Revista Brasileira de Informática na Educação, Volume 23, Número 1, 2015

Uma biblioteca gráfica para aprendizagem de estruturas de dados e algoritmos

A graphics API for learning data structures and algorithms

Jairo Francisco de Souza
Departamento de Ciência da Computação
Universidade Federal de Juiz de Fora
jairo.souza@ufjf.edu.br

Sandro Athaíde Coelho
Instituto de Ciências Exatas
Universidade Federal de Juiz de Fora
sandro.coelho@ice.ufjf.br

Resumo *As universidades estão começando a receber estudantes que fazem parte da geração Z, os quais têm demandado novas práticas de ensino. Entre as demandas trazidas por essa nova geração, estão a ansiedade por resultados imediatos e necessidade de recursos visuais para aprendizagem. Este artigo aborda uma estratégia de ensino e aprendizado de estruturas de dados apoiado pela biblioteca gráfica DSGraph. Esta estratégia tem sido utilizada na Universidade Federal de Juiz de Fora com o objetivo de adequar o ensino de conceitos abstratos como estruturas de dados e seus algoritmos para a nova geração de estudantes. A biblioteca permite, através da representação visual, a elucidação dos conceitos abordados durante o curso de estruturas de dados, aliado à participação ativa do aluno na construção e elaboração de suas soluções. Um estudo experimental foi realizado por dois anos com dois grupos de alunos. O primeiro grupo foi submetido à metodologia proposta neste artigo e outro grupo foi submetido à metodologia tradicional da disciplina. Os resultados mostram que a utilização da metodologia proposta neste trabalho traz resultados satisfatórios no índice de aprovação da turma e na satisfação do estudante.*

Palavras-Chave: *algoritmos, estruturas de dados, aprendizado visual.*

Abstract *Generation Z are the next generation of college students. These students have demanded new teaching practices. The use of visual resources for learning and the anxiety for immediate results are one of their demands. This paper discusses a strategy for teaching and learning data structures supported by the DSGraph library. This strategy has been used in Federal University of Juiz de Fora in order to adapt the teaching of abstract concepts such data structures and its algorithms for the new generation of students. The library aims, through visual representation, the elucidation of the concepts covered during the course of data structures, combined with active student participation in the construction and development of its solutions. An experimental study was carried out for two years with two groups of students. The first group was submitted to the proposed methodology and other group was submitted to the traditional methodology. The results show that the methodology proposed reaches satisfactory improvement of the approval rate and the students' satisfaction.*

Keywords: *algorithms, data structures, visual learning*

Recebido: 30 de Novembro de 2014 / Aceito: 13 Março de 2015
DOI: 10.5753/RBIE.2015.23.01.110

1. Introdução

As universidades estão começando a receber estudantes que fazem parte da geração Z (comumente chamada de "geração internet"). Esta nova geração de estudantes, nascidos após 1995, que cresceram com constante acesso à internet, que possuem acesso a tecnologias móveis e de diversas mídias sociais, têm motivado a discussão de novas práticas de ensino em universidades [16][5][15] assim como ocorreu nas escolas de ensino médio e fundamental [10][4][9]. Jane Hart [9] mostra que esta nova geração possui uma forma de aprendizado diferente das gerações anteriores e cita, entre outras características, a necessidade que esta nova geração possui de fazer descobertas através de experimentações, a necessidade de interação com outros aprendizes através do compartilhamento de suas ideias e descobertas, a gratificação imediata através de *feedbacks* e um melhor aprendizado através de recursos visuais.

No campo das Ciências Exatas e Engenharias, Chubin*et al* [5] discutem a necessidade de criação de formas inovadoras de ensino para manter esta nova geração de estudantes em faculdades de Engenharia. Os autores analisam a necessidade do ensino prático dentro de conteúdos teóricos, o uso de simulações, entre outras formas.

O ensino de estruturas de dados está presente no currículo da maioria, se não todos, os cursos da área da Computação (Ciência da Computação, Sistemas de Informação, Engenharia da Computação, Licenciatura em Computação e Engenharia de Software), além de alguns cursos da área da Engenharia como a Engenharia Civil, Mecânica, Produção, Computacional etc. Em 2013 a ACM publicou um novo currículo de referência para cursos de Computação [1]. O novo currículo define um Corpo de Conhecimento (*Body of Knowledge*) dividido em 18 áreas de conhecimento. O conhecimento em estruturas de dados está explicitamente descrito em nove das dezoito áreas de conhecimento.

Devido a sua importância, o ensino de estruturas de dados tem um papel fundamental nos cursos da área de Computação. Segundo Stillman & Peslak [18], esta é a primeira disciplina a qual o estudante aprende a necessidade do paradigma orientado a objetos, quando confrontado com questões relacionadas à modularidade e reuso do código. Além disso, programar estruturas de dados ajuda a desenvolver melhores habilidades de programação aos iniciantes. Ainda, construir estruturas abstratas de dados torna transparente aos novos programadores o funcionamento da máquina. Segundo Collins et al [6], o estudo de estruturas de dados permite construir a percepção de que o caminho óbvio e direto de solucionar problemas é quase sempre muito inferior às soluções que vão sendo estudadas em sala de aula.

Na Universidade Federal de Juiz de Fora (UFJF) tem-se observado uma dificuldade, por parte dos estudantes, de compreender a abstração apresentada em salas de aula e concretizar este conhecimento em aplicações mais complexas. Tal dificuldade permanece também nas atividades em laboratórios de programação, onde o estudante compreende o comportamento básico de algoritmos de manipulação das estruturas de dados, mas não consegue facilmente quebrar a barreira existente entre o conhecimento abstrato e o concreto. Além dessa dificuldade, a aprendizagem de estruturas de dados é dificultada por falhas no aprendizado de programação, como a manipulação de ponteiros, alocação de memória ou até dificuldades de entendimento lógico.

Este artigo aborda a estratégia utilizada na UFJF para melhorar a aprendizagem das estruturas de dados através da adoção de uma biblioteca gráfica para visualização de estruturas de dados e comportamento de seus algoritmos. Esta estratégia objetiva uma participação mais ativa do discente durante a programação de algoritmos e estruturas de dados, permitindo que o discente seja capaz de visualizar o comportamento do código implementado e consiga identificar, por si só, os erros que seu código possa conter.

A biblioteca, denominada DSGraph, foi desenvolvida na UFJF e o seu uso foi avaliado ao longo de dois anos em disciplinas de laboratório. Um estudo experimental foi realizado com dois grupos de estudantes. Um grupo foi submetido à aprendizagem com a DSGraph e outro grupo foi submetido à forma tradicional da disciplina.

Este artigo está organizado como se segue: a seção 2 apresenta as contribuições existentes na literatura para melhorar o ensino de estrutura de dados e algoritmos; a seção 3 apresenta a biblioteca gráfica para visualização de estruturas de dados criadas na UFJF e descreve as características que as difere das demais existentes; a seção 4 apresenta o estudo experimental realizado com os estudantes de programação da UFJF e mostra a eficácia da biblioteca no ensino e a seção 5 apresenta nossas conclusões.

2. O ensino de programação e seus desafios

O ensino de programação é central para os cursos da área de Computação, uma vez que a habilidade de programação do estudante de Computação será necessária na maior parte das disciplinas que cursará para completar o seu curso. Ainda, outros cursos, principalmente da área de Ciências Exatas, tem acrescentado disciplinas de programação na sua grade curricular. Essas grades visam, em alguns casos, suprir uma demanda cada vez maior da

sociedade por profissionais que consigam criar sua próprias soluções computacionais, independente da sua área de formação e, em outros casos, permitir que o estudante desenvolva seu raciocínio lógico para soluções de problemas. Assim, não há dificuldades de encontrarmos atualmente cursos de Química, Engenharias, Matemática, Física, entre outros, com disciplinas de programação na sua grade curricular.

As disciplinas básicas de programação geralmente são separadas em pelo menos duas partes. Num primeiro momento, é estimulado o raciocínio lógico através da apresentação e criação de algoritmos, permitindo que o estudante entenda como estruturar uma solução através de uma sequência de tarefas e de um fluxo de execução destas tarefas. Em outro momento, os estudantes são estimulados a perceber que cada problema permite diversas soluções e que a forma de manipular os dados podem influenciar na qualidade da solução. Neste momento, geralmente são apresentados conceitos como desempenho e alocação de recursos através do ensino de estruturas de dados.

Ainda, segundo Santos & Costa [14], a programação é um dos pontos chaves em um curso da área de Computação, pois é a atividade que supre o computador com meios para servir ao usuário. Seu uso adequado e o entendimento de conceitos, princípios, teorias e tecnologias podem conduzir ao desenvolvimento de produtos de software de qualidade, objetivo final, tanto quando se tem a computação como uma atividade meio como quando a computação é tida como uma atividade fim.

Diversas dificuldades, contudo, são encontradas por estudantes de programação, seja em cursos de Computação ou demais cursos. Segundo Dijkstra [7], a ausência de base lógico-matemática em anos anteriores à faculdade implica no despreparo do aluno, tornando o ensino de programação complexo e desmotivador. Estes alunos apresentam dificuldade em aplicar o raciocínio lógico pois estão acostumados a decorar conteúdos [10]. Em parte, contudo, a dificuldade de desenvolver o raciocínio lógico é decorrência da dificuldade encontrada pelos professores para acompanharem efetivamente as atividades laboratoriais de programação, dado o grande número de estudantes geralmente sob sua supervisão [22]. Esse despreparo do aluno o leva à falta de motivação ao se deparar com um erro no seu código [15]. Por conta disso, surge o desânimo baseado na crença de que a disciplina constituiu um obstáculo extremamente difícil de ser superado [15].

Segundo Pereira, Medeiros & Menezes [15], estudantes possuem dificuldade em programação por conta da falta de compreensão de abstrações. Segundo os autores, alunos de engenharia civil, quando vão criar seu primeiro projeto, uma casa, por exemplo, já passaram pela experiência de ter entrado em uma casa, sabem o que é uma porta e uma janela. Da mesma forma, um aluno de medicina já teve a experiência de ter entrado num consultório médico ou hospital. Por outro lado, um aluno que vai aprender a programar não tem noção clara do que é um sistema, um banco de dados, uma linguagem de programação, a não ser que já tenha trabalhado ou feito um curso de extensão.

O uso de ferramentas computacionais em sala de aula tem sido explorado para auxiliar no ensino de programação. Segundo [17], a elaboração de uma ferramenta computacional didática deve tornar o ensino do conteúdo abordado mais prático e abrangente, de forma a despertar o interesse do aluno, o seu espírito de pesquisa e a busca de informações que possam torná-lo um profissional crítico e de opinião sólida. Porém, além do uso da ferramenta computacional, o professor deve também utilizar sua criatividade e tentar resolver cada problema baseando-se em situações do cotidiano. Segundo [16], se isso for feito, o aluno começa a ter raciocínio lógico e ordem de pensamento.

O ensino de programação pode ser projetado para finalidades distintas: (1) aquele que objetiva unicamente o desenvolvimento do raciocínio lógico e utiliza uma linguagem de programação mais didática ou pseudocódigos ou (2) aquele que objetiva, além o desenvolvimento do raciocínio lógico, o entendimento mais profundo do funcionamento da máquina para criações de soluções complexas. O primeiro tipo de abordagem geralmente faz parte de cursos que pretendem formar profissionais para usar a computação como atividade-meio, enquanto a segunda faz parte da maioria dos cursos da área da Computação, os quais necessitam que o profissional possua domínio sobre o funcionamento da máquina e consiga criar soluções avançadas.

De acordo com os objetivos da disciplina, surgem restrições para o uso de uma ou outra ferramenta computacional de ensino. Para cursos que necessitam de uma abordagem mais profunda de programação, por exemplo, é comum a necessidade de explorar conceitos como aritmética de ponteiros, operações com binários e gerência dos recursos utilizados pela solução implementada. Em outros casos, torna-se ainda necessário que o estudante crie variações de uma mesma solução para fins de comparação, como diferentes implementações de uma estrutura de dados para analisar desempenho e gasto de memória. Nestes casos, torna-se relevante o uso de uma ferramenta computacional didática que permita ao estudante identificar problemas no comportamento do seu código, mas que possa ser integrada sem dificuldades ao código que o próprio estudante desenvolveu.

3. Trabalhos relacionados

A adoção de ferramentas computacionais para auxiliar no ensino de disciplinas de programação vem sendo proposta há algum tempo no meio acadêmico. Em especial, alguns autores propuseram recursos computacionais para permitir o ensino de programação de forma visual. Para Garcia *et al.* [9], o uso de movimento em tempo real, cores e sons enriquecem ainda mais o poder de comunicação. Vale ressaltar, contudo, que a ênfase na qualidade da animação gráfica para enriquecer essa abordagem pode levar os sistemas a um grau de complexidade que impede seu uso para fins de criação de animações pelos próprios aprendizes, limitando a sua utilização à interação com animações pré-determinadas.

Dois exemplos conhecidos que permitem a visualização do comportamento de algoritmos e estrutura de dados através de animações desenvolvidas pelos próprios aprendizes são o Scratch [14] e o ASTRAL [8].

O Scratch é uma ferramenta resultante de uma parceria entre a Portugal Telecom e o MIT. A ferramenta permite a criação de histórias, animações e jogos através de comandos pré-programados representados por figuras que podem ser agrupadas para gerar a animação desejada. O ambiente utiliza a linguagem LOGO [11], também criada para fins educacionais, sendo utilizado no ensino de programação para pessoas entre 8 e 16 anos de idade [12]. Com uma proposta similar ao Scratch, o ALICE [17] possui um ambiente para animações tridimensionais, também através de componentes de animação pré-programados, com o objetivo de ensinar programação para adolescentes.

O ASTRAL [8], por sua vez, é um ambiente de programação em PASCAL criado para ser utilizado em cursos superiores de estrutura de dados. A ferramenta possui um conjunto pré-determinado de animações, um ambiente para edição dos códigos que o aluno pode utilizar para montar e verificar o comportamento da estrutura de dados, com opções para submissão de arquivos de dados que podem ser lidos pelo programa para serem inseridos na estrutura de dados.

Alguns autores propõem ferramentas para o ensino de estruturas de dados de forma visual, permitindo animações mais detalhadas e explicativas dos algoritmos de manipulação das estruturas, porém sem permitir que o aprendiz implemente sua própria solução e a compare com a solução fornecida pela ferramenta. Essa abordagem passiva de aprendizagem é proposta pelas ferramentas EDDL [3], TED [7] e TBC-AED [13].

Segundo Garcia *et al.* [8], do mesmo modo que a experimentação durante a operação das animações enriquece o aprendizado mais do que a mera observação passiva delas, é de se esperar que, com um sistema onde a própria implementação das animações gráficas é facilitada a ponto de poder ser realizada pelo estudante, a absorção do funcionamento dos algoritmos seja ainda mais intensa. Dessa forma, consideramos que abordagens como a do ASTRAL e Scratch são mais enriquecedoras no processo de ensino e aprendizagem do que abordagens passivas como as fornecidas pelas ferramentas EDDL, TED e TBC-AED.

Neste sentido, nossa abordagem com a DSGraph difere das ferramentas EDDL, TED e TBC-AED por oferecer ao aluno a capacidade de construir seu próprio algoritmo. A DSGraph se difere do Scratch e do ALICE por objetivar o ensino de programação utilizando linguagens de programação de uso geral, ao invés de linguagens educacionais ou programação via componentes visuais. Além disso, a DSGraph auxilia no uso de recursos de mais baixo nível, como a manipulação de ponteiros e alocação de memória, permitindo que o estudante identifique erros no seu código em um desses processos.

Assim, a DSGraph compartilha algumas características com a ASTRAL, pois ambas são direcionadas para o ensino em cursos superiores. Ambas permitem o uso de linguagens de programação de uso geral, além de possibilitar que o aprendiz faça alterações no algoritmo com a finalidade de visualizar o comportamento do algoritmo na estrutura de dados. Por sua vez é interessante ressaltar que a DSGraph apresenta evoluções em algumas características, se comparado com o ASTRAL.

O ASTRAL é uma IDE. A DSGraph é uma biblioteca que o aprendiz insere como dependência no seu projeto. Dessa forma, o aprendiz pode adotar a IDE que for mais conveniente. A DSGraph foi desenvolvida para permitir simular um ambiente de desenvolvimento mais próximo à um ambiente profissional, onde o desenvolvedor usará uma IDE como Netbeans, Eclipse etc, e linguagens mais comerciais como C, C++ ou Java. De acordo com o plano de aula adotado no ensino de estruturas de dados, o professor pode fazer uso da DSGraph para que o aluno programe com estruturas heterogêneas em C (*struct*) ou pode ensinar conceitos como polimorfismo e herança através de classes em C++ ou Java.

O ASTRAL possui um ambiente mais controlado com o objetivo de deixar transparente ao aprendiz operações como entrada e saída de dados, sendo adequado quando o tutor deseja que o aprendiz mantenha sua atenção somente ao comportamento da estrutura. A visualização das estruturas de dados usando a DSGraph não impõe muitas restrições ao código do aprendiz, permitindo que ele possa implementar sua estrutura da forma que achar mais adequada, sendo aplicável quando o tutor queira proporcionar uma experiência de programação mais completa e próxima de um ambiente real.

Por fim, a DSGraph não é distribuída em módulos separados, uma para cada animação, como o ASTRAL. A biblioteca possui visualizadores para todas as estruturas dentro da mesma distribuição. Tal facilidade permite que em cursos avançados de estrutura de dados o aprendiz possa construir mais de uma estrutura no mesmo projeto e visualizar as suas partes, como é o caso de SkipLists [19]. Assim, a DSGraph permite uma maior liberdade de uso em diferentes linguagens e para diferentes implementações, uma maior flexibilidade para o estudante criar sua animação e visualizar partes da sua estrutura, além de uma maior integração ao código do estudante, visto que não é necessário utilizar uma sintaxe nova ou forçar o uso de códigos pré-implementados.

4. Visualizando estruturas de dados e algoritmos

A biblioteca DSGraph[1] é uma ferramenta que permite a visualização gráfica das estruturas de dados abstratas, buscando facilitar o ensino e aprendizagem das disciplinas de Algoritmos e Estrutura de Dados. Atualmente a biblioteca encontra-se disponível para projetos em linguagem C e C++. Uma versão para a plataforma Java está em desenvolvimento.

O projeto visa facilitar a aprendizagem de estruturas de dados e algoritmos por alunos de diversos cursos de graduação como Ciência da Computação, Sistemas de Informação, Licenciatura em Computação, e as várias Engenharias. Para facilitar o entendimento do código, a biblioteca permite que o aluno visualize o estado da sua implementação e, assim, entenda com mais facilidade o funcionamento do código construído.

A proposta é que o aprendiz, durante a construção de suas soluções, faça a importação da biblioteca, e através da utilização dos métodos providos pela mesma, analise graficamente o estado da sua solução. É importante ressaltar que a DSGraph não gera código para o aluno, mas somente dispõe de meios para que o mesmo visualize a o estado da estrutura por ele implementada.

No desenvolvimento do projeto DSGraph foi utilizada a biblioteca Allegro [2] que é completamente escrita em C e distribuída de forma livre para projetos em C e C++. A Allegro é uma biblioteca gráfica voltada para construção de jogos, possuindo suporte nativo a gráficos 2D e 3D, manipulação de imagens, texto, áudio, arquivo midi, suporte a DirectX e OpenGL além de ser multiplataforma. No projeto foi adotada a versão 4.0.

Os métodos da biblioteca permitem personalizar as cores e realçar os elementos em exibição, controlar o tempo de transição das animações e definir o tipo da estrutura, além de permitir exibir comentários na tela. Um das diretrizes definidas para o projeto foi fornecer métodos minimamente invasivos, facilitando o processo de desacoplamento da biblioteca do código do aprendiz. O foco foi evitar a poluição do código, visando não prejudicar a compreensão do aluno em questões fora do escopo do aprendizado de algoritmos e das estruturas de dados apresentadas nas aulas de laboratório.

Atualmente com a DSGraph é possível a visualização das seguintes estruturas de dados: árvore binária, árvore B, árvore B+, árvore QuadTree, árvore ternária, árvore Vermelho-Preto (ou rubro-negra), árvore PATRICIA, Tries Multiways, fila, lista circular, lista duplamente encadeada, lista simplesmente encadeada, lista com descritor, pilha, matrizes e vetores, além de ponteiros.

As seções seguintes apresentam a arquitetura da biblioteca e sua forma de uso.

4.1 Arquitetura

A arquitetura interna foi projetada para simplificar a inclusão de novas funcionalidades e, sem grandes modificações, as evoluções da Allegro. O projeto ainda abre a possibilidade de uma completa substituição da biblioteca gráfica sem a necessidade de alterações nos demais módulos internos.

Cada estrutura de dados suportada pela DSGraph possui heurísticas, que realizam a leitura da variável ou objeto passado pelo aluno via parâmetro. Esta leitura converte todo o conteúdo lido para uma segunda estrutura de dados – interna à DSGraph. Tal estrutura contém informações relevantes para a representação dos elementos na tela, como posicionamento, tamanho das caixas, cores e conectores – esse último, quando aplicável. Finalmente, esta estrutura é lida por um *wrapper*, invocando os métodos da Allegro que exibe os elementos na tela. A figura 7 ilustra em alto nível a divisão interna das camadas da DSGraph.

[1] http://dsgraph.sourceforge.net/

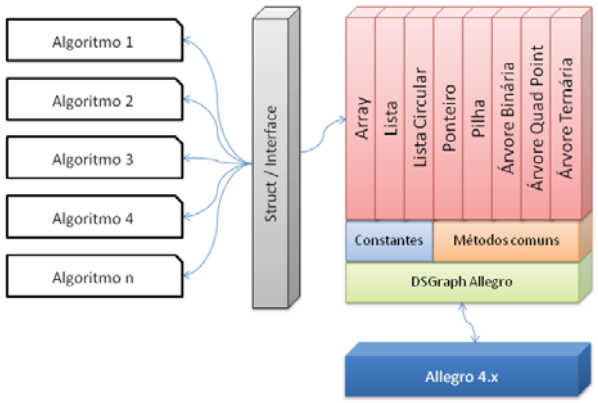

Figura 7: Arquitetura da DSGraph.

A biblioteca é extensível, de forma que novas visualizações de estruturas podem ser acrescentadas facilmente. Para tal, é necessário registrar uma nova constante no módulo Constantes e criar uma nova classe que implementa os métodos contidos no módulo "Métodos comuns". Ainda, visualizações alternativas para o mesmo tipo de estrutura de dado podem ser acrescentadas. A biblioteca está sendo disponibilizada como código aberto e o código está armazenado no repositório Git do projeto[2].

4.2 Utilização da DSGraph

A finalidade da biblioteca é permitir que o aprendiz consiga entender o comportamento da sua solução e ajudar na identificação de comportamentos inesperados. Para isso, o aprendiz pode incluir a biblioteca no seu código e chama-la em momentos específicos para montar a animação que desejar. Contudo, é importante que a criação das animações seja simples o bastante para não tirar o foco de aprendizagem do estudante. Ainda, é importante que a biblioteca não limite a solução do estudante, impondo restrições ao seu código.

Para demonstração do uso da DSGraph, toma-se, por exemplo, a implementação de uma árvore binária de busca. O exemplo de código em linguagem C a seguir retrata o uso da ferramenta para visualizar a estrutura implementada. O código de implementação da árvore é de responsabilidade do aluno, porém, alguns métodos na Figura 1 foram fornecidos pela DSGraph, os quais detalharemos a seguir.

[2] https://github.com/ufjf-dcc/dsgraph

```c
#include<stdio.h>
#include<stdlib.h>
#include<stdint.h>
//cabeçalhoDSGraph.h
#include "DSGraph.h"
// tipo fornecido pela biblioteca
typedef struct NODE_STRUCTURE NO;
//Inserir novo nó na árvore
NO *inserir (NO *pRaiz, int valor){....};
//Retirar um valor na árvore
No *remover (*pRaiz, int valor){....};
//Buscar nó em uma árvore binária de busca
No *buscar (NO *pRaiz, int valor){....};

int main( ) //DSGraph
{
    init ($ARVORE_BINARIA);
    showComment("TAD Árvore binária de busca");
    //Meu código para inserção na árvore
    NO *raiz = NULL;
    raiz = inserir(raiz, 15);
    raiz = inserir(raiz, 9);
    raiz = inserir(raiz, 20);
    raiz = inserir(raiz, 7);
    raiz = inserir(raiz, 11);
    raiz = inserir(raiz, 18);
    raiz = inserir(raiz, 22);
    raiz = inserir(raiz, 10);
    raiz = inserir(raiz, 13);
    raiz = inserir(raiz, 5);
    raiz = inserir(raiz, 8);
    //DSGraph
    setSleepTime(10);
    show(raiz,0);
    terminateDSGraph();
    return 0;
}
```

Figura 1: Inclusão dos métodos da DSGraph para visualizar árvores binárias de busca

O código da figura 1 possui três métodos (inserir, remover e buscar) que foram implementados por um aprendiz. O tipo de dado NODE_STRUCTURE é fornecido pela DSGraph caso seja implementado em linguagem C. Para linguagem C++, é fornecida uma classe abstrata que o aprendiz pode especializar com novos atributos e métodos. Ao final da implementação acima, o aprendiz decidiu visualizar se um dado conjunto de dados foi corretamente inserido na estrutura. No método *main()*, o aprendiz introduziu cinco métodos fornecidos pela DSGraph. As funcionalidades desses métodos são descritas abaixo:

- init() - inicializa a biblioteca com a estrutura a ser mostrada na tela. Um conjunto de constantes identifica cada tipo de estrutura. De acordo com a constante fornecida, a biblioteca utiliza um *template* correspondente para visualizar a estrutura.
- showComment() – imprime um texto na janela de exibição da estrutura.

- setSleepTime() – informa por quantos segundos a imagem será exibida. De acordo com o tempo especificado, o aprendiz pode controlar a velocidade da sua animação para cada exibição do estado da estrutura.
- show() - Exibe o estado da estrutura na janela de exibição. A chamada ao método *show()* pode ocorrer em diferentes trechos do código para que o aprendiz possa verificar o comportamento do seu algoritmo.
- terminateDSGraph() - Método responsável por descarregar da memória todos os recursos utilizados durante a exibição da estrutura.

A visualização gerada pelo código da figura 1 é apresentada pela figura 2.

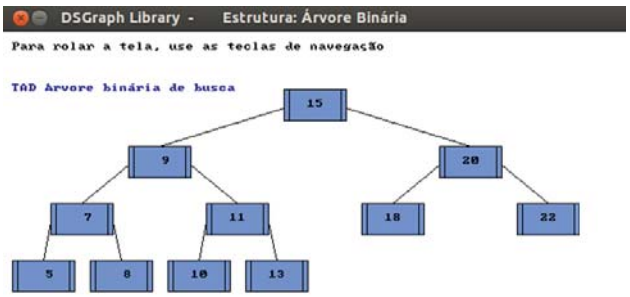

Figura 2: Visualização gráfica da árvore binária de busca com a DSGraph.

Erros de implementação podem ser facilmente identificados. Por exemplo, na figura 3 é apresentado uma árvore binária de busca que foi gerada com erro no método de inserção, o que acarretou no posicionamento incorreto do valor 20 na árvore.

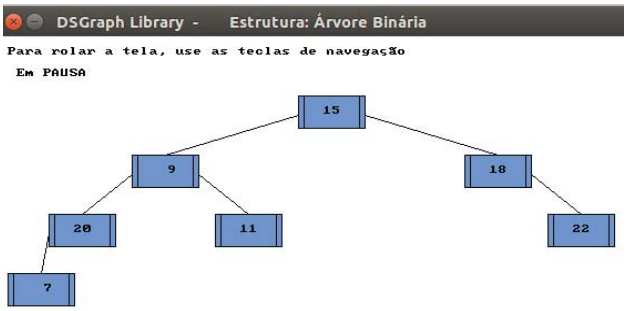

Figura 3: Visualização gráfica da árvore binária de busca com erro na inserção na sub-árvore esquerda do nó de valor 9.

A manipulação equivocada de ponteiros é um dos pontos dos maiores fatores causadores de erros cometidos em disciplinas práticas de programação. A DSGraph permite que o aprendiz identifique problemas na manipulação de ponteiros, como pode ser verificado na figura abaixo. Neste exemplo, a sub-árvore esquerda da raiz foi perdida ao armazenar no respectivo ponteiro um valor incorreto. Tal forma de verificação do comportamento do código é mais rápida para o aluno iniciante, uma vez que este nem sempre está familiarizado com ferramentas de depuração do código e, ainda que esteja, estas ferramentas geralmente possibilitam a observação de valores de variáveis dentro de um bloco sendo necessário verificar valores de ponteiros de cada nó. Com a DSGraph, o estudante pode verificar o conjunto de nós de uma única vez, se concentrando em identificar a anomalia e a sua solução.

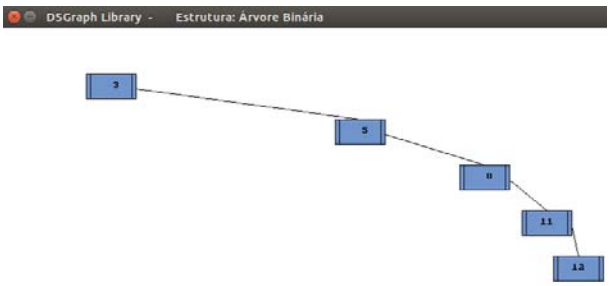

Figura 4: Visualização gráfica da árvore binária de busca com a DSGraph onde a sub-árvore esquerda da raiz foi perdida.

A biblioteca pode ser utilizada também para auxiliar na aprendizagem de algoritmos, principalmente quando estes manipulam estruturas mais simples como vetores e matrizes ou somente valores primitivos. Em disciplinas básicas de programação é comum a manipulação de vetores e matrizes para reforçar conceitos como laços e condicionais. A figura abaixo mostra a animação gerada por um estudante para visualizar o código criado para ordenar um vetor utilizando o algoritmo BubbleSort em ordem decrescente.

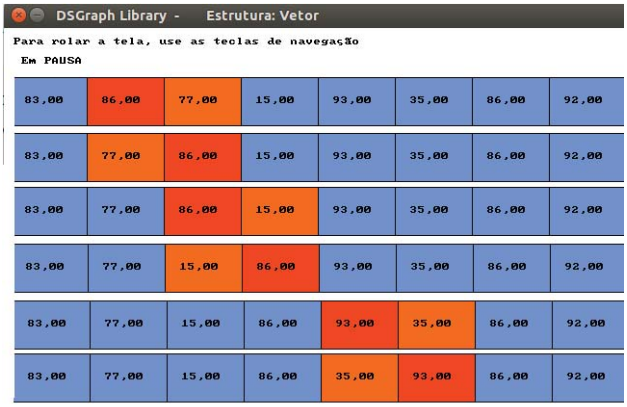

Figura 5: Animação criada por um estudante para verificar o comportamento do algoritmo de ordenação implementado.

Abaixo são apresentadas algumas das visualizações disponíveis para outras estruturas de dados, como a árvore B (figura 6), listas encadeadas com descritor (figura 7), árvore ternária (figura 8) e fila (figura 8).

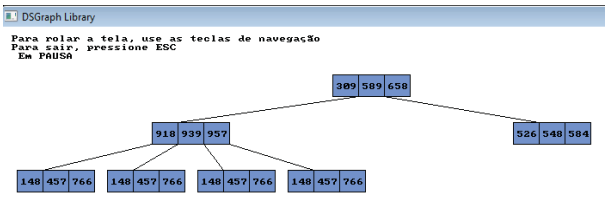

Figura 6: Visualização gráfica da árvore B com a DSGraph.

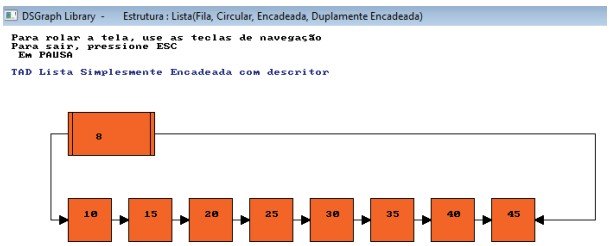

Figura 7: Visualização gráfica da lista encadeada com descritor na DSGraph.

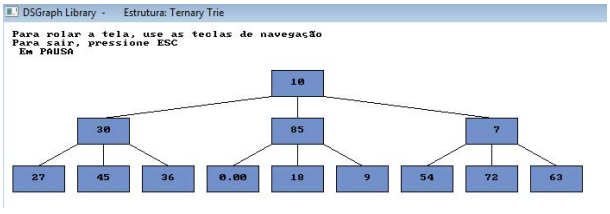

Figura 8: Visualização gráfica da árvore ternária com a DSGraph.

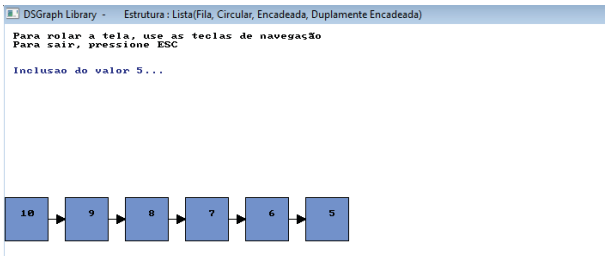

Figura 9: Visualização gráfica da fila com a DSGraph.

A biblioteca permite a visualização de estruturas que armazenam valores inteiros, ponto flutuante ou caracteres. Para projetos em C++ ou Java, é fornecido um tipo de dado (uma classe abstrata) e o aprendiz deve implementá-la para qualquer estrutura de dados. Assim, a biblioteca permite que o aprendiz crie sua própria estrutura e a visualize com os métodos padrões fornecidos.

5. Avaliação

Para avaliar a DSGraph, foi realizada uma pesquisa qualitativa com alunos da Universidade Federal de Juiz de Fora para verificar a aceitação da biblioteca pelos discentes e as consequências do seu uso. A pesquisa contou com a participação de alunos matriculados na disciplina Laboratório de Programação II no período de quatro semestres letivos (início de 2011 até final de 2012). Estes são alunos dos cursos de Ciência da Computação, Sistemas de Informação, Engenharia Sanitária e Ambiental e do Bacharelado Interdisciplinar em Ciências Exatas. Ao todo, participaram da pesquisa 242 alunos.

A disciplina de DCC107 - Laboratório de Programação II é uma disciplina prática que objetiva a implementação das estruturas de dados estudadas na disciplina teórica denominada DCC013 - Estrutura de Dados. Na disciplina DCC107 o aprendiz implementa, a cada semana, uma estrutura de dado e alguns algoritmos de manipulação com a supervisão de um professor e de monitores. O conteúdo da disciplina abrange a implementação de matrizes (esparsa, identidade, triangular), listas encadeadas, pilhas e árvores binárias de busca. Nas turmas avaliadas nesta pesquisa, foi utilizado a linguagem C++, IDE CodeBlocks e a biblioteca DSGraph. O professor de cada turma começa a aula indicando qual estrutura de dado e suas operações serão implementadas durante as duas horas de prática de laboratório. Os aprendizes iniciam o

trabalho criando um novo projeto no CodeBlocks e importando a DSGraph e uma classe de interface fornecida pelo professor contendo a assinatura dos métodos a serem implementados. Durante o trabalho de implementação, o aprendiz é incentivado a utilizar a DSGraph para verificar o estado da sua estrutura de dado e montar uma pequena animação para demonstrar o funcionamento correto dos métodos implementados.

Durante a prova final da disciplina, um questionário contendo perguntas relacionadas à DSGraph foi entregue a todos os 242 alunos. O questionário identifica alunos que estão fazendo a disciplina pela primeira vez e alunos repetentes, e contém cinco perguntas principais, sendo a última aplicada aos alunos que não cursavam a disciplina pela primeira vez:

Q1. Ao iniciar a disciplina, você conseguia entender com facilidade os algoritmos apresentados pelo professor na DSGraph?

Q2. A DSGraph te ajudou a compreender as estruturas de dados para essa prova?

Q3. Você recomendaria a DSGraph para outros alunos?

Q4. Foi possível construir seus algoritmos utilizando a DSGraph para entender o seu funcionamento?

Q5. Em comparação com períodos anteriores, você considera que sua compreensão de algoritmos melhorou por causa da DSGraph?

Todas as perguntas podiam ser respondidas marcando as opções "Concordo totalmente", "Concordo parcialmente", "Discordo", "Não sei/prefiro não opinar".

A figura 8 apresenta as respostas às perguntas 1 a 4. A figura mostra que os aprendizes, na sua maioria, consideraram que a biblioteca os ajudou a compreender os algoritmos e estruturas de dados e a recomendaria para outros aprendizes.

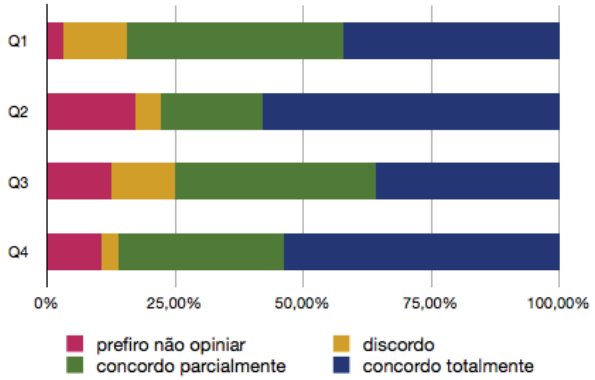

Figura 8: Respostas às perguntas 1 a 4 do questionário.

Para melhor analisar a percepção do aprendiz sobre o uso da biblioteca, foi analisada a resposta à questão Q5 dos alunos que não estão cursando a disciplina pela primeira vez e que cursaram a disciplina no semestre anterior sem a utilização da DSGraph. A figura 9 apresenta os resultados. Pode-se verificar que a maioria dos aprendizes consideraram que compreenderam melhor o conteúdo da disciplina ao utilizarem a biblioteca gráfica. Poucos foram os alunos que consideraram que essa abordagem não os ajudou a compreender o conteúdo da disciplina (2%) e houve uma perda de 11% dos participantes que não responderam a questão.

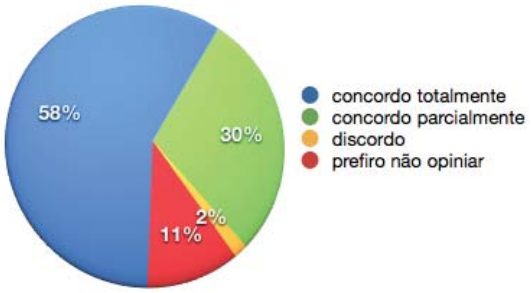

Figura 9: Estudantes que consideraram que a DSGraph auxiliou na compreensão de algoritmos em comparação ao período anterior.

Por fim, foram analisadas as taxas de aprovação nas disciplinas práticas utilizando os dados recuperados do Sistema de Gestão Acadêmica da UFJF ao longo dos 2 anos da pesquisa. Neste período, algumas turmas adotaram a prática convencional de ensino e foram consideradas como grupo controle desta pesquisa. Os professores de todas as turmas seguem o mesmo material de aula, o

mesmo cronograma de atividades e as mesmas avaliações.

A figura 10 apresenta uma comparação com dados de aprovação das disciplinas de Laboratório de Programação II nos períodos letivos de 2011.1, 2011.3, 2012.1 e 2012.3. Nas turmas denominadas como "convencionais" (em verde na figura) os aprendizes eram incentivados a programar seus algoritmos na IDE Codeblocks e linguagem C++, com o auxílio do professor, o qual se utilizava do quadro branco e projeções para explicar os algoritmos. Nas turmas que utilizaram a DSGraph (em azul na figura), os aprendizes recebiam o mesmo tratamento da turma convencional. Contudo, estes eram apresentados à DSGraph e eram incentivados a visualizar o comportamento dos seus algoritmos utilizando a biblioteca.

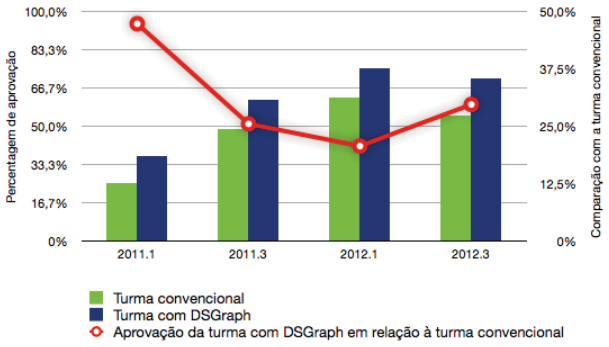

Figura 10: Comparação da taxa de aprovação nas turmas que utilizaram a DSGraph e o grupo controle.

A figura 10 mostra que houve uma maior aprovação (ver eixo Y à esquerda) nas turmas que utilizaram a DSGraph que nas turmas convencionais. A linha vermelha mostra que a aprovação nas turmas que utilizaram a DSGraph melhorou pelo menos em 20% (ver eixo Y à direita) em relação às turmas convencionais.

6. Conclusões

Conforme as universidades vêm recebendo uma nova geração de estudantes, torna-se imprescindível discutir e adaptar as práticas de ensino. Esta nova geração de estudantes, a primeira a nascer com acesso constante à internet, dispositivos móveis e acesso às redes sociais, são caracterizados por necessitarem de gratificação imediata através de *feedbacks*, fazer descobertas através de experimentações e aprender através de recursos visuais.

A abordagem de ensino de estrutura de dados praticada pelo Departamento de Ciência da Computação da UFJF prevê a necessidade de experimentação e a adoção de recursos visuais ao longo da elaboração de exercícios de programação para que o aprendiz possa visualizar tipos abstratos de dados e o comportamento da sua solução implementada. Este artigo apresentou a biblioteca DSGraph utilizada para auxiliar os aprendizes no aprendizado de estruturas de dados. A pesquisa realizada ao longo de quatro semestres letivos mostra que essa prática de ensino tem gerado resultados positivos.

A biblioteca DSGraph permite que o aprendiz crie suas animações para verificar o comportamento do seu algoritmo. Diferente de outras soluções, a DSGraph pode ser utilizada em qualquer ambiente de desenvolvimento para linguagem C, C++ ou Java, e permite que o aprendiz desenvolva a sua solução com seu próprio tipo abstrato de dado. Ao contrário do Scratch e ASTRAL que possuem uma IDE própria e, por esse motivo, mantém o ambiente mais controlado, a DSGraph permite que se simule em laboratório um ambiente mais próximo do ambiente que encontrará profissionalmente. Por outro lado, a flexibilidade que a DSGraph permite pode tornar o seu uso mais adequado para aprendizes que já possuem um conhecimento básico de programação.

Em relação ao ensino, o uso da DSGraph é menos intrusivo para o plano de aula do professor, uma vez que não impõe uma única linguagem de programação, não impõe uma nova IDE ao aluno e permite que o professor aborde todas as estruturas e algoritmos que planejava sem imposição de códigos próprios da biblioteca. Para o discente, a habilidade de visualizar sua estrutura ou algoritmo, auxilia na identificação de erros na manipulação de ponteiros ou alocação de memória, evitando a frustração frequente em programadores iniciantes quando não conseguem identificar o motivo dos resultados inesperados da sua solução.

Agradecimentos

Agradecemos aos alunos envolvidos na implementação da DSGraph, aos professores do Departamento de Ciência da Computação da UFJF que ministraram a dis-

ciplina DCC107 – Laboratório de Programação II e que adotaram o DSGraph como apoio às suas aulas, e aos alunos que aceitaram participar dessa pesquisa.

Referências

[1] ACM. Computer Science Curricula 2013 - Ironman Draft. Fevereiro de 2013. Disponível em http://www.computingportal.org/cs2013. Acesso: 20 de setembro de 2013.

[2] Allegro Community. The Allegro 5 Library Reference Manual. Acesso em 15 de Março de 2013. ONLINE. Disponível em: alleg.sourceforge.net/a5docs/refman/index.html.

[3] Azul, A. A. e Mendes, A. J. EDDL: Um Programa Didático sobre Estruturas de Dados Dinâmicas Lineares. 3o Simpósio Investigação e Desenvolvimento de Software Educativo – 1998. Évora, Portugal.

[4] Bekebrede, G., Warmelink, H.J.G, Mayer, I.S. Reviewing the need for gaming in education to accommodate the net generation. Vol 57 (2), 1521-1529 pp. Setembro de 2011.

[5] Chubin, D., Donaldson, K., Olds, Barbara., Fleming, L. Educating Generation Net—Can U.S. Engineering Woo and Win the Competition for Talent?. Journal of Engineering Education, Vol. 97 (3), 245-257 pp. Julio de 2008.

[6] Collins, W., Tenenberg, J., Lister, R., and Westbrook, S. 2003. "The role for framework libraries in CS2." In Proceedings of the 34th SIGCSE Technical Symposium on Computer Science Education (Reno, Nevada, USA, February 19 - 23, 2003). SIGCSE '03. ACM Press, New York, NY, 403-404.

[7] Dijkstra, E. W. On the Cruelty of Really Teaching Computing Science. Communication of ACM, 1398-1404, 1989.

[8] Flávio, D. TED - Tutorial de Estruturas de Dados, desenvolvido durante estágio supervisionado na Universidade do Vale do Itajaí – 2004. Disponível em http://www.tutorialdeestruturadedados.8m.com Acesso: 20 de setembro de 2013.

[9] Garcia, I. C., Rezende, P. J. & Calheiros, F. C. Astral: Um Ambiente para Ensino de Estruturas de Dados através de Animações de Algoritmos. Revista Brasileira de Informática na Educação, 1997.

[10] Gois, A., Pinho, A. Brasil é reprovado, de novo, em matemática e leitura. Disponível em http://www1.folha.uol.com.br/folha/educacao/ult305u351481.shtml. Acesso em 23 de novembro de 2014.

[11] Hart, J. Understanding Today's Learner. Learning Solutions, 1-11 pp. 2008.

[12] Kumar. The Net Generation's Informal and Educational Use of New Technologies. In Education. Vol 16 (1). 2010.

[13] LOGO, What is Logo?. Disponível em http://el.media.mit.edu/logo-foundation/logo/index.html. Acesso em 24 de setembro de 2013.

[14] Maloney, John, et al. "The scratch programming language and environment." ACM Transactions on Computing Education (TOCE) 10.4 (2010): 16.

[15] Pereira, P. S., Medeiros, M., Menezes, J. W. M. Análise Do Scratch Como Ferramenta De Auxílio Ao Ensino De Programação De Computadores. In: Anais do XL Congresso Brasileiro de Educação em Engenharia. Belém: UFPA, 2012.

[16] Rodrigues, M. C. Como Ensinar Programação?. Informática – Boletim Informativo Ano I no 01, ULBRA. Canoas, RS, Brasil, 2002.

[17] Santos, R. P. e Costa, H. A. X. TBC-AED: Um Software Gráfico para Apresentação de Algoritmos e Estruturas de Dados aos Iniciantes em Computação e Informática. I Congresso de Computação do Sul do Mato Grosso (COMPSULMT' 2005). Rondonópolis, MT, Brasil.

[18] Santos, R. P. e Costa, H. A. X. Análise de Metodologias e Ambientes de Ensino para Algoritmos, Estruturas de Dados e Programação aos iniciantes em Computação e Informática. Journal of Computer Science, v.5 (1), pp. 41-50, 2006.

[19] SCRATCH. About Scratch (Scratch Documentation Site). Disponível em: <http://info.scratch.mit.edu/About_Scratch>. Acesso em: 24 de setembro de 2013.

[20] Selwyn, N. Social Media in Higher Education. The Europa World of Learning. 2012

[21] So, H.J., Choi, H., Lim, W.Y., Xiong, Y. Little experiencewith ICT: Are they really the Net Generation student-teachers?. Computers & Education. Vol. 59 (4), 1234-1245 pp. Dezembro, 2012.

[22] Stephen Cooper, Wanda Dann, and Randy Pausch. Alice: a 3-D tool for introductory programming concepts. Journal of Computing in Sciences Colleges. 15, 5 (April 2000), 107-116.

[23] Stillman, Richard M., Alan R. Peslak. "Modern Data Structures: Experiences with a Flexible Approach to a Data Structures Course." Information Systems Education Journal. Vol 5 (3). 2007.

[24] Tobar, C. M., Rosa, J. L. G., Coelho, J. M. A. e Pannain, R. Uma Arquitetura de Ambiente Colaborativo para o Aprendizado de Programação. XII Simpósio Brasileiro de Informática na Educação (SBIE'2001). Vitória, ES, Brasil.

[25] William Pugh. Skiplists: a probabilistic alternative to balanced trees. Communications of ACM Vol 33, edição 6, páginas 668-676. Junho de 1990.