RÔMULO CÉSAR SILVA

SEQUENCIAMENTO ADAPTATIVO DE EXERCÍCIOS BASEADO NA CORRESPONDÊNCIA ENTRE A DIFICULDADE DA SOLUÇÃO E O DESEMPENHO DINÂMICO DO APRENDIZ

Tese apresentada como requisito para obtenção do grau de Doutor. Programa de Pós-Graduação em Informática, Setor de Ciências Exatas, Universidade Federal do Paraná.
Orientador: Prof. Dr. Alexandre I. Direne

CURITIBA

2015

S586s Silva, Rômulo César
Sequenciamento Adaptativo de Exercícios Baseado na Correspondência entre a Dificuldade da Solução e o Desempenho Dinâmico do Aprendiz/ Rômulo César Silva. – Curitiba, 2015.
82 f. : il. color. ; 30 cm.

Tese - Universidade Federal do Paraná, Setor de Ciências Exatas, Programa de Pós-graduação em Informática, 2015.

Orientador: Alexandre Ibrahim Direne .
Bibliografia: p. .

1. Calibração. 2. Estudantes - Avaliação. 3. Sistemas tutoriais inteligentes. I. Universidade Federal do Paraná. II.Direne, Alexandre Ibrahim. III. Título.

CDD: 371.334

Ministério da Educação
Universidade Federal do Paraná
Programa de Pós-Graduação em Informática

PARECER

Nós, abaixo assinados, membros da Comissão Examinadora da defesa do(a) aluno(a) de Doutorado em Ciência da Computação, Rômulo César Silva, avaliamos a de tese de doutorado intitulado "Sequenciamento Adaptativo de Exercícios Baseado na Correspondência entre a Dificuldade da Solução e o Desempenho Dinâmico do Aprendiz", cuja defesa pública, foi realizada no dia 21 de outubro de 2015. Após avaliação, decidimos pela: (X) **Aprovação** do(a) candidato(a). ( ) **Reprovação** do(a) candidato(a).

Curitiba, 21 de outubro de 2015.

Prof. Dr. Alexandre Ibrahim Direne
PPGINF – Orientador

Prof. Dr. Robinson Vida Noronha
UTFPR – Membro Externo

Prof. Dr. Diego Marczal
UTFPR – Membro Externo

Prof. Dr. Jomar Antonio Camarinha Filho
UFPR – Membro Externo

Prof. Dr. Andrey Ricardo Pimentel
PPGINF – Membro Interno

Prof. Dr. Eduardo Jaques Spinosa
PPGINF – Membro Interno

MINISTÉRIO DA EDUCAÇÃO - UNIVERSIDADE FEDERAL DO PARANÁ - Setor de Ciências Exatas - Coordenação de Pós-Graduação em Informática

"O aprendizado é o melhor substituto das ilusões pessoais."

(Waldo Vieira)

AGRADECIMENTOS

À minha esposa, Márcia Ebling, que sempre me incentivou e soube ser companheira nas horas difíceis.

À minha família, que sempre apoiou meus estudos.

Ao meu orientador, Alexandre Direne, por sua paciência, dedicação, entusiasmo, confiança nas minhas capacidades e fundamental colaboração nos momentos em que as situações mais complicadas emergiam da pesquisa.

A Diego Marczal, por sua parceria e ajuda na implementação da ADAPTFARMA.

À Ana Carla Borille e Valderez Luiz pelo empenho na elaboração e aplicação do objeto de aprendizagem aos alunos participantes da pesquisa.

Aos professores Paulo Guimarães e Ângelo Cabral do departamento de Estatística da UFPR, e seu aluno Bruno Camargo pela consultoria prestada.

Aos colegas Willian Zaleswki e Ricardo Tavares pelos momentos de descontração no laboratório do LIAMF.

SUMÁRIO

LISTA DE SIGLAS E ABREVIAÇÕES

ACT *Adaptive Control of Thought*

CBM *Constraint Based Modeling*

CT *Cognitive Tutor*

FARMA Ferrameta de Autoria para a Remediação de erros com Mobilidade na Aprendizagem

IA Inteligência Artificial

ITaG *Intelligent Tutoring and Games*

KC *Knowlegde Component*

KT *Knowledge Tracing*

MSA Método de Sequenciamento Adaptativo

MSD Método de Sequenciamento por grau de Dificuldade

MSP Método de Sequenciamento do Professor

MSR Método de Sequenciamento Randômico

Nic.br Núcleo de Informação e Coordenação do Ponto BR

OA Objeto de Aprendizagem

PFA *Performance Factor Analysis*

SIAI Sequenciador Inteligente de Atividades na Internet

STI Sistema Tutor Inteligente

TRI Teoria da Resposta ao Item

VPS *Vygotski Policy Sequencer*

WYSIWYG *What You See Is What You Get*

ZPDES *Zone of Proximal Development and Empirical Success*

LISTA DE FIGURAS

LISTA DE TABELAS

RESUMO

A perícia do aprendiz geralmente é desenvolvida através da resolução de exercícios que requerem um conjunto de habilidades avaliadas, tanto no sistema educacional de sala de aula convencional quanto em sistemas de apredizagem baseados em computador tais como Sistemas Tutores Inteligentes. Esta pesquisa propôs uma fórmula de *rating* para avaliação automática do desempenho do aluno, partindo-se do princípio de que o grau de dificuldade das questões pode ser medido pela taxa de alunos que as acertam/erram, sendo essa informação usada no cálculo de sua nota. Neste trabalho, os aspectos motivacionais em aprendizagem são considerados relevantes, sendo importante propor atividades adequadas ao nível da expertise do estudante, pois a apresentação de exercícios com grau de dificuldade muito abaixo (ou acima) do nível cognitivo do aprendiz pode causar entediamento (ou frustração), ocasionando o abandono da atividade proposta. Nesse sentido, este estudo também desenvolveu um algoritmo de sequenciamento adaptivo de exercícios que se baseia nos graus de dificuldade das questões, em que o sequenciamento é guiado pela performance dinâmica do aprendiz. Foi realizado um estudo empírico a partir de dados coletados de alunos reais que demonstrou a validade da fórmula de *rating*. Os algoritmos para cálculo do grau de dificuldades das questões e dos *ratings* dos alunos, bem como o algoritmo de sequenciamento adaptivo foram implementados efetuando-se alterações na ferramenta *web* de autoria de objetos de aprendizagem FARMA, gerando assim o ambiente ADAPTFARMA. Também foi realizada uma avaliação experimental da aprendizagem através de experimento estatístico comparativo entre diferentes modalidades de sequenciamento de exercícios usando como base um objeto de aprendizagem construído em ADAPTFARMA.

Palavras-chave: calibragem de exercícios, *rating*, Sistemas Tutores Inteligentes.

ABSTRACT

The expertise of learners is usually developed by solving problems that require a set of assessed skills. This is done in both conventional education schools and by applying advanced learning technologies, such as Intelligent Tutoring Systems. This research proposed a formula for automatic assessment of students, assuming that the degree of difficulty of the questions can be measured by counting the students that are successful and those who failed. This information is used to calculate their grade as a particular rating scale. Besides, the motivational aspects of learning are considered in depth. In this sense, it is important to propose activities according to the student's level of expertise, which is achieved through presenting students with exercises that are compatible with the difficulty degree of their cognitive skills. In doing so, both boredom and frustation can be avoided, as much as can be the withdrawal of the proposed activities on the part of students. An empirical study based on existing students data partially influenced the development of the first version of an adaptive algorithm for exercises sequencing, based on the difficulty degree of the questions. The sequencing process is guided by the learner's dynamic performance. An experiment has also been carried out with four maths classes of a local public school. Data collected from students' performance gains demonstrated the suitability of the *rating* formula. The algorithms for calculating and matching the difficulty degree of the questions and the students' *rating* were implemented by extending the *web* authoring tool of learning objects named FARMA, thus generating the ADAPTFARMA environment. Finally, conclusions and future research directions are described.

Keywords: exercises calibration, rating, Intelligent Tutoring Systems.

CAPÍTULO 1

INTRODUÇÃO

Este capítulo traz uma descrição do problema central e de sua relevância para a área de Informática na Educação, que lida com ferramentas inteligentes como elemento de suporte no mundo digital. É feita também uma exposição dos objetivos do trabalho de pesquisa de tese.

1.1 O problema central

Em pesquisa recente (ano base: 2013) [5] feita pelo Centro Regional de Estudos para o Desenvolvimento da Sociedade da Informação (Cetic.br), um departamento do Nic.br (Núcleo de Informação e Coordenação do Ponto BR), que implementa as decisões e projetos do Comitê Gestor da Internet do Brasil (Cgi.br), mostrou que quase todas as escolas públicas localizadas em áreas urbanas do país possuem computador (99%). O acesso à Internet já está presente em praticamente todas as escolas públicas com computador (95%). Entre os alunos que têm *tablet*, a proporção dos que levam o equipamento para a escola subiu de 9% para 25% em 2013. A pesquisa também mostrou que o computador está presente, em média, em 72% dos domicílios de alunos do último ano do Ensino Fundamental e, a proporção de alunos das escolas particulares que têm Internet em casa é de 93% enquanto dentre os alunos das escolas públicas é de 61%. Embora ainda existam deficiências quanto à infraestrutura de redes e largura de banda, esses dados mostram o grande potencial que se tem de abrangência da população estudantil do país em relação ao uso de computadores para fins educacionais.

Ao longo dos anos, a pesquisa sobre a construção e uso de softwares educativos se diversificou bastante, contemplando diferentes abordagens e objetivos, mas não necessariamente excludentes, tais como:

- **Objetos de Aprendizagem (OAs)**;

- **Sistemas de** *e-learning*;
- **Sistemas Tutores Inteligentes (STIs)**;
- **Jogos Educacionais ou** *serious games*.

Enquanto os OAs são auto-contidos em termos de conteúdo e têm como característica a busca pela reusabilidade/interoperabilidade [62], os STIs procuram formas de se adaptar às necessidades individuais do estudante [15]. Na abordagem de jogos educacionais procura-se ensinar ou reforçar determinado assunto exercitando alguma habilidade, valendo-se de características comuns a jogos como motivação e interação [6, 51].

De acordo com Anderson [2] uma das motivações das pesquisas em STIs são as evidências que mostram que o tutor humano particular é muito eficaz. As decisões didáticas, ao invés de serem codificadas de maneira fixa, podem ser derivadas das interações de regras especializadas ou estruturas de conhecimento similar para representar a perícia pedagógica do sistema [61], sendo que representações explícitas de conhecimento pedagógico criam o potencial para sistemas se adaptarem e melhorarem suas estratégias ao longo do tempo, e para componentes serem reusados em outros domínios.

Nwana [42] aponta duas razões principais porque os pesquisadores se interessam por STIs:

- necessidades de pesquisa: entender mais sobre os processos que contribuem para a interação educacional. Desde que STI está na interseção das principais disciplinas envolvidas, ele provê uma excelente base de teste para as várias teorias da Psicologia Cognitiva, IA e teoristas educacionais;
- necessidades práticas: razões econômicas e sociais, em virtude da possibilidade de tutoria um para um.

Vanlehn [60] observa que embora os STIs difiram bastante em domínios, interfaces de usuário, estruturas de software e bases de conhecimento, seus comportamentos na realidade são bastante similares, podendo ser descritos como tendo 2 laços ou *loops*: um *loop* externo, executado uma vez para cada tarefa, no qual uma tarefa geralmente consiste

em resolver um problema complexo de vários passos; e um *loop* interno, executado uma vez para cada passo tomado pelo estudante na construção da solução de uma tarefa. No *loop* interno, a competência do estudante pode ser avaliada e atualizado o modelo do estudante, que é usado pelo *loop* externo na escolha da próxima tarefa apropriada ao estudante.

Além disso, considerando os aspectos motivacionais em aprendizagem, é importante que o STI seja capaz de sugerir atividades adequadas ao nível da expertise do estudante. Isso porque a apresentação de exercícios com grau de dificuldade muito abaixo do nível cognitivo do aprendiz pode causar entediamento enquanto exercícios muito acima podem gerar desmotivação [46]. Ambas as situações podem gerar o abandono da atividade proposta.

Sendo assim, cabe ressaltar aqui a hipótese principal de pesquisa, que pode ser descrita tal como segue: é possível criar meios automáticos de avaliação do aluno e do grau de dificuldades das questões, que ao mesmo tempo servem de base para sequenciar tarefas e adequar o seu nível de dificuldade ao desempenho do estudante através de técnicas de Inteligência Artificial (IA).

1.2 Relevância do Problema

Na literatura, os trabalhos encontrados sobre sequenciamento automático de atividades, a granularidade do sequenciamento é no nível de objetos de aprendizagem (OAs) e/ou a validação do método de sequenciamento foi feita apenas com estudantes simulados. Busca-se com este estudo a proposição de técnicas de sequenciamento efetivas para o nível de exercícios tanto em STIs como OAs.

O sequenciamento automático de exercícios levando em conta o nível de expertise do aluno é relevante pelas seguintes razões:

- permite maior grau de adaptabilidade dos STIs e OAs, principalmente no que se refere à adaptação do grau de dificuldade à capacidade do estudante;
- permite a construção de ferramentas de autoria com maior usabilidade, através da

eliminação da necessidade do autor especificar as ordenações possíveis dos exercícios;

- diminui as horas de autoria necessárias à construção de STIs, reduzindo assim os seus custos.

Além disso, a correção automática dos exercícios aliada ao sequenciamento adequado pode contribuir na ampliação do potencial de avaliação formativa de ambientes de aprendizagem baseados em computador.

1.3 Contexto da Pesquisa

Dentro do âmbito do conteúdo da pesquisa, este estudo situa-se na área de IA aplicada à Educação, mais especificamente na área de Sistemas Tutores Inteligentes e Objetos de Aprendizagem. Ele compreende técnicas de avaliação automática do desempenho do estudante, estratégias tutoriais relacionadas à adaptação do sistema ao nível de perícia do aluno.

No âmbito institucional, este trabalho tem estreita relação com o projeto CONDIGITAL (Edital 001/07 MEC/MCT) do consórcio do Estado do Paraná, que envolveu o C3SL (Centro de Computação Científica e Software Livre) e o Instituto de Tecnologia para o Desenvolvimento (LACTEC) na Universidade Federal do Paraná (UFPR), o então Centro de Excelência em Tecnologia Educacional do Paraná (CETEPAR) e a Universidade Estadual de Londrina (UEL). Ele teve como objetivo principal contribuir para a melhoria e a modernização dos processos de ensino e aprendizagem da área de Matemática na rede de escolas públicas (e mesmo privadas). Após a finalização do projeto CONDIGITAL, foi desenvolvida a FARMA (Ferrameta de Autoria para a Remediação de erros com Mobilidade na Aprendizagem) [33], que serviu de base para implementação e testes das soluções propostas neste trabalho.

1.4 Objetivos

O objetivo geral deste estudo é propor técnicas para melhorar a adaptabilidade de STIs no que se refere ao sequenciamento de exercícios.

Objetivos específicos:

- pesquisar e propor técnicas para sequenciamento automático de exercícios em ordem crescente de dificuldade para STIs;
- propor um algoritmo para ordenação automática crescente em dificuldade de exercícios para promover adaptatividade dinâmica de STIs (sem a necessidade de interferência direta do professor);
- definir mecanismos de análise do grau de dificuldade de questões;
- definir uma fórmula para avaliação automática do desempenho do aluno na forma de *rating*;
- estender o ambiente sócio-interacionista FARMA [35], que será apresentada no Capítulo 3, para contemplar recursos de aprendizagem adaptativa que realizam alteração dinâmica da ordem (e até mesmo da quantidade) de exercícios de um aluno, individualmente;
- avaliar o desempenho da aprendizagem de alunos em turmas reais de escolas de ensino básico utilizando técnicas estatísticas de medição.

1.5 Organização da Tese

Este documento está organizado da seguinte forma: no Capítulo 2 é apresentada a resenha literária sobre os principais conceitos de STIs, principais técnicas de construção do modelo do estudante, formas de personalização dos STIs, diferentes abordagens para avaliação do estudante, métodos de calibragem e estratégias de sequenciamento de tarefas em STIs. No Capítulo 3 são abordados os conceitos fundamentais da solução proposta e algoritmos associados. No Capítulo 4 são mostrados os resultados de um estudo empírico realizado visando uma pré-validação da solução adotada para avaliação automática do aluno. No Capítulo 5 são apresentados a metodologia e análise dos resultados de um experimento estatístico realizado para avaliar as soluções propostas no Capítulo 3. E por último, no

Capítulo 6 são feitas a considerações finais e propostos trabalhos futuros do presente estudo.

CAPÍTULO 2

RESENHA LITERÁRIA

O objetivo deste capítulo é apresentar uma discussão dos principais pontos relacionados à modelagem e construção de STIs e avaliação do aluno.

2.1 Sistemas Tutores Inteligentes

A pesquisa e desenvolvimento de Sistemas Tutores Inteligentes (STIs) tem procurado unir técnicas de IA, Psicologia Cognitiva e teorias de aprendizagem educacional com o ojetivo de criar sistemas educacionais capazes de saber *o que* ensina, *para quem* ensina e *como* ensina [42]. A figura 2.1 mostra a relação entre essas áreas e a constituição de STIs.

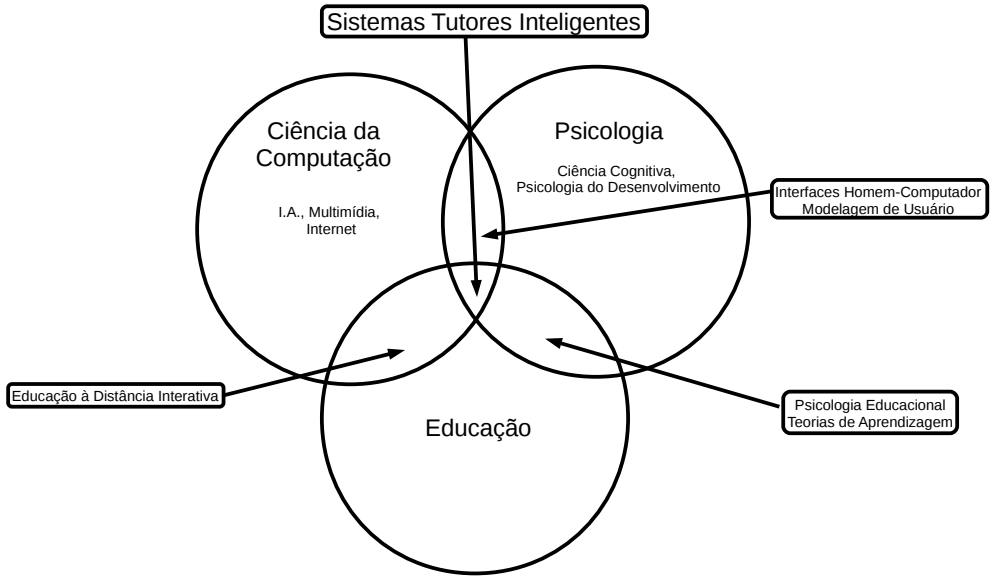


Figura 2.1: Constituição de um STI (adaptado de Woolf [63])

Tradicionalmente, um STI é definido como sendo composto de 4 módulos [61], con-

forme mostrado na figura 2.2:

- **Especialista (perito)**: cobre os fatos e regras do domínio particular a ser ensinado.
- **Estudante (aprendiz)**: representa dinamicamente o conhecimento e a capacidade do estudante.
- **Tutor (pedagógico)**: decide como o conteúdo deve ser apresentado ao estudante (tática pedagógica ou didática), adaptando-se à medida que o estudante interage com o sistema.
- **Interface**: é o componente de comunicação que controla a interação entre o estudante e o sistema.

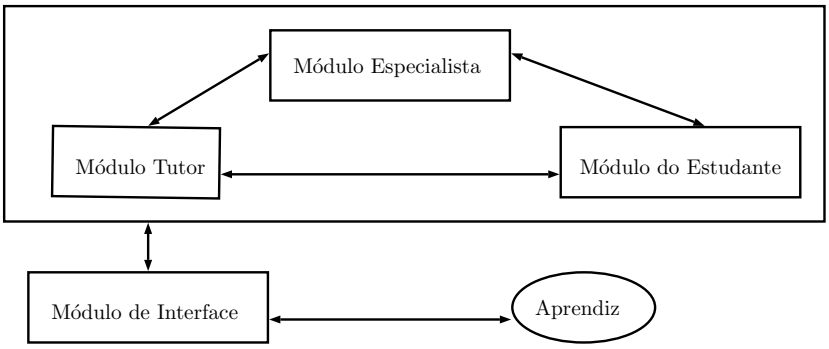


Figura 2.2: Arquitetura clássica de um STI (adaptado de Nkambou et al. [41])

O modelo do especialista, ou também chamado modelo do domínio pode cobrir diferentes papéis como: avaliar a performance do estudante ou detectar seus erros. Algumas vezes ele está organizado dentro de um currículo ou uma estrutura incluindo todos os elementos do conhecimento ligados de acordo com sequências pedagógicas. Cada unidade de conhecimento pode ser mais ou menos detalhada e o currículo pode ser organizado em um modelo dinâmico, de acordo com várias estruturas tais como: hierarquias, redes semânticas, *frames*, ontologias e regras de produção [41].

O modelo do estudante é considerado o componente central de um STI. Idealmente, deve conter tanto quanto possível todo o conhecimento sobre os estados cognitivo e afetivo do estudante e sua evolução à medida que o processo de aprendizagem avança. Geral-

mente, o modelo do estudante é visto como um modelo dinâmico que implementa várias funções. Wenger [61] atribuiu 3 funções principais ao modelo do estudante:

1. coletar dados implícitos e explícitos do estudante;
2. usar esses dados coletados para criar a representação do conhecimento do estudante e do processo de aprendizagem;
3. realizar algum tipo de diagnóstico levando em conta os dados coletados, seja do estado do conhecimento do estudante ou em termos de selecionar estratégias pedagógicas mais favoráveis para apresentação da informação de domínio subsequente para o estudante.

Já Self [56] identificou 6 papéis principais para o modelo do estudante:

1. **corretivo**: erradicar erros (*bugs*) no conhecimento do estudante;
2. **elaborativo**: corrigir o conhecimento incompleto do estudante;
3. **estratégico**: iniciar mudanças significativas na estratégia tutorial;
4. **diagnóstico**: diagnosticar erros ou conceitos equivocados no conhecimento do estudante;
5. **preditivo**: determinar a provável resposta do aluno às ações tutoriais;
6. **avaliativo**: avaliar o estudante ou o próprio STI.

Essas funções e papéis do modelo do estudante têm sido explorados ao longo dos anos da pesquisa na área de STIs.

O módulo tutor recebe entradas dos módulos de domínio e estudante e faz decisões sobre estratégias e ações pedagógicas. É responsável por tomar a decisão de intervir ou não na interação do estudante com o sistema, e em caso afirmativo, quando e como. O planejamento de conteúdo também é parte das funções do modelo tutor.

Decisões de tutoria idealmente deveriam ser refletidas em diferentes formas de interação com o estudante: diálogos socráticos, sugestões, *feedback* do sistema, etc. As

interações tutor/estudante ocorrem através do módulo de interface ou comunicação. Este componente dá acesso aos elementos do conhecimento do domínio através de múltiplas formas de ambientes de aprendizagem, incluindo simulações, hipermídia, micromundos, etc [41].

Nwana [42] aponta uma forte ligação entre a arquitetura e o paradigma pedagógico em STIs, mostrando que filosofias pedagógicas diferentes enfatizam módulos diferentes: domínio, estudante ou tutor. O projeto arquitetural de um STI em geral reflete essa ênfase, o que leva a uma variedade de arquiteturas, mas nenhuma delas, individualmente, capaz de suportar todas as estratégias de tutoria.

Além da visão clássica de STIs, os sistemas costumam oferecer serviços cuja implementação geralmente transcende os componentes individuais. Por exemplo, o serviço de diagnóstico é algumas vezes considerado parte do modelo do estudante, mas muitos pesquisadores o veem como uma função do componente tutor.

2.1.1 Modelagem do Estudante

Ao longo dos anos, a modelagem do estudante tem sido alvo de intensa pesquisa, abrangendo variados aspectos, tais como [21]:

- **modelagem de habilidades** (*skills*): como representar as diferentes habilidades do aprendiz?
- **emoções**: como detectar o estado emocional e o nível de atenção do estudante enquanto utiliza o sistema? Como avaliar e manter o engajamento do estudante dentro da atividade proposta?
- **metacognição e aprendizagem auto-regulada**: como levar o estudante a organizar os próprios processos cognitivos visando a realização da atividade proposta? Como fazer com que o estudante seja capaz de avaliar seu próprio conhecimento?
- **modelagem de longo prazo**: como ter interoperabilidade entre diferentes sistemas, tal que as informações de um modelo possam ser compartilhadas ou usadas em outros?

Em muitas visões idealizadas da área de STI, como na de Self [57], a criação de modelos do estudante mais precisos, entre outros benefícios, permite que a prática designada ao estudante seja baseada na avaliação de seu conhecimento, atendendo suas necessidades individuais. Além disso, modelos do estudante se tornaram um componente chave em STIs, incluindo fatores como detecção da ajuda apropriada e/ou de situações de comportamento fora da tarefa proposta [4]. As abordagens modernas sobre o assunto de deslocamento de um indivíduo em relação ao comportamento esperado parecem ser cada vez mais influenciadas por dados produzidos por outros aprendizes. Nesse sentido, quanto maior for o universo de amostragem de aprendizes, mais precisas devem se tornar as avaliações automáticas de um indivíduo.

Há várias técnicas para construção e/ou estruturação do modelo do estudante, que podem ser encontradas na literatura, entre as quais se destacam: regras de produção, redes bayesianas, métodos *fuzzy*, ontologias, algoritmos de aprendizagem de máquina (*machine learning*) e mineração de dados (*data mining*).

Uma rede bayesiana é um grafo orientado no qual cada nodo tem associado uma informação de probabilidade quantitativa. A topologia de uma rede especifica os relacionamentos de independência condicionada que ocorrem em algum domínio [49].

Redes Bayesianas têm sido usadas na modelagem de habilidades (*skills*) devido às seguintes características [21]:

- possuem uma representação gráfica intuitiva;
- é um formalismo que calcula probabilidades de nodos não observáveis a partir de evidências de nodos observáveis;
- possuem grande flexibilidade (qualquer nodo considerado sob uma nova evidência permite a propagação do seu efeito na forma de atualização das probabilidades dos demais);
- podem ser potencialmente derivadas de dados, reduzindo substancialmente a necessidade de engenharia de conhecimento;

- os cálculos de probabilidades podem ser feitos utilizando pacotes de software padrão;
- sua expressividade permite unificar em um mesmo *framework* conceitos corretos e equívocos do aluno.

No entanto, redes bayesianas podem conter um grande número de nodos escondidos (não observáveis), tornando-as extremamente complexas e, com isso, elevando sobremaneira o custo computacional do cálculo das probabilidades.

Um exemplo de STI que utiliza rede bayesiana é PAT2Math, sistema para o domínio de álgebra [53], no qual a rede é usada na implementação do modelo do estudante com o propósito de identificação do conhecimento do aluno em equações de 1^{o} grau. A rede, construída a partir de um mapa conceitual de operações de 1^{o} grau (soma, subtração, multiplicação, divisão, Mínimo Múltiplo Comum, Propriedade Distributiva, Princípio Aditivo e Princípio Multiplicativo), tem nodos representando as operações, as equações (exercícios) e os passos necessários para resolvê-las. Além das probabilidades de acerto e erro, os passos da equação possuem fatores de adivinhação (*guess*) e falta de atenção (*slip*). Um experimento prático conduzido com 24 alunos evidenciou que a rede proporciona resultado semelhante às conclusões de um especialista humano.

Outra técnica para tratamento probabilístico é a utilizada em shells de sistemas especialistas, tal como e-MYCIN. Nele, regras de produção clássicas de encadeamento regressivo são usadas para representação do conhecimento, estendidas com um modelo de implicação aproximado por meio de fatores numéricos que integram as noções de certeza e exatidão. Nessa abordagem, evidências são coletadas de forma interativa com o usuário, separadamente da confirmação ou não da verdade das hipóteses possíveis para um diagnóstico. Só depois disso é que a decisão final da hipótese mais provável ocorre com base na soma algébrica das evidências atualizadas com dados a favor e contrários a cada hipótese possível. Em outras palavras, a existência simultânea de evidências a favor e contra a mesma hipótese parece ser uma característica importante de qualquer raciocínio com fatores potencialmente inexatos ou incertos [17].

A construção de modelos do estudante pode ser feita a partir de técnicas de engenharia do conhecimento como entrevistas estruturadas ou de maneira automatizada como

técnicas de aprendizagem de máquina. Por um lado, as primeiras são bastante subjetivas e dependem da codificação de natureza dedutiva de expertise do domínio. Em geral, isso demanda muito tempo e se torna sujeito à omissão de importantes componentes do conhecimento. Já as técnicas automatizadas frequentemente apresentam o desafio de interpretação dos modelos gerados por algum método de natureza indutiva [30]. Um aspecto que reforça a vantagem de métodos automáticos é o fato de que muito da expertise humana se enquadra na categoria de conhecimento tácito. Por exemplo, em aprendizagem de línguas, os falantes nativos podem escolher corretamente as preposições, mas podem não saber porque as escolheram.

Dentre as diversas formas de representação do conhecimento em STIs, duas abordagens têm sido bastante difundidas [21]: tutores cognitivos (CT), *Cognitive Tutor*, do inglês, e modelagem baseada em restrições (CBM), *Constraint Based Modeling*, do inglês. Os tutores cognitivos representam conhecimentos procedurais mapeados em ações do estudante. Tutores baseados em restrições representam conhecimentos declarativos como restrições às respostas do estudante ou ao resultado das ações do estudante. Apesar das diferenças entre as 2 abordagens, um estudo comparativo [39] demonstrou que ambas atingem resultados similares em termos de aprendizagem humana.

Em tutores cognitivos, usualmente o modelo do estudante, denominado KT (*Knowledge Tracing*), é baseado em uma rede bayesiana dinâmica de 2 estados onde o desempenho do estudante é a variável observada e o conhecimento do estudante é a variável latente. Nesse modelo, há 2 parâmetros de desempenho: *slip* e *guess*, que se situam entre a performance do estudante e o conhecimento do estudante. O parâmetro *guess* representa o fato de que o estudante pode responder corretamente uma questão embora não domine a habilidade corretamente. O parâmetro *slip* reconhece que o estudante mesmo dominando corretamente uma habilidade, pode eventualmente cometer um erro por falta de atenção ou deslize. Além disso, há também 2 parâmetros relacionados à aprendizagem: o conhecimento inicial representado pela probabilidade de o estudante já ter a habilidade na primeira vez que utiliza o tutor, e a taxa de aprendizagem representada pela probabilidade do estudante adquirir a habilidade como resultado após uma oportunidade de

praticá-la.

Desde seu aparecimento, tem surgido diferentes variantes do KT, ou diferentes maneiras de construí-lo. Gong *et al.* [25] efetuaram um estudo comparativo entre diversas abordagens para montagem do modelo KT e estimativa de seus parâmetros. O estudo mostrou não haver diferença estatística significativa na acurácia preditiva entre a técnica de Análise de Fator de Performance (PFA - *Performance Factor Analysis* do inglês) e versões de KT que lidam com questões de múltiplas habilidades. A técnica PFA é uma abordagem competitiva que prevê a performance do estudante baseada na dificuldade do item e performances históricas do estudante. Nesse modelo é estimado para cada item um parâmetro representando sua dificuldade e também 2 parâmetros para cada habilidade (*skill*) refletindo o efeito de sucessos prévios e falhas prévias atingidas por aquela habilidade.

No que se refere à modelagem de habilidades, destacam-se as seguintes vertentes principais:

- considerar que cada exercício ou tarefa refere-se ao domínio de uma única habilidade. Essa vertente é a adotada pelo modelo padrão do *framework* KT.
- considerar que cada exercício ou tarefa pode requer o domínio de múltiplas habilidades. Nessa vertente, têm sido estudadas 2 abordagens:
  - o estudante deve dominar todas as habilidades requeridas no exercício para resolvê-lo corretamente.
  - a resolução correta ou incorreta do exercício é dominada pela habilidade mais fraca do estudante.

No caso do tratamento de múltiplas habilidades, uma questão pertinente é: se para um dado exercício ou tarefa, todas as habilidades requeridas são independentes ou se alguma habilidade influi na aquisição de outra habilidade. Com o objetivo de se evitar uma complexificação excessiva do modelo do estudante, em geral os sistemas existentes consideram as habilidades como independentes.

Cen *et al.* [11] e Gong *et al.* [25] propõem a utilização de modelos conjuntivos que assumem que o estudante deve dominar todas as habilidades requeridas em uma tarefa para realizá-la corretamente. Assim, seguindo a teoria de probabilidade clássica, a estimativa da probabilidade de realização correta da tarefa é feita pela multiplicação das probabilidades de que o estudante domine as habilidades requeridas.

Xu e Mostow [64] criaram um modelo de aquisição de habilidades denominado LR-DBN em que as probabilidades de transição entre estados de conhecimento sucessivos são calculadas usando regressão logística em todas as sub-habilidades. A regressão logística é uma técnica estatística que permite estabelecer relação entre uma variável resposta do tipo categórica e variáveis explicativas contínuas e/ou binárias. A utilização de regressão logística para modelagem de múltiplas habilidades visa evitar a subestimativa da probabilidade de acerto que ocorre geralmente em abordagens conjuntivas apresentadas anteriormente.

A Teoria da Resposta ao Item (TRI) é um modelo largamente estudado em psicometria, que assume que o sucesso de todos itens em um teste é determinado por uma única habilidade [3]. Essa habilidade, denotada por θ, é referida como sendo o traço latente. Além disso, também são importantes os parâmetros discriminação (a_i) e dificuldade do item (b_i) para determinar a probabilidade de sucesso em um único item, calculado por:

$$P(X_i|\theta) = \frac{1}{1 + e^{a_i(\theta - b_i)}}$$

.

A TRI estabelece que as chances de sucesso em uma tarefa (ou item) aumentam em função do nível de perícia em uma habilidade não observada (traço latente). Assim como um modelo de habilidades do estudante, a TRI pode ser considerada como um modelo de transferência, em que há um único nodo escondido (o traço latente) que está ligado a todos resultados observáveis, correspondendo ao sucesso ou à falha nas tarefas propostas. Portanto, qualquer observação de um resultado conduz à transferência de evidência a todos os outros itens através do traço latente.

O modelo básico da TRI leva em consideração uma única habilidade. Modelos da TRI que incluem uma ou mais dimensões, θ torna-se um vetor de traços latentes e o parâmetro a torna-se vetor de discriminação, ambos um por dimensão. Essas considerações aumentam consideravelmente a complexidade, o que explica a baixa utilização de modelos multidimensionais da TRI em ambientes de aprendizagem personalizada [21].

Enquanto vários STIs procuram meios para detecção e representação de erros e concepções equivocadas dos alunos, atualmente também vem crescendo o interesse de pesquisadores pela utilização direta de erros como fonte de material pedagógico, ao modo da técnica de exemplos errôneos, que consiste em uma descrição de como resolver um problema na qual um ou mais passos estão incorretos, sendo solicitado ao estudante encontrá-lo(s), explicá-lo(s) e/ou corrigi-lo(s) com o objetivo de aprender mais profundamente sobre o conteúdo do domínio e desenvolver habilidades metacognitivas. Isotani et al. [28] [36] realizaram um estudo sobre a aplicação da técnica de exemplos errôneos interativos no domínio matemático de decimais utilizando uma rede bayesiana para modelagem dos equívocos mais comuns do domínio, e encontraram evidências da técnica ser útil para a aprendizagem, especialmente quando o estudante tem tempo e oportunidades adequadas para refletir.

Similarmente às questões de adaptabilidade de STIs que usam exercícios convencionais, a utilização de exemplos errôneos apresenta o desafio de determinar quando, para quais estudantes e o tipo dos exemplos a serem apresentados.

2.1.2 Personalização em STIs

Um dos desafios na construção de STIs é torná-los capazes de se adequar às necessidades individuais de aprendizagem do estudante. Assim, a personalização pode ir desde simples preferências de interface até a detecção de estados emocionais do estudante, podendo afetar a estruturação de todos os módulos do STI.

Uma das formas mais comuns de personalização em STIs é o emprego de suportes (*scaffolding* do inglês) adaptivos. Segedy *et al.* [55] propuseram uma taxonomia para suportes adaptivos em sistemas de aprendizagem baseados em computador chamada

Suggest-Assert-Modify. Suportes do tipo sugestão proveem informação aos estudantes com o propósito de engajá-los em um comportamento específico. Suportes de asserção comunicam uma informação como sendo verdadeira que deve ser integrada com o seu atual entendimento. E suportes do tipo modificação altera aspectos da própria tarefa de aprendizagem.

No contexto da aprendizagem, é relevante considerar os diferentes níveis de conhecimento prévio e expertise dos estudantes, bem como suas habilidades cognitivas, estilos de aprendizagem e motivação, que em conjunto traduzem suas necessidades individuais. Nesse sentido, técnicas usadas em sistemas de recomendação também têm sido empregadas em STIs e em sistemas de *e-learning* para fins de adaptação às necessidades individuais do estudante [23, 10]. Existe um mapeamento similar entre modelagem do estudante em STIs e sistemas de recomendação onde estudante, tarefa (materiais), e performance (nota, pontuação) se torna usuário, item e *rating*, respectivamente [59].

Um exemplo de uso de técnicas de sistemas de recomendação é o sistema tutor de programação Protus [29], que faz uso de ontologias do domínio assim como de regras de sintaxe para personalizar dinamicamente o material de apoio ao aprendiz. Protus adota o modelo de estilos de aprendizagem Felder-Silverman, que categoriza cada aprendiz de acordo com 4 dimensões:

- processamento da informação (ativo/reflexivo).
- percepção da informação (sensorial/intuitivo).
- recepção da informação (visual/verbal).
- compreensão da informação (sequencial/global).

O sistema possui um módulo de recomendação, cuja finalidade é facilitar a adaptação automática do sistema tutor aos interesses e níveis de aprendizagem dos estudantes. O sistema divide o processo de adaptação em 3 etapas. Primeiro os estudantes são agrupados em *clusters* utilizando-se técnicas de mineração de dados de acordo com seu estilo de aprendizagem. Depois, padrões comportamentais são descobertos para cada estudante

usando um algoritmo de mineração de logs Web. Em seguida, uma lista de recomendação é criada de acordo com as necessidades detectadas do estudante.

Segal *et al.* [54] propõem um algoritmo denominado EduRank para personalizar conteúdo educacional para estudantes em sistemas *e-learning*, que combina algoritmos de filtragem colaborativa com a Teoria da Escolha Social [31]. O algoritmo constroi um *ranking* de dificuldade sobre questões para um estudante destino agregando o *ranking* de estudantes similares, medido pelos aspectos diferentes de suas performances em questões comuns passadas, tais como série, número de tentativas e tempo gasto na solução de questões.

Thai-Nghe [59] fez um amplo estudo sobre predição de perfomance do estudante. Especificamente, a previsão da performance tem por objetivo saber como os estudantes aprendem (em geral ou estritamente), o quão rapidamente ou lentamente se adaptam a novos problemas, ou ainda inferir os requisitos de conhecimento para solucionar os problemas, obtidos diretamente dos dados da performance do estudante. Além disso, eventualmente o sistema tenta descobrir se os estudantes realizariam as tarefas (problemas, exercícios) corretamente com algum nível de certeza, quando interagindo com os sistemas tutores. Outro benefício da previsão de performance é permitir que os instrutores possam auxiliar o estudante provendo *feedbacks* antecipadamente.

Desarkar and Sarkar [20] propõem um algoritmo de predição de *rating* chamado - PrefNMF-RP, que considera os *ratings* relativos dados pelos usuários para diferentes pares de itens. O algoritmo modela os usuários e itens usando um *framework* de fatoração de matriz. O modelo de usuários e itens aprendido é usado primeiro para prever a utilidade personalizada de um item para um usuário. Essa utilidade é então convertida para um valor de *rating* válido dentro de uma escala pré-definida empregando uma classificação personalizada. O algoritmo apresentou melhor performance que diferentes métodos baseados em filtragem colaborativa e fatoração de matriz.

Rioja *et al.* [47] desenvolveram um STI baseado em exercícios parametrizados e hierarquicamente interconectados. O sequenciamento dos exercícios é representado através de um grafo criado por tutor humano tal que as arestas representam transições entre

exercícios e são rotuladas com expressões booleanas contendo condições sobre a nota do estudante e número de tentativas. Embora o uso de exercícios parametrizados permitam instanciação automática, o grafo de sequenciamento entre eles com as restrições de transições devem ser configuradas por um especialista do domínio.

Um exercício parametrizado é um exercício que pode ser instanciado várias vezes com dados diferentes. Por exemplo, em álgebra, o problema de calcular as raízes de um polinômino do segundo grau poderia ser tratado como um exercício de 2 parâmetros x_1 e x_2 representando as raízes. Em uma primeira versão, seria apresentado o polinômino na forma $ax^2 + bx + c$ e seria solicitado ao estudante calcular as raízes. Em outra versão, poderiam ser fornecidas as raízes e solicitar qual o polinômio correspondente. Em ambas versões, em cada instanciação os valores dos parâmetros poderiam ser diferentes. O uso de exercícios parametrizados pode ser útil em situações em que se deseja certificar que o estudante assimilou determinado conceito, solicitando o mesmo tipo de exercício ou suas variações mais de uma vez, porém com dados diferentes. Além disso, o uso de diferentes versões do mesmo exercício pode ser considerado uma forma de personalização.

McNamara *et al.* [37] discutem uma área adicional de pesquisa chamada Tutoria Inteligente e Jogos (ITaG - *Intelligent Tutoring and Games* do inglês), que incorpora princípios de jogos em STIs e vice-versa. Essa área de pesquisa seria uma fusão dos princípios de aprendizagem e motivação para criar jogos aprimorados com princípios de aprendizagem, e STIs aprimorados com princípios de jogos. Alguns exemplos de funcionalidades baseadas em jogos são competição, pontuação ou níveis e dificuldade de tarefas. É esperado que essas e outras funcionalidades, ao sererem incluídas em STIs, possam aumentar os fatores motivacionais do estudante.

2.2 Avaliação do Aprendiz em STIs

Bloom *et. al.* [8] destacam 3 tipos de avaliação no processo de ensino-aprendizagem:

- **avaliação diagnóstica**: visa diagnosticar o domínio dos objetivos previstos e necessários para se iniciar uma atividade de ensino.

- **avaliação somativa**: descrição e classificação do aluno ao final de uma unidade, semestre ou ano, expresso em graus ou conceitos.

- **avaliação formativa**: ocorre durante o processo de ensino-aprendizagem, fornecendo *feedback* a professores e alunos acerca do que o aluno aprendeu, do que ainda precisa aprender, suas necessidades individuais e quais aspectos da instrução precisam ser modificados com o intuito de individualizar o atendimento e solucionar as falhas na aprendizagem.

Segundo Dutra *et. al.* [19], ainda hoje a avaliação formativa é pouco utilizada no contexto geral da Educação, mesmo sendo bastante pesquisada [58][40]. Sendo assim, supondo a predominância da utilização de avaliação somativa, o aumento da adaptabilidade de STIs pode ampliar o potencial de avaliação formativa, promovendo mudança qualitativa nas formas de se ensinar e se aprender na era digital.

Ferguson [24] aponta que não é possível observar o domínio de uma habilidade ou perícia do aluno diretamente, sendo este conhecimento normalmente inferido a partir das respostas dadas por ele a problemas que envolvam a utilização dessas habilidades.

Ao longo dos anos, têm surgido diferentes abordagens para avaliação do aluno, seja em ambientes *e-learning* ou STIs. As avaliações podem ser automáticas, parcialmente automáticas ou ainda feitas por um especialista ou tutor humano.

Pinkwart e Loll [45] apresentaram um estudo comparativo entre três abordagens de avaliação da qualidade das soluções dos estudantes, sendo que duas fazem uso de revisões por pares: a qualidade da solução de uma tarefa de um estudante é determinada heuristicamente por avaliações de outros estudantes. Na primeira abordagem, quando um estudante trabalha em uma tarefa e provê uma solução, depois é solicitado a ele avaliar algumas soluções alternativas. Uma avaliação terá um peso maior que outras quando o estudante que a faz tem um *rating* de qualidade mais alto quando comparado a outros. A segunda abordagem assume que um estudante que consegue classificar corretamente a qualidade de soluções alternativas, também está apto a prover uma solução de maior qualidade. Enquanto a primeira faz um uso clássico de revisão por pares com um valor base fixo para aquelas soluções que não foram avaliadas ainda, a segunda substitui este

valor base por um *rating* calculado dinamicamente. A terceira abordagem usa somente a fórmula de *rating* base. O processo de colaboração entre os usuários pode ser classificado como uma aplicação de Web Semântica Social, que tem o potencial de diminuir a carga de trabalho de tutores humanos e prover a possibilidade dos estudantes treinarem suas capacidades de criticidade.

Guzmán e Conejo [27] propõem um modelo de avaliação baseado na TRI para medir tanto o nível de conhecimento quanto os conceitos equivocados do estudante através de testes. O modelo se baseia na hipótese de que certas escolhas incorretas em questões de teste podem ser usadas para inferir os conceitos equivocados dos estudantes. Após a sessão de testes, o modelo do estudante é atualizado com informação a respeito do seu nível de conhecimento e também com dados sobre os conceitos equivocados assumidos por ele. O modelo de avaliação é composto por 3 submodelos:

- modelo do estudante, formado por uma camada de conceitos do domínio c_1, ..., c_N e uma camada de conceitos equivocados m_1,..., m_R;
- modelo de tarefas, consistindo de elementos através dos quais as evidências de conhecimento podem ser capturadas, isto é, exercícios, problemas ou questões de teste;
- modelo de evidência: que conecta os outros 2 modelos, e é usado para atualizar o modelo do estudante. O relacionamento entre questões e conceitos, questões e conceitos equivocados é modelado por curvas características da TRI.

Sales *et al.* [50] propõem um modelo de avaliação formativa para automatizar o acompanhamento qualitativo/quantitativo em ambientes virtuais de aprendizagem. O modelo faz uso de vetores na forma bidimensional e equações trigonométricas, em que os vetores-aprendizagem contém as avaliações em cada uma das atividades à distância do aluno tais como fóruns de discussão, tarefas, *wikis*, *chats* e também notas das atividades presenciais. A projeção horizontal do vetor expressa a nota naquela atividade e sua positividade enquanto a projeção vertical relaciona-se à negatividade do seu desempenho.

2.3 Calibragem de tarefas

Um aspecto importante na elaboração de tarefas para serem realizadas pelos estudantes é a sua calibragem, ou seja, a determinação do seu nível de dificuldade. Do mesmo modo que um tutor humano, é desejável que sistemas de aprendizagem por computador, incluindo STIs, sejam capazes de aferir o grau de dificuldades das tarefas que propõe.

Os pesquisadores Ravi e Sosnovsky propuseram um método para calibragem da dificuldade dos exercícios de STIs baseado na mineração de dados do *log* do estudante [46], o que apresentou um ganho considerável na precisão da predição do desempenho do estudante. O método consiste de 3 fases:

- filtragem e enriquecimento do *log* do estudante. Nessa fase, são excluidas do *log* informações desnecessárias ao algoritmo tais como *login/logout* do usuário e página apresentada/requisitada. Também é feito o enriquecimento do *log* incluindo-se informações sobre os conceitos subjacentes ao exercício alvo.

- estimação da habilidade do estudante. Utilizando o modelo clássico bayesiano KT em que a probabilidade de um estudante ter aprendido o conceito é calculada baseada na sequência de tentativas de resolução dos exercícios envolvendo o conceito. Tal probabilidade é usada como medida da habilidade do estudante em relação ao conceito, sendo essa informação acrescentada ao *log* de dados para obtenção do *log* enriquecido final.

- calibragem dos metadados dos exercícios. Nessa última fase, os eventos do *log* gerado são agrupados por exercícios, classificados de acordo com a habilidade do estudante e usados para se encaixar na curva sigmoide da TRI. O agrupamento dos eventos são feitos através do algoritmo de *clustering* padrão *k-means*.

Papoušek e Pelánek [43] pesquisaram a relação entre o engajamento do estudante e a dificuldade da tarefa no ensino de Geografia, e propuseram um mecanismo para o ajuste dinâmico da dificuldade. O estudo se baseia na Teoria do Fluxo e na hipótese do U invertido [1, 16], que estabelecem que as questões devem ter o nível de dificuldade

adequado, pois questões fáceis tendem a deixar os estudantes entediados e questões difíceis tendem a deixá-los frustados. Cada item é avaliado por 3 funções de pontuação, cada uma de acordo com um critério diferente, tal que o item com a soma mais alta é usado como candidato para ser perguntado. O aspecto da dificuldade é levado em conta no modelo do estudante, que usa uma combinação do sistema de *rating* Elo do jogo de xadrez e PFA. Cada item tem uma probabilidade estimada P_{est} que um estudante particular irá responder corretamente. A primeira função de pontuação depende da distância entre P_{est} e a taxa de sucesso P_{target}, dada por:

$$S_{prob}(P_{est}, P_{target}) = \begin{cases} \frac{P_{est}}{P_{target}} \, se \, P_{target} \geq P_{est} \\ \frac{1-P_{est}}{1-P_{target}} \, se \, P_{target} < P_{est} \end{cases}$$

A segunda função, baseada no tempo gasto desde a última pergunta, serve para penalizar os itens e é dada por: $S_{time}(t) = -1/t$, onde t é o tempo em segundos. A terceira função induz o sistem a pedir questões sobre novos itens, dada por: $S_{count}(n) = 1/\sqrt{1+n}$. A pontuação final é dado pela soma ponderada das pontuações individuais, sendo os pesos configurados manualmente a partir das experiências de uso do sistema ($W_{prob} = 1$, $W_{count} = 1$ e $W_{time} = 12$). Para avaliar a efetividade do algoritmo proposto, os autores compararam o *feedback* dos alunos com a sua performance, tendo obtido que a taxa média de sucesso não é afetada pelo ajuste, porém a experiência do estudante é diferente, já que 5% a mais consideraram a dificuldade apropriada quando o mecanismo está ligado.

Günel e Aiyslan [26] propõem um modelo para calcular dificuldade de questões em STIs usando a seguinte equação diferencial: $y(t) = \frac{y_0}{y_0+(1-y_0).e^{-kt}}$, onde $y_0 = 0.5$, $k = \frac{\beta-\alpha}{\alpha+\beta}$, sendo α e β o número de vezes que a questão foi respondida corretamente, e com falha, respectivamente. Os autores pontuam a necessidade de testar a fórmula com alunos reais para verificar sua efetividade.

2.4 Sequenciamento de tarefas

O sequenciamento de tarefas refere-se à ordem em que as diferentes atividades de aprendizagem (exercícios, leitura, interação com outros alunos) devem ser propostas aos alunos. É eminentemente uma atribuição do módulo tutor, mas tem estreita relação com a calibragem de tarefas e com a avaliação da performance do aprendiz. É um aspecto importante no desenvolvimento de sistemas educacionais, pois deve determinar o sequenciamento mais adequado possível do conteúdo a ser apresentado ao estudante visando uma aprendizagem mais efetiva.

Schatten and Schmidt-Thieme [52] apresentam o Vygotski Policy Sequencer (VPS), baseado no conceito Zona de Desenvolvimento Proximal proposto por Vygotski. Nessa abordagem, a fatoração de matriz , que é um método para previsão de *rating* do usuário, é combinada com uma política de sequenciamento. Isto é feito com o objetivo de selecionar a cada passo o conteúdo de acordo com o *score* previsto. A política de sequenciamento é tal que a definição explícita de dificuldade, como conceito, não é necessária. Além disso, é mostrado como usar estudantes simulados para testar as sequências antes delas serem aplicadas a estudantes reais. Os autores pontuam a necessidade de avaliar o método com estudantes reais para confirmar as suposições feitas sobre o processo de aprendizagem tal como a impossibilidade de se aprender a partir de um problema mais que as capacidades requeridas para solucioná-lo.

Clement *et al.* [14] propõem dois algoritmos para modelo tutor de STI. O primeiro, chamado RiARiT (*Right Activity at Right Time* do inglês), é baseado em técnicas usadas em problemas de alocação sequencial de recursos, conhecidas como *multi-arm bandits* na gíria americana da língua inglesa [9], tal que cada atividade envolve diferentes capacidades, referidas como componentes de conhecimento (do inglês *Knowledge Components* (KCs)). O modelo do estudante é uma generalização do modelo bayesiano KT. Enquanto no KT o nível de compentência do estudante (c_i) é representado por uma variável booleana, na abordagem KC ele é representado por um número real no intervalo [0..1]. Também há uma tabela que associa para cada atividade a o c_i de KC_i requerido para se obter sucesso, denotado por $q_i(a)$. Quando o estudante tem sucesso na atividade a, c_i está acima de $q_i(a)$,

caso contrário ele estará abaixo. Além disso, uma recompensa representando o progresso de aprendizagem é definida pela diferença entre $q_i(a)$ e c_i. O algoritmo inclui um conjunto de filtros que atuam como especialistas que monitoram o quanto de recompensa cada atividade está dando, e seleciona as atividades de ensino proporcionalmente ao progresso esperado de aprendizagem.

O segundo algoritmo, ZPDES (*Zone of Proximal Development and Empirical Success*) [14] é uma versão modificada de RiARiT, em que o cálculo da recompensa é mudado para remover a dependência do nível de competência estimado do estudante. A recompensa torna-se uma medida de como a taxa de sucesso está aumentando, provendo uma escolha mais preditiva de atividades. Uma limitação do estudo é a falta de validação com estudantes reais, sem fazer suposições a respeito de suas taxas de aprendizagem e níveis de compreensão.

Carvalho [18] apresenta o SIAI (Sequenciador Inteligente de Atividades na Internet), sistema desenvolvido para fazer acompanhamento pedagógico dos alunos e sequenciamento de atividades no nível de objetos de aprendizagem de acordo com o respectivo desempenho individidual. A modelagem do domínio é feita através de uma linguagem textual hierarquizada em que o conhecimento é estruturado em forma de árvore, enquanto as estratégias de ensino são codificadas em outra linguagem que referencia os elementos da primeira para fazer o sequenciamento indicado pelo especialista.

Champaign e Cohen [13] propuseram um algoritmo para selecionar o conteúdo apropriado a ser apresentado ao estudante baseado no uso de técnicas de filtragem colaborativa que identificam alunos similares a outros, e então preferencialmente recomenda os mesmos OAs aos estudantes similares encontrados. Dois pontos importantes a serem destacados nesse estudo são:

- a granularidade do sequenciamento (feita em nível de objeto de aprendizagem).
- a validação do algoritmo (feita com estudantes simulados).

O sequenciamento no nível de OA não faz alterações no próprio OA, considerando-o como uma unidade fechada ou "caixa preta", sendo geralmente usado para ordenação de

tópicos de currículos ou disciplinas. O sequenciamento no nível de exercícios ou nível de instrução em STIs tem sido realizado a partir da experiência pedagógica de especialistas da área sendo ensinada, geralmente sendo adotada uma ordem fixa.

A validação do algoritmo a partir de estudantes simulados, embora apresente limitações quanto às extrapolações que podem ser feitas a partir dos resultados encontrados, tem a vantagem de permitir investigar várias condições iniciais e seu impacto na calibragem do sistema.

Pimentel e Direne [44] apresentam uma ferramenta de autoria denominada SEQUENCE para medidas cognitivas usadas no ensino de conceitos visuais. A ferramenta permite que o autor do STI utilize a carga cognitiva de diagnóstico dos exemplares para a estruturação da sequência de sessões de ensino. Em um primeiro nível, o autor define o formato de uma sessão de ensino: número de imagens que a compõe, quais imagens, sua ordem de apresentação e parâmetros para controlar a apresentação das imagens. Em um outro nível, é possível descrever os valores que cada imagem de uma classe de anomalia possui, permitindo diferenciá-las qualitativa e quantitativamente. A carga cognitiva de uma imagem é calculada pela fórmula: $G = \frac{\alpha frequ+\beta sal+\gamma conf+\delta vtd+\theta dif+\omega stec}{\alpha+\beta+\gamma+\delta+\theta+\omega}$, onde $frequ$ se refere a frequência, sal é saliência, $conf$ é confiabilidade, vtd é visão tridimensional, $ddif$ é diagnósitico diferencial, e $stec$ é sinônimos técnicos. Os pesos $\alpha, \beta, \gamma, \delta, \theta$ e ω são definidos por um especialista. Um estudo de caso desmonstrou que a sequência definida por um especialista e a definida pela ferramenta de autoria foram relativamente semelhantes, demonstrando assim sua eficiência.

Outro aspecto relacionado ao sequenciamento de tarefas é a capacidade de previsão ou prospecção se determinado aluno está apto a responder corretamente uma questão, ou seja, solucionar um problema ou não, antes de ser apresentado a ele. Esse tipo de funcionalidade permite ao sistema apresentar exercícios mais fáceis ou difíceis ao estudante conforme seu *skill* ou *rating*. Cetintas *et al.* [12] propõem a utilização de uma abordagem de filtragem colaborativa temporal em um sistema tutor de matemática para esse fim. A técnica da filtragem colaborativa, bastante utilizada em sistemas de *e-commerce*, mostra como os usuários de um sistema estão associados com itens em uma aplicação visando

prever a utilidade dos itens para um usuário particular. Na abordagem da filtragem colaborativa temporal, a informação obtida a partir de múltiplas interações entre pares de problema/estudante é usada para comparar não somente a última performance do estudante sobre outros problemas, mas também obter curvas de aprendizagem através da comparação entre suas diversas tentativas sobre os mesmos problemas.

CAPÍTULO 3

CONCEITOS DA SOLUÇÃO

A expertise do estudante é geralmente desenvolvida através da resolução de exercícios que requerem um conjunto de habilidades avaliadas. Isso é feito tanto no sistema educacional de sala de aula convencional quanto em sistemas que fazem uso de computadores e aplicam técnicas de IA tais como Sistemas Tutores Inteligentes (STIs). Normalmente, os professores detectam equívocos conceituais dos alunos durante a realização de testes e exercícios. Dependendo de como a solução dada diverge da resposta correta ou solução esperada, dois estudantes que erraram a mesma questão podem ser pontuados diferentemente para aquela questão específica.

Outra condição que pode ser levada em conta em uma avaliação, considerando que todas questões estejam bem formuladas, é o fato de que quando um aluno acertou uma questão que a maioria do alunos erraram, além de indicar a perícia do aluno também indica que a referida questão provavelmente apresenta um grau de dificuldade maior. De modo análogo, quando um aluno errou uma questão que a maioria acertou, pode indicar a falta de perícia do aluno e também que a referida questão pode ser considerada de nível fácil.

Nesse sentido, é desejável que STIs saibam calibrar a dificuldade das questões ou atividades propostas de acordo com o nível de habilidade adquirido até o momento pelo aluno, efetuando um sequenciamento adequado das questões. A calibragem de questões também pode contribuir para evitar a desmotivação do aluno por entediamento ocorrida quando são apresentadas questões que o aluno considera de baixo desafio, ou por desistência quando são apresentadas questões com grau de dificuldade muito superior ao nível de conhecimento do aluno.

3.1 *Rating* do aluno

A utilização de sistemas de *rating* é uma forma bastante comum de classificação da perícia dos jogadores em jogos adversaristas como o xadrez. Em tais sistemas, o *rating* é um número real no intervalo $[minRating, maxRating]$, atualizado a cada torneio, tal que quanto maior o valor, maior a perícia do jogador. Além disso, em uma dada partida com jogadores J_a e J_b e respectivamente *ratings* R_{J_a} e R_{J_b}, se $R_{J_a} > R_{J_b}$, então a probabilidade de R_{J_a} vencer R_{J_b} é maior que a probabilidade de R_{J_b} vencer R_{J_a}. Assim, sistemas de *rating* também são frequentemente usados para prever resultados de partidas entre os jogadores.

Em STIs, os alunos não concorrem entre si. Porém são desafiados a realizar tarefas que ao mesmo tempo têm o propósito de avaliá-lo e de desenvolver sua perícia. E os resultados de seus desempenhos podem ser comparados e/ou usados para indicar futuras tarefas suas ou de outros estudantes.

Inspirada no sistema de *rating* do xadrez, este estudo propõe uma fórmula para fazer a classificação de um aluno levando em consideração os acertos, os erros e o número de tentativas, tanto do próprio aluno quanto dos demais. Por analogia, a fórmula foi elaborada como se houvesse implicitamente uma competição entre todos os alunos da turma.

Para a elaboração da fórmula foram adotadas as seguintes diretrizes:

- o *rating* do aluno é um valor real no intervalo [1..10];
- para uma dada questão, quanto mais alunos acertam, então menor o incremento do *rating* para os que acertaram e maior o decremento do *rating* para os que erraram. Assim, por hipótese, se uma questão é resolvida corretamente pela maioria, ela é considerada de nível fácil. Inversamente, se a maioria erra, ela é considerada de nível difícil;
- cada questão é pontuada no intervalo [0..10];
- alunos que resolvem uma questão corretamente na primeira tentativa devem ter uma

pontoação maior quando comparado a alunos que precisam de várias tentativas;

- no caso de errar uma questão, quanto maior o número de tentativas maior o decremento do *rating*;
- uma questão saltada (não resolvida) é considerada como erro.

3.1.1 Formalismo

$$R_J^q = R_J^{q-1} + Ak_1\alpha(10 - \frac{9T_J^q}{T_{med}^q}) - Ek_2\beta \times 10\frac{T_J^q}{T_{med}^q} \tag{3.1}$$

- R_J^q: *rating* do estudante J após responder a questão q;
- $A = 1$ e $E = 0$ se o estudante acertou q, caso contrário $A = 0$ e $E = 1$;
- T_J^q: número de tentativas sem sucesso do estudante J ao responder a questão q;
- T_{med}^q: mediana do número de tentativas erradas da questão q durante a sessão de exercícios;
- $\alpha = \frac{1}{N_a^q}$: ponderação do incremento de *rating*;
- $\beta = \frac{1}{N_e^q}$: ponderação do decremento de *rating*;
- N_a^q: número de estudantes que tiveram sucesso em responder a questão q;
- N_e^q: número de estudantes que erraram a questão q;
- k_1 e k_2: fatores multiplicadores de incremento e decremento de *rating*, respectivamente, calculados de acordo com R_J^{q-1} tal que $1 \leq R_J^{q-1} \leq 10$, and $k_1 = 1 - \frac{R_J^{q-1}}{10}$ e $k_2 = \frac{R_J^{q-1}-1}{10}$;
- $10 - \frac{9T_J^q}{T_{med}^q}$: nota na questão q em caso de acerto;
- $10\frac{T_J^q}{T_{med}^q}$: nota na questão q em caso de erro.

Além disso, não existe limite para o número de tentativas. Entretanto, se há mais de 10 tentativas, 10 é considerado para propósitos de cálculo. Os fatores k_1 e k_2 evitam que

o resultado da Equação 3.1 atinja os extremos do intervalo [1..10]. O *rating* inicial R_J^0 é 5.5 (ponto médio do intervalo de *rating*) para todos estudantes inicialmente, quando não se tem qualquer informação a seu respeito.

Os fatores de ponderação α e β representam a influência da dificuldade da questão no cálculo de *rating*, pois são inversamente proporcionais ao número de alunos que acertaram e erraram a questão, respectivamente. Assim, quanto mais alunos acertam uma questão q, menor será o valor de α para o estudante que acertou q e maior o valor de β para o estudante que errou q.

Outra maneira de se compreender o papel dos fatores α e β na fórmula é através da analogia com jogos adversaristas. Assim, o movimentado acertado do jogador sendo avaliado se distancia mais dos jogadores concorrentes quando eles fazem movimentos equivocados, ou se distancia menos quando eles também fazem movimentos acertados no jogo.

3.1.2 Algoritmo

Após o fechamento da sessão de exercícios, para cada questão são feitos os cálculos da mediana de tentativas (T_{med}^q), número de alunos que acertaram (N_a^q) e do número de alunos que erraram (N_e^q). Em seguida, o cálculo de *rating* consiste na aplicação iterativa da fórmula 3.1 considerando $R_J^0 = 5.5$ para todos os alunos.

3.1.3 Exemplo

A tabela 3.1 apresenta um exemplo hipotético considerando 7 questões com os respectivos número de alunos que acertaram, erraram, a mediana de tentativas, e as informações relacionadas ao desempenho do estudante nº9: o *status* (A=Acertou, E=Errou, S=Saltou), número de tentativas e o cálculo iterativo do seu *rating*. Nesse exemplo, a turma contém 30 alunos.

Tabela 3.1: Exemplo de *rating*

				Desempenho do estudante nº9		
Questão	N_a^q	N_e^q	T_{med}^q	*Status*	T_9^q	R_9^q
-						$R_9^0 = \mathbf{5.5}$
1	27	3	2	A	0	$R_9^1 = 5.5 + (1 - \frac{5.5}{10}) \times \frac{1}{27} \times (10 - \frac{9 \times 0}{2})$ $= \mathbf{5.6667}$
2	23	7	2	E	3	$R_9^2 = 5.6667 - (\frac{5.6667 - 1}{10}) \times \frac{1}{7} \times 10 \times \frac{3}{2}$ $= \mathbf{5.0000}$
3	17	13	4	A	2	$R_9^3 = 5.0000 + (1 - \frac{5.0000}{10}) \times \frac{1}{17} \times (10 - \frac{9 \times 2}{4})$ $= \mathbf{5.2941}$
4	25	5	2	A	0	$R_9^4 = 5.2941 + (1 - \frac{5.2941}{10}) \times \frac{1}{25} \times (10 - \frac{9 \times 0}{2})$ $= \mathbf{5.4824}$
5	12	18	3	S	0	$R_9^5 = 5.4824 - (\frac{5.4824 - 1}{10}) \times \frac{1}{18} \times 10 \times \frac{0}{3}$ $= \mathbf{5.4824}$
6	4	26	5	E	13	$R_9^6 = 5.4824 - (\frac{5.4824 - 1}{10}) \times \frac{1}{26} \times 10 \times \frac{10}{5}$ $= \mathbf{5.1376}$
7	20	10	2	A	1	$R_9^7 = 5.1376 + (1 - \frac{5.1376}{10}) \times \frac{1}{20} \times (10 - \frac{9 \times 1}{2})$ $= \mathbf{5.3807}$

3.2 Calibragem das questões

A calibragem de questões consiste em classificá-las de acordo com seu grau de dificuldade, podendo ser utilizada uma escala discreta ou contínua. Neste estudo foi adotada uma escala contínua dentro do intervalo [0..10], sendo que quanto maior o valor, mais difícil é considerada a questão.

A princípio, professores experientes são capazes de aquilatar o grau de dificuldade das questões que elaboram. Porém, à medida que o número de questões aumenta torna-se mais intricado colocá-las ordenadas por grau de dificuldade. Assim, embora a tarefa de classificação da dificuldade de uma questão possa ser deixada para o professor, é importante ter também instrumentos automáticos que efetuem a classificação automática.

Uma maneira simples de se calcular o grau de dificuldade de uma questão é analisar a taxa de erro/acerto após sua aplicação em uma turma de alunos.

3.2.1 Formalismo

Em uma primeira abordagem, o grau de dificuldade da questão q pode ser definido através dos detalhes da Equação 3.2 e seus parâmetros que se seguem:

$$D^q = \frac{N_e^q + N_s^q}{N_e^q + N_a^q + N_s^q} \times 10 \tag{3.2}$$

onde:

- D^q: grau de dificuldade da questão q após uma sessão de exercícios;
- N_e^q: número de estudantes que erraram q;
- N_a^q: número de estudantes que acertaram q;
- N_s^q: número de estudantes que saltaram q.

Em outra abordagem, usando somente o número de tentativas e considerando que o estudante geralmente tenta até conseguir obter a resposta correta, o grau de dificuldade de uma questão q pode ser definido pela Equação 3.3 e seus parâmetros que seguem:

$$D^q = \frac{\sum_{J=0}^{J=n} T_J^q}{N_e^q + N_a^q} \tag{3.3}$$

onde:

- D^q: grau de dificuldade da questão q após uma sessão de exercícios;
- T_J^q: número de tentativas do estudante J na questão q. Se o número de tentativas é maior que 10, então 10 é considerado como T_J^q;
- N_e^q: número de estudantes que erraram q;
- N_a^q: número de estudantes que acertaram q.

3.2.2 Algoritmo

O algoritmo do cálculo do grau de dificuldade de cada questão consiste simplesmente na aplicação da fórmula 3.2 ou 3.3 após o término da sessão de exercícios.

3.2.3 Exemplo

Para exemplificar o uso das fórmulas de cálculo do grau de dificuldade, considere a tabela 3.2, que mostra para 10 alunos e uma questão hipotética q: o *status* da resposta (A=Acertou; E=Errou e S=Saltou) e o número de tentativas sem sucesso respectivamente.

Considerando a fórmula 3.2, o grau de dificuldade da questão q será dado por: $\frac{(3+1)}{11} \times 10 = 3.6363$. Já utilizando a fórmula 3.3, o grau de dificuldade de q será dado por: $\frac{(0+8+10+7+1+9+10+2+8+4)}{10} = \mathbf{5.9}$.

3.3 Algoritmo para Sequenciamento Adaptativo de Exercícios

Um aspecto importante em STIs é como os exercícios devem ser sequenciados após serem calibrados tal que se encaixem com o nível de perícia do estudante. No início, o sistema não tem qualquer informação a respeito do estudante. Portanto, uma maneira natural é

Tabela 3.2: Exemplo de dados para uma questão hipotética q

Aluno J	*Status* q	T_J^q
1	A	0
2	E	8
3	A	13
4	A	7
5	A	1
6	A	9
7	E	15
8	S	0
9	A	2
10	E	8
11	A	4

colocá-los em ordem crescente de dificuldade. Entretanto, ainda assim existem questões em aberto como por exemplo: quantos exercícios devem ser apresentados? E quantas vezes um mesmo exercício pode ser apresentado?

Neste trabalho, é proposto um algoritmo para sequenciamento de exercícios em ordem crescente de dificuldade, combinado com um mecanismo similar à interpolação numérica, que segue as diretrizes abaixo:

- a calibragem dos exercícios deve ter sido efetuada previamente, seguindo a fórmula 3.2 ou 3.3 para cálculo do grau de dificuldade;
- uma sequência mínima de exercícios é definida, que será apresentada inicialmente ao estudante, que corresponde aos exercícios que serão resolvidos no caso do estudante responder sempre corretamente;
- a sequência mínima sempre começa com o exercício mais fácil e termina com o mais difícil;
- os exercícios de dificuldade intermediária na sequência mínima são distribuídos uniformemente entre o mais fácil e o mais difícil, considerando o tamanho do passo, denotado aqui por s e que se refere ao número de exercícios que podem ser saltados quando o estudante acerta um exercício. Assim, para um conjunto de n exercícios

$\{e_1, e_2, e_3, ..., e_n\}$ ordenados por grau de dificuldade, a sequência mínima será definida por $min\_seq =< e_1, e_{1+s}, e_{1+2s}, e_{1+3s}, ..., e_n >$;

- o número de tentativas que um estudante tem para resolver uma questão é limitado à média de tentativas, obtida durante a fase de calibragem;

- quando o número de tentativas excede o limite estabelecido, o próximo exercício a ser apresentado ao estudante é o que se encontra no ponto médio de dificuldade entre o exercício atual e o último respondido corretamente.

Por exemplo, considere um OA com 30 exercícios em ordem crescente de dificuldade $\{e_1, e_2, ..., e_{30}\}$ e $s = 4$. A sequência mínima de exercícios será $min\_seq =< e_1, e_5, e_9, e_{13}, e_{17}, e_{21}, e_{25}, e_{29}, e_{30} >$. A apresentação dos exercícios ao estudante, inicia-se seguindo a ordem da sequência mínima definida. Porém, por exemplo, se o estudante excede o número de tentativas estabelecido para e_9, então e_7 (ponto médio entre e_5 e e_9) é apresentado. Ao contrário do que ocorre durante a fase de calibragem dos exercícios, não é permitido ao estudante saltar exercícios. Além disso, se ele/ela erra continuadamente, a apresentação dos exercícios passa a ser sequencial.

O tamanho do passo s pode ser configurado pelo autor do OA e deve ser tal que o número de exercícos na sequência mínima tenha pelo menos 25% do total de exercícios. A quantidade de exercícios na sequência mínima é dada pela fórmula: $\lfloor \frac{n}{s} \rfloor + 1 + \lfloor \frac{n \mod s}{2} \rfloor$.

A sequência mínima define automaticamente o número mínimo de exercícios e quais são esses exercícios que deverão ser apresentados no caso do aluno acertar continuadamente.

O funcionamento do algoritmo pode ser representado usando uma árvore binária, em que um nó-filho à direita representa o próximo exercício quando o estudante respondeu corretamente ao exercício correspondente ao nó-pai dentro do limite de tentativas estabelecido. Um nó-filho à esquerda representa o próximo exercício quando o estudante erra o exercício representado pelo nó-pai, uma vez esgotado o limite de tentativas. Na Figura 3.1 é apresentada uma árvore de sequenciamento de exercícios considerando $s = 4$. A sequência mínima corresponde exatamente ao caminho da árvore de sequenciamento tomando-se sempre o nó-filho à direita a partir da raiz.

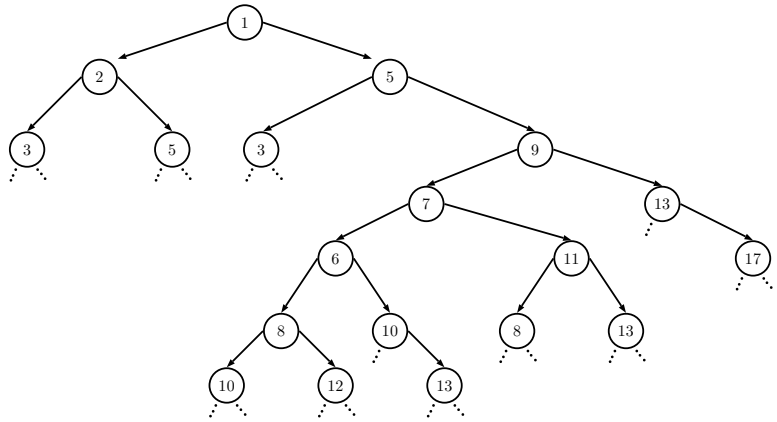


Figura 3.1: Árvore de Sequenciamento de Exercícios

A Figura 3.2 mostra o pseucódigo da função chamada $nextExercise$, que implementa o algoritmo de sequenciamento adaptativo de exercícios, retornando a próxima página de exercício para um dado estudante, representado pelo argumento st_id. Essa função faz uso das seguintes funções auxiliares, que localizam um exercício que atenda determinadas condições de acordo com o histórico de desempenho do estudante:

- $last\_exercise\_done(st\_id)$: retorna o último exercício respondido;
- $next\_exercise\_not\_done\_in\_min\_seq(st\_id)$: retorna o próximo exercício ainda não exibido da sequência mínima;
- $first\_exercise\_not\_done(st\_id)$: retorna o primeiro exercício não exibido ainda;
- $last\_ex\_correctly\_done\_in\_min\_seq(st\_id)$: retorna o último exercício da sequência mínima respondido corretamente;
- $last\_ex\_correctly\_done\_before(ex_{last}, st\_id)$: retorna o último exercício respondido corretamente, anterior ao exercício atual;
- $first\_not\_done\_exercise\_after(ex_{mid}, st\_id)$: retorna o primeiro exercício não apresentado após ex_{mid}.

Na função $nextExercise$, as variáveis ex_{last} e ex_{min_seq} contêm o último exercício apresentado ao estudante e o próximo exercício da sequência mínima, respectivamente. Já as variáveis ex_{low} e ex_{mid} armazenam o último exercício resolvido corretamente e o exercício

```
1: function NEXTEXERCISE(st_id)                ▷ Retorna o próximo exercício para o estudante st_id
2:     ex_last ← last_exercise_done(st_id)
3:     if ex_last is null then
4:         return 1                                                    ▷ Exibe o 1º exercício
5:     else if ex_last is correct then
6:         ▷ Vai para exercício mais difícil
7:         ex_min_seq ← next_exercise_not_done_in_min_seq(st_id)
8:         if ex_min_seq is null then                  ▷ Chegou ao fim da sequência mínima
9:             return ex_last                                  ▷ Permanece no último exercício
10:        else if ex_last + STEP_SIZE < ex_min_seq then
11:            ▷ Vai para o 1º exercício não respondido após o último respondido da sequência mínima
12:            return first_exercise_not_done(st_id)
13:        else
14:            return ex_min_seq
15:        end if
16:    else if NumAttempts(ex_last, st_id) = Average(ex_last) then
17:        ▷ Número de tentativas esgostado - ir para exercício mais fácil
18:        ex_low ← last_ex_correctly_done_in_min_seq(st_id)
19:        if ex_low is null then
20:            ex_low ← last_ex_correctly_done_before(ex_last, st_id) ▷ Estudante falhou nos exercícios da
   sequência mínima
21:        end if
22:        if ex_low is null then
23:            ex_low ← ex_last                ▷ Estudante falhou em todos os exercícios apresentados
24:        end if
25:        ex_mid = (ex_low + ex_last)/2 ▷ Calcula o índice do exercício de dificuldade média entre ex_low e ex_last
26:        return first_not_done_exercise_after(ex_mid, st_id)
27:    else
28:        ▷ Permanece no mesmo exercício
29:        return ex_last
30:    end if
31: end function
```

Figura 3.2: Função para calcular o próximo exercício

cujo grau de dificuldade está no ponto médio entre o último correto e o atual respondido incorretamente. Caso o aluno tenha falhado em todos os exercícios apresentados até o presente momento, a variável ex_{low} irá armazenar o último exercício apresentado.

3.4 Arquitetura e implementação da ADAPTFARMA

ADAPTFARMA é uma versão modificada de FARMA (Ferramenta de Autoria para a Remediação de erros com Mobilidade na Aprendizagem) [34, 35], uma ferramenta de autoria para construção de OAs de conceitos matemáticos. Do mesmo modo que em FARMA, em ADAPTFARMA um OA consiste de uma sequência de exercícios após uma introdução. A introdução é a parte teórica de um OA onde conceitos são definidos através de texto, imagens, audios e vídeos. Os exercícios contém a essência prática dos conceitos, com os

quais o aprendiz irá adquirir perícia através de tarefas de resolução de problemas. Adicionalmente, ADAPTFARMA tem integrado em seu arcabouço a implementação do cálculo de *rating* dos alunos, o grau de dificuldades das questões e o algoritmo de sequenciamento adaptativo de exercícios apresentados na seção anterior.

3.4.1 FARMA

A Figura 3.3 mostra a arquitetura funcionalista de FARMA, que tem 3 módulos principais: autoria, interação e monitoramento.

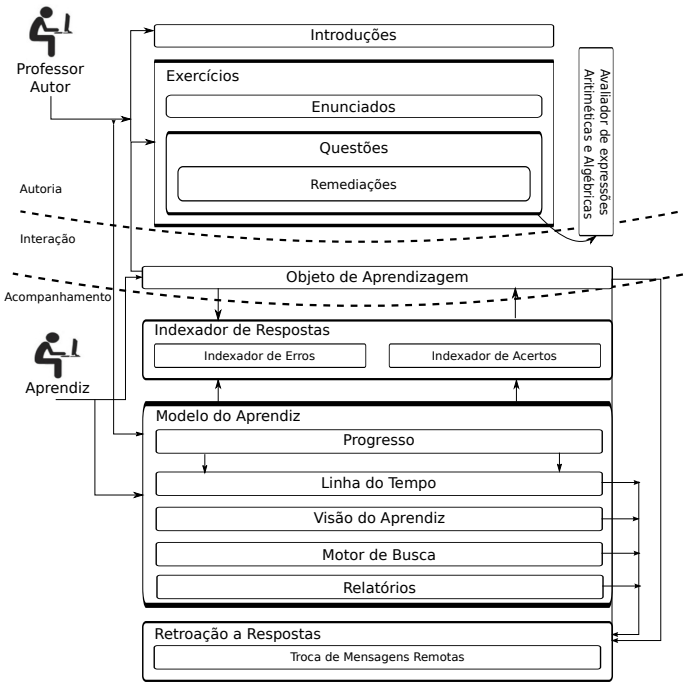


Figura 3.3: Arquitetura funcionalista de FARMA

3.4.1.1 Módulo de Autoria

O módulo de autoria provê as funcionalidades para construção de OAs. Ele é dividido em construtor de introduções (parte teórica) e construtor de exercícios (parte prática). Devido à natureza genérica da Matemática, cujas respostas dos exercícios podem ser expressões algébricas ou aritméticas, OAs podem ser construídos para várias áreas como Química, Física e Matemática entre outras.

Para construir uma introdução e seus exercícios correspondentes, FARMA oferece uma interface WYSIWYG (*What You See Is What You Get*), similar aos processadores de texto altamente interativos. Em relação aos exercícios, a ferramenta é bastante genérica, permitindo ao professor definir o número de questões relacionadas a cada exercício. Esta técnica é baseada no teoria ACT (*Adaptive Control of Thought*) [48]. Para cada questão, o professor-autor deve indicar uma solução de referência, que é a resposta correta da questão. FARMA permite que expressões algébricas e aritméticas sejam colocadas como solução de referência. Sob o modo estudante, a ferramenta lida automaticamente com equivalência entre a resposta do estudante e a solução de referência.

Além disso, FARMA permite ao professor-autor definir regras de remediação e mensagens para cada questão dos exercícios, aplicadas quando o estudante comete um erro. Para tal, é necessário definir o número de tentativas sem sucesso após o qual a mensagem de *feedback* deve ser exibida. Isto mostra o quanto é simples proporcionar *feedback* imediato em FARMA.

3.4.1.2 Módulo de Interação

O modulo de interação é a interface entre o aprendiz e o OA criado pelo professor. Esse módulo efetua a junção das introduções, exercícios, questões e remediação de erros para executar o OA, que consiste em uma sequência de páginas.

3.4.1.3 Módulo de Monitoramento

Esse módulo permite que o professor faça a avaliação formativa da aprendizagem individual do aluno através da retroação a erros, em que o professor pode ter acesso ao contexto exato da ocorrência de erro do aluno. O módulo está dividido nas seguintes partes principais:

- **Indexador de respostas**: salva todas respostas submetidas pelo estudante, sejam erradas ou corretas;

- **Modelo do Estudante**: a informação da interação de cada usuário com o OA particular, tais como:

  - **(a)** Progresso do Aprendiz, que analisa as respostas dos estudantes e infoma em procentagem, quando cada estudante falta para completar o OA;
  - **(b)** O *timeline* do Aprendiz, onde é possível ver todas interações em ordem cronológica;
  - **(c)** Visão do Estudante do OA, onde o professor tem exatamente a mesma visão do estudante;
  - **(d)** Motor de busca, no qual o professor, bem como estudantes podem pesquisar por erros;
  - **(e)** Relatório sobre interação, onde é possível ver de maneira tabulada a interação dos aprendizes com o OA, agrupada por: nome do estudante, exercícios, questão, resposta correta ou incorreta; e o número de tentativas da questão.

- **Retroação a respostas**, em que o indexador de respostas recupera dados sobre o aprendiz e permite o professor retornar para o contexto exato da resposta de um estudante específico.

A capacidade de retroagir ao contexto exato do erro cometido pelo estudante é uma funcionalidade diferenciada de FARMA, quando comparada a ferramentas similares, pois dá oportunidade ao professor de identificar os passos equivocados feitos pelo estudante para então lidar apropriadamente com as causas do erro. Além disso, FARMA permite ao professor ver uma interação completa do estudante com a ferramenta em ordem cronológica, na forma de uma *timeline*. Da mesma maneira, os aprendizes também podem retroagir ao contexto exato de qualquer uma das suas respostas corretas ou erradas, visando refletir sobre seus próprios passos da solução.

Quando um professor ou aluno retroage ao contexto de uma resposta, FARMA restaura exatamente a mesma visão do usuário ocorrida no exato momento em que ela foi inserida no sistema. Além disso, no aspecto colaborativo, é possível ao professor prover

feedback não automático, feito através do sistema de mensagens remotas do sistema. É importante mencionar que o mode de retroação difere das ações de desfazer (*undoing*) tão frequentemente disponíveis em interfaces WYSIWYG.

3.4.2 Esquema geral da ADAPTFARMA

A arquitetura básica de ADAPTFARMA é a mesma de FARMA, porém com modificações nos módulos de interação e monitoramento. Além disso o banco de dados foi alterado para armazenar os *ratings* dos alunos em cada objeto de aprendizagem, e estatísticas relativa às questões, como a média do número de tentativas e o grau de dificuldade.

No módulo de interação foi criada uma camada que encapsula a função $nextExercise$, apresentada na Figura 3.2. No caso de um exercício conter várias questões, seu grau de dificuldade é calculado pela média dos graus de dificuldade das questões que o compõem. E quando o aluno excede o número de tentativas de qualquer uma das suas questões, é apresentado o próximo exercício.

No módulo de monitoramento foram criados um relatório para visualização dos *ratings* dos estudantes e outro para visualização do grau de dificuldade das questões, dado um objeto de aprendizagem.

CAPÍTULO 4

ESTUDO EMPÍRICO

Para fazer uma validação inicial das fórmulas de *rating* 3.1 e grau de dificuldade 3.2 e 3.3, apresentadas no capítulo 3, foi realizado um estudo empírico utilizando dados de alunos reais, coletados por 2 objetos de aprendizagem (OAs) diferentes: um sobre Progressões Geométricas em Fractais e outro sobre Logaritmos.

4.1 OA Progressões Geométricas em Fractais

As páginas iniciais do OA contêm explicações sobre conceitos básicos de fractais e sua relação com o domínio de progressões geométricas. As páginas finais consistem de 6 exercícios no formato tabular, em que cada linha corresponde a uma iteração de fractal, devendo o aluno preencher colunas com valores ou expressões algébricas que correspondam à medida de lado, perímetro ou área da figura geométrica utilizando um teclado virtual. Cada exercício tem um número de células (questões) a serem preenchidas pelo aluno. A Figura 4.1 apresenta uma das páginas de exercício do OA Progressões Geométricas em Fractais.[1].

4.2 OA Invenção dos Logaritmos

O OA consite em páginas de exercícios, tal que é apresentada alguma propriedade de logaritmos e em seguida enunciados de questões em que a referida propriedade deve ser aplicada para encontrar a solução. A Figura 4.2 apresenta uma das páginas de exercícios do OA.

[1]Disponível em: `http://webeduc.mec.gov.br/portaldoprofessor/matematica/condigital2/fractal/fractal/index.html`

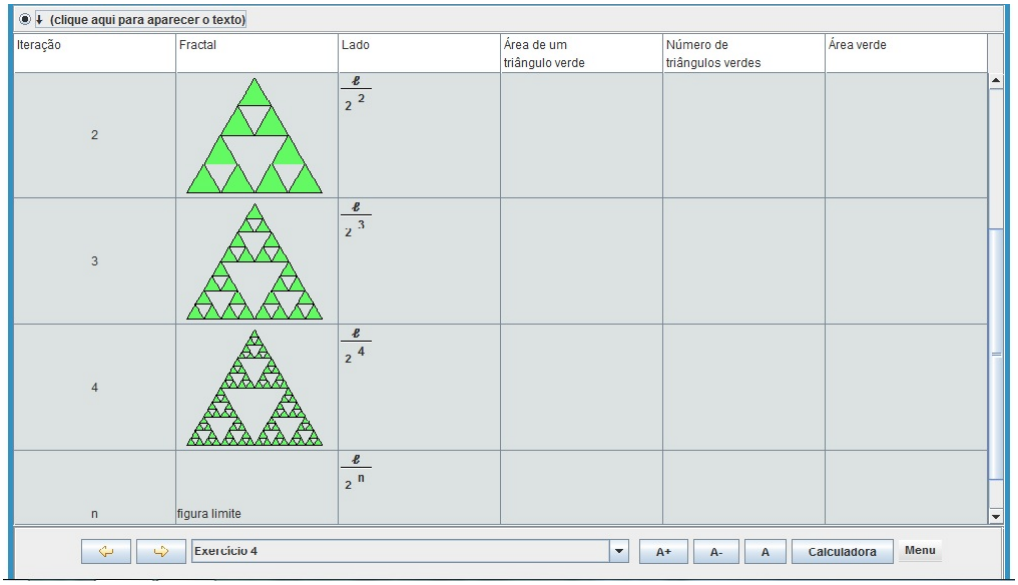


Figura 4.1: Página de Exercício do OA Progressões Geométricas em Fractais

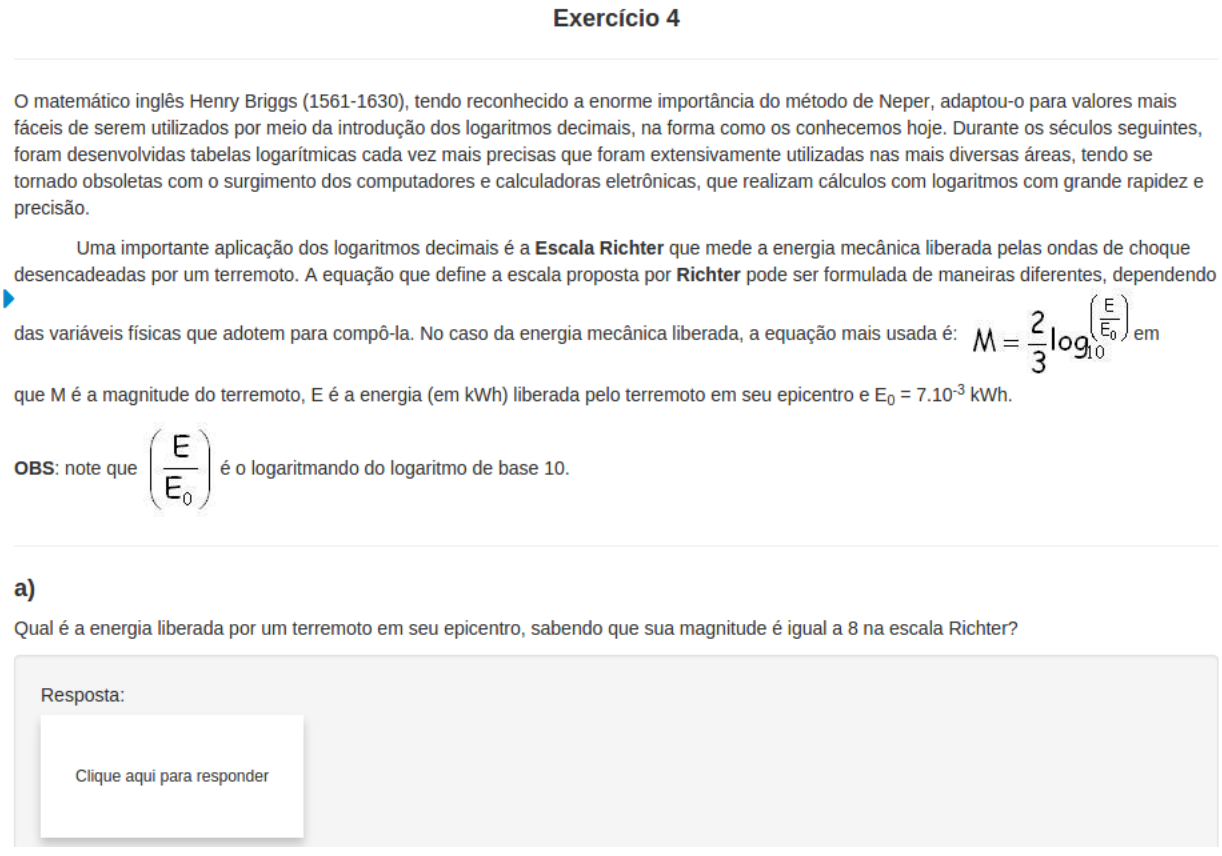


Figura 4.2: Página de Exercício do OA Invenção dos Logaritmos

4.3 Sujeitos

O OA Progressão Geométrica em Fractais foi aplicado em uma turma de 34 alunos do 1ºano de Engenharia. O OA Invenção dos Logaritmos foi aplicado em uma turma de 21 alunos do 1ºano do ensino médio do curso Técnico de Edificações.

4.4 Metodologia

Para avaliação da fórmula de *rating* 3.1 foi feita sua aplicação iterativamente para cada aluno seguindo a ordem de apresentação das questões no OA. Esse procedimento foi aplicado 2 vezes para todos os alunos, sendo a 1ªvez considerando o valor do *rating* inicial 1 e a 2ªvez considerando 5.5 (ponto médio do intervalo da escala de *rating*). O teste usando diferentes valores iniciais de *rating* permite testar a convergência da fórmula sobre dados reais. Os valores dos *ratings* finais podem ser visualizados nas tabelas 4.1 e 4.2. É possível notar que a ordem final dos alunos (*ranking*) para *rating* inicial 1 e 5.5 ficou idêntica para os dados coletados a partir do OA Invenção dos Logaritmos e praticamente a mesma para os dados obtidos a partir do OA Progressões Geométricas em Fractais.

Sobre os dados coletados do OA Invenção dos Logaritmos foi feita outra aplicação da fórmula 3.1 para avaliar sua adequação para uso no momento exato do registro de uma resposta, ao invés de somente ao final de uma sessão de exercícios. Assim, essa aplicação foi feita considerando a ordem de *timestamp* das respostas e trocando T^q_{med} por T^q_{max}. Nessas condições, o número de alunos que acertaram (N^q_a) e o número de alunos que erraram (N^q_e) usados correspondem somente às respostas registradas até o *timestamp* sendo analisado. Portanto, em um dado *timestamp* o total de alunos pode não corresponder à soma de N^q_a e N^q_e. Isso equivale analogamente em jogos à situação de quando nem todos jogadores ingressam no torneio ao mesmo tempo. A partir dos dados mostrados na tabela 4.3, é possível observar que a ordem final se mantém praticamente a mesma adotando *rating* inicial 1 ou 5.5.

A tabela 4.4 mostra uma comparação entre os *rankings* obtidos com o OA Invenção dos Logaritmos, considerando *rating* inicial 5.5, com e sem uso dos *timestamps*. É possível

Tabela 4.1: *Rating* final dos alunos - OA Fractais

user_id	**rating_inicial**	**rating_atual**	**user_id**	**rating_inicial**	**rating_atual**
1	1	1,01587	1	5,5	3,78548
18	1	1,54417	22	5,5	4,13436
37	1	1,74527	35	5,5	4,25925
22	1	1,88914	18	5,5	4,2969
4	1	2,06839	4	5,5	4,38951
35	1	2,16064	37	5,5	4,45217
23	1	2,88091	23	5,5	4,45936
26	1	3,10333	2	5,5	5,55684
2	1	3,34901	38	5,5	5,62763
32	1	3,46241	26	5,5	5,7379
38	1	4,0028	32	5,5	5,75749
16	1	4,02356	16	5,5	6,01131
28	1	4,6198	28	5,5	6,30853
13	1	5,14821	13	5,5	6,69221
39	1	6,13242	39	5,5	7,42517
15	1	6,67339	34	5,5	7,49184
9	1	6,70767	21	5,5	7,52258
21	1	6,78383	15	5,5	7,60694
36	1	6,88475	36	5,5	7,64642
34	1	7,00927	9	5,5	7,83071
11	1	7,52769	11	5,5	8,15244
8	1	7,75403	8	5,5	8,55832
14	1	8,23743	14	5,5	8,76219
6	1	8,44965	6	5,5	8,94474
12	1	8,50183	19	5,5	9,14668
7	1	8,6351	12	5,5	9,23329
27	1	8,68941	7	5,5	9,30111
19	1	8,84899	27	5,5	9,32854
41	1	8,9834	41	5,5	9,47957
29	1	9,25184	24	5,5	9,58812
24	1	9,2955	29	5,5	9,61674
17	1	9,4087	17	5,5	9,69705
20	1	9,4364	20	5,5	9,71128
10	1	9,48436	10	5,5	9,73606

Tabela 4.2: *Rating* final dos alunos - OA Logaritmos

user_id	**rating_inicial**	**rating_atual**	**user_id**	**rating_inicial**	**rating_atual**
9	1	0,906419992	9	5,5	5,453209877
10	1	1,132320046	10	5,5	5,566160202
3	1	1,222159982	3	5,5	5,61108017
5	1	1,428570032	5	5,5	5,714290142
1	1	1,490020037	1	5,5	5,745009899
14	1	1,946050048	14	5,5	5,97303009
7	1	2,069859982	7	5,5	6,034930229
18	1	2,131649971	18	5,5	6,065830231
19	1	2,299860001	19	5,5	6,14993
11	1	2,476589918	11	5,5	6,238289833
2	1	2,569269896	2	5,5	6,284629822
17	1	2,671430111	17	5,5	6,335710049
6	1	3,358839989	6	5,5	6,679419994
16	1	3,494159937	16	5,5	6,747079849
20	1	3,610199928	20	5,5	6,805099964
8	1	3,673670053	8	5,5	6,836840153
21	1	4,144430161	21	5,5	7,072219849
4	1	5,691760063	4	5,5	7,845880032
12	1	5,798719883	12	5,5	7,89936018
15	1	6,556920052	15	5,5	8,278459549
13	1	7,077330112	13	5,5	8,538660049

Tabela 4.3: *Rating* final do alunos usando *timestamp* - OA Logaritmos

user_id	**rating_inicial**	**rating_atual**	**user_id**	**rating_inicial**	**rating_atual**
8	1	1,317469954	8	5,5	1,317469954
19	1	1,423789978	19	5,5	1,438030005
11	1	1,450000048	11	5,5	1,450000048
5	1	1,5625	4	5,5	1,899999976
16	1	1,641680002	12	5,5	1,99000001
4	1	1,899999976	16	5,5	2,064960003
12	1	1,99000001	1	5,5	2,233730078
1	1	2,096839905	6	5,5	2,628420115
9	1	2,148149967	7	5,5	3,110860109
6	1	2,293869972	10	5,5	3,199310064
18	1	2,404949903	2	5,5	3,448479891
7	1	2,487309933	18	5,5	3,702100039
14	1	2,515369892	14	5,5	3,867660046
3	1	2,6329	9	5,5	3,876569986
10	1	2,723320007	3	5,5	4,066730022
20	1	3,293440104	20	5,5	5,36755991
2	1	3,448479891	5	5,5	5,78125
17	1	4,095099926	17	5,5	7,047550201
21	1	4,270319939	21	5,5	7,135159969
15	1	8,612170219	15	5,5	8,866629601
13	1	9,250740051	13	5,5	9,570010185

notar que a ordem final dos *ratings* são significativamente diferentes devido ao fato de que a atualização do *rating* do aluno no momento da gravação da resposta é feita apenas com as informações dos demais alunos que tenham respondido à mesma questão até o momento analisado.

Tabela 4.4: Comparação de *ratings* - OA Invenção dos Logaritmos

user_id	*rating sem timestamp*	**user_id**	*rating com timestamp*
9	5,453209877	8	1,317469954
10	5,566160202	19	1,438030005
3	5,61108017	11	1,450000048
5	5,714290142	4	1,899999976
1	5,745009899	12	1,99000001
14	5,97303009	16	2,064960003
7	6,034930229	1	2,233730078
18	6,065830231	6	2,628420115
19	6,14993	7	3,110860109
11	6,238289833	10	3,199310064
2	6,284629822	2	3,448479891
17	6,335710049	18	3,702100039
6	6,679419994	14	3,867660046
16	6,747079849	9	3,876569986
20	6,805099964	3	4,066730022
8	6,836840153	20	5,36755991
21	7,072219849	5	5,78125
4	7,845880032	17	7,047550201
12	7,89936018	21	7,135159969
15	8,278459549	15	8,866629601
13	8,538660049	13	9,570010185

Na tabela 4.5 são apresentados os graus de dificuldade de algumas questões do OA Progressão Geométrica em Fractais, calculados a partir da taxa de erros ($D^q = \frac{N_e^q + N_s^q}{N_e^q + N_a^q + N_s^q} \times 10$) e a partir da média de tentativas ($D^q = \frac{\sum_{J=0}^{J=n} T_J^q}{N_e^q + N_a^q}$), apresentadas no capítulo 3.

4.5 Análise dos Resultados

A partir dos resultados obtidos, foi possível notar que:

- A adoção de $R^0 = 1$ ou $R^0 = 5.5$ não afeta o *ranking*, e sim somente o valor final do *rating*. Devido a isso, neste trabalho resolve-se adotar $R^0 = 5.5$ por considerar o aluno como estando inicialmente no ponto médio, onde não se tem qualquer

Tabela 4.5: Grau de dificuldade das questões - OA Fractais

	Dificuldade	
Questão	**taxa de erro**	**Média de tentativas**
112	2,6086	1,2439
122	2,8260	1,0000
132	2,8260	1,1944
142	3,6956	2,9796
212	2,6086	1,2368
222	2,8260	1,0000
242	2,8260	1,0294
252	3,9130	1,2632
303	3,0434	1,1395
312	3,9130	1,0000
313	3,6956	1,9455
322	4,1304	1,0323
323	4,1304	1,2571
333	4,7826	1,3929
343	5	1,1786
352	5	1,2800
353	6,0869	1,6000

informação a seu respeito;

- a fórmula de *rating* é adequada para avaliação de cada aluno após uma sessão de exercícios;

- a fórmula de *rating* não é adequada para ser usada no momento exato de cada resposta, pois nessa situação conduz a resultados distorcidos do *ranking* final;

- o uso de T^q_{med} (mediana do número de tentativas) justifica-se pela exclusão de valores muito discrepantes entre os estudantes no número de tentativas de uma determinada questão;

- T^q_{max} (número máximo de tentativas) pode ser usada no lugar de T^q_{med}, desde que se estabeleça um valor limite, visando descartar valores discrepantes gerados por alunos que respondem várias vezes ao mesmo exercício de maneira descompromissada. A implementação do cálculo de T^q_{max} tem complexidade de tempo $O(n)$ enquanto a implementação clássica de T^q_{med}, que requer ordenação, tem complexidade $O(n \log n)$.

CAPÍTULO 5

AVALIAÇÃO EXPERIMENTAL DA APRENDIZAGEM

5.1 Experimento

5.1.1 Metas

Com a finalidade avaliar a aprendizagem proporcionada pela estratégia de sequenciamento descrita na seção 3.3 e encapsulada na implementação do algoritmo mostrado na figura 3.2, foi feito um experimento com alunos reais.

5.1.2 Sujeitos

O experimento foi conduzido com 4 turmas do ensino médio técnico de uma escola pública, sendo 2 turmas do curso Técnico em Química Industrial (turmas A e B) e 2 turmas do curso Técnico em Meio Ambiente (turmas C e D). A idade dos estudantes é entre 15 e 17 anos. A tabela 5.1 apresenta a quantidade de alunos que participaram em cada etapa do experimento, sendo a última coluna o número de alunos que participaram de todas as etapas.

Tabela 5.1: Quantidade de alunos participantes em cada etapa do experimento

Turma	**Qtd Alunos por Turma**	**Pré-teste**	**ADAPTFARMA**	**Pós-teste**	**Válidos**
A	33	31	30	32	29
B	35	32	33	34	30
C	33	33	30	32	30
D	35	32	32	35	30
					Total: 119

Foram retirados da análise do experimento os alunos que não realizaram o pré-teste ou pós-teste ou que não utilizaram a ferramenta, ficando 119 participantes válidos.

5.1.3 Material

Foi criado um objeto de aprendizagem sobre logaritmos na ADAPTFARMA com 10 páginas conceituais e 30 exercícios, pelo mesmo professor da disciplina de Matemática nas 4 turmas. O professor também foi o responsável pela elaboração e aplicação do pré-teste e pós-teste aplicado nas 4 turmas, valendo $1,5$ pontos cada, sendo a pontuação usada para compor a nota oficial do registro acadêmico. Ambos testes constam nos anexos A e B.

A ferramenta estava disponível *online* e cada turma teve 2 dias alternados com 2 horas-aulas de 50 min em laboratório de informática, equipado com computadores com acesso à Internet, para utilizar a ferramenta. Todos os estudantes estavam previamente cadastrados nas respectivas turmas com *login*/senha de acesso no ambiente ADAPTFARMA e puderam usá-lo fora do horário de aula.

5.1.4 Metodologia

Visando comparar diferentes estratégias de sequenciamento de exercícios, foram adotadas as seguintes diretrizes:

- os mesmos pré-teste e pós-teste foram aplicados às 4 turmas;
- o mesmo OA foi aplicado nas 4 turmas;
- o OA foi elaborado tal que todo exercício contém exatamente uma única questão;
- o uso da ferramenta ADAPTFARMA não era obrigatório, sendo posteriormente retirados da análise os dados referentes aos estudantes que não a utilizaram;
- para cada turma foi usada uma estratégia de sequenciamento diferente, distribuidas da seguinte maneira:
  - Turma A: Método de Sequenciamento Randômico (**MSR**);
  - Turma B: Método de Sequenciamento definido pelo Professor (**MSP**);

- Turma C: Método de Sequenciamento ordenado por grau de Dificuldade (**MSD**), calculado usando a fórmula 3.3 ($D^q = \frac{\sum_{J=0}^{J=n} T_J^q}{N_e^q + N_a^q}$);
- Turma D: Método de Sequenciamento Adaptativo (**MSA**), usando o algoritmo descrito na seção 3.3 e apresentado na figura 3.2.

Além disso, foi seguido o cronograma constante na tabela 5.2, tal que os dados coletados na turma A servissem de base para a calibragem das questões para posterior utilização nas estratégias MSD e MSA.

Tabela 5.2: Etapas do experimento

Turma / Estratégia	**Semana 1**	**Semana 2**	**Semana 3**	**Semana 4**
A /MSR	Pré-teste	ADAPTFARMA	Pós-teste	
B / MSP		Pré-teste	ADAPTFARMA	Pós-teste
C / MSD		Pré-teste	ADAPTFARMA	Pós-teste
D / MSA		Pré-teste	ADAPTFARMA	Pós-teste

Na tabela 5.3 é mostrada a correspondência das ordenações dos exercícios do OA para as estratégias cujo sequenciamento é fixo para todos os alunos, isto é: MSP, MSD e MSR. Assim, por exemplo: o 1º exercício na ordem fixada pelo professor corresponde ao 2º na ordem por grau de dificuldade e ao 7º exercício pela ordenação randômica.

5.1.5 Resultados

A partir dos dados coletados nas 4 turmas procedeu-se à análise estatística dos resultados através da aplicação de diferentes métodos estatísticos, adequados às condições das amostras obtidas. Em todos os testes estatísticos foram adotados o nível de significância de $\alpha = 0,05$.

5.1.5.1 Métodos de Sequenciamento

O teste estatístico de Shapiro-Wilk foi aplicado aos dados de diferença média entre as notas do pós-teste e pré-teste para fins de testar a condição de normalidade, confirmada

Tabela 5.3: Correspondência das ordenações dos exercícios

MSP	**MSD**	**MSR**
1^{o}	2^{o}	7^{o}
2^{o}	9^{o}	25^{o}
3^{o}	1^{o}	13^{o}
4^{o}	23^{o}	12^{o}
5^{o}	20^{o}	5^{o}
6^{o}	21^{o}	19^{o}
7^{o}	14^{o}	28^{o}
8^{o}	13^{o}	23^{o}
9^{o}	12^{o}	16^{o}
10^{o}	18^{o}	11^{o}
11^{o}	15^{o}	14^{o}
12^{o}	30^{o}	2^{o}
13^{o}	16^{o}	18^{o}
14^{o}	7^{o}	4^{o}
15^{o}	3^{o}	17^{o}
16^{o}	4^{o}	29^{o}
17^{o}	22^{o}	8^{o}
18^{o}	19^{o}	21^{o}
19^{o}	11^{o}	10^{o}
20^{o}	5^{o}	24^{o}
21^{o}	24^{o}	3^{o}
22^{o}	28^{o}	6^{o}
23^{o}	17^{o}	26^{o}
24^{o}	29^{o}	9^{o}
25^{o}	26^{o}	15^{o}
26^{o}	6^{o}	30^{o}
27^{o}	25^{o}	27^{o}
28^{o}	10^{o}	22^{o}
29^{o}	8^{o}	20^{o}
30^{o}	27^{o}	1^{o}

somente para a turma C (método MSD). Em função disso, foi aplicado o teste T pareado de Student para a turma C. Para as demais, foi aplicado o teste de Wilcoxon, que não exije que os dados tenham distribuição normal para sua aplicação. A tabela 5.4 mostra a diferença média entre o pós-teste e pré-teste e o p-valor obtido em cada teste estatístico. Os valores em negrito indicam que a hipótese nula, denota por H_0, foi aceita.

Tabela 5.4: Resultado individual dos métodos de sequenciamento

		p-valor		
Tipo de Sequenciamento	Diferença média entre notas	Shapiro-Wilk	teste T Pareado	Wilcoxon
MSR	0,1759	0,0041	—	0,0007
MSP	0,2033	0,0039	—	<0,0001
MSD	0,0233	**0,0827**	**0,5302**	—
MSA	0,1400	0,0428	—	0,0037

As hipóteses dos testes estatísticos da tabela 5.4 são as seguintes:

- Shapiro-Wilk:

  H_0: *A diferença média entre as notas do pós-teste e pré-teste segue a distribuição normal.*

  H_0 é rejeitada quando p-valor $< 0,05$. Assim, rejeita-se H_0 para os métodos MSR, MSP e MSA. Aceita-se que a diferença média entre as notas do pós-teste e pré-teste segue a distribuição normal somente para o método MSD.

- teste t Pareado :

  H_0: *Não há diferença significativa entre as notas do pós-teste e pré-teste.*

  H_0 é aceita quando p-valor $> 0,05$. Assim, não se pode afirmar que houve ganho na aprendizagem para o grupo de alunos em que foi utilizado o método MSD.

- Wilcoxon:

  H_0: *Não há diferença significativa entre as notas do pós-teste e pré-teste.*

  H_0 é rejeitada quando p-valor $< 0,05$. Assim, aceita-se que houve ganho de aprendizagem para os grupos de alunos em que foram utilizados os métodos MSR, MSP e MSA.

Na tabela 5.5 são mostrados mostrados a média, desvio-padrão e coeficiente de variação do pré-teste, pós-teste e diferença entre pré e pós-teste para cada método de sequenciamento separado e também avaliados em conjunto.

Tabela 5.5: Média, desvio-padrão e coeficiente de variação

	Sequenciamento	Qtd Alunos	Média	Desvio-padrão	Coeficiente de Variação
Pré-testes	Geral	119	0,9580	0,3153	32,91%
	MSR	29	1,0414	0,2758	26,49%
	MSP	30	0,9067	0,3850	42,46%
	MSD	30	0,9000	0,2533	28,14%
	MSA	30	0,9867	0,3235	32,79%
Pós-testes	Geral	119	1,0933	0,3290	30,10%
	MSR	29	1,2172	0,2941	24,16%
	MSP	30	1,1100	0,3336	30,05%
	MSD	30	0,9233	0,2897	31,37%
	MSA	30	1,1267	0,3393	30,12%
Diferenças do Pré-teste para o Pós-teste	Geral	119	0,1353	0,2169	160,32%
	MSR	29	0,1759	0,2231	126,84%
	MSP	30	0,2033	0,1866	91,77%
	MSD	30	0,0233	0,2012	862,18%
	MSA	30	0,1400	0,2207	157,61%

Existem diferentes testes estatísticos para efetuar comparações entre múltiplas amostras. Para a identificação do teste adequado, primeiro é preciso aplicar os testes Shapiro-Wilk e Bartlett que verificam a condição de normalidade e igualdade de variâncias das amostras, respectivamente. No caso de se confirmarem ambas as condições, deve-se aplicar o teste ANOVA, caso contrário deve-se aplicar o teste Kruskal-Wallis. Assim, foi aplicado o teste ANOVA para o pré-teste, por terem sido aceitas as hipóteses de normalidade e igualdade de variâncias. Para os dados do pós-teste e diferença entre pós-teste e pré-teste foi aplicado o teste Kruskal-Wallis por ter sido rejeitada a hipótese de normalidade. A tabela 5.6 apresenta o p-valor obtido para cada situação. Os valores em negrito indicam que a hipótese nula, denota por H_0, foi aceita.

As hipóteses para os testes estatísticos da tabela 5.6 são as seguintes:

- Shapiro-Wilk:

Tabela 5.6: Comparação entre os métodos de sequenciamento

	Pressupostos ANOVA			
Comparação	Shapiro-Wilk	Bartlett	ANOVA	Kruskal-Wallis
Pré-testes	**0,1159**	**0,1150**	**0,2539**	—
Pós-testes	0,0011	**0,7646**	—	0,0056
Diferenças do pós-teste para o pré-teste	0,0062	**0,7566**	—	0,0307

H_0: *Os dados seguem a distribuição normal.*

H_0 é rejeitada quando p-valor $< 0,05$. Assim, aceita-se que somente os dados do pré-teste seguem a distribuição normal.

- Bartlett:

  H_0: *As variâncias das amostras são iguais.*

  H_0 é aceita se p-valor $> 0,05$. Assim, aceita-se que as variâncias são iguais para os dados do pré-teste, do pós-teste e também para os dados das diferenças entre pós-teste e pré-teste.

- ANOVA:

  H_0: *Não há diferença significativa nos dados do pré-teste para os diferentes métodos de sequenciamento, isto é, as diferentes amostras proveem da mesma população.*

  H_0 é aceita quando p-valor $> 0,05$. Assim, aceita-se que não há diferença significativa nas notas do pré-teste dos diferentes métodos de sequenciamento.

- Kruskal-Wallis:

  H_0: *Não há diferença significativa entre os métodos de sequenciamento.*

  Rejeita-se H_0 quando p-valor $< 0,05$. Assim, aceita-se que há diferença significativa entre os métodos de sequenciamento tanto para os dados do pós-teste quanto para a diferença entre pós-teste e pré-teste.

5.1.5.2 Notas Pré-teste e Pós-teste

Na figura 5.1 são apresentados os gráficos do tipo diagrama de caixa (*bloxplot*) para as notas do pré-teste, pós-teste e diferença entre pré e pós-teste para cada um dos métodos de sequenciamento. Nesse tipo de diagrama, as informações estatísticas têm a seguinte representação:

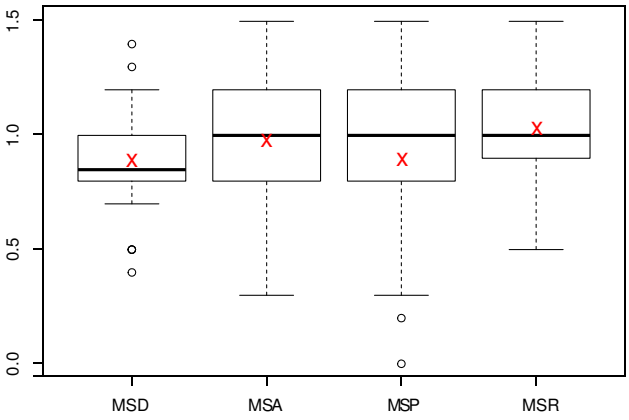
- mediana: linha que secciona o retângulo;
- média: x dentro do retângulo;
- quartis Q_1 e Q_3: formam a altura do retângulo;
- cauda da distribuição (*wisker*): comprimento das linhas fora do retângulo;
- valores atípicos: pontos fora dos limites indicados pela cauda da distribuição.

Observando a figura 5.1(a) e (b), é possível notar que a distribuição das notas do pré-teste para a turma em que foi utilizado MSD é a mais assimétrica, enquanto isso também ocorre para a distribuição das notas do pós-teste para MSR. Além disso, as medianas das notas do pós-teste para MSA, MSP e MSR são bastante próximas.

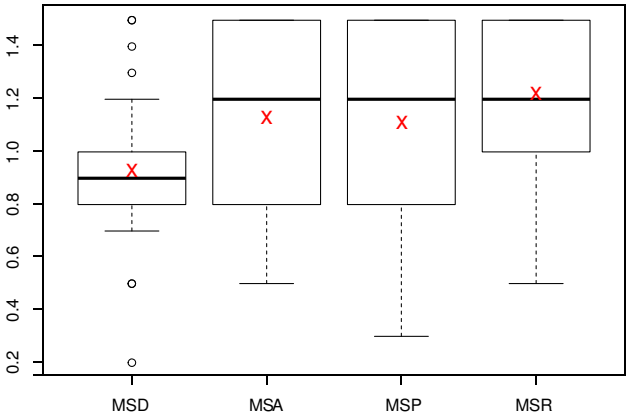
Analisando a figura 5.1(c), que apresenta a distribuição da diferença entre as notas do pós-teste e pré-teste, que reflete o ganho de aprendizagem, vê-se que MSA, MSP e MSR são similares enquanto MSD teve pior desempenho.

5.1.5.3 *Rating*

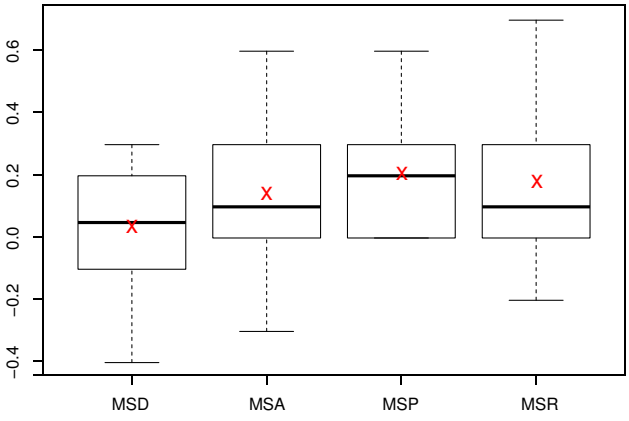
Os gráficos apresentados na figura 5.2 apresentam a relação entre a nota do pós-teste e o *rating* calculado para cada tipo de sequenciamento, e também o valor do coeficiente de Spearman (r), que avalia a correlação entre as duas variáveis. A correlação se mostrou fraca em todos os tipos de sequenciamento, pois os valores de r, indicados no canto inferior esquerdo dos gráficos, são menores que $0,5$.



(a) Pré-teste

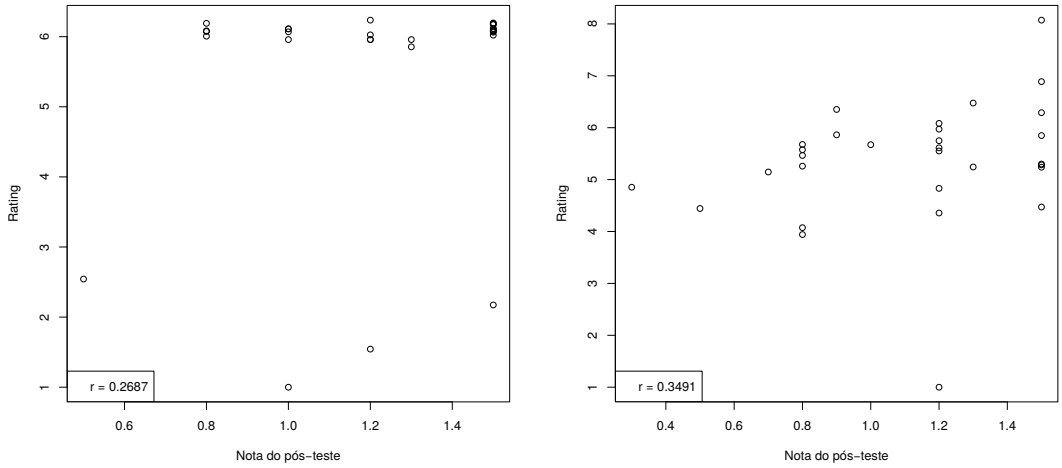


(b) Pós-teste

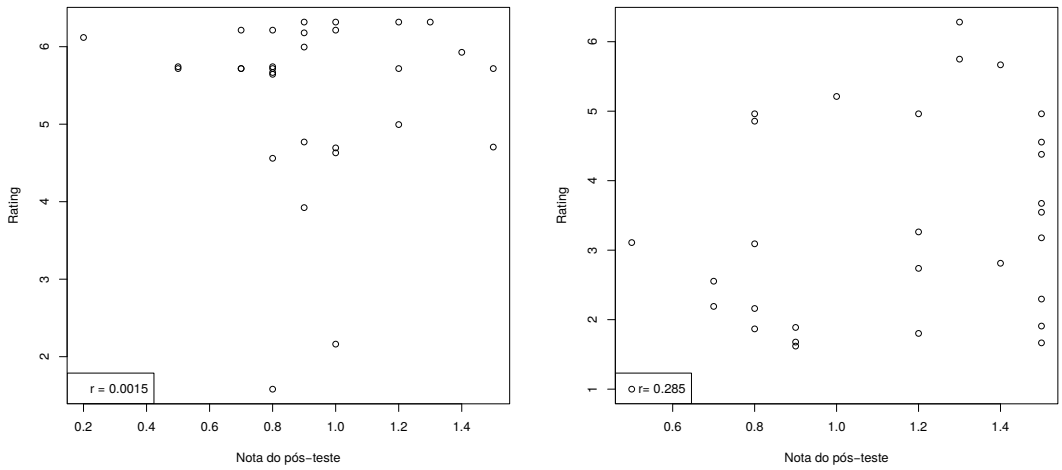


(c) Diferença entre Pré-teste e Pós-teste

Figura 5.1: Diagrama *Boxplot*



(a) MSR (b) MSP



(c) MSD (d) MSA

Figura 5.2: Nota do Pós-teste X *Rating*

5.2 Análise dos Resultados

5.2.1 Análise Quantitativa

Na avaliação individual dos métodos de sequenciamento de acordo com os p-valores encontrados e apresentados na tabela 5.4, todos os métodos, com exceção de MSD, tiveram aumento significativo na nota do pós-teste em relação à nota do pré-teste. Quando se compara os métodos de sequenciamento a partir dos p-valores obtidos e mostrados na tabela 5.6 é possível observar que:

- não se pode afirmar que há diferença significativa entre as 4 turmas para os dados do pré-teste, pois o método ANOVA obteve p-valor $= 0,2539$;
- há diferença significativa para os dados do pós-teste, pois p-valor $= 0,00579$ para o método Kruskal-Wallis;
- há diferença significativa para os dados da diferença entre pré e pós-teste, pois p-valor $= 0,03073$ para o método Kruskal-Wallis;
- MSR, MSP e MSA levam o estudante a obter uma melhor performance que MSD;
- a performance entre MSR, MSP e MSA foram similares.

Surpreendemente, MSR obteve a melhor performance enquanto MSD a pior. Isto contradiz um grande número de pesquisas referenciadas na literatura sobre práticas pedagógicas usando computadores [22] ou por outro lado, para desenvolvimento de habilidades de solução de problemas. Algumas razões que podem explicar esse fenômeno são:

- o problema da ordenação de exercícios é uma questão relevante que deve ser explorada mais detalhadamente para permitir a verificação da influência tácita do conhecimento contido na organização textual do problema;
- a falta de diferença significativa entre MSR, MSP e MSA é também confirmada por evidências achadas em pesquisas passadas, como no experimento descrito em [32, 38];

- o MSR pode ter obtido o melhor desempenho devido ao fato de que era permitido ao estudante navegar livremente entre os exercícios, podendo assim resolvê-los em uma ordem personalizada que julgasse adequada.

5.2.2 Limitações do Experimento

É importante ressaltar alguns aspectos relacionados ao controle do experimento:

- pré-testes e pós-testes: sua elaboração não foi acompanhada pelo pesquisador, sendo deixada a cargo do professor da disciplina. Porém, ficou acordado que seu conteúdo deveria versar estritamente sobre o mesmo assunto do OA. Foi solicitado e aceito pelo professor que ambos testes fizessem parte da avaliação oficial da disciplina;

- OA: o professor foi o responsável pela criação do OA sem interferência do pesquisador, sendo apenas acordado que o número mínimo de exercícios seria 30 e cada exercício deveria conter exatamente uma única questão;

- tempo disponível da ferramenta: inicialmente estava planejado que os alunos deveriam ter acesso ao ambiente ADAPTFARMA somente durante as aulas em laboratório. Porém, o professor da disciplina solicitou que os alunos pudessem usar a ferramenta fora do horário de aula, porque o tempo que poderia dispor em laboratório para essa atividade (4 horas-aulas distribuídas em dias alternados) não seria suficiente para a resolução de todos os exercícios;

- opcionalidade do uso da ferramenta: o uso do ambiente ADAPTFARMA era opcional. Caso o estudante não desejasse utilizá-lo, não sofreria nenhuma penalidade na nota do seu registro acadêmico;

- cópia indevida: não foi possível garantir a impossibilidade de um aluno passar respostas de exercícios para outro(s) aluno(s), já que era possível o uso do ambiente ADAPTFARMA fora do horário de aula.

CAPÍTULO 6

CONSIDERAÇÕES FINAIS E TRABALHOS FUTUROS

Em vários domínios do conhecimento, a perícia do estudante geralmente é desenvolvida através da resolução de exercícios que requerem um conjunto de habilidades avaliadas, inclusive em STIs. Este trabalho propôs um sistema de *rating* para avaliar automaticamente os estudantes. Dependendo do número de tentativas e o grau de dificuldade da questão, estudantes que acertam a mesma questão são pontuados diferentemente. Também foram apresentadas duas fórmulas de cálculo do grau de dificuldade de uma questão.

O sistema de *rating* implicitamente estabelece um *ranking* entre os estudantes e pode servir como ferramenta para o professor identificar quem precisa de mais assistência. Além disso, o professor também pode avaliar o grau de dificuldade das questões que elabora.

Aspectos relevantes em STIs foram abordados. Os exercícios devem ser apresentados aos estudantes em que ordem? Quantas vezes um mesmo exercício pode ser exibido? Este trabalho propôs um algoritmo para sequenciamento de exercícios que usa o grau de dificuldade combinado com um mecanismo similar à interpolação numérica, visando exibir um exercício com o grau de dificuldade adequado à performance do aluno. Esse algoritmo foi implementado para compor o ambiente ADAPTFARMA, uma ferramenta Web de autoria de OAs para criação e execução de OAs.

ADAPTFARMA está implementada de tal forma que é simples a alteração da estratégia de sequenciamento de exercícios. Aproveitando essa característica, foi realizado um experimento com alunos reais para testar diferentes estratégias de sequenciamento de exercícios. Os resultados do experimento mostraram que nem todas as estratégias de sequenciamento levaram a aumento significativo nas notas dos alunos, o que indica que o sequenciamento de exercícios é uma questão que merece ser melhor investigada visando a construção de STIs mais eficazes. A partir dos resultados do experimento, também foi observado um aumento significativo nas notas dos alunos que utilizaram o ambiente

ADAPTFARMA equipado com o algoritmo de sequenciamento proposto, mostrando sua efetividade.

6.1 Retrospectiva

Na resenha literária procurou-se apresentar uma visão geral dos componentes da arquitetura clássica de STIs, e principais técnicas usadas na construção do modelo do estudante. Também foi apresentado um estudo de trabalhos relacionados cujo foco é a personalização, avaliação do aprendiz, calibragem de tarefas e seu sequenciamento em STIs.

Em seguida, foram apresentados os conceitos fundamentais da solução: como calcular o *rating* dos alunos, a calibragem das questões, a proposta do algoritmo de sequenciamento adaptivo de exercícios, bem como a arquitetura e implementação da ferramenta ADAPTFARMA.

A validação inicial das fórmulas de *rating* e grau de dificuldade das questões foram feitas através de um estudo empírico, utilizando-se dados de alunos reais coletados a partir de trabalhos anteriores de outros pesquisadores [7, 33].

Posteriormente, foi realizado um experimento que contou com a participação de 119 alunos de uma escola pública do Ensino Médio. Ele teve a finalidade de avaliar a eficácia do algoritmo de sequenciamento adaptivo de exercícios aqui proposto na aprendizagem dos alunos. Os dados coletados foram analisados mediante testes estatísticos apropriados e apresentados neste trabalho.

O estudo realizado tem a limitação de que o *rating* calculado do aluno não é aproveitado como uma avaliação inicial em outros OAs que ele venha utilizar. É ainda necessário verificar a adequação da fórmula para essa situação. Outra limitação é em relação ao algoritmo de sequenciamento adaptativo de exercícios, que não tem um tratamento especial para considerar agrupamentos de subtópicos do assunto tratado no OA.

6.2 Trabalhos futuros

Ao longo da pesquisa, foi possível identificar aspectos a serem melhorados ou expandidos, incluindo novas funcionalidades e aplicações do estudo desenvolvido. São apresentados a seguir trabalhos futuros que podem ser realizados nesse sentido.

6.2.1 Implementação de MSA considerando Agrupamentos de Subtópicos

A implementação atual do MSA de exercícios não leva em conta os diferentes subtópicos abordados dentro de um mesmo OA, tratando todos os exercícios independentemente. Por exemplo, dentro do assunto "Propriedades de Logaritmos", pode-se elencar os seguintes subtópicos: propriedade do produto do logaritmo, propriedade do quociente do logaritmo, propriedade de mudança de base, propriedade da potência do logaritmo, propriedade da raiz de um logaritmo. Dependendo da quantidade de exercícios de cada subtópico e sua distribuição após a calibragem, pode ocorrer que algum subtópico não figure entre os exercícios que compõem a sequência mínima, e portanto não seja apresentado ao aluno. Esse tipo de situação pode ser contornado acrescentando-se ao algoritmo do MSA a informação do subtópico abordado em cada exercício e garantindo a participação de pelo menos um exercício de cada subtópico na sequência mínima. A implementação dessa funcionalidade requer alterações no módulo de autoria de ADAPTFARMA, que permitam ao autor especificar os subtópicos em cada exercício elaborado.

6.2.2 Personalização via Exercícios Parametrizados

Atualmente, o módulo de autoria de ADAPTFARMA é basicamente o mesmo de FARMA [33]. Agregar uma funcionalidade que permita a criação de exercícios parmetrizados traz os seguintes benefícios:

- ao invés de repetir exatamente o mesmo exercício nas suas diversas tentativas, a apresentação do mesmo exercício com dados diferentes pode contribuir para diminuir a sensação de monotonia do estudante, facilitando assim seu engajamento na

atividade proposta;

- se os dados parametrizados forem gerados de maneira aleatória, inibe-se o processo de cópia indevida ("cola") entre os alunos;
- torna-se mais fácil confirmar se o estudante adquiriu certas habilidades ao ter a possibilidade de repetir o mesmo exercício com dados diferentes;
- permite um certo grau de individualização ao se ter a possibilidade de apresentar o mesmo problema de maneira diferente dependendo de quantos parâmetros são informados.

6.2.3 Montagem Automática de OAs

Na versão atual de ADAPTFARMA, os exercícios estão atrelados ao OA, não podendo ser visualizados ou consultados isoladamente. Porém, com algumas alterações no modelo de banco de dados será possível armazená-lo associado ao tópico ou subtópico em um repositório compartilhado, possibilitando agregar ao módulo de autoria a funcionalidade de geração automática de OAs a partir dos seguintes parâmetros fornecidos pelo professor-autor:

- os tópicos ou subtópicos com o respectivo número de exercícios;
- o intervalo do grau de dificuldade $[d_{min}, d_{max}]$ dos exercícios que irão compor o OA;
- o tempo médio máximo de resolução dos exercícios.

Esse tipo de funcionalidade permite diminuir bastante o tempo de autoria bem como diversificar os OAs gerados para o mesmo domínio.

Neste trabalho, foi implementado o mecanismo para calibragem automática dos exercícios utilizando a fórmula 3.3 para o cálculo do grau de dificuldade, atualizado a cada aplicação do OA. Além disso, ADAPTFARMA registra o *timestamp* de cada uma das respostas dos alunos, tornando assim possível calcular o tempo gasto para resolução de cada questão pela diferença entre os *timestamps* de uma questão e sua imediata anterior resolvida. Portanto, para a implementação automática de OAs ainda será necessário criar mecanismo

para efetuar o cálculo do tempo médio de resolução e fazer alterações no módulo de autoria para a inclusão da informação de subtópicos conceituais associados a cada exercício.

BIBLIOGRAFIA

[1] Sami Abuhamdeh e Mihaly Csikszentmihalyi. The importance of challenge for the enjoyment of intrinsically motivated, goal-directed actitivies. *Personality and Social Psychology Bulletin*, 38(3):317–330, 2012.

[2] John R. Anderson. *Psicologia Cognitiva e suas Implicações Experimentais*. LTC, 2004.

[3] Frank B. Baker. *The Basics of Item Response Theory*. ERIC Clearinghouse on Assessment and Evaluation, University of Wiscosin, 2001.

[4] Ryan S.J.d. Baker, Adam B. Goldstein, e Neil T. Heffernan. Detecting the Moment of Learning. *LNCS*, 6094(PART I):25–34, 2010. Springer-Verlag Berlin Heidelberg.

[5] Alexandre F. Barbosa, editor. *Pesquisa sobre o uso das tecnologias de informação e comunicação nas escolas brasileiras: TIC Educação 2013*. Comitê Gestor da Internet no Brasil, São Paulo, 2014.

[6] Paulo E. Battistella, Christiane G. von Wangenheim, e João M. Fernandes. Como jogos educacionais são desenvolvidos? Uma revisão sistemática da literatura. *WEI - XXII Worshop sobre Educação em Computação - CSBC 2014*, páginas 1445–1455. SBC, 2014.

[7] Gustavo Bazzo, Alexandre Direne, e Diego Marczal. Classificação automática de erros de aprendizes humanos do processo de indução analítica. *Anais do XXII SBIE - XVII WIE*, páginas 130–139, 2011.

[8] Benjamin S. Bloom, J. Thomas Hastings, e George F. Madaus. *Manual de Avaliação Formativa e Somativa do Aprendizado Escolar*. Pioneira, São Paulo, 1983.

[9] Sébastien Bubeck e Nicolò Cesa-Bianchi. Regret Analysis of Stochastic and Nonstochastic Multi-armed Bandit Problems. *Foundations and Trends in Machine Learning*, 5(1):1–122, 2012.

[10] Hao Cen. *Generalized Learning Factors Analysis: Improving Cognitive Models with Machine Learning*. PhD Thesis, Carnegie Mellon University, abril de 2008.

[11] Hao Cen, Kenneth Koedinger, e Brian Junker. Comparing Two IRT Models for Conjunctive Skills. volume 5091 of *Lecture Notes in Computer Science*, páginas 796–798. Springer Berlin Heidelberg, julho de 2008.

[12] Suleyman Cetintas, Luo Si, YanPing Xin, e Casey Hord. Predicting Correctness of Problem Solving in ITS with a Temporal Collaborative Filtering Approach. Vincent Aleven, Judy Kay, e Jack Mostow, editores, *Intelligent Tutoring Systems*, volume 6094 of *Lecture Notes in Computer Science*, páginas 15–24. Springer Berlin Heidelberg, 2010.

[13] John Champaign e Robin Cohen. A Model for Content Sequencing in Intelligent Tutorign Systems Based on the Ecological Approach and Its Validation Through Simulated Students. páginas 486–491. Association for the Advancement of Artificial Intelligence (AAAI), 2010.

[14] Benjamin Clement, Didier Roy, e Pierre-Yves Oudeyer. Online Optimization of Teaching Sequences with Multi-Armed Bandits. J. Stamper, Z. Pardos, M. Mavrikis, e B. M. McLaren, editores, *Proceedings of the 7th International Conference on Educational Data Mining*, páginas 269–272, 2014.

[15] Cristina Conati. Intelligent Tutoring Systems: New Challenges and Directions. *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI'09)*, páginas 2–7, San Francisco, 2009.

[16] Mihaly Csikszentmihalyi. *A Descoberta do Fluxo: a Psicologia do Envolvimento com a Vida Cotidiana*. Rocco, Rio de Janeiro, 1999.

[17] Randall Davis e Bruce Buchanan. Production Rules as a Representation for a knowledge-Based Consultation Program. *Artificial Intelligence*, (8):15–45, 1977. North-Holland.

[18] Marco Aurelio de Carvalho. SIAI - Sequenciador Inteligente de Atividades na Internet. *Anais do XXII SBIE - XVII WIE*, páginas 50–59, 2011.

[19] Renato Luís de Souza Dutra, Liane Margarida Rockenbach Tarouco, e Liliana Passerino. Avaliação Formativa usando Objetos de Aprendizagem SCORM. *Novas Tecnologias na Educação*, 6(1), julho de 2008.

[20] Maunendra Sankar Desarkar e Sudeshna Sarkar. Rating Prediction Using Preference Relations Based Matrix Factorization. Eelco Herder, Kalina Yacef, e Stephan Weibelzahl, editores, *Workshop and Poster Proceedings of the 20th Conference on User Modeling, Adaptation, and Personalization (UMAP2012)*, volume 872. http://http://ceur-ws.org/, julho de 2012.

[21] Michel C. Desmarais e Ryan S. J. d. Baker. A Review of Recent Advances in Learner and Skill Modeling in Intelligent Learning Environments. *User Modeling and User-Adapted Interaction*, 22(1-2):9–38, 2011.

[22] Alexandre Direne. Authoring Intelligent Systems for Teaching Visual Concepts. *International Journal of Artificial Intelligence in Education*, 1(4):3–14, 1990.

[23] Mingyu Feng. *Towards Assessing Students' Fine Grained Knowledge: Using an Intelligent Tutor for Assessment*. Phd thesis, Worcester Polytechnic Institute, 2009.

[24] Kimberly Ferguson, Beverly Woolf, e Andy Barto. Improving Intelligent Tutoring Systems: Using Expectation Maximization to Learn Student Skill Levels. Technical Report TR-06-09, University of Massachusetts Amherst, Amhrest, MA, 2009.

[25] Yue Gong, Joseph E. Beck, e Neil T. Heffernan. Comparing Knowledge Tracing and Performance Factor Analysis by Using Multiple Model Fitting Procedures. *LNCS*, 6094(PART I):35–44, 2010. Springer-Verlag.

[26] Korhan Günel e Rifat Asliyan. Determining Difficulty of Questions in Intelligent Tutoring Systems. *The Turkish Online Journal of Educational Technology - TOJET*, volume 8, 2009.

[27] Eduardo Guzmán e Ricardo Conejo. Measuring Misconceptions Through Item Response Theory. Cristina Conati, Neil Heffernan, Antonija Mitrovic, e M. Felisa Verdejo, editores, *Proceedings of 17th International Conference on Artificial Intelligence in Education (AIED 2015)*, volume 9112 of *Lecture Notes in Artificial Intelligence*, páginas 608–611. Springer International Publishing Switzerland, junho de 2015.

[28] Seiji Isotani, Deanne Adams, Richard E. Mayer, Kelley Durkin, Bethany Rittle-Hohnson, e Bruce M. McLaren. Can Erroneous Examples Help Middle-School Students Learn Decimals? volume 6964 of *Lecture Notes in Computer Science*, páginas 181–195, Palermo, 2011. Springer Berlin Heidelberg.

[29] Aleksandra Klašnja-Milićević, Boban Vesin, Mirjana Ivanocić, e Zoran Budimac. E-learning personalization based on hybrid recommendation strategy and learning style identification. *Computers & Education*, 56(3):885–899, 2011.

[30] Nan Li, William W. Cohen, e Kenneth R. Koedinger. Discovering Student Models with a Clustering Algorithm Using Problem Content. S. K. D'Mello, R. A. Calvo, e A. Olney, editores, *Proceedings of the 6th international conference on Educational Data Mining (EDM2013)*, páginas 98–105, Memphis-TN, julho de 2013. International Educational Data Mining Society.

[31] Christian List. Social Choice Theory. Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Winter 2013 edition, 2013. http://plato.stanford.edu/archives/win2013/entries/social-choice/.

[32] Nigel Major e Han Reichgelt. COCA: A Shell for Intelligent Tutoring Systems. Claude Frasson, Gilles Gauthier, e Gordon McCalla, editores, *Intelligent Tutoring Systems*, volume 608 of *Lecture Notes in Computer Science*, páginas 523–530. Springer Berlin Heidelberg, 1992.

[33] Diego Marczal. FARMA: Uma ferramenta de autoria para objetos de aprendizagem de conceitos matemáticos. Tese de doutorado, Universidade Federal do Paraná, UFPR, 2014.

[34] Diego Marczal e Alexandre Direne. Um arcabouço que enfatiza a retroação a contextos de erro na solução de problemas. *Revista Brasileira de Informática na Educação*, 19(01):63, 2011.

[35] Diego Marczal e Alexandre Direne. FARMA: Uma ferramenta de autoria para objetos de aprendizagem de conceitos matematicos. *Anais do Simposio Brasileiro de Informatica na Educacao*, volume 23, 2012.

[36] Bruce M. McLaren, Deanne Adams, Kelley Durkin, George Goguadze, Richard E. Mayer, Bethany Rittle-Hohnson, Sergey Sosnovsky, Seiji Isotani, e Martin van Velsen. To Err is Human, to Explain and Correct Is Divine: A Study of Interactive Erroneous Examples with Middle School Math Students. volume 7563 of *Lecture Notes in Computer Science*, páginas 222–235, Saarbrücken, 2012. Springer Berlin Heidelberg.

[37] Danielle S. McNamara, G. Tanner Jackson, e Arthur Graesser. Intelligent Tutoring and Games (ITaG). H Chad Lane, Amy Ogan, e Valerie Shute, editores, *AIED 2009 Workshops Proceedings Volume 3: Intelligent Educational Games*, páginas 1–10, julho de 2009.

[38] Antonija Mitrovic. An intelligent SQL tutor on the web. *International Journal of Artificial Intelligence in Education*, 13(3):173–197, 2003.

[39] Antonija Mitrovic, Kenneth R. Koedinger, e Brent Martin. A Comparative Analysis of Cognitive Tutoring and Constraint-Based Modeling. Peter Brusilovsky, Albert Corbett, e Fiorella Rosis, editores, *User Modeling 2003, 9th International Conference (UM 2003)*, volume 2702, páginas 313–322, Johnstown-PA, junho de 2003. Springer Berlin Heidelberg.

[40] David J. Nicol e Debra Macfarlane-Dick. Formative assessment and self-regulated learning: a model and seven principles of good feedback pratice. *Studies in Higher Education*, 31(2):199–218, abril de 2006.

[41] Roger Nkambou, Jacqueline Bourdeau, e Riichiro Mizoguchi. *Advances in Intelligent Tutoring Systems*, volume 308 of *Studies in Computational Intelligence*. Springer, 2010.

[42] Hyacinth S. Nwana. Intelligent Tutoring Systems: an overview. *Artificial Intelligence Review*, 4(4):251–277, 1990.

[43] Jan Papoušek e Radek Pelánek. Impact of Adaptive Educational System Behaviour on Student Motivation. volume 9112 of *Lectures Notes in Artificial Intelligence*, páginas 348–357, Madrid, junho de 2015. Springer.

[44] Andrey R. Pimentel e Alexandre I. Direne. Medidas cognitivas para o ensino de conceitos visuais com sistemas tutores inteligentes. Raul Sidnei Wazlawick, editor, *Revista Brasileira de Informática na Educação*, volume 1, 1997.

[45] Niels Pinkwart e Frank Loll. Comparing Three Approaches to Assess the Quality of Students' Solutions. Darina Dicheva, Riichiro Mizoguchi, e Niels Pinkwart, editores, *AIED 2009 Workshops Proceedings Volume 2, SWEL'09: Ontologies and Social Semantic Web for Intelligent Educational Systems Intelligent Educational Games*, páginas 81–85, julho de 2009.

[46] Gautham Adithya Ravi e Sergey Sosnovsky. Exercise Difficulty Calibration Based on Student Log Mining. F. Mödritscher, V. Luengo, E. Lai-Chong Law, e U. Hoppe, editores, *Proceedings of DAILE'13: Workshop on Data Analysis and Interpretation for Learning Environments*, Villard-de-Lans (France), janeiro de 2013.

[47] Rosa M. García Rioja, Sergio Gutierrez Santos, Abelardo Pardo, e Carlos Delgado Kloos. A Parametric Exercise Based Tutoring System. *33rd Annual Frontiers in Education, 2003 (FIE 2003)*, volume 3, páginas S1B_20–S1B_26, novembro de 2003.

[48] Steven Ritter, John R. Anderson, Kenneth R. Koedinger, e Albert Corbett. Cognitive Tutor: Applied research in mathematics education. *Psychonomic bulletin & review*, 14(2):249–255, 2007.

[49] Stuart J. Russell e Peter Norvig. *Artificial Intelligence: A Mordern Approach*. Prentice Hall in Artificial Intelligence. Pearson Education, Upper Saddle River, New Jersey, 2010.

[50] Gilvandenys Leite Sales, Giovanni Cordeiro Barroso, e José Marques Soares. Um Indicador de Aprendizagem Não-linear para EaD online Fundamentado no Modelo de Avaliação Learning Vectors (LV). *Anais do XXII SBIE - XVII WIE*, páginas 1713–1722, 2011.

[51] Rafael Savi e Vania Ribas Ulbricht. Jogos Digitais Educacionais: Benefícios e Desafios. *RENOTE - Novas Tecnologias na Educação*, 6(2), dezembro de 2008.

[52] Carlotta Schatten e Lars Schmidt-Thieme. Adaptive Content Sequencing without Domain Information. *6th International Conference on Computer based Education*, abril de 2014.

[53] Henrique M. Seffrin, Geiseane Rubi, e Patricia Jaques. Uma Rede Bayesiana aplicada à Modelagem do Conhecimento Algébrico do Aprendiz. *II Congresso Brasileiro de Informática na Educação (CBIE2013), XXIV Simpósio Brasileiro de Informática na Educação (SBIE 2013)*, (CBIE):597–606, novembro de 2013.

[54] Avi Segal, Ziv Katzir, Kobi Gal, Guy Shani, e Bracha Shapira. EduRank: A Collaborative Filtering Approach to Personalization in E-learning. J. Stamper, Z. Pardos, M. Mavrikis, e B.M. B. M. McLaren, editores, *Proceedings of the 7th International Conference on Educational Data Mining*, páginas 68–75, 2014.

[55] James R. Segedy, Kirk M. Loretz, e Gautam Biswas. Suggest-Assert-Modify: A Taxonomy of Adaptive Scaffolds in Computer-Based Learning Environments. Gautam Biswas, Roger Azevendo, Valerie Shute, e Susan Bull, editores, *AIED 2013*

Workshops Proceedings Volume 2: Scaffolding in Open-Ended Learning Environments (OELEs), páginas 73–80, julho de 2013.

[56] John Self. Student models: What use are they? *Artificial Intelligence in Education*, 1988.

[57] John Self. Bypassing the Intractable Problem of Student Modelling. (41):1–26, 1990. AAI/AI-ED.

[58] Maddalena Taras. Assessment - Summative and Formative - Some Theoretical Reflections. *British Journal of Educational Studies*, 53(4):466–478, dezembro de 2005.

[59] Nguyen Thai-Nghe. *Predicting Student Performance in a Intelligent Tutoring System.* Tese de Doutorado, Department of Computer Science, Information Systems and Machine learning Lab (ISMLL), University of Hildesheim, Germany, 2011.

[60] Kurt Vanlehn. The behavior of tutoring systems. *International Journal of Artificial Intelligence*, 16(3):227–265, 2006.

[61] Etienne Wenger. *Artificial Intelligence and Tutoring Systems: Computational and Cognitive Approaches to the Communication of knowledge.* Morgan Kaufmann, 1987.

[62] David A. Wiley. *The Instructional Use of Learning Objects: Online Version*, capítulo Connecting learning objects to instructional design theory: a definition, a metaphor, and a taxonomy. 2000. http://reusability.org/read/chapters/wiley.doc.

[63] Beverly Park Woolf. *Building Intelligent Interactive Tutors: Student-centered strategies for revolutionizing e-learning.* Morgan kaufmann, 2008.

[64] Yanbo Xu e Jack Mostow. Using Logistic Regression to Trace Multiple Subskillsin a Dynamic Bayes Net. páginas 241–245, Eindhoven, Holanda, julho de 2011. International Educational Data Mining Society.

APÊNDICE A

CÓDIGO SQL PARA CÁLCULO DE *RATING*

```
 ------------------------------------------------------------------------
-- Routine DDL
-- ------------------------------------------------------------------------
DELIMITER $$
CREATE PROCEDURE `calcula_rating_aluno_questao`(IN id_aluno BIGINT(20),
                                                IN id_questao_final INT(11))
BEGIN
  DECLARE qtd_tentativas_aluno int(11);
  DECLARE qtd_max_tentativas_questao int(11);
  DECLARE qtd_med_tentativas_questao float;
  DECLARE qtd_alunos_acertaram int;
  DECLARE qtd_alunos_erraram int;
  DECLARE questao_atual int;
  DECLARE total_alunos int;
  DECLARE acertou_questao int;
  DECLARE A int; -- 1 quando acertou e 0 quando errou
  DECLARE E int; -- 1 quando errou  e  1 quando errou
  DECLARE rating_anterior float;
  DECLARE rating_novo float;
  DECLARE k1 float;  -- fator de incremento de rating
  DECLARE k2 float;  -- fator de decremento de rating
  DECLARE no_more_rows BOOLEAN;
  DECLARE msgTraceLog varchar(500);

  DECLARE cur_questao cursor for
  select id_questao from maxiambiental3.dificuldade_questao
   where id_questao <= id_questao_final
   order by id_questao;

  DECLARE CONTINUE HANDLER FOR NOT FOUND
    SET no_more_rows = TRUE;

  select count(distinct id) into total_alunos
  from maxiambiental3.users;

  select rating_atual into rating_anterior
  from maxiambiental3.studentrating
  where user_id = id_aluno;
```

```
  OPEN cur_questao;
the_loop: LOOP
  FETCH cur_questao into questao_atual;
  IF no_more_rows THEN
        CLOSE cur_questao;
        LEAVE the_loop;
  END IF;

  select COALESCE(count(id),0) into qtd_tentativas_aluno
  from maxiambiental3.erros
  where user_id = id_aluno and id_questao = questao_atual;

  select COALESCE(count(distinct user_id),0) into qtd_alunos_acertaram
  from maxiambiental3.hits
  where id_questao = questao_atual;
  set qtd_alunos_erraram = total_alunos - qtd_alunos_acertaram;

  -- obter a mediana do n\'famero de tentativas
  select COALESCE(qtd_med_tentativas,0) into qtd_med_tentativas_questao
  from med_tentativas
  where id_questao = questao_atual;

  IF qtd_med_tentativas_questao is null
     THEN set qtd_med_tentativas_questao = 1;
  END IF;
  IF qtd_med_tentativas_questao = 0
     THEN set qtd_med_tentativas_questao = 1;
       set qtd_tentativas_aluno = 1;
  END IF;
  IF qtd_med_tentativas_questao > 10 -- valor discrepante
     THEN
         set qtd_med_tentativas_questao = 10;
  END IF;
  IF qtd_tentativas_aluno > 10
     THEN
         set qtd_tentativas_aluno = 10;
  END IF;
  IF qtd_alunos_acertaram = 0
     THEN
      set qtd_alunos_acertaram = 1;
  END IF;
  IF qtd_alunos_erraram = 0
     THEN
          set qtd_alunos_erraram = 1;
  END IF;
  select COALESCE(count(id),0) into acertou_questao
  from maxiambiental3.hits
  where user_id = id_aluno and id_questao = questao_atual;
```

```
  IF acertou_questao > 0 -- aluno acertou a questao
     THEN
       set A = 1;
       set E = 0;
  ELSE
    set A = 0;
    set E = 1;
   END IF;
  SET k1 = (1-rating_anterior/10);
  SET k2 = ((rating_anterior - 1)/10);
  SET rating_novo = rating_anterior +
                    A*k1*(1/qtd_alunos_acertaram)*
                    (10 - 9*qtd_tentativas_aluno/qtd_med_tentativas_questao) -
                    E*k2*(1/qtd_alunos_erraram)*
                    (10*qtd_tentativas_aluno/qtd_med_tentativas_questao);
  set rating_anterior = rating_novo;
 END LOOP the_loop;
  update maxiambiental3.studentrating
    set rating_atual = rating_novo
   where user_id = id_aluno;
END
}
```

APÊNDICE B
CÓDIGO SQL PARA CÁLCULO DE *RATING* CONSIDERANDO *TIMESTAMP*

```
-- -----------------------------------------------------------------------------
-- Routine DDL
-- -----------------------------------------------------------------------------
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE `calcula_rating_ordem_tempo`()
BEGIN
  DECLARE id_aluno_param VARCHAR(50);
  DECLARE qtd_tentativas_aluno int;
  DECLARE qtd_max_tentativas_questao int;   -- maior número de tentativas
                                            -- até o momento
  DECLARE qtd_med_tentativas_questao float; -- mediana do número de tentativas
                                            -- até o momento
  DECLARE qtd_alunos_acertaram int;
  DECLARE qtd_alunos_erraram int;
  DECLARE ch_resposta_atual int;
  DECLARE ord_questao_atual varchar(2);
  DECLARE respostaCorreta VARCHAR(5);
  DECLARE total_alunos int;
  DECLARE dataOcorrenciaAtual DATE;
  DECLARE horaOcorrenciaAtual TIME;
  DECLARE A int; -- 1 quando acertou e 0 quando errou
  DECLARE E int; -- 1 quando errou  e  1 quando errou
  DECLARE rating_anterior float;
  DECLARE rating_novo float;
  DECLARE k1 float;  -- fator de incremento de rating
  DECLARE k2 float;  -- fator de decremento de rating
  DECLARE no_more_rows BOOLEAN;

  DECLARE cur_questao cursor for
  select ch_resposta, ord_questao, ID_Aluno, DataOcorrencia,
         HoraOcorrencia, Correta
  from invencaologaritmos.respostasexercicios
  order by DataOcorrencia, HoraOcorrencia;

  DECLARE CONTINUE HANDLER FOR NOT FOUND
    SET no_more_rows = TRUE;

  update ratingaluno
    set rating_inicial = 5.5,
```

```
  rating_atual = 5.5;
  delete from invencaologaritmos.tracelog;

  select count(distinct ch_aluno) into total_alunos
  from invencaologaritmos.ratingaluno;

OPEN cur_questao;
the_loop: LOOP
  FETCH cur_questao into ch_resposta_atual, ord_questao_atual,
        id_aluno_param, dataOcorrenciaAtual, horaOcorrenciaAtual,
        respostaCorreta;
  IF no_more_rows THEN
        CLOSE cur_questao;
        LEAVE the_loop;
  END IF;
  select rating_atual into rating_anterior
  from invencaologaritmos.ratingaluno
  where ch_aluno = id_aluno_param;

  select COALESCE(count(ch_resposta),0) into qtd_tentativas_aluno
  from invencaologaritmos.respostasexercicios
  where ID_Aluno = id_aluno_param and ord_questao = ord_questao_atual
        and Correta = 'FALSE' and DataOcorrencia <= dataOcorrenciaAtual
        and HoraOcorrencia <= horaOcorrenciaAtual;

  select COALESCE(count(distinct ID_Aluno),0) into qtd_alunos_acertaram
  from invencaologaritmos.respostasexercicios
  where ord_questao = ord_questao_atual and Correta = 'TRUE'
        and DataOcorrencia <= dataOcorrenciaAtual
        and HoraOcorrencia <= horaOcorrenciaAtual;

  select COALESCE(count(distinct id_aluno),0) into qtd_alunos_erraram
  from invencaologaritmos.respostasexercicios
  where ord_questao = ord_questao_atual and Correta = 'FALSE'
        and DataOcorrencia <= dataOcorrenciaAtual
        and HoraOcorrencia <= horaOcorrenciaAtual;

  select COALESCE(max(qtd),0) from
                (select count(ch_resposta) as qtd, ID_Aluno
                 from respostasexercicios
                 where ord_questao = ord_questao_atual
                       and Correta = 'FALSE'
                       and DataOcorrencia <= dataOcorrenciaAtual
                       and HoraOcorrencia <= horaOcorrenciaAtual
                 group by ID_Aluno) xxx into qtd_max_tentativas_questao;

  IF qtd_max_tentativas_questao is null
     THEN set qtd_max_tentativas_questao = 1;
```

```
  END IF;
  IF qtd_max_tentativas_questao = 0
     THEN set qtd_max_tentativas_questao = 1;
       set qtd_tentativas_aluno = 1;
  END IF;
  IF qtd_max_tentativas_questao > 10 -- valor discrepante
     THEN
         set qtd_max_tentativas_questao = 10;
  END IF;
  IF qtd_tentativas_aluno > 10
     THEN
         set qtd_tentativas_aluno = 10;
  END IF;
  IF qtd_alunos_acertaram = 0
     THEN
   set qtd_alunos_acertaram = 1;
  END IF;
  IF qtd_alunos_erraram = 0
     THEN
          set qtd_alunos_erraram = 1;
  END IF;

  IF respostaCorreta = 'TRUE'  -- aluno acertou a questao
     THEN
       set A = 1;
       set E = 0;
  ELSE
    set A = 0;
    set E = 1;
  END IF;

  SET k1 = (1-rating_anterior/10);
  SET k2 = ((rating_anterior -1)/10);

  SET rating_novo = rating_anterior +
                    A*k1*(1/qtd_alunos_acertaram)*
                    (10-9*qtd_tentativas_aluno/qtd_max_tentativas_questao) -
                    E*k2*(1/qtd_alunos_erraram)*
                    (10*qtd_tentativas_aluno/qtd_max_tentativas_questao);

  update invencaologaritmos.ratingaluno
    set rating_atual = rating_novo
  where ch_aluno = id_aluno_param;
END LOOP the_loop;

END
}
```

ANEXO A

PRÉ-TESTE

	CENTRO ESTADUAL DE EDUCAÇÃO PROFISSIONAL DE CURITIBA **CURSO TÉCNICO EM QUÍMICA** **AVALIAÇÃO DE MATEMÁTICA**
PROFESSOR(A)	1a. AVALIAÇÃO 4o. BIMESTRE
Valor da avaliação: 1,5	TURMA: 1QAT1 / DATA: 14/10/14
Nome:_____ No.____	**Nota:**
Duração: 100 minutos	Consulta: ()Sim (x)Não / Calculadora: ()Sim (x)Não

Diretivas: Cada item desta prova vale 0,1 ponto. A interpretação faz parte da prova, não sendo permitida perguntas durante este período. As respostas devem ser dadas a caneta (azul ou preta); caso seja resolvida à lápis, não haverá direito a revisão de prova. Não é permitido o uso de corretivos, o que anulará a resposta. As questões deverão apresentar a resolução de modo organizado no corpo da prova.

1) Usando a definição, calcule o valor dos seguintes logaritmos:

$a)\ \log_3 81$

$b)\ \log_5 125$

$c)\ \log 100.000$

$d)\ \log_6 216$

$e)\ \log 0,01$

$f)\ \log_9 \frac{1}{27}$

$g) log_{0,2} \sqrt[3]{25}$

2) Calcule:

a) o logaritmo de 4 na base $\frac{1}{8}$.

b) o logaritmo de $\sqrt{3}$ na base 27.

c) o logaritmo de 0,125 na base 16.

d) o logaritmo de 7 na base $\sqrt[5]{7}$

3) Obtenha em cada caso o valor de x:

$a)\ \log_5 x = \log_5 16$

$b)\ \log_3(4x-1) = \log_3 x$

$c)\ \log_{\frac{1}{2}} x = -2$

$d)\ \log_x 2 = 1$

ANEXO B

PÓS-TESTE

	CENTRO ESTADUAL DE EDUCAÇÃO PROFISSIONAL DE CURITIBA **CURSO TÉCNICO EM QUÍMICA** **AVALIAÇÃO DE MATEMÁTICA**

PROFESSOR(A)		2a. AVALIAÇÃO 4o. BIMESTRE
Valor da avaliação: 1,5	TURMA: 1MAT1	DATA: 20/11/14
Nome:_____ No.____		**Nota:**
Duração: 50 minutos	Consulta: ()Sim (x)Não	Calculadora: ()Sim (x)Não

Diretivas: Cada item desta prova vale 0,1 ponto. A interpretação faz parte da prova, não sendo permitida perguntas durante este período. As respostas devem ser dadas a caneta (azul ou preta); caso seja resolvida à lápis, não haverá direito a revisão de prova. Não é permitido o uso de corretivos, o que anulará a resposta. As questões deverão apresentar a resolução de modo organizado no corpo da prova.

1) Usando a definição, calcule o valor dos seguintes logaritmos:

a) $log_2 64$

b) $log_5 625$

c) $log_{36}\sqrt{6}$

d) $log_{1,25} 0{,}6$

e) $\log 1000000$

f) $log_8 16$

g) $\log 0{,}001$

2) Calcule:

a) o logaritmo de 9 na base $\frac{1}{27}$.

b) o logaritmo de $\sqrt{2}$ na base 32.

c) o logaritmo de 0,125 na base 16.

d) o número cujo logaritmo em base 3 vale -2

3) Obtenha em cada caso o valor de x:

a) $logx^2 = 4x$

b) $log_7(3x-1) = log_7(2x+5)$

c) $log_{12}(x-4) = log_{12}(2x-6)$

d) $log_5(2x+1) = log_5(x+3)$