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

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 - cpgfacom@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.

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

The undersigned hereby certify they have read and recommend to the PPGCO for acceptance the thesis entitled “An Approach to the Person alized Learning Objects Recommendation Problem as a Set Cov ering Problem Using Ontologies and Metaheuristics” submitted by “Clarivando Francisco Beliz´ario J´unior” as part of the requirements for obtaining the degree of Doctor in Computer Science.

Uberlˆandia, July 29, 2024

Supervisor:

Prof. Dr. Fabiano Azevedo Dor¸ca
Universidade Federal de Uberlˆandia (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. Jos´e Palazzo Moreira de Oliveira
Universidade Federal do Rio Grande do Sul (UFRGS)

Profa. Dra. M´arcia Aparecida Fernandes
Universidade Federal de Uberlˆandia (UFU)

Prof. Dr. Renan Gon¸calves Cattelan
Universidade Federal de Uberlˆandia (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 var ious 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 be longs 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 rec ommender 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

List of Tables

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

Chapter 1

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 con tent 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 con tent 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 Ob jects 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 combina tion of two or more techniques (hybrid recommendation) (BURKE, 2007; BARRAGÁNS MARTÍNEZ *et al.*, 2010; CHOI *et al.*, 2012; TARUS; NIU; YOUSIF, 2017). These tech niques suffer from the rating sparsity (ZHAO *et al.*, 2015) and cold-start (ADOMAVI CIUS; 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 ge netic 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 com munication 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 recommen dation 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 tech niques. 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 infer ence 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 differ ent areas of knowledge through the reuse of Web content and that meets the stu dents' 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 deliv ery 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 gamifica tion that will be able to tutor students, their learning and mediate more engaging dialogues, motivating and facilitating student learning and reducing possible peda gogical difficulties.

1.5 Thesis organization

The rest of this work is organized as follows. The background of this research is pre sented 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.

Chapter **2**

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. Ontolo gies 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

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 oper ators. 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 con texts. 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 iden tifier, 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, cre ation 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 inter activity 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: "*is part of* ", "*is based on*" and "*is required by*".
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 manage ment systems.

The SCORM standard aims at portability by allowing different types of learning ob jects to communicate efficiently with the various existing e-learning platforms. The stan dard 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 stu dent'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 Silver man (1988), is widely regarded as one of the most comprehensive frameworks for mod eling 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 In put 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 un derstand information in linear steps, with Global learners, who grasp concepts more holis tically. 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-

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 feed back and dynamic adaptation, providing detailed explanations and guided questions to enhance student understanding. Both ITSs underscore the impact of adaptive and con textualized 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 bore dom. 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 empha size 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

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 calcu lating 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, of fering a more individualized approach that respects different learning rhythms and styles. Through continuous analysis of performance data, recommender systems can identify ar eas 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 learn ing 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 learn ing object makes it possible to bring ITS interventions into the context of RS, enabling personalized intervention recommendations. Additionally, the proposed recommender sys tem 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 moti vation. 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 re wards. 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 tan gible or symbolic, positively reinforce desired behaviors and keep participants engaged. In the educational context, gamification has been shown to be a powerful tool for im proving student motivation, increasing knowledge retention, and promoting active learn ing (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 fos ter 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 gamifica tion 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. Addition ally, 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 ap plied to a wide range of applications (LAN; DEPUY; WHITEHOUSE, 2007), such as crew scheduling in railway and airlines (HOUSOS; ELMROTH, 1997; CAPRARA; FIS CHETTI; 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 x 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.

Chapter **3**

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 ($i = 1, ..., m$) 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 x 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:

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

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

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

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

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

where $p(i)$ represents the cost of introducing the i-th LO in the sequence of LOs recom mended 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 pro cesses 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 fol lowed by local search strategies. Among the evolutionary optimization techniques, they used GA and the particle swarm method, which are executed in parallel by the Evo lutionary 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 con ceptualization”, 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	Ontology or Semantic Web technologies	Web content reuse	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 *et al.* (2019)	Yes	Yes	Greedy alg.	No	No
Belizário Júnior *et al.* (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 *et al.* (2017)	No	Yes	SWRL	No	No
Abech *et al.* (2016)	Yes	Yes	SWRL and SPARQL	No	No
Pereira *et al.* (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 *et al.* (2018)	Yes	No	Association rules, CBF and CF	No	No

Klašnja-Milićević, Vesin and Ivanović (2018)

Tarus, Niu and Kalui (2018) No Yes Social tagging and SPM

and CF
No No No No

No No Context awareness, SPM No No

Wan and Niu (2020) No No SPM and self-organization

Yin *et al.* (2021) No No No No Yes Colace *et al.* (2018) Yes Yes LDA algorithm No Yes Katz *et al.* (2021) No No No No Yes Nguyen *et al.* (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 rec ommendation 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 personaliza tion 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 fil tering 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 recom mendation 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 com bines computational context, SPM and CF algorithms for the recommendation of educa tional 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 to gether 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 sys tems 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 rep resentation 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, loca tion) 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 recommen dation 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, affec tion 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 gam ification 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 com petence 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.

Chapter **4**

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 jus tified 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 objec tives and curriculum requirements. While generative AI offers advanced capabilities, the simplicity and efficiency of intent-based chatbots make them a practical choice for educa tional 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 cus tomized, 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 ed ucation 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 con cepts 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

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.

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 stan dard, 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 Owlready2[1]library is a Python module that makes it easy to manipulate ontolo gies 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, includ ing 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/

 Table 3 –

Resource types recommended for each learner profile

Rule

Dimensi

input
input
input
understanc
understanc
understanc
understanc
understanc
understanc
understanc
perceptic
perceptic
perceptic
perceptic
perceptic
perceptic
perceptic
processir
processir
processir
processir
processir
processir

processing | reflecti

LOs
Learning re source type

1 additionalReading 2 forumActivity

3 animation
4 example

5 realLifeApplication 6 additionalReading 7 animation
8 exercise
9 reflectionQuiz
10 selfAssessment
11 additionalReading 12 exercise
13 reflectionQuiz
14 animation
15 example
16 exercise
17 realLifeApplication 18 animation
19 exercise
20 forumActivity
21 selfAssessment
22 additionalReading 23 example
24 reflectionQuiz

4.2 Improvements in Ontology for individualized in tervention 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.

Table 4 – Two new SWRL rules

Rule	Meaning
Rule 33	
IdealLOs(**?idealLO**) ∧ hasState(**?idealLO**, activeIdealLO) ∧ Relation_7(?rel) ∧ hasRelation(**?idealLO**, ?rel) ∧ Resource(?res) ∧ hasResource(?rel, ?res) ∧ Identifier(?ideideal) ∧ hasIdentifier(?res, ?ideideal) ∧ hasEntry_(?ideideal, **?intentName**) ∧ Intents(**?intent**) ∧ hasName(**?intent**, **?intentName**) ∧ hasConcept(**?intent**, **?keyword**) ∧ PermanentLOs(**?lo**) ∧ General_1(?gen) ∧ hasGeneralData(**?lo**, ?gen) ∧ hasKeyword(?gen, **?keyword**) → SuggestedLOs(**?lo**)	Rule 34
	IdealLOs(**?idealLO**) ∧ hasState(**?idealLO**, activeIdealLO) ∧ Relation_7(?rel) ∧ hasRelation(**?idealLO**, ?rel) ∧ Resource(?res) ∧ hasResource(?rel, ?res) ∧ Identifier(?ideideal) ∧ hasIdentifier(?res, ?ideideal) ∧ hasEntry_(?ideideal, ?intentName) ∧ Intents(**?intent**) ∧ hasName(**?intent**, ?intentName) ∧ PermanentLOs**?lo**) ∧ hasLearningObject(**?intent**, **?lo**) → QualityLOs(**?lo**)
	IF there exists an active **?idealLO** such that

?idealLO has relation with an intent **?intent** **?idealLO** has relations with an intent *AND* **?intent** has concept **?keyword**, which **?intent** *AND* **?intent** has learning object is covered by a permanent LO **?lo** *THEN* **?lo** *THEN* **?lo** is a quality LO **?lo** is a suggested LO

IF there exists an active **?idealLO** such that

4.3 Learning Object Recommendation Problem de fined 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:

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

The $diss(O_{ideal}, O_j)$ value is inversely proportional to the degree of similarity between O_{ideal} and O_j. The result of $diss(O_{ideal}, O_j)$ 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):

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

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, *diff icult* = 0.75, *verydiff icult* = 1) that corresponds to the $\alpha_{i(j)}$ of Eq. (4).

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

$$c_j = diss(O_{ideal}, O_j) + (1 - P^L_j) \quad (5)$$

In this case, the calculation of the cost c_j takes into account the prediction P^L_j 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^L_j. 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:

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

where the *max* operator is a weight given to P^L_j 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^L_j 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^L_j. 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^L_j. 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 pro cedure (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 $C = \{1, ..., n\}$, 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 x 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_roots*
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: $D \leftarrow \{j \mid \forall j \in C, j$ 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: $candidates \leftarrow heuristic\_search(matrix, costs, top_k, num\_roots, cost_{j_{max}}, uncov_{j_{max}})$
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^0 is a work-in-progress solution of the Lorp algo rithm, 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 pro cedure *heuristic_search*(*matrix, costs,* top_k*, num_roots,* $cost_{jmax}$). 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

Algorithm 2 heuristic_search procedure

Input: *matrix*, *costs*, top_k, *num_roots*, $cost_{jmax}$, $uncov_{jmax}$
Output: *branches*

1: $k \leftarrow |top_k|$
2: *trees* ← [ ]
3: *size_trees* ← 0
4: *candidates* ← [ ]
5: *size_candidates* ← 0
6: **if** *num_roots* > *k* **then**
7: *num_roots* ← *k*
8: **end if**
9: **for** *j* from 1 to *num_roots* **do**
10: *new_cost* ← *costs*[*top_k*[*j*]]
11: $uncov_j \leftarrow \{i \mid \forall i \in uncov_{jmax}, matrix[i][j] = 0\}$
12: *new_branch* ← [[*j*], *new_cost, uncov_j*]
13: **if** $uncov_j$ **then**
14: *trees* ← *insertion_sort*(*trees, new_branch*)
15: *size_trees* ← *size_trees* + 1
16: **else**
17: *candidates* ← *insertion_sort*(*candidates, new_branch*)
18: *size_candidates* ← *size_candidates* + 1
19: **end if**
20: **end for**
21: **while** *trees* 6= Ø **do**
22: *head_branch* ← *tree*[0] . Get first branch: [*branch_nodes, cost, uncov_rows*] 23: *trees* ←

remove_head(*trees*) . Remove first branch 24: *size_trees* ← *size_trees* − 1
25: *j* ← *head_branch*[0][−1] + 1 . Gets last element of *branch_nodes* and add 1 26: **while** *j* <= *k* **do**
27: *new_branch_nodes* ← *head_branch*[0] + [*j*]
28: *new_cost* ← *head_branch*[1] + *costs*[*top_k*[*j*]]
29: **if** *new_cost* > $\delta * cost_{jmax}$ **then**
30: *j* ← *j* + 1
31: continue . Go to the next iteration of the loop 32: **end if**
33: $uncov_j$ ← {*i* | ∀ *i* ∈ uncov of *head_branch*, *matrix*[*i*][*j*] = 0}
34: **if** $uncov_j$ and *new_cost* > $cost_{jmax}$ **then**
35: *j* ← *j* + 1
36: continue . Go to the next iteration of the loop 37: **end if**
38: *new_branch* ← [*new_branch_nodes, new_cost,* $uncov_j$]
39: **if** $uncov_j$ **then**
40: *trees* ← *insertion_sort*(*trees, new_branch*)
41: *size_trees* ← *size_trees* + 1
42: **else**
43: *candidates* ← *insertion_sort*(*candidates, new_branch*)
44: *size_candidates* ← *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 $\delta * cost_{jmax}$, 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.	Goodbye It's never a farewell. After this function, a logoff will be performed.
Welcome If you feel like having a quick chat, just use this function.	I want to give you a rating. 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.
	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.	about any bug you encountered, which will be resolved quickly.
Practice Sends the user links to questions about the C programming language.	Clear Doubt Clear all your doubts about C with this function.
Report Bug You will receive a link to a form where you can send us a detailed message	

I want to know my profile. I want to practice a bit.	
Report bug.	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	5 pts	
Answer a quiz question (easy)	At least one question	1 pt	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		3 pts	If you evaluate the returned learning object, you will receive a 2-point bonus
Learn something new			

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 ques tions 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.

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 impor tant 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.