Múltiplas Representações Externas no Suporte à Aquisição de Conhecimento em Programação de Computadores

Multiple External Representations to Support Knowledge Acquisition in Computer Programming

Eleandro Maschio

Coordenação do Curso de Tecnologia em Sistemas para Internet

Universidade Tecnológica Federal do Paraná (UTFPR)

Guarapuava, Paraná, Brasil

eleandro@hotmail.com

Alexandre Ibrahim Direne

Departamento de Informática

Universidade Federal do Paraná (UFPR)

Curitiba, Paraná, Brasil

alexnd@inf.ufpr.br

Resumo *O presente artigo concentra-se em descrever como o conhecimento de alto nível sobre aspectos de experiência humana pode ser modelado, representado e, então, interpretado a fim de suportar interações de ensino e aprendizagem em Programação de Computadores. O esforço de pesquisa incide, especificamente, no problema de prover uma epistemologia para a descrição do conhecimento sobre enunciados e estágios de conhecimento de aprendizes, tratando-o por meio de (1) um método baseado em grafos genéticos para gerenciar a complexidade na autoria de enunciados e (2) um processo geral para a modelagem dinâmica do conhecimento de aprendizes através da sobreposição das capacidades do domínio previamente descritas. Ambos são apoiados por ferramentas de software implementadas e integradas ao ambiente MÚTIPLA. A avaliação do método e das ferramentas aqui propostos foi realizada por um experimento de impacto na aprendizagem e por um estudo empírico sobre a generalidade dos grafos genéticos como linguagem de autoria de um arcabouço unificado, destinado ao desenvolvimento de perícias em programação de computadores.*

Palavras-Chave: *ensino de Programação de Computadores, representação de conhecimento, sistemas tutores inteligentes, ambientes interativos de aprendizagem, múltiplas representações externas.*

Abstract *This paper describes how high-level knowledge about human expertise features can be modeled, represented and further interpreted to support learning and tutoring interactions in Computer Programming. We focused up, specifically, on the problem of providing an epistemology for describing knowledge about exercise statements and students' states of belief, treating it through (1) a method based on genetic graphs for managing the complexity of courseware authoring of problem statements and (2) a general process for dynamic modeling a learner's knowledge by overlaying it against the domain expertise features. The method and the model are both supported here by implemented prototype software tools that integrate the educational environment MULTIPLA. To evaluate the method and tools, an experiment and an empirical study have been carried out, focusing on learning impact and generality of genetic graphs as an authoring language to provide an unified expert-student framework for developing skills.*

Keywords: *teaching of Computer Programming, knowledge representation, intelligent tutoring systems, interactive learning environments, multiple external representations.*

1 Introdução

O presente artigo concentra-se em descrever como o conhecimento de alto nível sobre aspectos de experiência humana pode ser modelado, representado e, então, interpretado a fim de suportar interações de ensino e aprendizagem em Programação de Computadores.

As disciplinas introdutórias de Programação de Computadores concentram grande responsabilidade em um curso de Ciência da Computação. Cabem-lhes a introdução dos princípios e consequente iniciação ao desenvolvimento de perícias na área de programação. Geralmente isso se faz em tempo restrito, caso considerada a exigência cognitiva da aprendizagem, mediante a intercalação de abordagens teóricas e práticas. O legado do trabalho nas citadas disciplinas (tanto competências quanto deficiências) desencadeia-se por entre todo o currículo do curso e tende a desembocar na prática profissional do egresso como um fator predominante.

Contudo, conforme revisão feita sobre as pesquisas da área, pouco se investigou acerca do desenvolvimento de habilidades específicas do estereótipo de um programador perito. Nesse sentido, [10] manifesta-se contra tal desatenção ressaltando a íngreme curva de aprendizado enfrentada por um iniciante que objetiva proficiência na área. Corroborar tal fato a vasta literatura nas estantes de livrarias e na web que anuncia ensinar programação em poucos dias, se não em horas.

O citado estudo expõe maiores agravantes quando se trata não da suficiência, mas da excelência em determinado domínio de conhecimento. A exemplo disso, [4] documenta uma pesquisa, realizada com estudantes da Academia de Música de Berlim, que compara o tempo destinado à prática com o nível atingido pelos indivíduos. Constatou-se que todos começaram a tocar por volta dos cinco anos de idade. Nos primeiros anos, praticavam de duas a três horas semanais. Em torno dos oito anos de idade, diferenças reais apareceram. Aqueles que terminaram como melhores de suas turmas começavam a praticar mais do que o restante: 6 horas semanais aos 9 anos, 8 horas aos 12, 16 horas aos 14, aumentando para mais de 30 horas semanais de prática aos 20 anos. Nesta idade, os instrumentistas de elite totalizavam uma média de 10.000 horas de prática musical. Em contraste, aqueles que eram somente bons acumularam 8.000 horas e os que não se destacaram, pouco mais de 4.000 horas.

Assim, [10] aponta a prática deliberada como elemento decisivo no desenvolvimento de perícias em uma grande variedade de áreas. Sugere o desafio de tarefas que vão um pouco além da habilidade corrente do aprendiz, a fim de que ele tente desempenhá-las, analisar a própria

performance durante e depois de realizá-las para, então, corrigir possíveis erros. Outro estudo [7] reforça que um aprendizado mais efetivo requer tarefas bem definidas, com nível de dificuldade apropriado ao aprendiz em particular, bem como *feedback* informativo, oportunidade para repetição e correção de erros.

Dessa forma, o sequenciamento dos enunciados¹ a serem oferecidos ao aprendiz consta como um dos elementos pedagógicos cruciais ao desenvolvimento de perícias. Assim, prover uma modelagem precisa, tanto do conhecimento do domínio quanto do aprendiz, para subsidiar decisões nesse âmbito, perfaz uma contribuição relevante na área de Inteligência Artificial (IA). Ainda convém destacar que a modelagem de aprendizes não se trata de um conceito recente no campo de Sistemas Tutores Inteligentes (STIs). Contudo, pesquisas anteriores tendiam a se concentrar em questões pedagógicas em detrimento das psicológicas. Como consequência, os esforços se destinaram à proposição de arcabouços que estabelecessem relações de arquitetura entre os modelos pedagógico e do aprendiz.

Em Programação de Computadores, adicionalmente, percebe-se que pouco se fez acerca do suporte à aquisição de conhecimento no entremeio de princípios e perícias [18]. Algumas tentativas [15, 17, 19], no entanto, foram desempenhadas a fim de dar assistência, tanto passiva quanto ativamente, ao aprendiz em ambientes interativos no domínio. Especificamente em caráter passivo, houve construção de depuradores que se encaixassem no ensino de programação e permitissem ao aprendiz conduzir experimentos de execução simbólica do código-fonte.

Porém, ainda resta um enorme vazio de pesquisa original a ser coberto para oferecer acompanhamento ao progresso das capacidades do aprendiz, como também para monitorar adequadamente experimentos através de representações compatíveis com as habilidades que se desenvolvem. Um caminho promissor foi proposto sob a denominação de Múltiplas Representações Externas (MREs) [1], tendo considerável sucesso na tentativa de correlacionar descrições de conhecimento internas com externas em ambientes de aprendizagem, inteligentes ou não.

Além de ressaltar tal compatibilidade, influenciando assim uma recente linha de projeto dos ambientes interativos de aprendizagem, a utilização de um conjunto multirrepresentacional no ensino tende a desembocar numa abordagem metacognitiva [9] por parte do instrutor e do aprendiz. Em teorias cognitivas aplicadas à educação, isso geralmente implica em uma divisão mais equitativa

¹ Termo usado neste documento para referir-se, de maneira genérica, a exemplos e exercícios de um determinado domínio de conhecimento.

de responsabilidades e aprimora o desempenho naquilo que se ensina.

A partir disso, em contexto mais amplo, pode ser visto como desafio encontrar meios efetivos que propiciem o aprendizado, principalmente em Programação de Computadores. Na atualidade, os aprendizes têm o cotidiano imerso em tecnologias das mais variadas e sessões tradicionais de ensino mostram-se insuficientes para atraí-los e motivá-los ao estudo. Entretanto, pela característica imersiva, o ensino baseado em computadores tende a sobrepor muitas das abordagens obsoletas. Como consequência, a superação do platô constatado na modelagem do aprendiz tende a se mostrar naturalmente como o passo seguinte das pesquisas em Inteligência Artificial e Educação.

2 Definições

A problemática discutida embasa-se, conforme detalhado na sequência, nos conceitos concernentes à aquisição de princípios e desenvolvimento de perícia, múltiplas representações externas e sequenciamento de enunciados.

2.1 Aquisição de Princípios e Desenvolvimento de Perícia

O processo de aprendizagem em domínios de natureza prática e complexa fundamenta-se, segundo [8], na aquisição de conhecimentos sobre princípios e consequente desenvolvimento destes até consolidarem perícia na área. A aquisição de princípios corresponde à assimilação de conhecimento formal (científico, em alguns casos) repassado inicialmente na formação do aprendiz. Princípios são os fundamentos do domínio ao qual se referem. O desenvolvimento de perícia, por sua vez, diz respeito à construção do conhecimento experiencial, ou seja, a conquista de habilidade pela prática. Abrange a integração dos princípios aprendidos com a experiência obtida até o momento e, ainda possivelmente, campos distintos de conhecimento.

Embora o conhecimento de princípios seja semelhante tanto em peritos (experts) quanto em aprendizes, o conhecimento experiencial apresenta várias e contundentes diferenças. Para tanto, o desenvolvimento de perícia ocorre por meio da exposição do aprendiz a casos exemplares e não exemplares [8, 3]. Cada caso ao qual se expõe o aprendiz é compreendido como uma oportunidade para o desenvolvimento de perícia. A aprendizagem, portanto, acontece pela indução a partir desses casos.

Convém evidenciar que modelos para descrição de perícias não devem ser confundidos com os formalismos empregados por arquiteturas tutoriais para a definição de currículos, como taxonomias e partonomias [6]. Tais modelos consistem em formalismos ainda mais gerais,

sendo as taxonomias e as partonomias um mero subconjunto do universo de abstração que pode ser obtido por meio de dimensões de perícia apropriadamente definidas em um STI.

Conforme a abordagem descrita, o processo de aprendizagem em Programação de Computadores também pode ser visto como a aquisição de conhecimentos sobre os princípios de lógica de programação e consequente desenvolvimento destes até consolidarem perícias na área. Distinguem-se, os princípios das perícias [9], porquanto os primeiros tangem à compreensão de instruções primitivas isoladas, atendo-se à rigidez léxica, sintática e semântica determinada pela linguagem que se usa. As perícias, por sua vez, abrangem o encadeamento e o aninhamento de duas ou mais daquelas instruções a fim de compor um algoritmo. Assim, a análise e a resolução de problemas através de algoritmos são alicerçados pela soma dos respectivos princípios e perícias. No entanto, a tarefa de programar impõe uma alta carga cognitiva sobre os iniciantes, pois exige a aplicação de ambas as categorias de conhecimento na externalização do código-fonte. Como consequência, os aprendizes incorrem em erros frequentes que antagonizam a evolução desses indivíduos enquanto programadores.

Dentro disso, [12] identificou um conjunto de características do estereótipo de um programador perito, a saber:

- (a) precisão sintática;
- (b) precisão semântica;
- (c) identificação de estruturas principais no programa-fonte (busca por palavra-chave);
- (d) simulação mental dos estados do computador durante a execução;
- (e) catálogo de erros;
- (f) mapeamento mental das estruturas do programa;
- (g) checagem de pré-condições;
- (h) análise do problema;
- (i) integração dos subproblemas;
- (j) generalização da solução;
- (k) reutilização de soluções já conhecidas; e
- (l) catálogo de soluções.

Contudo, desde então, inexistem publicações que enfoquem o desenvolvimento de perícia em Programação de Computadores. Frente a isso, a corrente pesquisa retoma os citados estudos a fim de incorporar técnicas de autoria que permitam a definição formal de perícias em domínios de natureza prática e complexa. Tais esforços

podem levar a modelagem dinâmica do aprendiz um passo adiante nas pesquisas da área.

2.2 Múltiplas Representações Externas

O termo Representação Externa (RE) [2], no escopo desta pesquisa, refere-se ao uso de técnicas para representar, organizar e apresentar conhecimento. Remete, portanto, a uma ampla variedade de representações que compreende desde modelos proposicionais/sentenciais até modelos gráficos/diagramáticos. Preenchendo a lacuna entre esses extremos, há representações intermediárias que associam elementos gráficos e textuais (e.g. tabelas, que denotam matrizes graficamente). Logo, são exemplos de RE: sentenças em linguagem natural, sentenças em linguagens formais (como lógica de primeira-ordem), tabelas, listas, grafos, mapas, projetos, diagramas, animações e até mesmo as realidades aumentada e virtual.

Segundo definição [11], qualquer RE deve ser descrita em termos:

- (a) do mundo representado;
- (b) do mundo representante;
- (c) de quais aspectos do mundo representado estão sendo expressos;
- (d) de quais aspectos do mundo representante compõem a modelagem;
- (e) da correspondência entre os dois mundos.

Como consequência do bom emprego de REs nas tarefas do dia-a-dia, tais técnicas tornaram-se um suporte valioso à resolução de uma diversidade de problemas de caráter mais formal, bem como um poderoso auxílio para diminuir a carga cognitiva requerida, sendo empregadas em várias áreas do conhecimento humano.

Além do supracitado, há evidências [20] expondo vantagens da utilização de REs como suporte tanto ao ensino quanto à resolução de problemas. Nesse sentido, há duas décadas as pesquisas se concentram na combinação de diferentes REs para intermediar o aprendizado. Múltiplas Representações Externas (MREs) [1], denominação dada ao conjunto, tem se mostrado um nicho próspero no atendimento às necessidades cognitivas de aprendizes diversos.

Houve outras pesquisas abordando exclusivamente a potencialidade e os possíveis problemas associados à aprendizagem com, simultaneamente, mais de uma modalidade de RE [21]. Elas sustentam que MREs relacionadas resultam em um aprendizado mais flexível e intuitivo, dando espaço a diferentes ideias e estratégias. Contudo, também se constatou que aprendizes podem não ter êxito em se beneficiarem das vantagens anteditas, pois elas nem sempre são facilmente atingidas.

Estudos específicos [21, 2] relataram o emprego de REs em ambientes interativos de aprendizagem destinados à resolução de problemas. Constataram que sistemas como Bridge, Geometry Tutor e GIL valiam-se de grafos de prova como recurso metacognitivo, tendo-os como meio para tornar o processo de planejamento mais evidente. Foram também relacionados, pelos mesmos estudos, ambientes em que o raciocínio com REs era essencial à atividade de ensino subjacente. Hyperproof, para lógica de primeira-ordem, e Algebra (Word Problem Tutor), para habilidades básicas de manipulação algébrica, são exemplos disso. Finalmente, na mesma linha de estimular o pensamento reflexivo por meio de uma abordagem de causa-efeito, o ambiente SimForest, que simula o crescimento de árvores em florestas, inovou ao introduzir em ambientes interativos de aprendizagem os conceitos de caixa-preta e caixa-de-vidro para permitir que o aprendiz ajustasse o nível de detalhamento da representação dos mecanismos que conduzem a simulação.

Constatou-se também a incidência de MREs como recurso de depuração e/ou visualização em ambientes não-educacionais [13]. Evidência disso são os termos “visualização de algoritmos”, “visualização de programas” e “visualização de *software*”, todos análogos e referentes à representação gráfica de aspectos do código textual.

2.3 Sequenciamento de Enunciados

Diante da perspectiva do desenvolvimento de perícias, [12] descreve medidas cognitivas no ensino de Programação de Computadores. São uma forma de se obter metac conhecimento acerca dos enunciados de uma base de conhecimento de um STI. Funcionam como um forte elo entre o Modelo de Domínio e o Modelo do Aprendiz nestes sistemas.

As citadas medidas têm o papel de quantificar cognitivamente um enunciado de Programação de Computadores. Ocupam-se de avaliar o quanto determinado enunciado exige de um aprendiz em termos de conhecimentos adquiridos e capacidades desenvolvidas na progressão do aprendizado. Nesse sentido, o mesmo estudo define a carga cognitiva de um enunciado como a própria capacidade em exercitar o aprendiz na construção de um programa de computador. Portanto, a carga cognitiva pode ser dividida em alguns componentes que individualmente se responsabilizam por medir um tipo de contribuição/exigência proporcionado pelo enunciado ao aprendiz.

A complexidade do software é um dos componentes sugeridos pelo citado estudo. Todavia, como os programas destinados a iniciantes são majoritariamente pequenos, a complexidade pode ser melhor determinada usando linhas de código (LOC) em conjunto com complexidade estrutural (CE) e respectivos detalhes de implementação (DI). A complexidade estrutural avalia o número de es-

estruturas de decisão e de repetição, bem como o grau de dependência entre as mesmas estruturas (aninhamentos). Por sua vez, os detalhes de implementação consideram a quantidade de condições específicas que dificultam o cumprimento do enunciado.

Além dos componentes que concernem à complexidade, cada enunciado contribui para que o aprendiz desenvolva características do estereótipo de um programador perito (conforme Seção 2.1). Por consequência, os outros componentes da carga cognitiva de um enunciado baseiam-se e têm medidas definidas a partir desse grupo de características. Com isso, podem quantificar a contribuição/exigência de um enunciado frente a cada um dos atributos do estereótipo.

Portanto, a carga cognitiva de um enunciado constitui-se da ponderação das medidas cognitivas que o compõem. Embora cada medida contribua com uma parcela desta carga cognitiva, isso não ocorre de maneira uniforme [16]. Diferentes medidas possuem pesos distintos na composição da carga cognitiva de uma instância. Os pesos são valorados por meio de informações obtidas de especialistas no ensino de Programação de Computadores.

Evidencia-se então que estas medidas cognitivas fornecem subsídios adequados para a escolha do próximo enunciado a ser trabalhado em uma sessão de ensino, colaborando para que a progressão do aprendizado seja mais concisa e coesa. Assim, há utilidade prática na classificação de um enunciado, por grau de exigência/contribuição, diante de um elenco preciso de medidas cognitivas. Uma galeria de enunciados poderá ser catalogada e ordenada formalmente frente às dimensões que denotam o (sub)conjunto de medidas cognitivas consideradas.

Parte da importância desse procedimento pode ser constatada quando um problema de programação significativamente mais complexo do que o anterior é proposto a um aprendiz. O fato pode levar à ocorrência de erros múltiplos, impedindo o sucesso do indivíduo por longos períodos de tempo.

Partindo desse pressuposto, um fator adverso em ambientes de ensino destinados à Programação de Computadores reside na extrema rigidez na ordenação das respectivas galerias de enunciados. Mesmo que, mediante decisões fundamentadas, os projetistas e construtores de tais ambientes tenham alcançado determinada sequência de ensino, não há garantias de que a sequência privilegie toda a heterogeneidade de perfis e competências de diferentes aprendizes. Portanto, pode-se deduzir que a sucessão dos enunciados atende a um grupo específico de aprendizes, geralmente de desempenho entre baixo e mediano, e negligencia aqueles que não se enquadram no estereótipo idealizado.

Uma contribuição anterior [14] antecipou a complexidade dos enunciados como um dos principais componentes de motivação do indivíduo no aprendizado. Em consonância, [12] explica que “a repetição sistemática de enunciados completamente diferentes, porém, de graus de complexidade aproximadamente iguais pode elevar consideravelmente a autoconfiança do aprendiz, mantendo-o motivado e produtivo”.

Logo, seria não somente desejável, como também necessária, a flexibilização dos ambientes de ensino para que se adaptassem às competências de cada aprendiz. Nessa direção, foi desenvolvida a ferramenta de autoria Sequence [12], inicialmente concebida para a área médica e depois adaptada ao ensino de Programação de Computadores. Baseia-se em medidas cognitivas e avalia um enunciado mediante a complexidade (LOC, CE e DI) aliada às perícias de precisão sintática e semântica, análise do problema, reutilização de soluções e simulação mental. Com isso, consegue sugerir uma sequência inicial de enunciados e então adaptá-la à situação corrente do usuário mediante informações oriundas do Modelo do Aprendiz no STI.

Entretanto, consta-se que ainda permanece um vasto espaço para pesquisa, pois faltam estudos que se estendam na focalização das características cognitivas que compõem a complexidade de um enunciado de Programação de Computadores. Ademais, seria interessante que diferentes pesquisas reavivassem o estado da arte desta área e, como consequência, guinassem os ambientes de ensino destinados à Programação de Computadores a uma nova e promissora geração.

3 Arcabouço Conceitual e Ferramentas de Software

Discorre-se, nesta seção, sobre métodos e ferramentas propostos como contribuição no auxílio ao desenvolvimento de perícias em Programação de Computadores.

3.1 O Ambiente Interativo de Aprendizagem MÚTIPLA

O protótipo MÚTIPLA (Multiplicador de Linguagens Tipificado em Plataforma de Aprendizagem) [9] consiste em um ambiente interativo de aprendizagem destinado à aquisição de conhecimento na fronteira entre princípios e perícias de Programação de Computadores (Figura 1). Permite que o aprendiz construa algoritmos na representação gráfica de fluxograma e observe a geração automática do código equivalente na linguagem Pascal. Além disto, também há a possibilidade de que o aprendiz conduza experimentos de execução simbólica do algoritmo construído. Para este protótipo, trabalhou-se somente com um subconjunto da linguagem Pascal: os tipos boo-

leano, inteiro, real e caractere; as instruções de entrada, saída e atribuição; a estrutura de seleção *se (if)*; e as es-

truturas de repetição *enquanto-faça (while-do)*, *repita-até (repeat-until)* e *para (for)*.

Figura 1: Tela do ambiente interativo de aprendizagem MÚTIPLA

Constrói-se um algoritmo através da seleção dos elementos de fluxograma em *menus* de contexto e da digitação das respectivas expressões em caixas de diálogo. Cada alteração feita no fluxograma é automaticamente transposta para o código-fonte em Pascal, exibido no lado esquerdo. A declaração de variáveis foi alocada em uma tela independente.

O MÚTIPLA oferece duas opções de execução: em modo passo-a-passo e temporizado. Ambas realçam o elemento de fluxograma e a linha de código onde se encontra o fluxo de controle, bem como sincronizam estas informações com a visualização da janela de saída e das células de memória (valores das variáveis).

Também se evidencia que tanto a construção quanto a execução dos algoritmos são assistidas por múltiplas representações externas que atuam como ferramentas de visualização de diferentes aspectos destes algoritmos. Além daquelas acima mencionadas, destacam-se: cliques de correspondência entre a notação de fluxograma e o código em Pascal (o inverso também ocorre), recolhimento e expansão de blocos de elementos, cores distintas para cada categoria de elementos, comentários textuais e pontos de interrupção.

O conjunto multirrepresentacional definido remete às prescrições do *framework* conceitual DeFT (*Design, Functions, Tasks*), destinado ao ensino com MREs e descrito em [1]. Portanto, apresenta-se a correspondência entre alguns recursos semânticos entre as REs e as funções propostas na taxonomia funcionalista de Ainsworth, subjacente ao DeFT. Convém esclarecer que uma mesma RE pode desempenhar mais de um papel (ou função) em uma determinada situação de aprendizado.

- (1) *Construção de Compreensão Aprofundada.* A função que mais notoriamente embasa o MÚTIPLA classifica-se como abstração, visto que a RE de fluxograma apresenta mais baixo nível abstrato do que o código em Pascal. Com isso, ao passo que o aprendiz desenvolva habilidades, mesmo que parciais, de construção de algoritmos em fluxogramas, o micromundo subsidiará a aquisição de conhecimento sobre a tarefa correlata numa linguagem de alto nível (Pascal). Para isto, o aprendiz contará com uma série de instrumentos, como a geração automática do algoritmo textual equivalente e a visualização de aspectos dinâmicos (e também estáticos) de correspondência entre es-

tas duas REs. As duas outras funções que tangem à construção de compreensão aprofundada (denominadas relação e extensão) também ocorrem no conjunto de MREs estabelecido, porém não desempenham papel tão importante quanto a abstração.

- (2) *Restrição de Interpretação.* Nesta classe, a função que mais se sobressai é chamada de restrição por propriedades herdadas. Assim, mesmo na ocorrência do pior caso, ou seja, quando o aprendiz não tem familiaridade com nenhuma das duas formas de representação de um algoritmo, a mais específica destas (notação de fluxograma) deverá ajudar a restringir as ambiguidades daquela mais genérica (código em Pascal). Além disso, ambas teriam possíveis ambiguidades esclarecidas por outras REs do conjunto.
- (3) *Papéis Complementares.* A abordagem adotada pelo protótipo enquadra-se nesta classe como informações complementares diferentes. Isso acontece porque, conforme dito, os processos intrínsecos a qualquer uma das duas formas de representação adotadas é exatamente o mesmo, mas as informações contidas em cada uma das descrições de algoritmo são fundamentalmente diferentes. O conjunto de cores adotado e o destaque de variáveis com valores alterados também se incluem nesta mesma subclasse.

Investiu-se também na avaliação formal do MÚLTIPLA junto a aprendizes, no intuito de aferir os benefícios de utilização do protótipo desenvolvido. Como resultado subsequente, a coleta de informações a partir deste uso inicial tem validado a hipótese teórica que embasou o protótipo, como também forneceu indicativas para o aperfeiçoamento do ambiente.

Os resultados alcançados motivaram a pretensão de investir em técnicas de Inteligência Artificial que elevassem o nível do apoio fornecido pelo MÚLTIPLA de sintático para semântico. Investiu-se, assim, em técnicas associadas que propiciassem determinada automatização na sequência de enunciados a serem explorados, de acordo com as subseções seguintes.

3.2 Modelagem do Aprendiz e Sequenciamento de Enunciados

Conforme anteriormente abordado, consta-se o sequenciamento de enunciados fornecidos ao aprendiz como um dos elementos cruciais ao desenvolvimento de perícias. Para tanto, uma contribuição relevante consiste na modelagem apurada do conhecimento do domínio e do aprendiz, a fim de subsidiar tal processo de decisão. Nesse sentido, a critério de exemplo, os enunciados seguintes

estão ordenados conforme a carga cognitiva exigida na resolução, da menos complexa para a mais complexa.

Enunciado 1

Fornecida a idade do indivíduo, responda se a maioria já foi atingida.

Enunciado 2

Fornecida a idade do indivíduo, responda se o voto é obrigatório. No Brasil, a obrigatoriedade da votação incide sobre os cidadãos entre 18 e 70 anos.

Enunciado 3

Fornecida a idade do indivíduo, responda se o voto é obrigatório, facultativo ou não permitido. No Brasil, a obrigatoriedade da votação incide sobre os cidadãos entre 18 e 70 anos. O voto é facultativo para aqueles que têm entre 16 e 18 anos, ou mais de 70 anos.

Supondo a resolução do segundo enunciado pelo aprendiz, seria interessante dispor de mecanismos em um STI que, conforme a performance na resolução, pudessem retroceder ao primeiro enunciado ou avançar ao terceiro enunciado. Tais mecanismos somente são possíveis mediante uma modelagem precisa e convencionada que situacione o perfil do aprendiz frente ao conhecimento do domínio. Ademais, é necessário que se classifique cada enunciado conforme a fração exercitada deste conhecimento.

Considerando isto, decidiu-se, nesta pesquisa, descrever o conhecimento do aprendiz através do Modelo de Sobreposição (*Overlay*) [5]. A abordagem consiste na representação do aprendiz como um subconjunto do conhecimento do domínio. Através da sobreposição destes conjuntos, fica evidente quais objetivos devem ser atingidos pelo aprendiz. Todavia, implica que a representação de conhecimento de ambos os modelos sejam comparáveis, ou a mesma.

Assim, buscou-se uma representação que pudesse descrever perícias no domínio de Programação de Computadores e que, frente ao Modelo de Sobreposição, denotasse a incidência do conhecimento do aprendiz sobre o conhecimento do domínio. A correlação entre as diversas perícias e subperícias também deveria ficar evidente. Frente a isso, o grafo genético [5], técnica de representação escolhida, responde às necessidades expostas e também privilegia o Modelo de Sobreposição. Trata-se, resumidamente, de um arcabouço para representação do conhecimento. Nesse tipo específico de grafo, o conhecimento do domínio (ou de um perito) é constituído por um conjunto de fatos ou regras (vértices) interconectados por relações (arestas) que evoluem com o tempo. Um exemplo de aplicação do Modelo Sobreposição se dá quando um aprendiz de Programação de Computadores interage com um sistema tutor. À medida em que esse aprendiz fornece código-fonte como solução de proble-

mas, o sistema identifica entradas mais elaboradas, com maiores graus de correção e refinamento. Isso condiz com a natureza de monitoramento de longo prazo que eu sistema deve conduzir em domínios de aquisição de habilidades procedimentais.

A Figura 2 ilustra um exemplo simplificado de representação, em um grafo genético, das perícias (denotadas pelos vértices) que constituem um domínio de conhecimento, assim como a relação (arestas) entre elas. O conhecimento do aprendiz é modelado como um subconjunto (ou subgrafo) das perícias mapeadas no domínio. Todas as perícias pendentes ao aprendiz são evidenciadas pela sobreposição dos dois grafos. Assim, através dos mesmos aspectos de representação externa, tem-se realizado (em **negrito**) o estágio atual do aprendiz frente ao conhecimento do domínio, ou seja, a sobreposição propriamente dita.

Figura 2: Sobreposição do Modelo do Aprendiz frente ao conhecimento do domínio

A partir dos dois modelos, do aprendiz e do domínio, pode-se prover um processo de busca heurística que oriente a indicação de regiões do grafo a serem exploradas

pelo aprendiz. Tal indicação de perícias leva à prioridade de aquisição das mesmas pelo aprendiz, conforme o seu estágio atual de conhecimento (Figura 3).

Figura 3: Indicação de regiões do grafo a serem exploradas

Em adição, empregou-se o mesmo mapeamento de perícias como um gabarito para os enunciados a serem propostos pelo aprendiz. Cada enunciado contribui para o desenvolvimento de perícias específicas do aprendiz, podendo também ser visto como um subgrafo do conhecimento do domínio, conforme Figura 4.

Dessa maneira, os enunciados podem ser catalogados de acordo com o gabarito. Trata-se de uma extensão original para o trabalho de [5].

Isto posto, o Modelo do Aprendiz adquire caráter dinâmico por ser alimentado com a avaliação do cumprimento dos enunciados propostos. Supõe-se a avaliação por um especialista externo (humano), diante do gabarito de perícias fornecido pelo enunciado. O procedimento documenta o progresso do aprendiz e consequente evolução da estrutura do grafo genético. O especialista externo pode dar lugar a uma ferramenta de diagnóstico.

Figura 4: Catálogo de Enunciados

Outra possibilidade, alcançada pela corrente pesquisa e garantida pela representação em grafo, consiste em sintetizar o Modelo do Aprendiz de um grupo (sala de aula) e assim ter um espelho da ênfase com que as perícias são ensinadas pelo instrutor. Com isso, pode-se notar que um instrutor se sobressai no ensino de determinadas perícias, ou que um segundo negligencia algumas outras.

Convém salientar que, nos exemplos dados, as perícias são sinalizadas como desenvolvidas ou não-desenvolvidas. Entretanto, estuda-se a implementação do desenvolvimento gradativo das perícias pelo aprendiz por meio de uma escala entre tais extremos. Dessa forma, a progressão seria gradual e mais fiel ao denotar o percentual aprendido/cumprido de cada perícia.

Também podem ser fornecidos detalhes da dimensão temporal do progresso do aprendiz, ou seja, apresentar um histórico do desenvolvimento das perícias em um período específico. Remetendo-se à escala de progressão gradual, recém citada, há âmbito para representações externas bastante informativas. Por exemplo, exibir os vértices sendo (mais) coloridos à proporção do tempo, em uma animação. Adicionalmente, o mesmo recurso é também aplicável à sintetização do Modelo do Aprendiz de um grupo. Logo, é possível visualizar a progressão do aprendizado de uma sala inteira, ou de um curso inteiro.

Por fim, enfatiza-se que a modelagem do aprendiz e do conhecimento do domínio através da sobreposição abre uma vasta gama de possibilidades de aplicação em um STI. Nenhuma das ideias apresentadas nesta seção é complexa em si, entretanto se mostram bastante úteis para guiar o Modelo Pedagógico ou orientar instrutores

(humanos) em sessões de ensino.

3.3 Ferramentas de Autoria

O processo de autoria descrito é suportado por duas ferramentas implementadas. A primeira concentra-se na descrição do conhecimento do domínio mediante o detalhamento das perícias componentes. Conforme a Figura 5, trata-se de uma ferramenta de autoria que utiliza visualmente a representação de grafo genético. As opções de edição são acessadas por *menus* de contexto. Permite-se basicamente: (a) a inserção, edição e remoção de perícias; (b) o estabelecimento das relações evolucionárias entre perícias; (c) a definição de uma perícia inicial; e (d) a validação do modelo descrito quanto à alcançabilidade das demais perícias a partir daquela inicial. O arquivo com as informações do modelo é salvo em formato XML.

Figura 5: Ferramenta para a descrição do conhecimento do domínio

A segunda ferramenta, usando a mesma representação gráfica, destina-se à catalogação e elicitação de enun-

ciados descritos como subgrafos do conhecimento de domínio formalizado pela primeira (Figura 6). Isto é,

possibilita que sejam demarcadas as perícias específicas para as quais cada enunciado contribui. A sobreposição do enunciado enfatiza a fração do conhecimento exercitada. Em adição, pode-se inspecionar a quantidade de enunciados que exercitam determinada perícia, como

também se existem perícias não abordadas pelo conjunto (catálogo) de enunciados corrente. No mesmo sentido, foi implementada uma verificação mais profunda sobre a coesão do catálogo e consequente requisição da autoria de enunciados adicionais.

Figura 6: Ferramenta para a elicitación e catalogación de enunciado

4 Experimento de Avaliação

O escopo desta etapa residiu em mensurar/quantificar a contribuição do ambiente MÚLTIPLA para com o aprendizado em Programação de Computadores. A pesquisa foi realizada no Departamento de Ciência da Computação na Universidade Estadual do Centro-Oeste¹ (Unicentro). Consistiu em um projeto credenciado junto à Pró-Reitoria de Pesquisa e Pós-Graduação da universidade mediante aprovação do departamento mencionado. Houve também autorização e apoio dos acadêmicos envolvidos para que a coleta fosse conduzida.

¹ Câmpus Santa Cruz, Guarapuava, Paraná.

4.1 Sujeitos

Foram sujeitos desta pesquisa os alunos das turmas A e B da disciplina de Programação de Computadores I, da primeira série do curso de Ciência da Computação da Universidade Estadual do Centro-Oeste. Os alunos foram classificados, por um pré-teste (de acordo com critérios relatados na sequência), em dois grupos sem intersecção: o grupo de controle (GC) e o grupo experimental (GE).

Grupo de controle: composto por 10 alunos, presenciou as aulas regulares da disciplina (que contemplam teoria e prática, tendo sido ministradas em laboratório) e utilizou dos recursos de monitoria, bem como

atendimento docente na condição de atividades complementares.

Grupo experimental: constituído por outros 10 alunos, os quais participaram das aulas regulares e, em horário extra, de atividades dirigidas no ambiente MÚLTIPLA.

Os grupos de controle e experimental foram compostos de alunos provenientes tanto da turma A quanto da B. Portanto, receberam tratamento imparcial quanto ao conteúdo ministrado em carga-horária letiva e quanto às avaliações.

Dentro disso, havia ainda a incongruência de que o grupo experimental, sendo submetido a atividades extras, teria melhor desempenho independentemente da utilização do MÚLTIPLA. Como o ambiente contempla somente exercícios que eram resolvidos com o compilador Pascal em carga-horária letiva, orientou-se os alunos do grupo experimental de que resolver os mesmos exercícios também em horário de aula seria facultativo. Em contraste, apenas o grupo de controle participou das atividades de monitoria e de atendimento docente ofertadas no semestre.

Para a análise deste estudo, documentam-se os resultados dos 10 alunos de cada grupo. Em separado, consideram-se também os resultados somente dos alunos que não desistiram da disciplina ou do curso.

4.2 Instrumentos

Os instrumentos utilizados para a observação e análise foram:

- **Pré-teste:** avaliação de habilidades pré-algorítmicas;
- **Médias da disciplina:** primeiro semestre, segundo semestre e médias anuais;
- **Faltas:** primeiro semestre, segundo semestre e total anual;
- **Desistência:** da disciplina e do curso;
- **Pós-teste:** saldo de aprovações, exames e reprovações.

4.3 Procedimentos

O MÚLTIPLA abrange a exploração dos assuntos de manipulação de variáveis, instruções de entrada e de saída, estruturas condicionais e de repetição. Todos compreendidos no primeiro semestre do programa correspondente à referida disciplina.

Ambos os grupos assistiram às aulas da disciplina em horário regular. Tanto as aulas teóricas quanto as práticas

foram ministradas em um dos laboratórios de informática do curso. A sequência de ensino para as aulas compunha-se da exposição e explicação do conteúdo, com o uso de apresentações digitais e da lousa, bem como a resolução de exercícios (implementação de programas) na linguagem Pascal.

Em particular, o grupo de controle teve participação nos horários destinados à monitoria e naqueles de atendimento docente. Sendo o curso de Ciência da Computação oferecido em período integral, é tão frequente a presença dos acadêmicos em tais atividades que são vistas como uma extensão dos horários de aula propriamente ditos.

O recurso de monitoria foi ofertado em regime de 6 horas-aula semanais, totalizando uma carga-horária de 114 horas-aula (19 semanas). O estudante monitor dispunha do laboratório de informática com lousa, assim como ocorria nas aulas da disciplina, e auxiliava no entendimento do conteúdo, como também na resolução de exercícios na linguagem Pascal.

Por sua vez, o atendimento docente, realizado pelo ministrante da disciplina, compreendia 4 horas-aula semanais e acumulou 76 horas-aula naquele semestre. A assistência igualmente incidia sobre a compreensão do conteúdo e de exercícios relacionados. Era prestada no laboratório de informática nas mesmas condições que a monitoria.

O grupo experimental, exclusivamente, participou de encontros em que atividades orientadas ao ambiente MÚLTIPLA eram prescritas. Ao todo foram feitos cinco encontros de 2 horas-aula cada. Portanto, totalizaram-se 10 horas-aula de trabalho com o grupo experimental, equivalente a 7,35% da carga-horária da disciplina (136 horas-aula). Os encontros ocorreram todos no primeiro semestre, com periodicidade variável de acordo com o tempo disponível dos alunos (calendário de provas e trabalhos). Durante aquele semestre, não houve frequência (mas inexistiu restrição) do grupo nos horários destinados à monitoria ou ao atendimento docente institucionalizado. Os encontros consistiram nas únicas atividades complementares em Programação de Computadores.

4.4 Resultados Coletados

Os resultados provieram dos instrumentos de observação e análise especificados na Seção 4.2. São interpretados conforme segue.

4.4.1 Pré-Teste

Todos os alunos da disciplina foram sujeitos a um teste com o objetivo de sondagem e critério para constituição dos grupos. O teste enfocou a avaliação de habilidades pré-algorítmicas, ou seja, conhecimentos que indi-

cassem a estruturação de raciocínio lógico conforme o que demanda a prática de Programação de Computadores. Objetivou-se avaliar tais conhecimentos sem explicitamente se remeter a instruções específicas de linguagem de programação.

O teste aplicado foi composto por dezesseis questões que avaliavam implicitamente diferentes tópicos contemplados pelo ambiente e pela ementa da disciplina. Cada questão foi criteriosamente escolhida para remeter a um tópico específico de Programação de Computadores. Os alunos tiveram o tempo de 4 horas-aula para resolvê-lo.

Com os resultados do pré-teste, teve-se condições de classificar os alunos em dois grupos bastante parecidos, ou seja, equiparáveis (fenômeno estatístico denominado homocedasticidade). Além da média de ambos os grupos serem idênticas (algo difícil de ocorrer, pois se buscava médias próximas), cuidou-se para que houvesse paridade de alunos entre os grupos. A paridade ocorre no sentido de que dois alunos de desempenho próximos são preferencialmente alocados em grupos distintos, desde a dupla de melhor desempenho até aquela de pior. A decisão de qual aluno da dupla ficaria em determinado grupo foi também influenciada pela disponibilidade de horário para os encontros periódicos para utilização do ambiente MÚLTIPLA.

Os resultados do pré-teste são apresentados pela Tabela 1.

	GE	GC
Média aritmética	5,0309	5,0309
Desvio padrão	1,6676	1,9235
Mediana	5,1531	4,8500
Menor Valor	1,5156	1,2125
Maior Valor	7,2750	7,8813
Amplitude	5,7594	6,6688

Tabela 1: Avaliação de Problemas Pré-Algorítmicos

4.4.2 Médias do Primeiro Semestre

A média do primeiro semestre foi composta por duas avaliações presenciais e dois trabalhos correlatos às avaliações. O grupo experimental apresentou média semestral 13,6% superior à obtida pelo grupo de controle. Exceituando-se um aluno que desistiu do curso, componente do grupo de controle, a superioridade decrementa para 3,2%. Não houve desistentes no grupo experimental.

No entanto, merece registro o percentual de crescimento dos grupos com relação ao pré-teste aplicado. O grupo experimental apresentou crescimento médio de 32,91%, com crescimento máximo de 163,92%. O grupo de controle, por sua vez, apresentou crescimento médio de 4,74% e crescimento máximo de 128,57%. Com a exceção do aluno desistente, o crescimento médio do grupo de controle alcança 11,80%.

As referidas informações estão justapostas pelas tabelas 2, 3 e 4.

las 2, 3 e 4.

	Média	Crescimento	% Crescimento
Média aritmética	6,1900	1,1581	32,91%
Desvio padrão	1,8150	0,9961	49,31%
Mediana	6,3500	1,2906	22,79%
Menor Valor	3,6000	-0,3406	-8,64%
Maior Valor	9,3000	2,4844	163,92%
Amplitude	5,7000	2,8250	172,56%

Tabela 2: Grupo Experimental - Médias do Primeiro Semestre

	Média	Crescimento	% Crescimento
Média aritmética	5,4500	0,4181	4,74%
Desvio padrão	3,0457	2,2211	52,37%
Mediana	4,6000	-0,0406	-0,03%
Menor Valor	0,5000	-2,5531	-58,76%
Maior Valor	9,7000	5,4563	128,57%
Amplitude	9,2000	8,0094	187,33%

Tabela 3: Grupo de Controle - Médias do Primeiro Semestre

	Média	Crescimento	% Crescimento
Média aritmética	6,0000	0,5438	11,80%
Desvio padrão	2,6519	2,3178	50,25%
Mediana	5,0000	0,2656	7,58%
Menor Valor	2,6000	-2,5531	-49,55%
Maior Valor	9,7000	5,4563	128,57%
Amplitude	7,1000	8,0094	178,12%

Tabela 4: Grupo de Controle - Médias do Primeiro Semestre (Desistentes Exceituados)

4.4.3 Médias do Segundo Semestre

O cálculo da média do segundo semestre baseou-se somente em avaliações presenciais escritas. O grupo experimental teve 2 desistentes. O grupo de controle somou 4 desistentes. Os resultados das médias do segundo semestre levou em conta apenas os alunos que continuaram frequentando a disciplina.

Estima-se que, embora o MÚLTIPLA incidisse mais sobre os conteúdos do primeiro semestre, os resultados de um aprendizado produtivo são mais visíveis no segundo semestre, pois o ensino de Programação de Computadores não ocorre de forma compartimentada. A proficiência no segundo semestre é o produto de todos os conteúdos anteriormente vistos na disciplina.

O grupo de experimental teve média 59,21% superior do que o grupo de controle. O crescimento agora é relativo à média do primeiro semestre. Dentro disso, o grupo experimental apresentou crescimento médio de 10,96%, contra um decréscimo do grupo de controle de 32,50%.

Percebe-se ainda que os valores do grupo experimental são menos discrepantes do que aqueles do grupo de controle. Chama também atenção o fato de que a mediana do grupo experimental seja 8,15, enquanto a do grupo de controle constitui-se em 4,35.

Estas informações estão apresentadas nas tabelas 5 e 6.

	Média	Crescimento	% Crescimento
Média aritmética	7,5625	0,8500	10,96%
Desvio padrão	2,3157	1,2433	24,06%
Mediana	8,1500	1,0000	14,84%
Menor Valor	2,4000	-1,6000	-40,00%
Maior Valor	9,5000	2,2000	40,00%
Amplitude	7,1000	3,8000	80,00%

Tabela 5: Grupo Experimental - Médias do Segundo Semestre (Desistentes Excetuados)

	Média	Crescimento	% Crescimento
Média aritmética	4,7500	-1,3667	-32,50%
Desvio padrão	3,5664	1,7025	40,50%
Mediana	4,3500	-1,4000	-27,38%
Menor Valor	0,0000	-3,7000	-100,00%
Maior Valor	10,0000	1,0000	11,11%
Amplitude	10,0000	4,7000	111,11%

Tabela 6: Grupo de Controle - Médias do Segundo Semestre (Desistentes Excetuados)

4.4.4 Médias Anuais

A média anual de um aluno é dada sobre a média aritmética simples das semestrais. Apresentam-se aqui quatro tabelas que contrastam, além dos dois grupos, a exceção dos alunos desistentes. Nas duas primeiras, que incluem todos os 10 alunos de cada grupo, percebe-se 43,5% de superioridade na média do grupo experimental. O crescimento apontado é de 26,43% para o grupo experimental e de -22,50% para o grupo de controle.

Outros fatos relevantes, superiores no grupo de controle, são: (a) mediana: 6,85 contra 4,15; (b) menor nota: 1,8 em contraste com 0,3; e (c) maior crescimento: 111,13% contra 34,32%.

Todas essas informações são mostradas pelas tabelas 7 e 8.

	Média	Crescimento	% Crescimento
Média aritmética	6,1400	1,1081	26,43%
Desvio padrão	2,7359	1,8633	49,27%
Mediana	6,8500	1,8109	31,04%
Menor Valor	1,8000	-2,5500	-54,32%
Maior Valor	9,3000	2,7375	111,13%
Amplitude	7,5000	5,2875	165,46%

Tabela 7: Grupo Experimental - Médias Anuais

	Média	Crescimento	% Crescimento
Média aritmética	4,2800	-0,7519	-22,50%
Desvio padrão	2,9914	1,8616	40,50%
Mediana	4,1500	-1,2406	-24,44%
Menor Valor	0,3000	-3,8531	-75,26%
Maior Valor	9,5000	1,8375	34,32%
Amplitude	9,2000	5,6906	109,57%

Tabela 8: Grupo de Controle - Médias Anuais

Os mesmos cálculos foram feitos desconsiderando os alunos que desistiram da disciplina ou do curso, conforme tabelas 9 e 10. Notaram-se diferenças ainda maiores quanto à superioridade do grupo experimental: (a) média: 31,4% superior; (b) mediana: 7,55 contra 5,35; e (c) maior crescimento: 111,13% contra 30,31%.

	Média	Crescimento	% Crescimento
Média aritmética	7,1625	1,9715	46,40%
Desvio padrão	1,9056	0,4388	29,01%
Mediana	7,5500	1,9813	38,56%
Menor Valor	3,2000	1,3313	19,96%
Maior Valor	9,3000	2,7375	111,13%
Amplitude	6,1000	1,4063	91,17%

Tabela 9: Grupo Experimental - Médias Anuais (Desistentes Excetuados)

	Média	Crescimento	% Crescimento
Média aritmética	5,4500	-0,7135	-15,17%
Desvio padrão	2,9838	2,1366	37,67%
Mediana	5,3500	-1,2594	-19,85%
Menor Valor	1,3000	-3,8531	-74,77%
Maior Valor	9,5000	1,8375	30,31%
Amplitude	8,2000	5,6906	105,08%

Tabela 10: Grupo de Controle - Médias Anuais (Desistentes Excetuados)

4.4.5 Saldo de Aprovações, Reprovações e Exames

Tanto o grupo experimental quanto o grupo de controle tiveram 2 alunos para exame (com a exclusão dos desistentes). Embora o número de alunos em exame seja o mesmo, no grupo experimental apenas 3 obtiveram reprovação direta, contra 5 do grupo de controle. Ao final, o grupo de controle contou com 33,33% de aprovação, enquanto o grupo experimental totalizou 87,50% de aprovação. Tais informações foram todas compiladas nas tabelas 11, 12, 13 e 14.

	GE		GC	
	Alunos	% Alunos	Alunos	% Alunos
Aprovados	5	50%	2	20%
Reprovados	3	30%	5	50%
Em Exame	2	20%	3	3%

Tabela 11: Saldo de Aprovações, Reprovações e Exames

	GE		GC	
	Alunos	% Alunos	Alunos	% Alunos
Aprovados	5	62,50%	2	33,33%
Reprovados	1	12,50%	2	33,33%
Em Exame	2	25,00%	2	33,33%

Tabela 12: Saldo de Aprovações, Reprovações e Exames (Desistentes Excetuados)

	GE		GC	
	Alunos	% Alunos	Alunos	% Alunos
Aprovados	7	70%	2	20%
Reprovados	3	30%	8	80%

Tabela 13: Saldo de Aprovações e Reprovações

	GE		GC	
	Alunos	% Alunos	Alunos	% Alunos
Aprovados	7	87,50%	2	33,33%
Reprovados	1	12,50%	4	66,67%

Tabela 14: Saldo de Aprovações, Reprovações e Exames (Desistentes Excetuados)

4.4.6 Média do Percentual de Faltas

Um provável indicativo da motivação dos alunos que compuseram o grupo experimental reside no índice de faltas constatado (vide Tabela 15). Os alunos do grupo de controle faltaram 24,15% a mais do que aqueles do grupo experimental.

	GE	GC
Primeiro Semestre	10,00%	15,79%
Segundo Semestre	26,00%	28,00%
Ano Letivo	17,06%	21,18%

Tabela 15: Média do Percentual de Faltas

Sendo a carga horária da disciplina 136 horas-aula, destas, 76 foram ministradas no primeiro semestre e 60 no segundo semestre. Exige-se, ao menos, 75% de frequência.

4.4.7 Desistência da Disciplina

Há alto índice de desistência das disciplinas e do curso no primeiro ano de Ciência da Computação. Tal fato remeteu ao estudo e aperfeiçoamento da grade curricular do curso por parte da Coordenação do Departamento de Computação. Os docentes observaram existir grande exigência das disciplinas das áreas de Matemática e Física, que ultrapassam Programação de Computadores I em termos de carga-horária, mas desmotivam o aluno porquanto (em perspectiva discente) assumem o papel de descaracterizar o curso em um primeiro contato.

A despeito dessa situação, ainda se percebeu diferenças entre os alunos do grupo experimental e os do outro grupo. Além do número de alunos desistentes ser menor no grupo experimental (Tabela 16), o momento da desistência ocorreu posteriormente neste grupo se comparado ao de controle. Em média, os alunos do grupo experimental assistiram 77,94% das aulas da disciplina antes de desistirem (suficiente para aprovação por frequência, inclusive), enquanto os alunos do grupo de controle desistiram em média em 70,96% da carga-horária da disciplina (conforme Tabela 17).

	GE		GC	
	Alunos	% Alunos	Alunos	% Alunos
Desistência da Disciplina	1	10%	3	30%
Desistência do Curso	1	10%	1	10%

Tabela 16: Desistência da Disciplina

	GE		GC	
	Aula	% C/H	Aula	% C/H
Última aula assistida	106	77,94%	97	70,96%

Tabela 17: Última Aula Assistida Por Alunos Desistentes (Média)

5 Considerações Finais e Trabalhos Futuros

O método delineado propõe uma abordagem formal para a estruturação do conhecimento do domínio de Programação de Computadores, como também mecanismos para suportar a descrição simbólica dos aspectos de perícia na modelagem dinâmica de aprendizes. Procurou-se prover soluções específicas, partindo dos aspectos teóricos apresentados, tais como uma modelagem detalhada que evidenciasse a evolução do aprendiz frente ao conhecimento do domínio, e também a contribuição individual de cada enunciado para o referido progresso. Para isso, foram revisitados a técnica de grafo genético e o Modelo de Sobreposição [5]. As ferramentas de autoria e de aprendizagem, cujo conjunto foi denominado MÚLTIPLA, encontram-se em estágio avançado de implementação e permitem que seja experienciado todo um ciclo de desenvolvimento voltado a materiais didáticos para Programação de Computadores.

Para fundamentar afirmações sobre a generalidade do MÚLTIPLA, trabalha-se na avaliação das ferramentas e na definição das capacidades do paradigma de programação imperativista (como tipicamente nas linguagens Pascal e C). Os procedimentos de avaliação têm se concentrado na potencialidade dos formalismos de autoria, bem como na efetividade do aprendizado proporcionado. Os resultados positivos obtidos até então sugerem que as múltiplas representações externas [1] e os grafos genéticos formam um arcabouço pedagógico adequando no contexto de ambientes inteligentes.

Esforços subsequentes se concentrarão em avaliar formalmente o processo de autoria implementado pela pesquisa e depois em aprimorar as potencialidades do arcabouço proposto. Destaca-se a validação de duas hipóteses, a saber: **(a)** o grafo genético tem caráter geral suficiente, como linguagem, para o desenvolvimento de uma ampla gama de ambientes destinados ao processo de ensino e aprendizagem; e **(b)** as ferramentas interativas que suportam o processo de autoria mostram-se apropriadas no sentido de proporcionar maior rapidez e de também prover um arcabouço unificado para domínios de natureza prática e complexa.

Referências

- [1] S. Ainsworth. DeFT: A Conceptual Framework for Considering Learning with Multiple Representations. *Learning and Instruction*, 16(3):183–198, 2006.
- [2] R. Cox e P. Brna. Supporting the Use of External Representations in Problem Solving: The Need for Flexible Learning Environments. *Journal of Artificial Intelligence in Education*, 6(2/3):239–302, 1995.
- [3] A. I. Direne e D. Scott. Identifying the Component Features of Expertise in Domains of Complex Visual Recognition. *Information Technology Research Institute Technical Report Series*, University of Brighton, Lewes Road, Brighton, UK, 2001.
- [4] M. Gladwell. *Outliers: The Story of Success*. Little, Brown and Company, New York, NY, United States, 2008.
- [5] I. P. Goldstein. The Genetic Graph: A Representation for the Evolution of Procedural Knowledge. *International Journal of Man-Machine Studies*, 11(1):51–77, Jan. 1979.
- [6] R. W. Howard (ed.). *Concepts and Schemata*. Cassell Educational, Ann Arbor, MI, United States, 1990.
- [7] J. Lave. *Cognition in Practice: Mind, Mathematics and Culture in Everyday Life*. Cambridge University Press, Cambridge, UK, 1988.
- [8] A. Lesgold, H. Rubinson, P. Feltovitch, R. Glaser, D. Klopfer, e Y. Wang. Expertise in a Complex Skill: Diagnosing X-ray Pictures. In *The Nature of Expertise*, 311–342, Lawrence Erlbaum Associates, Hillsdale, NJ, United States, 1988.
- [9] E. Maschio e A. I. Direne. Múltiplas Representações Externas para o Ensino de Programação de Computadores. Workshop de Ambientes de Apoio à Aprendizagem de Algoritmos e Programação. In *Anais do XVIII Simpósio Brasileiro de Informática na Educação (SBIE/2007)*, São Paulo, SP, Brasil, 2007. Sociedade Brasileira de Computação.
- [10] P. Norvig. *Teach Yourself Programming in Ten Years*. <http://norvig.com/21-days.html>, Dez. 2014.
- [11] S. Palmer. Fundamental Aspects of Cognitive Representation. In *Cognition and Categorization*, páginas 259–303, Lawrence Erlbaum Associates, Hillsdale, NJ, United States, 1978.
- [12] A. R. Pimentel e A. I. Direne. Medidas Cognitivas no Ensino de Programação de Computadores com Sistemas Tutores Inteligentes. *Revista Brasileira de Informática na Educação (RBIE)*, 3:17–24, 1998.
- [13] P. Romero, R. Cox, B. du Boulay, e R. Lutz. A Survey of External Representations Employed in Object-oriented Programming Environments. *Journal of Visual Languages and Computing*, 14(5):387–419, Out. 2003.
- [14] T. del Soldado e B. du Boulay. Implementation of Motivational Tactics in Tutoring Systems. *Journal of Artificial Intelligence in Education*, 6(4):337–378, 1995.
- [15] M. Mozina, M. Guid, A. Sadikov, V. Groznik e I. Bratko. Goal-Oriented conceptualization of procedural knowledge. In *Anais da 11th International Conference on Intelligent Tutoring Systems (ITS/2012)*, 286–291, Creta, Grécia, 2012.
- [16] G. Wagner. SIMURENA: A Web Portal for Open Educational Simulation. In C. Laroque and J. Himmelsbach, R. Pasupathy, O. Rose, A. Uhrmacher (Eds.). In *Anais da Winter Simulation Conference (WSC/2012)*, 1–12, New Jersey, EUA, 2012.
- [17] A. Kutzke e A. Direne. Mediação do Erro na Educação: um Arcabouço de Sistema para a Instrumentalização de Professores e Alunos. In *Anais do XXIV Simpósio Brasileiro de Informática na Educação (SBIE/2014)*, da Sociedade Brasileira de Computação, Dourados, MS, 737–746, 2014.
- [18] A. Kutzke e A. Direne. FARMA-ALG: An Application for Error Mediation in Computer Programming Skill Acquisition. In *Anais da 17th International Conference on Artificial Intelligence in Education (IAED/2015)*. Lecture Notes in Computer Science, Springer, 690–693, 2015.
- [19] A. Direne, W. da Silva, F. Silva, L. Peres, A. Kutzke, D. Marczał, G. Barros, L. Moura, G. Bazzo. Aprofundamento da Mobilidade Tecnológico-Educacional por meio de Jogos Intelectivos como Facilitadores da Comunicação Professor-Aluno em Redes Virtuais de Ensino. In *I Workshop de Desafios da Computação Aplicada à Educação (DesafIE! 2012)*, Curitiba-PR, 1–10, 2012.
- [20] S. Ainsworth. The Educational Value of Multiple Representations When Learning Complex

Scientific Concepts. In J. K. Gilbert & M. Reiner & M. Nakhlel (Eds.), *Visualization: Theory and Practice in Science Education*, 191–208, New York: Springer, 2008.

- [21] P. Bivall, S. Ainsworth e L. Tibell. Do Haptic Representations Help Complex Molecular Learning? *Science Education*, 95(4), 700-719, 2011.

Artigo referente à seguinte tese, que obteve primeiro lugar no Concurso de Teses, Dissertações e Trabalhos de Conclusão de Curso em Informática na Educação (CTD-IE) de 2014:

E. Maschio. Modelagem do Processo de Aquisição de Conhecimento Apoiado por Ambientes Inteligentes. Tese de doutorado, Universidade Federal do Paraná, Agosto 2013.