Um Estudo sobre a Aprendizagem de Lógica de Programação Utilizando Programação por Demonstração

Cláudia Ferreira[1], Flávio Gonzaga[2,3], Rodrigo Santos[3]

[1]FAGOC, Faculdade Governador Ozanam Coelho, Brasil
CEP 36500-000 – Ubá, MG

[2]UNIFAL, Universidade Federal de Alfenas, Brasil
CEP 37130-000 – Alfenas, MG

[3]COPPE/UFRJ, Universidade Federal do Rio de Janeiro, Brasil
Caixa Postal 68511 – CEP 21945-970 – Rio de Janeiro, RJ

claudiacaroline.ferreira@gmail.com, {fbgonzaga, rps}@cos.ufrj.br

Abstract. *Considering the studies accomplished in the last years, teaching and learning process of programming logics is a challenge in Computer Science. In this sense, techniques such as Programming by Demonstration (PBD) can be useful to explore the motivation and maximize the students' performance. This paper presents a study to evaluate the PBD effects in learning process of programming logics considering students within an age range of* 14-17 years. *The results show that PBD contributes for the logical reasoning development, including students that not use video-game and/or computer for a long time.*

Resumo. *Com base em estudos realizados nos últimos anos, percebe-se que o processo de ensino e aprendizado da lógica de programação é um desafio para estudantes de Computação. Nesse sentido, técnicas como Programação por Demonstração podem ser úteis para explorar a motivação e maximizar o desempenho dos estudantes. Dessa forma, o presente trabalho visa apresentar um estudo realizado para avaliar os efeitos desta técnica no aprendizado de lógica de programação, considerando estudantes com idade entre 14 e 17 anos. Os resultados mostram indícios de que esta técnica contribui para o desenvolvimento do raciocínio lógico, inclusive para estudantes que não utilizam vídeo-game e/ou computador por muito tempo semanalmente.*

1. Introdução

O processo de ensino e aprendizagem de lógica de programação é considerado um desafio para estudantes de Computação, apesar dos numerosos esforços de pesquisa para melhorar esse processo [Baeza-Yates, 2000] [Tobar *et al.*, 2001] [Haden & Mann, 2003] [Neves & Coello, 2006] [Costa *et al.*, 2010]. A carga de conceitos abstratos nos primeiros anos dos cursos da área é significativa e pode ser decisiva para a motivação dos estudantes [Santos *et al.*, 2008a]. Além disso, programar não é uma tarefa trivial, pois requer o entendimento de uma linguagem específica, de algoritmos, da lógica da programação e das regras de negócios [Schots *et al.*, 2009]. Percebe-se que os estudantes acabam não querendo aprender uma nova linguagem, devido às dificuldades no entendimento, ao desconforto proporcionado e ao tempo requerido [Ben-Ari, 2001] [Pimentel *et al.*, 2003] [Coura, 2006]. Isso pode conduzir a altas taxas de reprovação e à

desistência de cursos na área, devido à diminuição da auto-estima e à geração de apatia, mesmo diante de atividades laboratoriais de programação [Santos *et al.*, 2008b].

Esses problemas fazem parte dos grandes desafios de educação em Computação para os próximos dez anos, conforme apresentado em [McGettrick *et al.*, 2004]. Este documento destaca o fato de que os estudantes percebem os cursos na área de Computação como "dominados" pela programação e aponta o desafio da inovação em reconhecer e acomodar competências e habilidades de estudantes, devido à amplitude de pesquisa e de aplicação da área. Essas questões estão diretamente relacionadas com os grandes desafios da pesquisa em Computação no Brasil, pois se concentram na busca de soluções melhor sedimentadas para o processo de formação de seus recursos humanos. Mais especificamente, artigos recentes enfatizam o problema da falta de qualidade em software e consideram sua implicação na indústria brasileira de desenvolvimento para exportação, atingindo o desafio *desenvolvimento tecnológico de qualidade* [SBC, 2006].

Estudos vêm sendo desenvolvidos a fim de tornar a aprendizagem de lógica de programação algo mais intuitivo. De acordo com Silva (2007), jogos interativos que exploram aspectos de interface gráfica proporcionam um meio lúdico para o entendimento e construção dos primeiros programas de computador, ao explorarem aspectos de cognição e de colaboração. Por outro lado, o processo de desenvolvimento de jogos pode ser utilizado como um atrativo para o ensino de lógica de programação. Nesse contexto, a Programação por Demonstração (*Programming By Demonstration* – PBD) é uma técnica que visa aproximar o usuário cada vez mais do ambiente de programação, sem que seja necessário aprender uma linguagem específica. A aplicação de PBD permite que os usuários, por meio de um ambiente baseado em interface gráfica, aprendam a programar por meio de exemplos, especificando "o que" deverá ser feito, sem ter a preocupação de "como" será feito.

Segundo Smith (2000), os estudantes que aprendem a programar utilizando um sistema de PBD tendem a obter melhores resultados do que aqueles que aprenderam pelo método tradicional (quadro-negro). Motivado por estes resultados, este artigo tem o objetivo de apresentar um estudo conduzido com estudantes de 14 a 17 anos (Ensino Médio) para verificar os efeitos da execução de um curso de curta duração de lógica de programação utilizando PBD sobre o aprendizado de algoritmos em Portugol (pseudocódigo) [Ferreira, 2008]. A faixa etária escolhida, além de se aproximar da idade dos estudantes ingressantes nos cursos de ensino superior, contribui ainda para analisar estratégias que fomentem o interesse de alunos, bem como a sua permanência e a afinidade para o desenvolvimento de habilidades de programação.

O artigo está organizado da seguinte forma: a Seção 2 discorre sobre a fundamentação teórica e apresenta a ferramenta *StageCast Creator* e o compilador *G-Portugol*, juntamente com a interface *Notepad++*, utilizados no estudo realizado; a Seção 3 descreve o planejamento, a execução e a análise dos resultados do estudo; e a Seção 4 apresenta algumas considerações finais e possibilidades de trabalhos futuros.

2. Fundamentação Teórica e Ferramentas Utilizadas para PBD

A PBD foi proposta com base na utilização de sistemas de recuperação de informações de banco de dados, onde o usuário era aquele que não tinha conhecimentos de programação. O banco de dados era conhecido como *Query By Example* (QBE) [Zloof,

1981]. O uso da PBD busca trazer vantagens e benefícios ao estimular o raciocínio lógico por possuir aplicações em: (i) escolas com crianças, com o intuito de estimular o raciocínio e a interatividade nas aulas com computadores; (ii) faculdades, durante o aprendizado das disciplinas de programação, algoritmos e tópicos relacionados, uma vez que não trabalha diretamente com linhas de código (faz-se apenas o uso da interface); e (iii) cursos técnicos, para formar profissionais qualificados para o mercado de trabalho. Com base em estudos prévios [Smith, 2000], percebe-se que a PBD auxilia no desenvolvimento do raciocínio seqüencial e lógico, e favorece o raciocínio cientifico.

A PBD pode ser aplicada em diversos contextos e possui um alto potencial na área de Informática na Educação e, como exemplo, destaca-se o *StageCast Creator*[1]. O *StageCast Creator* usa a técnica de PBD, cujo objetivo inicial era tornar constante o uso de computadores na educação, e trata-se de um software educacional voltado para o ensino da programação baseada em lógica, associada a exemplos (i.e., desenvolvimento de jogos sem a necessidade de implementar código). O *StageCast Creator* é uma ferramenta que reúne as tecnologias de PBD e regras visuais do tipo antes-depois, mostrando apenas o estado inicial e o estado final da regra. Essas regras possuem características diferentes das linguagens de programação tradicionais, onde o usuário precisa entender toda a sintaxe, por exemplo, o que deve passar como parâmetro, onde e quando usar ponto e vírgula etc. Dessa forma, programar no *StageCast Creator* consiste em mostrar ao personagem como efetuar a ação pretendida, a partir de uma situação inicial. A Figura 1 mostra como criar uma regra para um personagem na ferramenta.

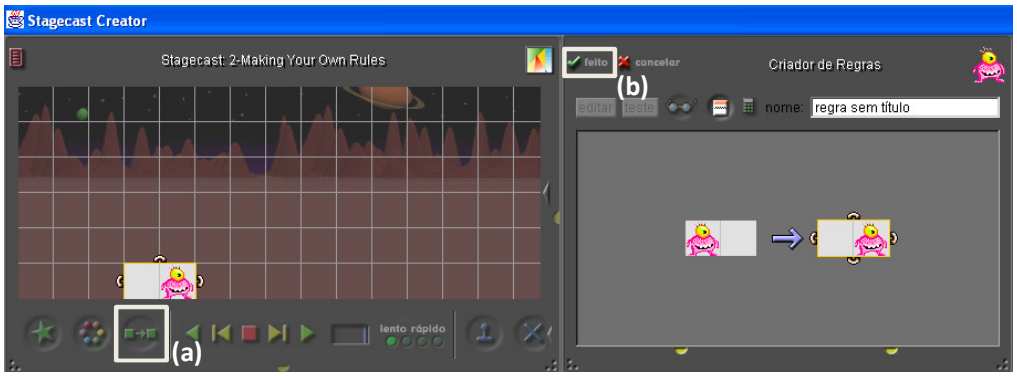

Figura 1 – Criação de regras: ativando (a) e concluindo a regra (b)

No momento em que a ferramenta de criação de regras é ativada (Figura 1.a), ao clicar no personagem, a tela de fundo apresenta uma aparência quadriculada. Uma vez ativada a criação, o próximo passo é definir a área de atuação, ação esta que é feita manipulando uma das alças ao redor do personagem, que permite variações quanto à direção. Na Figura 1, o foco está sendo estendido para a direita. A finalização da regra ocorre ao se clicar no botão *feito* (Figura 1.b). A regra mostrada no exemplo diz que sempre que houver um espaço vazio à direita do personagem, ele passa a ocupar esse espaço. Assim, o personagem irá se movimentar na direção estabelecida até não encontrar nenhum obstáculo. Além disso, o *StageCast Creator* disponibiliza ainda mais duas características: (i) sub-rotinas, onde as regras podem ser agrupadas para execução seqüencial ou aleatória; e (ii) propriedades, isto é, atributos variáveis para que os objetos possam ser modificados e avaliados a qualquer momento.

[1] A ferramenta *StageCast Creator* pode ser obtida em http://www.stagecast.com/creator.html.

Uma vez estudados os conceitos de lógica de programação por meio do *StageCast Creator*, pode-se utilizar o *G-Portugol*[2], que consiste em um compilador para a linguagem Portugol, muito utilizada para o ensino dos primeiros algoritmos com os estudantes de Computação. Um ambiente simples para a implementação desses algoritmos é o *Notepad++*[3], que oferece suporte para o desenvolvimento do código, permitindo a chamada do compilador *G-Portugol* a partir da própria interface. Na Figura 2.a, apresenta-se um código em Portugol no *Notepad++*; ao ser executado, o programa gerado através do *G-Portugol* escreverá na tela "Ola Mundo!" (Figura 2.b).

No contexto deste trabalho foram utilizadas as ferramentas apresentadas. A escolha foi feita com base no trabalho de [Coura, 2006]. Outras alternativas ao *StageCast Creator* foram analisadas, com destaque para os ambientes *Scratch*[4] e *Alice*[5]. A principal desvantagem em se usar o *Scratch* no presente trabalho se deve ao fato de que embora a ferramenta permita a criação de novas imagens para se usar no desenvolvimento, a mesma não dispõe de muitas imagens por padrão ao se instalar. Sendo o curso de PBD oferecido um curso de curta duração, contar com imagens pré-desenvolvidas no ambiente foi um atrativo a mais na escolha do *StageCast Creator*. O ambiente *Alice*, por sua vez, oferece muitas imagens por padrão, mas o desenvolvimento no ambiente é baseado no Paradigma Orientado a Objetos. Sendo o desenvolvimento no *StageCast Creator* mais próximo do paradigma estruturado, a sua utilização também foi preferida em comparação com o *Alice*, considerando as diretrizes tradicionalmente adotadas para o ensino dos primeiros algoritmos [Santos *et al.*, 2008b].

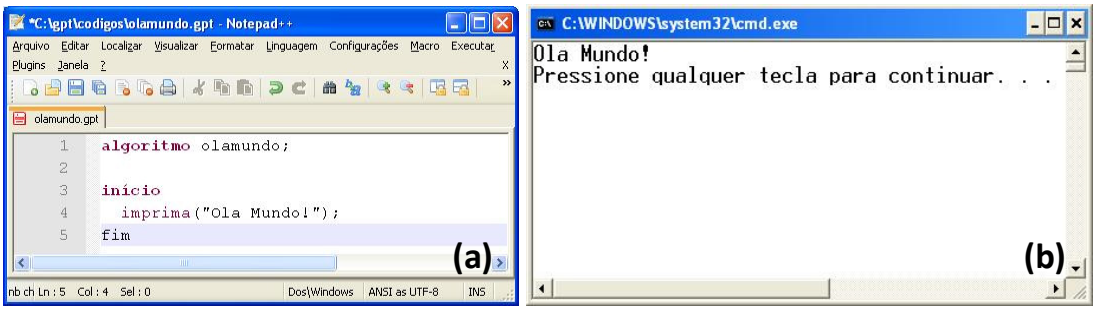

Figura 2 – Interface do *Notepad++* e a execução de um programa (*G-Portugol*)

3. Desenvolvimento do Estudo

Foi delineado um estudo segundo a abordagem GQM (*Goal-Question-Metric*) [Basili *et al.*, 1994], cujo objetivo é *analisar* o aprendizado de lógica de programação *com o propósito de* melhorar este processo *com respeito à* eficácia da aprendizagem de algoritmos em Portugol ao utilizar a técnica de PBD junto ao processo tradicional de ensino deste tópico *sob o ponto de vista de* professores de Algoritmos e Programação *no contexto de* um curso introdutório de lógica de programação para alunos de 14 a 17 anos. A principal motivação deste estudo está no fato de que os problemas no processo de ensino e aprendizagem de algoritmos e programação estão relacionados ao entendimento das abstrações de lógica pelos estudantes. Nesse contexto, o estudo consistiu na aplicação da estratégia supracitada com turmas de alunos de Ensino Médio

[2] O compilador *G-Portugol* pode ser obtido em: http://gpt.berlios.de/site/.
[3] O *Notepad++* pode ser obtido em: http://notepad-plus.sourceforge.net/br/site.htm.
[4] O *Scratch* pode ser obtido em: http://scratch.mit.edu/
[5] O *Alice* pode ser obtido em: http://www.alice.org/

em sala de aula (*in vivo*) com o acompanhamento dos pesquisadores/professores autores deste artigo. A partir da definição do objetivo, um estudo piloto (Seção 3.1) foi executado inicialmente, visando calibrar o plano do estudo. Após os ajustes realizados a partir dos problemas detectados, o estudo foi executado (Seção 3.2) e os seus resultados puderam ser analisados (Seção 3.3).

3.1. Estudo Piloto

Inicialmente, executou-se um estudo piloto, quando foram realizadas algumas visitas em escolas públicas para selecionar estudantes (i.e., os participantes do estudo). Os estudantes selecionados responderam a um questionário de caracterização do participante, que contemplava os seguintes aspectos: (i) idade (14 a 17 anos); e (ii) quantidade média de horas de uso de vídeo-game e/ou computador por semana. Uma vez respondido o questionário, os estudantes foram divididos em dois grupos semelhantes, formados por estudantes com perfis equilibrados (idade e uso de computador e vídeo-game). Com essa formação, o Grupo 1 foi selecionado para o curso de PBD, cuja carga horária total foi de 6 horas, distribuídas em 2 horas diárias por 3 dias. O curso foi ministrado por um dos pesquisadores envolvidos no Laboratório de Informática da Faculdade Governador Ozanam Coelho, com o uso de quadro branco e *data-show*. Cada estudante utilizou um computador individualmente para aprender a usar e criar jogos na ferramenta *StageCast Creator*. O detalhamento do perfil dos participantes é mostrado na Tabela 1 e o conteúdo ministrado no curso de PBD é exibido na Tabela 2.

Tabela 1 – Perfil dos participantes do estudo (por grupos)

	Grupo 1		Grupo 2	
	Idade	Quantidade de uso de computador/vídeo-game por semana (horas)	Idade	Quantidade de uso de computador/vídeo-game por semana (horas)
	16	9	16	35
	16	21	16	5
	16	26	15	28
	15	21	15	23
	15	14	15	2
	15	5	15	14
	15	80	15	35
	15	0	15	3
	15	7	15	42
	15	4	15	22
	15	36	14	4
	14	27	14	60
	14	12	14	7
	14	14	14	7
Média	**15,00**	**19,71**	**14,85**	**20,5**

Após o término do curso de PBD para o Grupo 1, ambos os grupos tiveram uma semana de curso de Portugol. Este curso foi ministrado por um outro pesquisador envolvido neste artigo, de forma que este não tivesse qualquer informação relativa sobre o grupo que havia feito o curso de PBD. O curso de Portugol foi realizado em sala de aula com quadro-negro (método tradicional). Os estudantes não dispunham de computador e os conteúdos trabalhados foram a estruturação de algoritmos e o desenvolvimento do raciocínio lógico. O conteúdo do curso é apresentado na Tabela 3. Após o término do curso, todos os estudantes foram submetidos a uma avaliação escrita.

Nesta avaliação, foram tratadas questões da organização seqüencial do raciocínio lógico. Um exemplo de questão é apresentado na Figura 3.

Tabela 2 – Conteúdo do Curso de PBD

Dias do Curso	Conteúdo
1ª	Apresentação do *StageCast Creator*, do *Player* do StageCast, execução de alguns exemplos.
2ª	Criação de um novo jogo, movimentos do personagem, interação do personagem com o ambiente do jogo.
3ª	Ordenação da regras do jogo, conclusões.

Tabela 3 – Conteúdo do Curso de Portugol

Dias do Curso	Conteúdo
1ª	Apresentação, estrutura básica de um algoritmo.
2ª	Conceito de variáveis e constantes, desenvolvimento de algoritmos de exemplo, estruturas condicionais simples.
3ª	Estruturas condicionais compostas, exercícios feitos em papel, conclusões.

Fazer um algoritmo que leia a nota de um aluno e imprima a mensagem de acordo com o valor da nota:

Comente cada linha do código.

1 Se nota >= 7,0 imprima “aprovado”
2 Se 5,0 <= nota < 7,0 imprima “prova final”
3 Se nota < 5,0 imprima “reprovado”

Figura 3 – Um exemplo de questão da avaliação escrita aplicada

Após a correção da avaliação, pôde-se constatar que, de forma geral, os alunos souberam organizar o raciocínio. No entanto, em função de não terem usado um ambiente de programação (e.g., *Notepad*++), os resultados produzidos ficaram um pouco distantes do padrão dos algoritmos desenvolvidos nesses ambientes. Assim, não foi possível fazer uma análise conclusiva. Outro aspecto observado foi a grande evasão do curso (aproximadamente 68%): dos 14 estudantes do Grupo 2, apenas 2 compareceram para fazer a avaliação e, do Grupo 1 (grupo que recebeu o curso de PBD antes), 7 compareceram. Nesse caso, a utilização da ferramenta *StageCast Creator* pode ter gerado uma motivação para o curso de Portugol. Dessa forma, para a execução do estudo, decidiu-se pela utilização do *Notepad*++ associado ao compilador *G-Portugol* (apresentados na Seção 2) e com a ministração dos cursos em quadro-branco.

3.2. Execução do Estudo

Após os ajustes realizados com base no estudo piloto, uma nova visita às escolas públicas foi realizada, e alunos com idade entre 14 e 17 anos foram selecionados para a execução do estudo. Nenhum dos participantes do estudo piloto foi selecionado, a fim de favorecer a validade interna do estudo (i.e., reduzir o viés ao se preocupar se o tratamento realmente causa o resultado). Analogamente ao estudo piloto, os participantes foram divididos em 2 grupos, gerados com perfis semelhantes. Apenas para diferenciar dos termos já usados na designação dos grupos da Seção 3.1, os grupos do

estudo serão referenciados como Grupo 1' e Grupo 2'. O Grupo 1' recebeu o treinamento em *StageCast Creator* com duração de 6 horas, distribuídas em 2 horas durante 3 dias, como o Grupo 1 da Seção 3.1. Após o término deste treinamento, os grupos foram reunidos em uma única turma para receberem o curso de Portugol, de maneira semelhante ao que foi desenvolvido no estudo piloto.

Na formação dos Grupos 1' e 2', 132 estudantes se inscreveram. Após a divisão igualitária dos grupos, no início do curso, o total de participantes dos Grupos 1' e 2' sofreu uma redução significativa para 60 estudantes no total, permanecendo 40 no Grupo 1' e 20 no Grupo 2'. Durante o curso, houve 18 desistências, sendo que dos 42 participantes que concluíram o curso no laboratório, 32 foram do Grupo 1', enquanto os outros 10 eram do Grupo 2'.

A avaliação aplicada ao final do estudo ocorreu de forma prática no Laboratório de Informática da Faculdade Governador Ozanam Coelho, realizada individualmente pelos participantes, consistindo na implementação de 4 algoritmos: (i) "alô mundo"; (ii) soma de dois números informados pelo usuário; (iii) cálculo da média de quatro números informados pelo usuário; e (iv) informação se um usuário é menor, adulto ou idoso a partir da idade fornecida. Os programas desenvolvidos visavam avaliar os seguintes aspectos, que foram ensinados no curso de Portugol: (i) raciocínio lógico na construção de algoritmos; (ii) variáveis; e (iii) estruturas condicionais.

Com relação à correção das avaliações, foi adotada a seguinte distribuição dos pontos, de acordo com a dificuldade dos algoritmos: (i) "alô mundo": 1 ponto; (ii) "soma": 2 pontos; (iii) "média": 3 pontos; e (iv) idade: 4 pontos. Para o programa cujo algoritmo possuía o pensamento lógico correto, mas que não compilava, e para aquele que compilava, mas que não funcionava como o esperado (atendendo parcialmente às exigências do exercício), foi atribuído a metade da pontuação. Na avaliação, apenas 6 participantes do Grupo 2' compareceram, enquanto que do Grupo 1', esse número foi 18. Embora os grupos estivessem equilibrados com relação à idade e ao tempo de uso de vídeo-game e/ou computador, em virtude das desistências, os grupos que realizaram a avaliação não estavam equilibrados nesses aspectos.

3.3. Análise de Resultados

Com o objetivo de fazer uma análise coerente, optou-se por analisar 6 participantes do Grupo 1', que tivessem perfis semelhantes aos 6 do Grupo 2' que realizaram a avaliação. Contudo, selecionar 6 participantes do Grupo 1' com perfis semelhantes poderia não resultar em uma análise coerente, dado que possivelmente poderia ocorrer de uma amostra de 6 participantes apresentar um desempenho melhor se comparada ao perfil de outra amostra, por exemplo, de desempenho pior. Numericamente falando, a partir de um grupo com 18 membros, 18.564 sub-grupos diferentes com 6 membros podem ser formados por combinação. Foi desenvolvido então um pequeno programa que obtivesse todos os sub-grupos do Grupo 1' com 6 membros. Para cada um dos 18.564 sub-grupos obtidos, foi verificado automaticamente se o sub-grupo possuía um perfil semelhante ao Grupo 2', que apresentava as seguintes características: (i) idade média: 15 anos; e (ii) quantidade de horas semanais de uso de vídeo-game e/ou computador: 16,67. Assim, foram considerados os sub-grupos com perfis que se enquadrassem nos intervalos: (i) $14 \leq$ idade ≤ 16; e (ii) $15{,}67 \leq$ horas vídeo-game e/ou computador $\leq 17{,}67$.

Dos 18.564 sub-grupos possíveis, apenas 2.036 se encaixaram no perfil acima (i.e., semelhantes ao Grupo 2'). Para cada um dos 2.036 sub-grupos, computou-se a média das notas das avaliações dos participantes, e comparou-se com a média das notas dos participantes do Grupo 2' (7,75). Os resultados foram os seguintes: (i) 81,5% dos 2.036 sub-grupos do Grupo 1' obtiveram uma nota média acima de 7,75; e (ii) 18,5% dos 2.036 sub-grupos do Grupo 1' obtiveram uma nota média abaixo de 7,75. Com base nestes resultados, pode-se verificar que existem indícios de que o uso da PBD para tornar o aprendizado da lógica de programação pode melhorar a compreensão deste conteúdo, ainda que seja oferecido como um curso de curta duração anterior a uma disciplina de introdução à programação (i.e., fundamentos de algoritmos), de maneira que não prejudique o planejamento da disciplina e ao mesmo tempo motive os estudantes no desenvolvimento de habilidades de raciocínio lógico e de programação.

Adicionalmente, foi analisado se os participantes que obtiveram as maiores notas nas avaliações (independente do grupo) possuíam também um uso médio de vídeo-game e/ou computador maior do que aqueles que obtiveram notas menores. Verificou-se, então, que os resultados estavam concentrados em duas faixas: aqueles que obtiveram acima de 9,0 na avaliação, e aqueles que ficaram abaixo de 7,5 (Figura 4). A Figura 5 mostra a distribuição dos participantes por nota, considerando os 12 participantes que utilizam mais de 20 horas semanais em média de vídeo-game e/ou computador, e os 12 que utilizam menos de 20 horas semanais em média. Como pode ser observado, dos 12 alunos que utilizam mais de 20 horas vídeo-game e/ou computador, 75% tiraram nota 10,0 na prova, enquanto que no outro grupo esse índice foi de 33,33%.

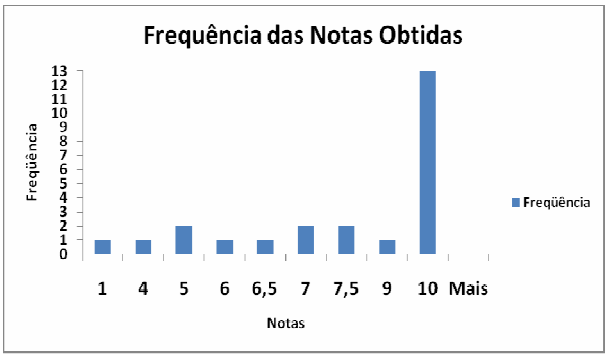

Figura 4 – Freqüência das notas das avaliações dos participantes

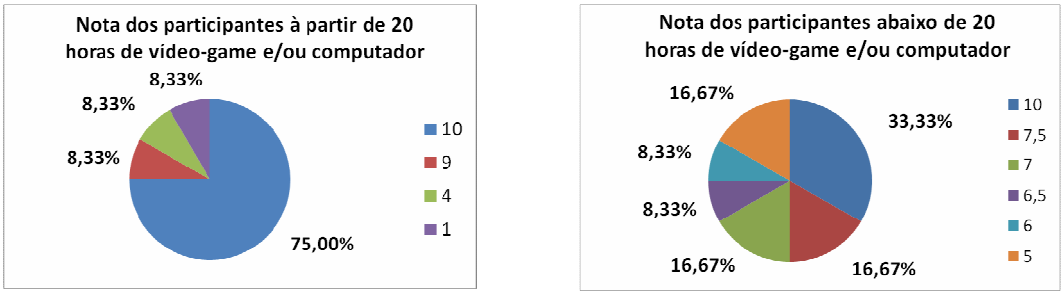

Figura 5 – Notas dos participantes que utilizam *menos* e *mais* de 20 horas semanais de vídeo-game e/ou computador

Esses resultados confirmam estudos já executados, que mostram que a noção de algoritmo está presente implicitamente nos jogos. Através dos jogos, os alunos se sentem

mais motivados na realização de tarefas e, conseqüentemente, desenvolvem o raciocínio, sobretudo quando os jogos agregam um conjunto de elementos multimídia que prendem mais a atenção do que questões em papel ou no quadro [Johnson, 2005].

Voltando o foco para os Grupo 2', dos 6 participantes, 3 alcançaram nota máxima na prova (50%), sendo que estes utilizam vídeo-game e/ou computador por mais de 20 horas semanais. Contudo, os outros 3 participantes obtiveram notas inferiores a 6,5, sendo que estes utilizam em média 6,33 horas semanais de vídeo-game e/ou computador (e todos utilizam menos de 10 horas por semana). Com os bons resultados obtidos com o Grupo 1', verifica-se que o curso de PBD pode contribuir para formar uma visão mais clara de conceitos de lógica de programação nos participantes que utilizam menos horas de vídeo-game e/ou computador. Dessa forma, o curso de PBD seria altamente recomendado para turmas com estudantes que apresentassem este perfil.

4. Conclusão

Conforme tradicionalmente discutido na literatura, o processo de ensino e aprendizagem de lógica de programação é alvo de pesquisas recorrentes devido às dificuldades e problemas observados por professores sobre o desempenho dos estudantes, e a técnica de PBD pode agregar contribuições positivas neste contexto. Dessa forma, o presente artigo apresentou um estudo conduzido com estudantes de 14 a 17 anos para verificar os efeitos da execução de um curso de curta duração de lógica de programação utilizando PBD sobre o aprendizado de algoritmos em Portugol. O estudo seguiu um planejamento previamente definido e calibrado por meio de um estudo piloto.

A partir dos resultados coletados, verificou-se que cursos preliminares de PBD de curta duração, associados ao uso da ferramenta *StageCast Creator*, podem gerar resultados interessantes para o aprendizado posterior de lógica de programação em Portugol. Os resultados permitem observar indícios de que alunos com perfis semelhantes aprendem a programar por demonstração antecipadamente e se saem melhor no aprendizado posterior de lógica de programação. Isso sugere a inserção de cursos de PBD como um pré-requisito interessante para melhorar a motivação e o desempenho dos estudantes que irão estudar algoritmos e programação. Além disso, estes resultados são ainda melhores para aqueles que dedicam mais tempo semanalmente ao uso de vídeo-game e/ou computador.

Como trabalho futuro, um estudo semelhante pode ser conduzido com estudantes de escolas públicas e com estudantes de escolas particulares, com o objetivo de verificar se os efeitos são convergentes ou divergentes, e ainda com estudantes de turmas de primeiro período de cursos de Ciência da Computação e de Sistemas de Informação, visando fortalecer os resultados observados no estudo – a validade externa (i.e., a busca por evidências a partir dos indícios detectados). Neste cenário, pode-se obter grupos de participantes com menor taxa de evasão, e resultados mais concretos podem ser vislumbrados a partir das avaliações da própria disciplina no decorrer do período.

Referências

Baeza-Yates, R.A. (2000) "Teaching Algorithms", In: *ACM SIGACT News*, v. 4, n. 26, 51-59.

Basili, V.R.; Shull, F.; Lanubile, F. (1999) "Building Knowledge through Families of Experiments". *IEEE Transactions on Software Engineering*, v. 25, n. 4 (Jul-Aug), 456-473.

Ben-Ari, M. (2001) "Constructivism in Computer Science Education", In: *Journal of Computers in Mathematics and Science Teaching*, v. 20, n. 1, 45-73.

Costa, H.A.X.; Santos, R.P.; Werner, C.M.L. (2010) "Uma Análise do Processo de Ensino e Aprendizagem de Engenharia de Software: Desafios e Soluções no Contexto Brasileiro", In: *Proceedings of the XI International Conference on Engineering and Technology Education*, Ilhéus, BA, Brasil, 367-371.

Coura, D.P. (2006) "*Produzindo Animações através da Programação por Demonstração*". Dissertação (Mestrado). Programa de Pós-Graduação em Ciência da Computação, Universidade Federal de Viçosa, Viçosa, MG, Brasil, 105p.

Ferreira, C.C. (2008) "*Estudo sobre a Aprendizagem de Lógica de Programação Usando Programação por Demonstração*". Monografia (Projeto Final). Faculdade Governador Ozanam Coelho, Ubá, MG, Brasil, 49p.

Haden, P.; Mann, S. (2003) "The Trouble with Teaching Programming", In: *Proceedings of the 16th Annual NACCQ*, Palmerston North, New Zealand.

Johnson, S. (2005) "*Surpreendente!: a televisão e o videogame nos tornam mais inteligentes*". Campus/Elsevier, 216p.

McGettrick, A.; Boyle, R.; Ibbett, R.; Lloyd, J.; Lovegrove, G.; Mander, K. (2004) "*Grand Challenges in Computing – Education*". The British Computer Society, 26p.

Neves, M.F.; Coello, J.M.A. (2006) "OntoRevPro: Uma Ontologia sobre Revisão de Programas para o Aprendizado Colaborativo de Programação Java", In: *Anais do XVII Simpósio Brasileiro de Informática na Educação*, Brasília, DF, Brasil, 569-578.

Pimentel, E.P.; França, V.F.; Omar, N. (2003) "Avaliação Contínua da Aprendizagem, das Competências e Habilidades em Programação de Computadores", In: *Anais do IX Workshop sobre Educação em Computação*, *XXIII Congresso da Sociedade Brasileira de Computação*, Campinas, SP, Brasil, 105-116.

Santos, R.P.; Costa, H.A.X.; Resende, A.M.P.; Souza, J.M. (2008a) "O Uso de Ambientes Gráficos para Ensino e Aprendizagem de Estruturas de Dados e de Algoritmos em Grafos", In: *Anais do XVI Workshop sobre Educação em Computação*, *XXVIII Congresso da Sociedade Brasileira de Computação*, Belém, PA, Brasil, 157-166.

Santos, R.P.; Vivacqua, A.S.; Souza, J.M.; Costa, H.A.X. (2008b) "Uma Proposta de Cenário para Ensino de Algoritmos e Programação com Contribuições de Cooperação, Colaboração e Coordenação", In: *Anais do XVI Workshop sobre Educação em Computação*, *XXVIII Congresso da Sociedade Brasileira de Computação*, Belém, PA, Brasil, 218-227.

SBC. (2006) "*Grandes Desafios da Pesquisa em Computação no Brasil – 2006-2016*". Relatório da Sociedade Brasileira de Computação, 22p.

Schots, M.; Santos, R.P.; Mendonça, A.P.; Werner, C.M.L. (2009) "Elaboração de um Survey para a Caracterização do Cenário de Educação em Engenharia de Software no Brasil", In: *Anais do II Fórum de Educação em Engenharia de Software*, *XXIII Simpósio Brasileiro de Engenharia de Software*, Fortaleza, CE, Brasil, 57-60.

Silva, C.E.M. (2007) "A Importância dos Jogos Eletrônicos como Elemento de Apoio à Produção, Resgate e Valorização Cultural", In: *Anais do XII Congresso de Ciências da Comunicação na Região Sudeste*, Juiz de Fora, MG, Brasil.

Smith, D.C. (2000) "Building Personal Tools by Programming". *Communications of the ACM*, v. 43, n. 8 (Aug), 92-95.

Tobar, C.M.; Rosa, J.L.G.; Coelho, J.M.A.; Pannain, R. (2001) "Uma Arquitetura de Ambiente Colaborativo para o Aprendizado de Programação", In: *Anais do XII Simpósio Brasileiro de Informática na Educação*, Vitória, ES, Brasil, 21-23.

Zloof, M.M. (1981) "QBE/OBE: A Language for Office and Business Automation". *IEEE Computer*, v. 14, n. 5 (May), 13-22.