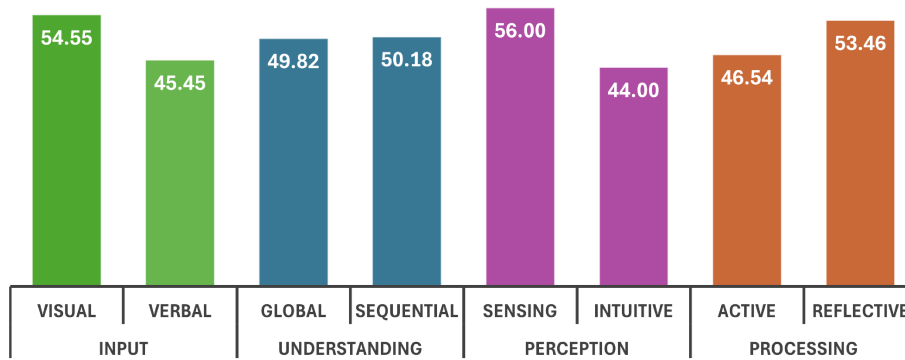


Figure 24 – Average scores for learning styles

5.2.2 Learner satisfaction measure

To validate the usability of the recommender system and the level of student satisfaction, the Satisfaction and Usability questionnaire was applied (see Appendix C). Figure 25 displays the results of this evaluation on the Likert scale (JOSHI *et al.*, 2015), which ranges from 1 (completely disagree) to 5 (completely agree) so that the intervals [1, 1.8], (1.8, 2.6], (2.6, 3.4], (3.4, 4.2] and (4.2, 5] correspond to “completely disagree”, “disagree”, “neutral”, “agree” and “completely agree”, respectively.

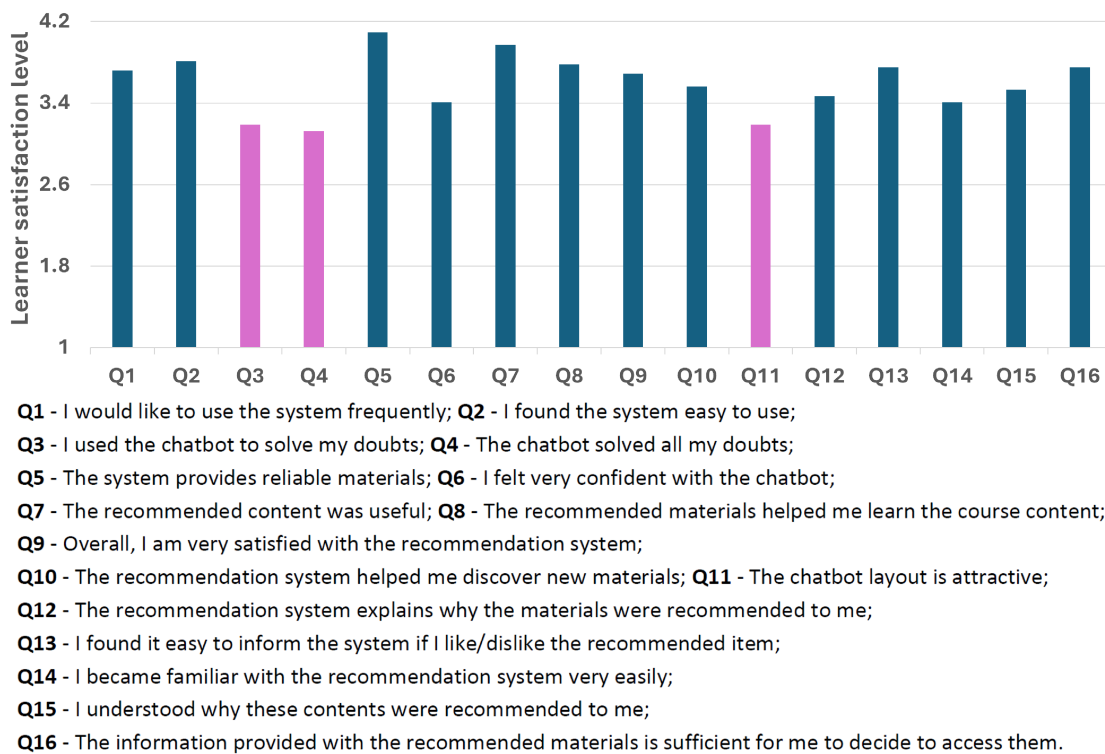


Figure 25 – Learner satisfaction with the usability of RS and chatbot

The height of the bars indicates the average satisfaction level of the class of 32 students concerning each of the 16 statements. The majority of the students expressed satisfaction

and approved of the system's usability, agreeing with 13 positive statements. However, they remained neutral regarding three statements that suggest certain limitations of the system in addressing all student doubts (Q3, Q4) and in presenting an attractive layout (Q11).

Of the six educational resources corresponding to the aggregated interventions, four were evaluated by the students, of which two received opinions. The average evaluation score was 4.2 and the two student opinions were: "I found it extremely intuitive and organized. The explanation was very clear and direct. I loved it!!" and "I found the material very succinct, easy to understand, good for beginners". Thus, the students evaluated the recommended interventions well, suggesting that the materials are reliable. On the other hand, the number of interventions was insufficient, as they were created only for the content of the students' first test of the semester. Consequently, they did not address all the students' doubts, which is why the students were neutral in statements Q3 and Q4. In addition to the interventions, the LOs recommended for learning new concepts were also well received by the students, supporting the second hypothesis that "Recommended LOs meet the students' knowledge and learning style and are useful for assisting them in learning and problem solving."

Final Remarks

In this work, we proposed an approach that uses collaborative filtering and the reuse of web content for the recommendation of LOs based on concepts. Our approach reduces the cold-start and rating sparsity problems through the ontology-based recommendation, which uses an ontology to model students' knowledge and learning resources. We formulated the LORP as a covering problem so that the recommendation of LOs took into account the concepts that the student needs to learn. In addition, we implemented the hint type in an ontology for a more fine-grained recommendation of LOs, which is combined with the reuse of web content to overcome the lack of refined concepts in RSs and the low content diversity of ITSs, respectively.

A set of 24 benchmark instances created to simulate a real scenario was used to execute the experiments. When the RS considers the variables P (collaborative filtering) and I (intervention LOs) in calculating the cost of the solutions, the quality of the solutions improves in terms of the average rating and the number of expected interventions according to experimental results. The Exact, Greedy and Lorp algorithms are good strategies to find these solutions for instances that simulate the educational context. If the best runtime is the priority, then the best algorithm is the Greedy, but it finds the best solution in only 37.5% of instances, while JPSO and Lorp are exact in 66.7% and 41.7% of the instances, respectively. Thus, the Lorp algorithm presents better solutions than the greedy one, increasing the execution time slightly, while GA and JPSO can consume a lot of time for some instances in the educational context.

Moreover, the implementation of the Anya chatbot within the Moodle environment demonstrates the practical applicability and integration of our RS in real-world educational settings. Incorporating gamification elements in Anya increased student engagement and provided an innovative way to track and enhance learning progress. The chatbot's ability to interact naturally with students and provide timely recommendations based on their individual needs and learning styles highlights the potential for combining advanced algorithms with intuitive interfaces. These results emphasize the importance of integrating pedagogical methods with modern technology to enhance learning experiences.

6.1 Main Contributions

In this section, we highlight the main contributions of our research, focusing on the development and implementation of a hybrid recommender system that combines CF and ontology-based techniques to personalize learning object recommendations according to students' needs. When the LOs do not cover all the concepts that the student needs to learn, additional educational resources are sought on the Web to satisfy the student's needs. The key contributions include:

- ❑ **Ontology-Based Knowledge Modeling:** An ontology was implemented to store metadata of LOs associated with the knowledge domain, avoiding the costly task of creating a complete knowledge domain. This ontology supports the RS in delivering context-specific and personalized learning interventions.
- ❑ **Formulation of LORP as a Coverage Problem:** We formalized the Learning Object Recommendation Problem (LORP) as a set covering problem (SCP), integrating CF and fine-grained LOs to address the specific concepts students need to learn.
- ❑ **Development of a Fine-Grained Recommender System:** Our RS provides personalized recommendations of fine-grained LOs, such as hints, from diverse knowledge areas. This approach enhances the granularity and relevance of recommendations, improving upon existing systems that do not combine fine-grained concepts with multi-disciplinary content.
- ❑ **Integration of an Intelligent Chatbot with Gamification:** The creation of the Anya chatbot, which utilizes gamification techniques to engage and motivate students, represents a significant advancement. This intelligent interface tutors students, facilitates more engaging dialogues, and supports the teaching-learning process by reducing pedagogical difficulties.
- ❑ **Validation and Usability:** The proposed RS was validated within the Moodle environment with first-year Computer Science students. The results indicated high levels of student satisfaction and usability, with positive feedback on the recommendations provided by the system.

6.2 Limitations and Future Work

Building on the foundations laid by this research, several avenues for future work are proposed. Future research should focus on expanding the ontology to cover a broader range of subjects and concepts. This expansion will enable the RS to provide even more

comprehensive and diversified learning support. Incorporating more sophisticated gamification elements, such as adaptive difficulty levels, personalized rewards, and interactive storytelling, can further enhance student engagement and motivation. Improving the NLP capabilities of the chatbot to better understand and respond to complex student queries will make the interaction more natural and effective. In addition, conducting longitudinal studies to assess the long-term impact of the RS on student learning outcomes, retention rates, and academic performance will provide deeper insights into its effectiveness and areas for improvement.

One significant limitation of this research is the scalability of the ontology and the recommender system. While the current model performs well within the scope of the study, expanding the ontology to accommodate a wider array of subjects and concepts may pose challenges in terms of data management and system performance. Future work should explore scalable solutions and optimization techniques to ensure that the system can handle increased complexity without compromising speed or accuracy.

Additionally, in future work, probabilistic inference can be explored to determine students' learning styles. Techniques such as Bayesian networks and Gaussian mixture models can be utilized to estimate the likelihood of various learning preferences based on observable behaviors and interactions. This approach has the potential to significantly enhance the personalization of educational content by dynamically adapting to the inferred learning styles, thereby improving student engagement and learning outcomes.

Furthermore, other personality traits of students can also be explored. This research primarily focused on certain learning styles, but incorporating a broader range of personality traits, such as those defined by the Big Five personality traits (openness, conscientiousness, extraversion, agreeableness, and neuroticism) and self-regulated learning behaviors, could provide a more holistic understanding of student needs. By integrating these factors, the recommender system could offer more personalized and effective learning interventions.

Moreover, the current study did not extensively address the diversity of learning environments and contexts. Future research should investigate how the RS performs across different educational settings, including online, hybrid, and traditional classroom environments. This would help in understanding the adaptability and effectiveness of the system in varied contexts, thereby broadening its applicability and impact.

Additionally, ethical considerations and data privacy concerns were beyond the primary scope of this research but are crucial for the broader implementation of RS in educational settings. Future studies should rigorously examine the ethical implications of using student data for personalized learning and ensure robust privacy protection measures are in place. This will not only enhance trust among users but also align with regulatory standards and best practices in data management.

Lastly, the integration of more advanced analytics and reporting tools could signif-

icantly benefit educators and administrators by providing deeper insights into student progress and learning patterns. Future developments should focus on creating user-friendly dashboards and analytical tools that allow stakeholders to make data-driven decisions to further improve educational outcomes.

6.3 Contributions to Bibliographic Production

During the doctoral studies, we have made some contributions to the field through important publications. These publications collectively demonstrate the significant strides made in the field of personalized education through the innovative application of bio-inspired algorithms, Semantic Web technologies, and ontology-based systems. Below are summaries of four significant articles that showcase the breadth and depth of this research:

- **Avanços na Recomendação Personalizada de Objetos de Aprendizagem Através da Utilização de Meta-heurísticas Clássicas Associadas aos Problemas de Cobertura de Conjuntos e de Máxima Cobertura: Uma Análise Experimental** (BELIZÁRIO JÚNIOR *et al.*, 2020): A limitation of recommender systems is their inability to ensure that the recommended LOs cover all the concepts a student needs to learn. To address this challenge, this paper formulates the Learning Object Recommendation Problem (LORP) as a Set Covering Problem and a Maximum Covering Problem, both of which are NP-Hard class problems. Greedy and evolutionary metaheuristics were appropriately adapted, resulting in a promising approach to solving the LORP. This approach provides more personalized content for students by using an ontology that models their knowledge, needs, learning styles, and search parameters.

- **Personalized Recommendation of Learning Objects Through Bio-inspired Algorithms and Semantic Web Technologies: an Experimental Analysis** (PEREIRA JÚNIOR *et al.*, 2020): The growing need to utilize the Web as a learning resource while providing personalized recommendations presents a significant challenge. In this context, this paper introduces an approach that integrates Semantic Web technologies with bio-inspired algorithms to perform personalized recommendations of LOs using local repositories and Web resources. Web resources are retrieved and structured as LOs, enabling the automatic generation of metadata and reducing the workload for course tutors. Experiments were conducted to determine the most suitable bio-inspired evolutionary algorithm for this task. Additionally, the quality of recommendations, considering both local repositories and the Web, was examined. Initial experiments evaluating the efficiency of the proposed approach have demonstrated promising results.

- **Solving the Individualized Instructional Content Delivery Problem Using Ontology and Metaheuristics on the Set Covering Problem: An Experimental Analysis** (BELIZÁRIO JÚNIOR *et al.*, 2023): Intelligent Tutoring Systems that utilize a step-by-step problem-solving approach are limited in terms of compatible content. Conversely, recommender systems can suggest various content types but lack the detailed granularity of concepts inherent in step-by-step approaches. This paper addresses this challenge by proposing a method to recommend instructional content from diverse knowledge domains while integrating the refined concepts of ITSs. The instructional content delivery problem is formulated as a set covering problem, classified as NP-hard. We demonstrate that a PSO-based algorithm is a suitable candidate for solving LORP, offering better runtime performance than the exact algorithm and superior solutions compared to the greedy heuristic. By leveraging CF and an ontology that models students' knowledge, learning styles, and search parameters, this approach provides more individualized content.

- **Advances in personalised recommendation of learning objects based on the set covering problem using ontology** (BELIZÁRIO JÚNIOR *et al.*, 2024): Loop-based ITSs support the learning process through a step-by-step problem-solving approach. However, a limitation of ITSs is that only a limited amount of content is compatible with this method. On the other hand, recommender systems can suggest various types of content but often ignore the fine-grained concepts essential to the step-by-step approach. This work addresses this challenge by proposing a method to recommend learning objects from various knowledge domains while incorporating the detailed concepts of ITSs. We formulate the learning object recommendation problem as a set covering problem, which is classified as NP-hard. An exact algorithm and a greedy heuristic were adapted to this context, resulting in a promising approach to solve these problems, as demonstrated by the results. This approach provides more personalized content for students through the use of CF and an ontology that models their knowledge, learning styles, and search parameters.

6.4 Acknowledgments

I would like to express my sincere gratitude to Coordination for the Improvement of Higher Education Personnel—Brazil (CAPES)—Finance Code 001 and the National Council for Scientific and Technological Development (CNPq; Grant 402431/2021-9) for their financial support throughout my research. Their funding was crucial in enabling the progress and completion of this doctoral thesis.

Bibliography

ABECH, M. *et al.* A model for learning objects adaptation in light of mobile and context-aware computing. **Personal and Ubiquitous Computing**, Springer-Verlag, v. 20, n. 2, p. 167–184, 2016. Available at: <<https://doi.org/10.1007/s00779-016-0902-3>>.

ACAMPORA, G.; GAETA, M.; LOIA, V. Exploring e-learning knowledge through ontological memetic agents. **IEEE Computational Intelligence Magazine**, IEEE, v. 5, n. 2, p. 66–77, 2010. Available at: <<https://doi.org/10.1109/MCI.2010.936306>>.

ACAMPORA, G. *et al.* Optimizing learning path selection through memetic algorithms. In: INTERNATIONAL JOINT CONFERENCE ON NEURAL NETWORKS, 2008, Hong Kong, China. **Proceedings...** Washington, USA: IEEE Computer Society, 2008. p. 3869–3875. Available at: <<https://doi.org/10.1109/IJCNN.2008.4634354>>.

_____. An adaptive multi-agent memetic system for personalizing e-learning experiences. In: INTERNATIONAL CONFERENCE ON FUZZY SYSTEMS, 2011, Taipei, Taiwan. **Proceedings...** Washington, USA: IEEE Computer Society, 2011. p. 123–130. Available at: <<https://doi.org/10.1109/FUZZY.2011.6007519>>.

ADOMAVICIUS, G.; TUZHILIN, A. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. **IEEE transactions on knowledge and data engineering**, IEEE, v. 17, n. 6, p. 734–749, 2005. Available at: <<https://doi.org/10.1109/TKDE.2005.99>>.

ADVANCED DISTRIBUTED LEARNING INITIATIVE. **Sharable Content Object Reference Model (SCORM)**. Alexandria, VA, USA, 2001. Available at: <<http://www.adlnet.gov>>.

AYEDOUN, E.; HAYASHI, Y.; SETA, K. Adding communicative and affective strategies to an embodied conversational agent to enhance second language learners' willingness to communicate. **International Journal of Artificial Intelligence in Education**, Springer, v. 29, n. 1, p. 29–57, 2019. Available at: <<https://doi.org/10.1007/s40593-018-0171-6>>.

BAJENARU, L.; BOROZAN, A.-M.; SMEUREANU, I. Using ontologies for the e-learning system in healthcare human resources management. **Informatica Economica**, INFOREC Association, v. 19, n. 2, p. 15–24, 2015. Available at: <<https://doi.org/10.12948/issn14531305/19.2.2015.02>>.

- BAKER, R. S. *et al.* Better to be frustrated than bored: The incidence, persistence, and impact of learners' cognitive–affective states during interactions with three different computer-based learning environments. **International Journal of Human-Computer Studies**, Elsevier, v. 68, n. 4, p. 223–241, 2010. Available at: <<https://doi.org/10.1016/j.ijhcs.2009.12.003>>.
- BALAJI, S.; REVATHI, N. A new approach for solving set covering problem using jumping particle swarm optimization method. **Natural Computing**, Springer, v. 15, n. 3, p. 503–517, 2016. Available at: <<https://doi.org/10.1007/s11047-015-9509-2>>.
- BALAS, E.; CARRERA, M. C. A dynamic subgradient-based branch-and-bound procedure for set covering. **Operations Research**, INFORMS, v. 44, n. 6, p. 875–890, 1996. Available at: <<https://doi.org/10.1287/opre.44.6.875>>.
- BARRAGÁNS-MARTÍNEZ, A. B. *et al.* A hybrid content-based and item-based collaborative filtering approach to recommend TV programs enhanced with singular value decomposition. **Information Sciences**, Elsevier, v. 180, n. 22, p. 4290–4311, 2010. Available at: <<https://doi.org/10.1016/j.ins.2010.07.024>>.
- BEASLEY, J. E.; CHU, P. C. A genetic algorithm for the set covering problem. **European journal of operational research**, Elsevier, v. 94, n. 2, p. 392–404, 1996. Available at: <[https://doi.org/10.1016/0377-2217\(95\)00159-X](https://doi.org/10.1016/0377-2217(95)00159-X)>.
- BELIZÁRIO JÚNIOR, C. F. **Reúso de conteúdo da Web na recomendação personalizada de objetos de aprendizagem: uma abordagem baseada em um algoritmo genético, tecnologias da Web Semântica e uma ontologia**. Dissertation (Master of Science), 2018. Available at: <<https://doi.org/10.14393/ufu.di.2018.988>>.
- BELIZÁRIO JÚNIOR, C. F.; DORÇA, F. Uma abordagem para a criação e recomendação de objetos de aprendizagem usando um algoritmo genético, tecnologias da web semântica e uma ontologia. In: **Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)**. Fortaleza, CE: [s.n.], 2018. v. 29, p. 1533.
- BELIZÁRIO JÚNIOR, C. F. *et al.* Avanços na recomendação personalizada de objetos de aprendizagem através da utilização de meta-heurísticas clássicas associadas aos Problemas de Cobertura de Conjuntos e de Máxima Cobertura: Uma análise experimental. In: SBC. **Anais do XXXI Simpósio Brasileiro de Informática na Educação**. 2020. p. 1383–1392. Available at: <<https://doi.org/10.5753/cbie.sbie.2020.1383>>.
- _____. Advances in personalised recommendation of learning objects based on the set covering problem using ontology. **International Journal of Learning Technology**, Inderscience Publishers (IEL), v. 19, n. 1, p. 25–57, 2024. Available at: <<https://doi.org/10.1504/IJLT.2024.10063339>>.
- _____. Solving the individualized instructional content delivery problem using ontology and metaheuristics on the set covering problem: An experimental analysis. In: SBC. **Anais do XXXIV Simpósio Brasileiro de Informática na Educação**. 2023. p. 1202–1214. Available at: <<https://doi.org/10.5753/sbie.2023.234482>>.
- BERNERS-LEE, T.; HENDLER, J.; LASSILA, O. The semantic web. **Scientific american**, JSTOR, v. 284, n. 5, p. 34–43, 2001. Available at: <<https://doi.org/10.1038/scientificamerican0501-34>>.

- BILAL, N.; GALINIER, P.; GUIBAULT, F. A new formulation of the set covering problem for metaheuristic approaches. **ISRN Operations Research**, Hindawi Publishing Corporation, v. 2013, 2013. Available at: <<https://doi.org/10.1155/2013/203032>>.
- BIRJALI, M.; BENI-HSSANE, A.; ERRITALI, M. A novel adaptive e-learning model based on Big Data by using competence-based knowledge and social learner activities. **Applied Soft Computing**, Elsevier, v. 69, p. 14–32, 2018. Available at: <<https://doi.org/10.1016/j.asoc.2018.04.030>>.
- BOOCH, G.; RUMBAUGH, J.; JACOBSEN, I. Uml: unified modeling language. **Versão**, 1997.
- BURKE, R. Hybrid web recommender systems. **The adaptive web**, Springer, p. 377–408, 2007. Available at: <https://doi.org/10.1007/978-3-540-72079-9_12>.
- CAPRARA, A.; FISCHETTI, M.; TOTH, P. A heuristic method for the set covering problem. **Operations research**, INFORMS, v. 47, n. 5, p. 730–743, 1999. Available at: <<https://doi.org/10.1287/opre.47.5.730>>.
- CHOI, K. *et al.* A hybrid online-product recommendation system: Combining implicit rating-based collaborative filtering and sequential pattern analysis. **electronic commerce research and applications**, Elsevier, v. 11, n. 4, p. 309–317, 2012. Available at: <<https://doi.org/10.1016/j.elerap.2012.02.004>>.
- 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. Available at: <<https://doi.org/10.1016/j.tele.2017.02.004>>.
- CHVATAL, V. A greedy heuristic for the set-covering problem. **Mathematics of operations research**, INFORMS, v. 4, n. 3, p. 233–235, 1979. Available at: <<https://doi.org/10.1287/moor.4.3.233>>.
- COLACE, F. *et al.* Chatbot for e-learning: A case of study. **International Journal of Mechanical Engineering and Robotics Research**, v. 7, n. 5, p. 528–533, 2018. Available at: <<https://doi.org/10.18178/ijmerr.7.5.528-533>>.
- DEBORAH, L. J.; BASKARAN, R.; KANNAN, A. Learning styles assessment and theoretical origin in an E-learning scenario: a survey. **Artificial Intelligence Review**, Springer, v. 42, n. 4, p. 801–819, 2014. Available at: <<https://doi.org/10.1007/s10462-012-9344-0>>.
- FALCI, S. H. *et al.* A Low Complexity Heuristic To Solve a Learning Objects Recommendation Problem. In: **IEEE. 2019 IEEE 19th ICALT**. 2019. v. 19, p. 49–53. Available at: <<https://doi.org/10.1109/ICALT.2019.00022>>.
- FELDER, R. M.; SILVERMAN, L. K. Learning and teaching styles in engineering education. **Engineering education**, v. 78, n. 7, p. 674–681, 1988.

- FERREIRA, H. N. *et al.* Integração de técnicas de inteligência artificial para modelagem probabilística do estudante em ambientes virtuais de aprendizagem. **Tecnologias, Sociedade e Conhecimento**, v. 10, n. 2, p. 38–67, 2023. Available at: <<https://doi.org/10.20396/tsc.v10i2.18365>>.
- FISHER, M. L.; KEDIA, P. Optimal solution of set covering/partitioning problems using dual heuristics. **Management science**, INFORMS, v. 36, n. 6, p. 674–688, 1990. Available at: <<https://doi.org/10.1287/mnsc.36.6.674>>.
- GAETA, M. *et al.* An Approach To Personalized e-Learning. **Journal of Education, Informatics & Cybernetics**, v. 11, n. 1, 2013.
- GAREY, M. R.; JOHNSON, D. S. **Computers and intractability: A guide to the theory of np-completeness**. New York: W. H. Freeman & Co., 1979. 340 p. (Series of books in the mathematical sciences).
- GOLAB, L. *et al.* Size-constrained weighted set cover. In: IEEE. **2015 IEEE 31st Intern. Conf. on Data Engineering**. 2015. p. 879–890. Available at: <<https://doi.org/10.1109/ICDE.2015.7113341>>.
- GRAESSER, A. C. Conversations with autotutor help students learn. **International Journal of Artificial Intelligence in Education**, Springer, v. 26, n. 1, p. 124–132, 2016. Available at: <<https://doi.org/10.1007/s40593-015-0086-4>>.
- GRAESSER, A. C. *et al.* Intelligent tutoring systems with conversational dialogue. **AI magazine**, v. 22, n. 4, p. 39–39, 2001.
- GRAF, S.; KINSHUK; IVES, C. A flexible mechanism for providing adaptivity based on learning styles in learning management systems. In: IEEE. **2010 10th IEEE International Conference on Advanced Learning Technologies**. 2010. p. 30–34. Available at: <<https://doi.org/10.1109/ICALT.2010.16>>.
- GRUBER, T. R. A translation approach to portable ontology specifications. **Knowledge acquisition**, Elsevier, v. 5, n. 2, p. 199–220, 1993. Available at: <<https://doi.org/10.1006/knac.1993.1008>>.
- HARTLEY, J.; SLEEMAN, D. Towards more intelligent teaching systems. **International Journal of Man-Machine Studies**, v. 5, n. 2, p. 215–236, 1973. ISSN 0020-7373. Available at: <<https://www.sciencedirect.com/science/article/pii/S0020737373800331>>.
- HORROCKS, I. *et al.* SWRL: A semantic web rule language combining OWL and RuleML. **W3C Member submission**, v. 21, n. 79, p. 1–31, 2004.
- HOUSOS, E.; ELMROTH, T. Automatic optimization of subproblems in scheduling airline crews. **Interfaces**, INFORMS, v. 27, n. 5, p. 68–77, 1997. Available at: <<https://doi.org/10.1287/inte.27.5.68>>.
- JEEVAMOL, J.; RENUMOL, V. An ontology-based hybrid e-learning content recommender system for alleviating the cold-start problem. **Education and Information Technologies**, Springer, v. 26, n. 4, p. 4993–5022, 2021. Available at: <<https://doi.org/10.1007/s10639-021-10508-0>>.
- JOSHI, A. *et al.* Likert scale: Explored and explained. **British journal of applied science & technology**, v. 7, n. 4, p. 396–403, 2015.

KATZ, S. *et al.* Linking dialogue with student modelling to create an adaptive tutoring system for conceptual physics. **International Journal of Artificial Intelligence in Education**, Springer, v. 31, n. 3, p. 397–445, 2021. Available at: <<https://doi.org/10.1007/s40593-020-00226-y>>.

KENNEDY, J.; EBERHART, R. Particle swarm optimization. In: IEEE. **Proceedings of ICNN'95-International Conference on Neural Networks**. 1995. v. 4, p. 1942–1948. Available at: <<https://doi.org/10.1109/ICNN.1995.488968>>.

KIRYAKOVA, G.; ANGELOVA, N.; YORDANOVA, L. Gamification in education. In: **Proceedings of 9th international Balkan education and science conference**. [S.l.: s.n.], 2014. v. 1, p. 679–684.

KLAŠNJA-MILIĆEVIĆ, A.; VESIN, B.; IVANOVIĆ, M. Social tagging strategy for enhancing e-learning experience. **Computers & Education**, Elsevier, v. 118, p. 166–181, 2018. Available at: <<https://doi.org/10.1016/j.compedu.2017.12.002>>.

LAN, G.; DEPUY, G. W.; WHITEHOUSE, G. E. An effective and simple heuristic for the set covering problem. **European journal of operational research**, Elsevier, v. 176, n. 3, p. 1387–1403, 2007. Available at: <<https://doi.org/10.1016/j.ejor.2005.09.028>>.

LIMONGELLI, C.; GASPARETTI, F.; SCIARRONE, F. Wiki course builder: a system for retrieving and sequencing didactic materials from wikipedia. In: IEEE. **2015 International Conference on Information Technology Based Higher Education and Training (ITHET)**. 2015. p. 1–6. Available at: <<https://doi.org/10.1109/ITHET.2015.7218041>>.

LTSC. Standard for Learning Object Metadata (IEEE 1484.12.1). **Learning Technology Standards Committee**, Nova York, 2002. Available at: <https://webpages.uidaho.edu/fritz/edui/LOM_1484_12_1_v1_Final_Draft.pdf>.

MEDIO, C. D. *et al.* MoodleREC: A recommendation system for creating courses using the moodle e-learning platform. **Computers in Human Behavior**, Elsevier, v. 104, p. 1–14, 2020. Available at: <<https://doi.org/10.1016/j.chb.2019.106168>>.

MENDES, T. *et al.* Uso de sistemas de gamificação no combate a evasão de cursos de graduação da área de exatas. In: **Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)**. [S.l.: s.n.], 2019. v. 30, p. 733.

MITCHELL, S.; OSULLIVAN, M.; DUNNING, I. PuLP: a linear programming toolkit for python. **The University of Auckland, Auckland, New Zealand**, v. 65, 2011.

MONTGOMERY, D. **Design and Analysis of Experiments**. New Jersey, USA: John Wiley & Sons, 2012. 730 p.

MOREIRA, S. *et al.* Uma experiência de gamificação no ensino com o ambiente classcraft: análise da motivação dos estudantes. In: SBC. **Simpósio Brasileiro de Informática na Educação (SBIE)**. 2022. p. 403–414. Available at: <<https://doi.org/10.5753/sbie.2022.224735>>.

MORENO, A. *et al.* Sigtur/e-destination: ontology-based personalized recommendation of tourism and leisure activities. **Engineering applications of artificial intelligence**, Elsevier, v. 26, n. 1, p. 633–651, 2013. Available at: <<https://doi.org/10.1016/j.engappai.2012.02.014>>.

MURRAY, T. Authoring intelligent tutoring systems: An analysis of the state of the art. **International Journal of Artificial Intelligence in Education (IJAIED)**, v. 10, p. 98–129, 1999.

MUSLIU, N. Local search algorithm for unicost set covering problem. In: SPRINGER. **International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems**. 2006. p. 302–311. Available at: <https://doi.org/10.1007/11779568_34>.

NGUYEN, H. D. *et al.* Intelligent tutoring chatbot for solving mathematical problems in high-school. In: IEEE. **2019 11th International Conference on Knowledge and Systems Engineering (KSE)**. 2019. p. 1–6. Available at: <<https://doi.org/10.1109/KSE.2019.8919396>>.

NOY, N. F.; MCGUINNESS, D. L. **Ontology Development 101: A guide to creating your first ontology**. Stanford, USA, 2001. (Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880).

NWANA, H. S. Intelligent tutoring systems: an overview. **Artificial Intelligence Review**, Springer, v. 4, n. 4, p. 251–277, 1990. Available at: <<https://doi.org/10.1007/BF00168958>>.

OUF, S. *et al.* A proposed paradigm for smart learning environment based on semantic web. **Computers in Human Behavior**, Elsevier, v. 72, p. 796–818, 2017. Available at: <<https://doi.org/10.1016/j.chb.2016.08.030>>.

PASHLER, H. *et al.* Learning styles: Concepts and evidence. **Psychological science in the public interest**, SAGE Publications Sage CA: Los Angeles, CA, v. 9, n. 3, p. 105–119, 2008. Available at: <<https://doi.org/10.1111/j.1539-6053.2009.01038.x>>.

PATEL, F. *et al.* Combating depression in students using an intelligent chatbot: a cognitive behavioral therapy. In: IEEE. **2019 IEEE 16th India Council International Conference (INDICON)**. 2019. p. 1–4. Available at: <<https://doi.org/10.1109/INDICON47234.2019.9030346>>.

PEREIRA, C. K. *et al.* BROAD-RSI—educational recommender system using social networks interactions and linked data. **Journal of Internet Services and Applications**, SpringerOpen, v. 9, n. 7, p. 1–28, 2018. Available at: <<https://doi.org/10.1186/s13174-018-0076-5>>.

PEREIRA JÚNIOR, C. *et al.* Personalized recommendation of learning objects through bio-inspired algorithms and semantic web technologies: an experimental analysis. In: SBC. **Anais do XXXI Simpósio Brasileiro de Informática na Educação**. [S.l.], 2020. p. 1333–1342.

RAMIREZ-ARELLANO, A.; BORY-REYES, J.; HERNÁNDEZ-SIMÓN, L. M. Learning Object Retrieval and Aggregation Based on Learning Styles. **Journal of Educational Computing Research**, SAGE Publications Sage CA: Los Angeles, CA, v. 55, n. 6, p. 757–788, 2017. Available at: <<https://doi.org/10.1177/0735633116681303>>.

RECTOR, A. Representing Specified Values in OWL: “value partitions” and “value sets”. **Technical Report Note 17, W3C, Semantic Web Best Practices and Deployment Working Group (May 2005)**, 2005. Available at: <<http://www.w3.org/TR/swbp-specified-values/>>.

REN, Z. *et al.* A fast and efficient ant colony optimization approach for the set covering problem. In: IEEE. **2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)**. [S.l.], 2008. p. 1839–1844.

REN, Z.-G. *et al.* New ideas for applying ant colony optimization to the set covering problem. **Computers & Industrial Engineering**, Elsevier, v. 58, n. 4, p. 774–784, 2010. Available at: <<https://doi.org/10.1016/j.cie.2010.02.011>>.

RUAN, S. *et al.* Quizbot: A dialogue-based adaptive learning system for factual knowledge. In: **Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems**. [s.n.], 2019. p. 1–13. Available at: <<https://doi.org/10.1145/3290605.3300587>>.

RUOTSALO, T. *et al.* SMARTMUSEUM: A mobile recommender system for the Web of Data. **Journal of Web Semantics**, Elsevier, v. 20, p. 50–67, 2013. Available at: <<https://doi.org/10.1016/j.websem.2013.03.001>>.

RYAN, R. M.; DECI, E. L. Self-determination theory and the facilitation of intrinsic motivation, social development, and well-being. **American psychologist**, American Psychological Association, v. 55, n. 1, p. 68, 2000. Available at: <<https://doi.org/10.1037//0003-066X.55.1.68>>.

SHISHEHCHI, S. *et al.* Ontological approach in knowledge based recommender system to develop the quality of e-learning system. **Australian Journal of Basic and Applied Sciences**, v. 6, n. 2, p. 115–123, 2012.

SMUTNY, P.; SCHREIBEROVA, P. Chatbots for learning: A review of educational chatbots for the facebook messenger. **Computers & Education**, Elsevier, v. 151, p. 1–11, 2020. Available at: <<https://doi.org/10.1016/j.compedu.2020.103862>>.

SOLAR, M.; PARADA, V.; URRUTIA, R. A parallel genetic algorithm to solve the set-covering problem. **Computers & Operations Research**, Elsevier, v. 29, n. 9, p. 1221–1235, 2002. Available at: <[https://doi.org/10.1016/S0305-0548\(01\)00026-0](https://doi.org/10.1016/S0305-0548(01)00026-0)>.

SOLOMAN, B. A.; FELDER, R. M. Index of learning styles questionnaire. **NC State University**. Available online at: <https://learningstyles.webtools.ncsu.edu/> (last visited on 12.06.2024), v. 70, 2005.

TARUS, J. K.; NIU, Z.; KALUI, D. A hybrid recommender system for e-learning based on context awareness and sequential pattern mining. **Soft Computing**, Springer, v. 22, n. 8, p. 2449–2461, 2018. Available at: <<https://doi.org/10.1007/s00500-017-2720-6>>.

- TARUS, J. K.; NIU, Z.; KHADIDJA, B. E-learning recommender system based on collaborative filtering and ontology. **International Journal of Computer and Information Engineering**, World Academy of Science, Engineering and Technology, v. 11, n. 2, p. 256–261, 2017.
- TARUS, J. K.; NIU, Z.; YOUSIF, A. A hybrid knowledge-based recommender system for e-learning based on ontology and sequential pattern mining. **Future Generation Computer Systems**, Elsevier, v. 72, p. 37–48, 2017. Available at: <<https://doi.org/10.1016/j.future.2017.02.049>>.
- TUKEY, J. W. Comparing Individual Means in the Analysis of Variance. **Biometrics**, [Wiley, International Biometric Society], v. 5, n. 2, p. 99–114, 1949. Available at: <<https://doi.org/10.2307/3001913>>.
- VALERIANO, E.; CORRÊA, A.; POZZEBON, E. O sistema tutor inteligente mazk no ensino fundamental i. In: **Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)**. [S.l.: s.n.], 2019. v. 30, n. 1, p. 616.
- VANETTI, M. *et al.* Content-based filtering in on-line social networks. In: SPRINGER. **International Workshop on Privacy and Security Issues in Data Mining and Machine Learning**. 2010. p. 127–140. Available at: <http://dx.doi.org/10.1007/978-3-642-19896-0_11>.
- VANLEHN, K. The behavior of tutoring systems. **International journal of artificial intelligence in education**, IOS Press, v. 16, n. 3, p. 227–265, 2006.
- VANLEHN, K. *et al.* The andes physics tutoring system: Lessons learned. **International Journal of Artificial Intelligence in Education**, IOS Press, v. 15, n. 3, p. 147–204, 2005.
- VASKO, F. J.; WILSON, G. R. Using a facility location algorithm to solve large set covering problems. **Operations Research Letters**, Elsevier, v. 3, n. 2, p. 85–90, 1984. Available at: <[https://doi.org/10.1016/0167-6377\(84\)90047-6](https://doi.org/10.1016/0167-6377(84)90047-6)>.
- VASKO, F. J.; WOLF, F. E.; STOTT, K. L. Optimal selection of ingot sizes via set covering. **Operations Research**, INFORMS, v. 35, n. 3, p. 346–353, 1987. Available at: <<https://doi.org/10.1287/opre.35.3.346>>.
- VICCARI, R. *et al.* The obaa proposal for learning objects supported by agents. In: MASEIE WORKSHOP–AUTONOMOUS AGENTS AND MULTI-AGENT SYSTEMS (AAMAS), 9., 2010, Toronto, Canadá. **Proceedings...** Toronto, Canadá: IFAAMAS, 2010.
- WAN, S.; NIU, Z. A hybrid e-learning recommendation approach based on learners' influence propagation. **IEEE Transactions on Knowledge and Data Engineering**, IEEE, v. 32, n. 5, p. 827–840, 2020. Available at: <<https://doi.org/10.1109/TKDE.2019.2895033>>.
- WANG, R.-L.; OKAZAKI, K. An improved genetic algorithm with conditional genetic operators and its application to set-covering problem. **Soft computing**, Springer, v. 11, n. 7, p. 687–694, 2007. Available at: <<https://doi.org/10.1007/s00500-006-0131-1>>.

- WARNCKE-WANG, M.; COSLEY, D.; RIEDL, J. Tell me more: An actionable quality model for wikipedia. In: INTERNATIONAL SYMPOSIUM ON OPEN COLLABORATION, 9., 2013, Hong Kong, China. **Proceedings...** New York: ACM, 2013. Available at: <<https://doi.org/10.1145/2491055.2491063>>.
- WENGER, E. **Artificial intelligence and tutoring systems: computational and cognitive approaches to the communication of knowledge**. [S.l.]: Los Altos (Calif.): M. Kaufmann, 1987.
- XIAO, J. *et al.* A personalized recommendation system with combinational algorithm for online learning. **Journal of Ambient Intelligence and Humanized Computing**, Springer, v. 9, n. 3, p. 667–677, 2018. Available at: <<https://doi.org/10.1007/s12652-017-0466-8>>.
- XU, Y. *et al.* Same benefits, different communication patterns: Comparing children’s reading with a conversational agent vs. a human partner. **Computers & Education**, Elsevier, v. 161, p. 1–17, 2021. Available at: <<https://doi.org/10.1016/j.compedu.2020.104059>>.
- YAGIURA, M.; KISHIDA, M.; IBARAKI, T. A 3-flip neighborhood local search for the set covering problem. **European Journal of Operational Research**, Elsevier, v. 172, n. 2, p. 472–499, 2006. Available at: <<https://doi.org/10.1016/j.ejor.2004.10.018>>.
- YIN, J. *et al.* Conversation technology with micro-learning: The impact of chatbot-based learning on students’ learning motivation and performance. **Journal of Educational Computing Research**, SAGE Publications Sage CA: Los Angeles, CA, v. 59, n. 1, p. 154–177, 2021. Available at: <<https://doi.org/10.1177/0735633120952067>>.
- ZHAO, X. *et al.* A hybrid approach of topic model and matrix factorization based on two-step recommendation framework. **Journal of Intelligent Information Systems**, Springer, v. 44, n. 3, p. 335–353, 2015. Available at: <<https://doi.org/10.1007/s10844-014-0334-3>>.

Appendix

APPENDIX **A**

Grouped interventions

The three figures below correspond to each page of didactic and structured educational content that brings together coherently organized interventions.



SUMÁRIO

DEFINIÇÃO.....	1
UTILIDADE.....	1
ATENÇÃO.....	1
EXEMPLO.....	2
DÚVIDAS.....	3

DEFINIÇÃO

Através de uma condição um determinado bloco de código vai se repetir até que essa condição seja falsa.

UTILIDADE

Usando tal ferramenta é possível colocar determinada parte do código em um loop. Dessa forma, facilitando diversas operações como somas consecutivas, por exemplo.

ATENÇÃO

- Utilize for quando você sabe o número exato de vezes que o loop deve ser executado.
- Utilize while quando a condição precisa ser verificada antes de cada iteração.
- Utilize do-while quando a condição precisa ser verificada após cada iteração.
- Certifique-se de que a condição do loop eventualmente se torne falsa. Caso contrário, o loop continuará indefinidamente, o que pode causar problemas.
- Se a condição dentro dos loops for falsa logo de início, o código dentro do bloco não será executado.

Figure 26 – Educational content with interventions (Part 1 of 3)



Funcionamento e Sintaxe

For:

```
int main() {  
  
    /*  
    for(inicialização; condição; atualização)  
    */  
    for(int i = 0; i < 5; i++)  
    {  
        //Será mostrado 5 vezes na tela  
        puts("Dentro do For");  
    }  
  
    return 0;  
}
```

While:

```
int main() {  
    int i = 5;  
    //Dentro do parênteses vai a condição  
    //Enquanto ela for verdade o while se repetirá  
    while(i < 10)  
    {  
        //Essa frase será mostrada 5 vezes  
        puts("Dentro do while");  
        i++;  
    }  
  
    return 0;  
}
```



Do while:

```
int main() {
    int i = 5;

    //Esse bloco de código vai acontecer pelo menos uma vez
do
{
    //Essa frase será mostrada 1 vez
    puts("Dentro do while");
    i++;
} while(i != 6); //Dentro do parênteses vai a condição
//Enquanto ela for verdade o while se repetirá

    return 0;
}
```

? DÚVIDAS

Dúvidas frequentes

- **Qual é a estrutura de um loop for?** Um loop for em C tem três componentes principais: inicialização, condição e atualização. Certifique-se de que todos os três estejam corretamente definidos. Por exemplo, `for(i = 0; i < 10; i++)`.
- **O que é um loop infinito e por que geralmente é considerado um erro de programação?** Um loop infinito é um loop que nunca termina porque sua condição de término nunca é atendida. Isso é considerado um erro porque leva a programas que nunca terminam ou consomem recursos excessivos.

APPENDIX **B**

RS documentation

The RS developed in Python has two APIs: Ontosearch and Lors. Below is the Swagger documentation of the Ontosearch and Lors modules.

Swagger Supported by SMARTBEAR /apispec_1.json **Explore**

API ONTO-SEARCH 1.0.0

[Base URL: localhost:5000]
/apispec_1.json

API Rest desenvolvida para o Sistema de Recomendação de Objetos de Aprendizagem

[Terms of service](#)
[Contact the developer](#)
Apache 2.0

Schemes: HTTP **Authorize**

User

- POST** /v1/users/register Cadastra um novo usuário

Estudante

- GET** /v1/students Retorna todos os estudantes do banco
- GET** /v1/students/profile/{id_student} Retorna o perfil (FSLSM) de um dado estudante

Questões do Index of Learning Styles (ILS)

- GET** /v1/students/questions_to_student Retorna questões do questionário ILS ainda não respondidas pelo estudante

Busca questões respondidas do Index of Learning Styles (ILS)

- GET** /v1/students/answer/{id_student} Retorna respostas ao questionário ILS dadas pelo estudante

Figure 29 – Ontosearch API - Swagger documentation (Part 1 of 2)

Salva respostas do Index of Learning Styles (ILS) ▼		
POST	<code>/v1/students/answer/{id_student}</code>	Salva na ontologia respostas do questionário ILS dadas pelo estudante 
Objeto de Aprendizagem (OA) ▼		
GET	<code>/v1/learn_objects/{id_lo}</code>	Retorna o OA dado o nome da instância 
PUT	<code>/v1/learn_objects/{id_lo}</code>	Atualiza OA na ontologia 
POST	<code>/v1/learn_objects/{id_lo}</code>	Salva OA na ontologia 
GET	<code>/v1/learn_objects</code>	Retorna lista de OAs dados os nomes das instâncias 
Busca na Wikipédia ▼		
GET	<code>/v1/search_wiki</code>	Retorna as seções de páginas wiki relacionadas a concepts 
Oauth 2.0 ▼		
POST	<code>/v1/oauth/token</code>	Autorização OAuth 2.0
POST	<code>/v1/oauth/refresh</code>	Autorização OAuth 2.0
DELETE	<code>/v1/oauth/logout</code>	Regovar autorização OAuth 2.0 
Backup da ontologia ▼		
GET	<code>/v1/ontology/backup/{critical_backup}</code>	Salva a ontologia atual na Azure Cloud 

Figure 30 – Ontosearch API - Swagger documentation (Part 2 of 2)

Swagger
supported by SMARTBEAR

/apispec_1.json Explore

API LORS - Learning Object Recommendation System ^{1.0.0}

[Base URL: localhost:80]
/apispec_1.json

API Rest - Sistema de Recomendação de Objetos de Aprendizagem

[Terms of service](#)
[Contact the developer](#)
Apache 2.0

Schemes
HTTP Authorize

Recomendação de Objetos de Aprendizagem (OAs)

- POST** /v1/recommendation/{id_student}/{recomm_type} Recomenda OAs

Estudante

- GET** /v1/student/rating Retorna todos os OAs do banco avaliados pelos estudantes
- POST** /v1/student/rating/{id_student} Salva os ratings e os comentários sobre os OAs (id_los) avaliados pelo estudante (id_student)
- PUT** /v1/student/rating/{id_student}/{id_lo}/{score} Atualiza o score de um OA avaliado pelo estudante
- DELETE** /v1/student/rating/{id_rating} Deleta uma avaliação (rating) do banco de dados pelo id_rating

Figure 31 – Edulors API - Swagger documentation (Part 1 of 3)

Objeto de Aprendizagem (OA)			▼
GET	/v1/learning_objects	Retorna todos OAs do banco	🔒
GET	/v1/learning_objects/ids	Retorna OAs do banco dada uma lista de ids	🔒
GET	/v1/learning_objects/{id_lo}	Retorna o OA dado o id	🔒
DELETE	/v1/learning_objects/{id_lo}	Deleta o OA de um dado id	🔒
PUT	/v1/learning_objects/{id_lo}/{video}	Atualiza OA na ontologia	🔒
POST	/v1/learning_objects/{video}	Salva OA na ontologia	🔒
Conceitos			▼
GET	/v1/learning_objects/concepts	Retorna todos os conceitos do banco	🔒
POST	/v1/learning_objects/concepts	Salva Conceito no banco	🔒
PUT	/v1/learning_objects/concepts/{id_concept}	Atualiza conceito no banco de dados	🔒
DELETE	/v1/learning_objects/concepts/{id_concept}	Deleta um conceito do banco de dados pelo id	🔒
Tipos de Recursos de Aprendizagem			▼
GET	/v1/learning_objects/resource_types	Retorna todos os tipos de recursos de aprendizagem do banco	🔒
POST	/v1/learning_objects/resource_types	Salva tipo de recurso no banco	🔒
PUT	/v1/learning_objects/resource_types/{id_resource}	Atualiza tipo de recurso no banco de dados	🔒
DELETE	/v1/learning_objects/resource_types/{id_resource}	Deleta um tipo recurso de aprendizagem do banco de dados pelo id	🔒

Figure 32 – Edulors API - Swagger documentation (Part 2 of 3)

Gamificação			▼
GET	/v1/gamification/score	Retorna pontuação, nível e conquistas de todos os estudantes	🔒
GET	/v1/gamification/score/{id_student}	Retorna pontuação, nível e conquistas do estudante	🔒
PUT	/v1/gamification/score/{id_student}	Atualiza pontuação, nível e conquistas do estudante	🔒
GET	/v1/gamification/questions	Retorna todos OAs do tipo questão do banco	🔒
POST	/v1/gamification/questions	Salva OA do tipo questão na ontologia e banco	🔒
GET	/v1/gamification/questions/{gamification}	Retorna questões ainda não respondidas corretamente pelo estudante	🔒
POST	/v1/gamification/answers /{id_student}	Salva os ids dos OAs (do tipo questão) com as respectivas respostas dadas pelo estudante	🔒
Feedback			▼
GET	/v1/feedback/evaluations	Retorna todas as avaliações do chatbot realizadas pelos estudantes	🔒
POST	/v1/feedback/evaluations	Salva a avaliação do chatbot realizada pelo estudante	🔒
PUT	/v1/feedback/evaluations/{id_evaluation}	Atualiza avaliação do chatbot	🔒
DELETE	/v1/feedback/evaluations/{id_evaluation}	Deleta uma avaliação do chatbot	🔒
POST	/v1/feedback/student_log	Salva log (intent, entity, setença/conversa) dos estudantes	🔒

Figure 33 – Edulors API - Swagger documentation (Part 3 of 3)

APPENDIX **C****Questionnaire**

Below is the questionnaire related to “RS usability and student satisfaction”

QUESTIONÁRIO 1: Usabilidade e Satisfação					
Por favor, pontue as afirmações abaixo sobre o Sistema de Recomendação de Objetos de Aprendizagem e sua interface (chatbot)					
	Discordo totalmente	Discordo	Neutro	Concordo	Concordo totalmente
1. Eu gostaria de usar o sistema frequentemente.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
2. Eu achei o sistema fácil de usar.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
3. Eu usei o chatbot para solucionar as minhas dúvidas.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
4. O chatbot solucionou todas as minhas dúvidas.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
5. O sistema fornece materiais confiáveis.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
6. Eu me senti muito confiante com o chatbot.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
7. O conteúdo recomendado foi útil.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
8. Os materiais recomendados me ajudaram a aprender o conteúdo do curso.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
9. Em geral, estou muito satisfeito com o sistema de recomendação.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
10. O sistema de recomendação me ajudou a descobrir novos materiais.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
11. O layout do chatbot é atrativo.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
12. O sistema de recomendação explica por que os materiais foram recomendados para mim.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
13. Eu achei fácil informar ao sistema se eu gosto/não gosto do item recomendado.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
14. Eu me familiarizei com o sistema de recomendação muito facilmente.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
15. Eu entendi por que estes conteúdos foram recomendados para mim.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
16. As informações fornecidas com os materiais recomendados são suficientes para eu tomar a decisão de abri-los.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Gostaria de deixar alguma observação?					

Figure 34 – Usability and satisfaction at the end of the term

Annex

Index of Learning Styles Questionnaire

Below are the 44 questions of the questionnaire created by Solomon and Felder (2005) to assess learner preferences in the four dimensions of the FSLSM proposed by Felder and Silverman (1988): Input (Visual and Verbal), Organization (Sequential and Global), Perception (Sensitive and Intuitive) and Processing (Active and Reflective).

1. I understand something better after I
 - try it out.
 - think it through.
2. I would rather be considered
 - realistic.
 - innovative.
3. When I think about what I did yesterday, I am most likely to get
 - a picture.
 - words.
4. I tend to
 - understand details of a subject but may be fuzzy about its overall structure.
 - understand the overall structure but may be fuzzy about details.
5. When I am learning something new, it helps me to
 - talk about it.
 - think about it.
6. If I were a teacher, I would rather teach a course
 - that deals with facts and real life situations.

- that deals with ideas and theories.
7. I prefer to get new information in
- pictures, diagrams, graphs, or maps.
 - written directions or verbal information.
8. Once I understand
- all the parts, I understand the whole thing.
 - the whole thing, I see how the parts fit.
9. In a study group working on difficult material, I am more likely to
- jump in and contribute ideas.
 - sit back and listen.
10. I find it easier
- to learn facts.
 - to learn concepts.
11. In a book with lots of pictures and charts, I am likely to
- look over the pictures and charts carefully.
 - focus on the written text.
12. When I solve math problems
- I usually work my way to the solutions one step at a time.
 - I often just see the solutions but then have to struggle to figure out the steps to get to them.
13. In classes I have taken
- I have usually gotten to know many of the students.
 - I have rarely gotten to know many of the students.
14. In reading nonfiction, I prefer
- something that teaches me new facts or tells me how to do something.
 - something that gives me new ideas to think about.
15. I like teachers
- who put a lot of diagrams on the board.

-
- who spend a lot of time explaining.
16. When I'm analyzing a story or a novel
- I think of the incidents and try to put them together to figure out the themes.
- I just know what the themes are when I finish reading and then I have to go back and find the incidents that demonstrate them.
17. When I start a homework problem, I am more likely to
- start working on the solution immediately.
- try to fully understand the problem first.
18. I prefer the idea of
- certainty
- theory.
19. I remember best
- what I see.
- what I hear.
20. It is more important to me that an instructor
- lay out the material in clear sequential steps.
- give me an overall picture and relate the material to other subjects.
21. I prefer to study
- in a study group.
- alone.
22. I am more likely to be considered
- careful about the details of my work.
- creative about how to do my work.
23. When I get directions to a new place, I prefer
- a map.
- written directions.
24. I learn
- at a fairly regular pace. If I study hard, I'll "get it."

- in fits and starts. I'll be totally confused and then suddenly it all "clicks."
25. I would rather first
- try things out.
 - think about how I'm going to do it.
26. When I am reading for enjoyment, I like writers to
- clearly say what they mean.
 - say things in creative, interesting ways.
27. When I see a diagram or sketch in class, I am most likely to remember
- the picture.
 - what the instructor said about it.
28. When considering a body of information, I am more likely to
- focus on details and miss the big picture.
 - try to understand the big picture before getting into the details.
29. I more easily remember
- something I have done.
 - something I have thought a lot about.
30. When I have to perform a task, I prefer to
- master one way of doing it.
 - come up with new ways of doing it.
31. When someone is showing me data, I prefer
- charts or graphs.
 - text summarizing the results.
32. When writing a paper, I am more likely to
- work on (think about or write) the beginning of the paper and progress forward.
 - work on (think about or write) different parts of the paper and then order them.
33. When I have to work on a group project, I first want to
- have "group brainstorming" where everyone contributes ideas.

- brainstorm individually and then come together as a group to compare ideas.
34. I consider it higher praise to call someone
- sensible.
 - imaginative.
35. When I meet people at a party, I am more likely to remember
- what they looked like.
 - what they said about themselves.
36. When I am learning a new subject, I prefer to
- stay focused on that subject, learning as much about it as I can.
 - try to make connections between that subject and related subjects.
37. I am more likely to be considered
- outgoing.
 - reserved.
38. I prefer courses that emphasize
- concrete material (facts, data).
 - abstract material (concepts, theories).
39. For entertainment, I would rather
- watch television.
 - read a book.
40. Some teachers start their lectures with an outline of what they will cover. Such outlines are
- somewhat helpful to me.
 - very helpful to me.
41. The idea of doing homework in groups, with one grade for the entire group,
- appeals to me.
 - does not appeal to me.
42. When I am doing long calculations,
- I tend to repeat all my steps and check my work carefully.

- I find checking my work tiresome and have to force myself to do it.
43. I tend to picture places I have been
- easily and fairly accurately.
 - with difficulty and without much detail.
44. When solving problems in a group, I would be more likely to
- think of the steps in the solutions process.
 - think of possible consequences or applications of the solution in a wide range of areas.

SWRL rules

Below are the 32 SWRL rules of the ontology initially proposed in (BELIZÁRIO JÚNIOR; DORÇA, 2018).

1. Student(?student), IdealLOs(?idealLO), hasState(?idealLO, activeIdealLO), isRecommendedFor(?idealLO, ?student), Profile(?profile), hasProfile(?student, ?profile), hasProcessing(?profile, activeProcessing) → ListResourcesIdealLO(animation)
2. Student(?student), IdealLOs(?idealLO), hasState(?idealLO, activeIdealLO), isRecommendedFor(?idealLO, ?student), hasProfile(?student, ?profile), hasPerception(?profile, sensing) → ListResourcesIdealLO(animation)
3. Student(?student), IdealLOs(?idealLO), hasState(?idealLO, activeIdealLO), isRecommendedFor(?idealLO, ?student), Profile(?profile), hasProfile(?student, ?profile), hasProcessing(?profile, reflective) → ListResourcesIdealLO(reflectionQuiz)
4. Student(?student), IdealLOs(?idealLO), hasState(?idealLO, activeIdealLO), isRecommendedFor(?idealLO, ?student), Profile(?profile), hasProfile(?student, ?profile), hasPerception(?profile, intuitive) → ListResourcesIdealLO(additionalResource)
5. Student(?student), IdealLOs(?idealLO), hasState(?idealLO, activeIdealLO), isRecommendedFor(?idealLO, ?student), Profile(?profile), hasProfile(?student, ?profile), hasProcessing(?profile, reflective) → ListResourcesIdealLO(example)
6. Student(?student), IdealLOs(?idealLO), hasState(?idealLO, activeIdealLO), isRecommendedFor(?idealLO, ?student), Profile(?profile), hasProfile(?student, ?profile), hasUnderstanding(?profile, sequential) → ListResourcesIdealLO(additionalResource)
7. Student(?student), IdealLOs(?idealLO), hasState(?idealLO, activeIdealLO), isRecommendedFor(?idealLO, ?student), Profile(?profile), hasProfile(?student, ?profile), hasProcessing(?profile, activeProcessing) → ListResourcesIdealLO(selfAssessment)
8. Student(?student), IdealLOs(?idealLO), hasState(?idealLO, activeIdealLO), isRecommendedFor(?idealLO, ?student), Profile(?profile), hasProfile(?student, ?profile), hasInput(?profile, verbal) → ListResourcesIdealLO(additionalResource)
9. Student(?student), IdealLOs(?idealLO), hasState(?idealLO, activeIdealLO), isRecommendedFor(?idealLO, ?student), Profile(?profile), hasProfile(?student, ?profile), hasProcessing(?profile, activeProcessing) → ListResourcesIdealLO(forumActivity)

10. Student(?student), IdealLOs(?idealLO), hasState(?idealLO, activeIdealLO), isRecommendedFor(?idealLO, ?student), Profile(?profile), hasProfile(?student, ?profile), hasPerception(?profile, sensing) → ListResourcesIdealLO(realLifeApplication)
11. Student(?student), IdealLOs(?idealLO), hasState(?idealLO, activeIdealLO), isRecommendedFor(?idealLO, ?student), Profile(?profile), hasProfile(?student, ?profile), hasProcessing(?profile, reflective) → ListResourcesIdealLO(additionalResource)
12. Student(?student), IdealLOs(?idealLO), hasState(?idealLO, activeIdealLO), isRecommendedFor(?idealLO, ?student), Profile(?profile), hasProfile(?student, ?profile), hasInput(?profile, verbal) → ListResourcesIdealLO(forumActivity)
13. Student(?student), IdealLOs(?idealLO), hasState(?idealLO, activeIdealLO), isRecommendedFor(?idealLO, ?student), Profile(?profile), hasProfile(?student, ?profile), hasProcessing(?profile, activeProcessing) → ListResourcesIdealLO(exercise)
14. Student(?student), IdealLOs(?idealLO), hasState(?idealLO, activeIdealLO), isRecommendedFor(?idealLO, ?student), Profile(?profile), hasProfile(?student, ?profile), hasPerception(?profile, sensing) → ListResourcesIdealLO(exercise)
15. Student(?student), IdealLOs(?idealLO), hasState(?idealLO, activeIdealLO), isRecommendedFor(?idealLO, ?student), Profile(?profile), hasProfile(?student, ?profile), hasPerception(?profile, sensing) → ListResourcesIdealLO(example)
16. Student(?student), IdealLOs(?idealLO), hasState(?idealLO, activeIdealLO), isRecommendedFor(?idealLO, ?student), Profile(?profile), hasProfile(?student, ?profile), hasUnderstanding(?profile, sequential) → ListResourcesIdealLO(exercise)
17. Student(?student), IdealLOs(?idealLO), hasState(?idealLO, activeIdealLO), isRecommendedFor(?idealLO, ?student), Profile(?profile), hasProfile(?student, ?profile), hasPerception(?profile, intuitive) → ListResourcesIdealLO(exercise)
18. Student(?student), IdealLOs(?idealLO), hasState(?idealLO, activeIdealLO), isRecommendedFor(?idealLO, ?student), Profile(?profile), hasProfile(?student, ?profile), hasUnderstanding(?profile, sequential) → ListResourcesIdealLO(selfAssessment)
19. Student(?student), IdealLOs(?idealLO), hasState(?idealLO, activeIdealLO), isRecommendedFor(?idealLO, ?student), Profile(?profile), hasProfile(?student, ?profile), hasUnderstanding(?profile, global) → ListResourcesIdealLO(realLifeApplication)
20. Student(?student), IdealLOs(?idealLO), hasState(?idealLO, activeIdealLO), isRecommendedFor(?idealLO, ?student), Profile(?profile), hasProfile(?student, ?profile), hasPerception(?profile, intuitive) → ListResourcesIdealLO(reflectionQuiz)
21. Student(?student), IdealLOs(?idealLO), hasState(?idealLO, activeIdealLO), isRecommendedFor(?idealLO, ?student), Profile(?profile), hasProfile(?student, ?profile), hasUnderstanding(?profile, global) → ListResourcesIdealLO(example)
22. Student(?student), IdealLOs(?idealLO), hasState(?idealLO, activeIdealLO), isRecommendedFor(?idealLO, ?student), Profile(?profile), hasProfile(?student, ?profile),

hasInput(?profile, visual) → ListResourcesIdealLO(animation)

23. Student(?student), IdealLOs(?idealLO), hasState(?idealLO, activeIdealLO), isRecommendedFor(?idealLO, ?student), Profile(?profile), hasProfile(?student, ?profile), hasUnderstanding(?profile, sequential) → ListResourcesIdealLO(reflectionQuiz)
24. Student(?student), IdealLOs(?idealLO), hasState(?idealLO, activeIdealLO), isRecommendedFor(?idealLO, ?student), Profile(?profile), hasProfile(?student, ?profile), hasUnderstanding(?profile, sequential) → ListResourcesIdealLO(animation)
25. IdealLOs(?idealLO), hasState(?idealLO, activeIdealLO), ListResourcesIdealLO(?res_type), PermanentLOs(?lo), Educational_5(?edu), hasEducationalData(?lo, ?edu), hasLearningResourceType(?edu, ?res_type) → SuggestedLOs(?lo)
26. IdealLOs(?idealLO), hasState(?idealLO, activeIdealLO), General_1(?gen_ideal), hasGeneralData(?idealLO, ?gen_ideal), hasKeyword(?gen_ideal, ?keyword), PermanentLOs(?lo), General_1(?gen), hasGeneralData(?lo, ?gen), hasKeyword(?gen, ?keyword) → SuggestedLOs(?lo)
27. IdealLOs(?idealLO), hasState(?idealLO, activeIdealLO), General_1(?gen_ideal), hasGeneralData(?idealLO, ?gen_ideal), hasTitle(?gen_ideal, ?title), PermanentLOs(?lo), General_1(?gen), hasGeneralData(?lo, ?gen), hasTitle(?gen, ?title) → SuggestedLOs(?lo)
28. IdealLOs(?idealLO), hasState(?idealLO, activeIdealLO), Educational_5(?edu_ideal), hasEducationalData(?idealLO, ?edu_ideal), hasSemanticDensity(?edu_ideal, ?sem_den), PermanentLOs(?lo), Educational_5(?edu), hasEducationalData(?lo, ?edu), hasSemanticDensity(?edu, ?sem_den) → SuggestedLOs(?lo)
29. IdealLOs(?idealLO), hasState(?idealLO, activeIdealLO), Educational_5(?edu_ideal), hasEducationalData(?idealLO, ?edu_ideal), hasDifficulty(?edu_ideal, ?diff), PermanentLOs(?lo), Educational_5(?edu), hasEducationalData(?lo, ?edu), hasDifficulty(?edu, ?diff) → SuggestedLOs(?lo)
30. IdealLOs(?idealLO), hasState(?idealLO, activeIdealLO), Educational_5(?edu_ideal), hasEducationalData(?idealLO, ?edu_ideal), hasInteractivityLevel(?edu_ideal, ?int_lev), PermanentLOs(?lo), Educational_5(?edu), hasEducationalData(?lo, ?edu), hasInteractivityLevel(?edu, ?int_lev) → SuggestedLOs(?lo)
31. IdealLOs(?idealLO), hasState(?idealLO, activeIdealLO), Educational_5(?edu_ideal), hasEducationalData(?idealLO, ?edu_ideal), hasInteractivityType(?edu_ideal, ?int_type), PermanentLOs(?lo), Educational_5(?edu), hasEducationalData(?lo, ?edu), hasInteractivityType(?edu, ?int_type) → SuggestedLOs(?lo)
32. TemporaryLOs(?tempLO), General_1(?gen_temp), hasGeneralData(?tempLO, ?gen_temp), Identifier(?ide_temp), hasIdentifier(?gen_temp, ?ide_temp), hasEntry_(?ide_temp, ?uri), PermanentLOs(?lo), General_1(?gen), hasGeneralData(?lo, ?gen), Identifier(?ide), hasIdentifier(?gen, ?ide), hasEntry_(?ide, ?uri) → SuggestedLOs(?lo)

Metadata generation for wiki sections

In this annex, we illustrate the operation of the automatic metadata generation process developed during the master's degree (BELIZÁRIO JÚNIOR, 2018).

C.1 Quality assessment of wiki sections

Wikipedia is a great digital encyclopedia. On its main page, consulted on August 30, 2018, it appears that it has a collection of more than 5 million and 708 thousand articles in English and 1 million articles in Portuguese. After almost 4 years, on June 16, 2022, Wikipedia reaches 6 million and 519 thousand articles in English and 1 million and 93 thousand articles in Portuguese. Wikipedia content may be copied, modified and redistributed under a Creative Commons BY-SA license. The Wikipedia community can assess the quality of articles with almost a dozen labels that serve to rank articles from the worst to the best, but manually labeling these articles is an unfeasible process, given that, in recent years, the amount of Wikipedia articles has grown exponentially and, associated with this, articles are edited all the time, which can change the quality of the article. To deal with this problem, several researches have been developed for the automatic evaluation of wiki articles.

Warncke-Wang, Cosley and Riedl (2013) used a classifier based on a decision tree to identify the quality of Wikipedia articles. The authors showed that by extracting only five features from wiki pages it is possible to obtain significant results. These characteristics are readily available and defined by the authors as follows:

1. Completeness = $0.4 * \text{NumBrokenWikilinks} + 0.4 * \text{NumWikilinks}$;
2. Informativeness = $0.6 * \text{InfoNoise} + 0.3 * \text{NumImagens}$ (InfoNoise is the content of the page itself, without programming code and without stopwords);
3. NumHeadings: the number of sections in the wiki article;
4. ArticleLength: the length of the article;

5. NumReferences/ArticleLength: a measure of the number of citations by counting `<ref>` tags, which are used in footnote citations. The division by article size (ArticleLength) is important to reduce the correspondence (the larger the article, the greater the number of citations) that exists between the two variables.

The classifier presented by Warncke-Wang, Cosley and Riedl (2013) can classify wiki articles into seven different classes: FA, GA, A, B, C, Start and Stub, all defined by Wikipedia itself. The authors divided the articles into two large classes: GoodEnough articles (containing FA, GA and A) and NeedsWork (containing B, C, Start and Stub).

In the master's dissertation, we mapped these classes in values according to Table 16. Thus, it is possible to use Eq. (4). The classes in the table are ordered by their quality standard. Articles with the highest quality level (FA) receive a maximum value equal to 1 and articles with no quality receive a score of 0.

Table 16 – Wikipedia quality classes

Class	Criteria	Value
FA (Featured Articles)	It contains the highest quality articles, they are well-written, understandable, neutral (no opinions), have reliable sources, are stable (the content does not need constant changes), follow a good structure (with abstract, division into sections and well-formatted citations), have appropriate images and adequate size (focus on the main topic without unnecessary content).	1
A	The article is well organized and essentially complete. Only minor stylistic issues and other details need to be fixed before submission as an FA article.	0.85
GA (Good Articles)	The article requires minor changes to achieve a better quality level.	0.70
B	The article is almost complete and without major problems, but it requires additional changes to reach the standard of good article (GA).	0.50
C	The article is substantial but lacks important content or contains irrelevant material. The article cites more than one reliable source and is more developed in style, structure, and quality than the Start class, but fails one or more of the B class criteria.	0.30
Start	It's an article under construction and quite incomplete, but it has a significant amount of good content. Does not follow the Wikipedia Manual of Style. May or may not cite reliable and appropriate sources.	0.15
Stub	The article has a very basic description of the topic, and all articles of very poor quality belong to this class.	0

The quality of wiki articles corresponds to one of the p parameters defined by Eq. (4). Let α_q be a quality parameter and, to use it in Eq. (4), the $\alpha_{q_{ideal}}$ and α_{q_j} of the j -th LO

compared to the ideal LO are defined below:

- a) $\alpha_{q_{ideal}} = 1$ (The ideal LO certainly has the highest level of quality, belonging to the FA class);
- b) α_{q_j} is the value corresponding to the quality rating received by the j -th LO according to Table 16.

The solution to the original problem is to recommend a group of LOs with the fewest differences from the ideal LO. Thus, the higher the quality of an LO, the greater the chances of it being recommended, as the difference between it and the ideal LO is smaller.

C.2 Identification of wiki section concepts

The return of a Wikipedia search are wiki pages, whose sections are candidates for LOs. For the sections of these pages to actually become LOs, it is necessary that they deal with the concepts chosen by the user, that is, the concepts that the student is expected to learn.

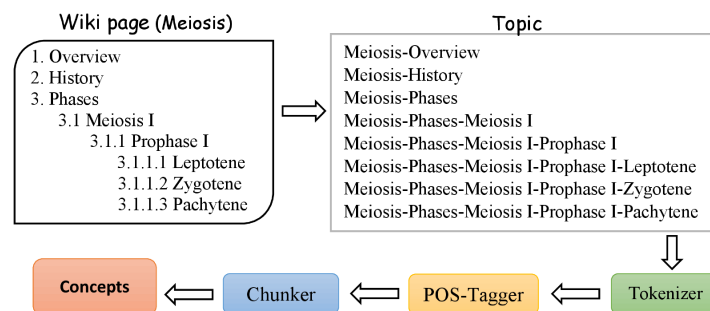


Figure 35 – Processing a Wikipedia summary for concept generation

First, concepts are extracted from the wiki sections to confront them with the concepts entered by the user in the ideal LO. The method used to extract concepts from wiki sections (see Fig. 35) is based on the intuition that the section title itself plus the hierarchy of titles in which it is inserted can be used to extract the concepts it deals with. In this case, after pointing out the grammatical class (POS-tagging) of each word found in this hierarchy of titles by the tokenizer, the chunker is used to form each Noun Phrase (NP). An NP can be any type of noun, including nouns interspersed with prepositions and characterized by adjectives. Although some undesirable nouns can be selected as concepts, the titles of sections almost always contain the main concepts related to them. For example, the topic “Meiosis-Phases-Meiosis I-Prophase I” is related to the concepts Meiosis, Phases, Meiosis I and Prophase I.

This conversion process is performed with the aid of two Python natural language processing libraries. The spaCy library¹ is used in the Tokenizer and POS-Tagger steps, and the NLTK library² is used in the Chunker step.

In addition, we use the Wikipedia API³, a Python module that supports extracting sections, titles, links and categories from Wikipedia. In fact, it is possible to extract almost every part of a wiki page using simple commands. This API can also be used to search for wiki pages, using the page method, which expects the title of the page being searched for as a parameter. In the Master's, the Wikipedia API was mainly used to:

- a) The search for the wiki page associated with each concept of the ideal LO not covered by the permanent LOs of the ontology;
- b) Extracting the titles of the wiki sections to fill in the title field (1.2 IEEE-LOM) and the entry field (1.1.2 IEEE-LOM), which corresponds to the URI of the wiki section.

¹ Available at: <https://spacy.io/>

² Available at: <https://www.nltk.org/>

³ Available at: <https://pypi.org/project/Wikipedia-API/>