Aplicação Web para o Ensino de Árvore Binária de Busca

Gedilson Pessoa da Silva, Fernando Luiz de Oliveira, Madianita Bogo Marioti, Heloise Acco Tives Leão , Fabiano Fagundes

Departamento de Computação – Centro Universitário Luterano de Palmas

{gedilsonpessoa,nandoluiz.br,madianitab,heloise.acco,thilfa}@gmail.com

Resumo. *A proposta deste trabalho foi apresentar uma ferramenta online que trabalha os conceitos de árvore binária, bem como elementos de seu funcionamento, oferecendo ao usuário a possibilidade de criar, inserir, excluir, buscar nós, percorrer e ilustrar graficamente os conceitos relacionados ao tema. Para o desenvolvimento da ferramenta foi utilizado o framework Bootstrap para criação da interface, Javascript para implementação dos códigos e, para desenhar os elementos gráficos, foi utilizado o framework svg.js. O ambiente foi desenvolvido tendo como objetivo principal servir como ferramenta auxiliar no estudo e aprendizado de árvore binária permitindo ao estudante criar, de forma dinâmica, nós e executar funções relacionadas ao conceito abordado.*

1. Introdução

As árvores binárias de busca são estruturas de dados não lineares. Para encontrar valores contidos em sua estrutura percorrem-se vários caminhos em diferentes direções. Uma das principais características desta estrutura refere-se ao fato de ser uma estrutura de dados hierárquica, permitindo, assim, organizar e buscar elementos de forma mais rápida se comparada às demais estruturas de dados.

Trata-se de uma estrutura de dados muito utilizada na computação e, segundo Lafore (1999, p. 285), a estrutura combina vantagens do array ordenado, pois permite realizar buscas de forma rápida e da lista encadeada, permitindo assim realizar operações de inserção e exclusão de itens rapidamente. Sua estrutura pode ser utilizada para representar expressões matemáticas e estruturas de pastas, bem como na criação de jogos, na inteligência artificial, em interfaces gráficas e em diversas outras situações. A estrutura é formada basicamente por um conjunto finito de elementos, geralmente representados por círculos e nomeados de nós. O primeiro elemento desse conjunto recebe o nome de raiz e é a partir desse nó que desencadeia toda a estrutura da árvore.

As diversas formas de armazenamentos de dados em estruturas de dados são apresentadas e compreendidas na disciplina Estrutura de Dados. Através desta disciplina é possível estudar, compreender os conceitos comuns e particulares de cada estrutura e implementá-los para melhor assimilação.

Este trabalho apresenta uma ferramenta online que permite ao docente da disciplina Estrutura de Dados apresentar a seus alunos a estrutura de árvore binária de busca de forma dinâmica, lúdica e intuitiva, possibilitando, com isso, visualizar o conceito teórico de forma aplicada. A ferramenta permite ao aluno aprender de forma gradual, obtendo informações teóricas relacionadas ao assunto, como: profundidade, grau da árvore, quantidade de nós e demais informações pertinentes à estrutura em questão.

As próximas seções descrevem alguns dos conceitos de árvores binárias que são apresentados através da ferramenta, seguidos da descrição das tecnologias envolvidas em seu desenvolvimento bem como alguns pontos-chaves de sua implementação. Por fim são apresentadas algumas considerações sobre o desenvolvimento do trabalho e as referências bibliográficas utilizadas.

2. Referencial Teórico

De acordo com Veloso (1986), **árvore** pode ser definida como uma estrutura de dados que se caracteriza pela relação existente entre os dados, denominados nós, ou seja, é uma relação de hierarquia onde um conjunto de dados é hierarquicamente subordinado a outro. Segundo Horowitz (1984), entende-se por árvore um conjunto finito de um ou mais nós de maneira que, obrigatoriamente, um deles seja denominado raiz, de modo que cada nó da árvore é a raiz de uma subárvore. Assim, uma árvore é formada por subárvores que respeitam as mesmas regras em todos os níveis. Em uma definição simples e objetiva, árvore é um conjunto de nós que se relacionam, possuindo uma hierarquia de pai para filho. Pode-se facilmente visualizar, como exemplificam Szwarcfiter e Markenzon (1994), o conceito de árvore em um organograma de uma empresa, que é representado de forma hierarquizada.

Nó é um ponto de conexão que liga as ramificações de uma árvore. **Grau de um nó** (ou grau de saída de um nó) é a quantidade de filhos deste nó, seus descendentes, ou seja, a quantidade de ramificações que saem deste nó. Segundo Lafore (1999, p. 289), "um nó que não tem filhos é chamado de nó folha ou simplesmente folha", ou seja, um nó folha tem grau zero. Um nó de uma árvore é irmão de outro nó se os dois tiverem o mesmo pai, que é o nó ascendente. Na árvore todo nó, exceto o nó raiz, possui apenas um único nó ascendente.

Entende-se por **árvore binária**

> um conjunto finito de elementos que está vazia ou é particionado em três subconjuntos disjuntos. O primeiro subconjunto contém um único elemento, chamado raiz da árvore. Os outros dois subconjuntos são em si mesmos árvores binárias, chamadas subárvores esquerda e direita da árvore original. Uma subárvore esquerda ou direita pode estar vazia. (TENENBAUM; LANGSAM; LANGSAM, 1995, p. 303).

Assim, tem-se árvores binárias como estruturas do tipo árvore, onde o grau de cada nó é menor ou igual a dois (VELOSO, 1986). Como o grau de uma árvore "é fornecido pelo nó que tem maior grau" (LOPES, 1999, p. 21), pode-se afirmar que uma árvore binária é uma árvore com, no máximo, grau 2.

De acordo com Lafore (1999, p. 290), o **nível** de um nó particular se refere a quantas gerações o nó está da raiz. Assim, o nível de um nó é definido através da sua distância até o nó raiz. Considerando o nível da raiz igual a zero, o nível de um dado nó será definido através do nível do seu nó pai mais 1.

A **altura da árvore** é definida através da distância do nó raiz até o filho, descendente, mais distante, assim, pode-se entender como a quantidade de interações necessárias até se chegar ao nó mais distante da raiz.

Segundo Tenenbaum; Langsam; Langsam (1995, p. 306), "se todo nó que não é folha numa árvore binária tiver sub-árvores esquerda e direita não-vazias, a árvore será considerada uma **árvore estritamente binária**". Ascencio; Araújo (2010, p. 284) apresentam uma **árvore binária completa** como a "árvore em que todos os nós com menos de dois filhos ficam no último e no penúltimo nível". Uma árvore pode ser considerada estritamente binária e completa quando for uma **árvore cheia**, ou seja, todos os nós do penúltimo nível, tiverem 2 filhos.

São três os **percursos** tradicionais em uma árvore binária: **pré-ordem**, **in-ordem** e **pós-ordem**.

> Em cada um desses métodos, não é preciso fazer nada para percorrer uma árvore binária vazia. Todos os métodos são definidos recursivamente, de modo que percorrer uma árvore binária envolve visitar a raiz e percorrer suas subárvores esquerda e direita. A única diferença entre os métodos é a ordem na qual essas três operações são efetuadas Tenenbaum; Langsam; Langsam (1995, p. 323)

ISSN: 2447-0767

No percurso pré-ordem visita-se a raiz, percorre-se a subárvore esquerda em pré-ordem e em seguida a subárvore direita em pré-ordem. No percurso in-ordem percorre-se a subárvore esquerda em in-ordem, visita-se a raiz e percorre-se a subárvore direita em in-ordem. Por fim, no percurso pós-ordem percorre-se a subárvore esquerda em pós-ordem, em seguida percorre-se a subárvore direita em pós-ordem e visita-se a raiz.

Uma **árvore binária de busca** é uma árvore binária na qual para todo nó, os valores dos nós de sua subárvore à esquerda são menores que a informação ali armazenada e os valores dos nós de subárvore a direita são maiores. Assim, para localizar dentro de uma árvore binária de busca o nó de menor valor, partindo do princípio que todo nó do lado esquerdo é menor que seu nó pai, realiza-se o percurso visitando todos os nós das subárvores esquerdas subsequentes até chegar ao nó folha, ou seja, último nó esquerdo. A mesma ideia é adotada para localizar o nó de maior valor, porém visitando todos os nós das subárvores direita subsequentes até chegar ao nó folha.

Para inserir um valor na árvore o primeiro passo é verificar se o valor já existe em sua estrutura, ou seja, para toda nova inserção é realizada uma busca pelo valor a ser inserido e, caso o valor não exista, é verificado se o valor é menor ou maior que valor do último nó verificado e insere-o na subárvore da esquerda ou direta, respeitando a organização dos elementos na árvore.

Estes e outros conceitos foram traduzidos em ações na aplicação web que permite ao usuário inserir, remover, buscar valor e percorrer uma árvore binária de busca além de obter informações sobre cada um de seus nós e sobre a própria árvore, como altura, nível, quantidade de filhos, dentre outros. Os passos para implementação desta ferramenta são descritos a seguir.

3. Materiais e métodos

Para compreensão e elaboração do referencial teórico foram realizadas pesquisas em livros impressos, livros digitais encontrados na Internet, artigos, materiais didáticos e vídeos que abordam o assunto relacionado à árvore binária, buscando, assim, o entendimento aprofundado sobre a estrutura. Além do referencial teórico sobre árvore binária, também foram realizadas pesquisas que contemplam as seguintes tecnologias: HTML5 – *HyperText Markup Language*; Javascript; Bootstrap (*framework front-end* utilizado para criar interfaces web no padrão W3C); *framework* SVG (svg.js): (para manipular o componente SVG do HTML tornando possível interagir e animar os nós da árvore na área de desenho) e; CSS – *Cascading Style Sheets* (utilizada para estilizar os elementos do HTML).

As tecnologias mencionadas foram utilizadas para desenvolver o ambiente que permitiu desenhar árvores binárias. A interface da ferramenta foi desenvolvida utilizando o *framework* Bootstrap. A utilização do *framework* permitiu criar uma interface prática, otimizada e responsiva, ou seja, permitiu ao aplicativo ter sua interface ajustada automaticamente a diversos formatos e tamanho de monitores.

Todo o ambiente, controles, módulo de desenho e demais recursos da ferramenta executam em *front-end*, ou seja, na máquina do usuário. Para implementar a ferramenta, foi utilizada a IDE JetBrains PyCharm 2018.1.4. A IDE foi escolhida por ser compatível com as tecnologias apresentadas, além de apresentar uma interface amigável e conter recursos adicionais que facilitarão a implementação.

Os processos para desenvolvimento deste projeto foram sistematizados em: preparação do ambiente de desenvolvimento, módulo de desenho de árvore binária de busca e módulo de informação. A ferramenta permite desenhar somente o tipo de árvore binária de busca.

Assim, inicialmente foi preparado o ambiente de desenvolvimento. Nesta primeira etapa, as ferramentas necessárias para desenvolvimento da ferramenta foram devidamente instaladas e configuradas. A segunda etapa no projeto foi desenvolver o módulo que permitisse desenhar árvore binária de busca, ou seja, a interface da ferramenta. Na tela da

ferramenta estão disponíveis controles que permitem criar os nós e as arestas que compõem a árvore binária de busca.

A terceira etapa foi o desenvolvimento do módulo de informação. Nesta etapa foram implementadas as classes, objetos e funções necessárias para criar e analisar a estrutura de dados desenhada na tela. Além dos modelos citados anteriormente, foi implementada em Javascript a estrutura da árvore binária de busca que reflete o desenho criado na área de desenho. Todas as alterações realizadas sobre o desenho criado na área de desenho serão transferidas para a árvore binária de busca implementada em Javascript. Esta estrutura de dados implementada em Javascript é utilizada para obter informações relacionadas à árvore binária de busca.

Durante todas as etapas que contemplam o módulo de desenho e módulo de informação, foram realizados testes de funcionalidade buscando corrigir eventuais erros existentes na codificação. Com isso, foi possível analisar se as informações retornadas pelo sistema, através do módulo de informação, estavam corretas.

4. Resultados

A figura 1 a seguir apresenta a estrutura da aplicação para melhor entendimento do funcionamento dos módulos.

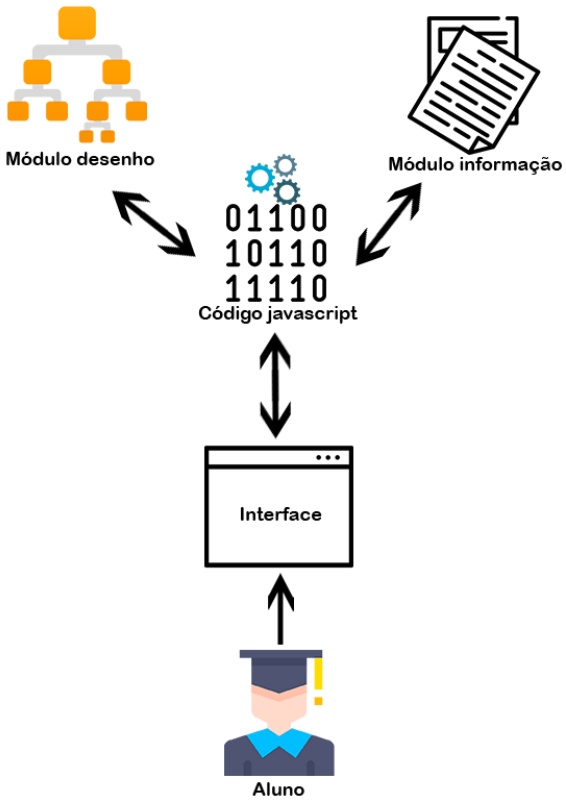

Figura 1. Estrutura da aplicação

Como se pode observar na Figura 1, o aluno/usário tem acesso a interface da aplicação por meio de um navegador web. O módulo de desenho disponibiliza ferramentas que permitem ao uusário criar a estrutura da árvore gráfica com base na estrutura criada no código Javascript. O módulo informação realiza processamentos sobre a estrutura da árvore em código Javascript, extraindo as informações necessárias e apresentado estas informações ao usuário.

A aplicação foi desenvolvida para ser utilizada através de navegadores web. As tecnologias escolhidas têm como objetivo simplificar o uso da ferramenta, pois são nativas

ISSN: 2447-0767

nos principais navegadores existentes. A tela principal da aplicação está dividida em três partes:

- menu principal: disponibiliza recursos para realizar operações, como: busca em pré-ordem, em-ordem e pós-ordem, destacar nós folhas e destacar o menor e o maior nó da árvore. Além desses recursos o usuário terá acesso ao menu que permitirá ter acesso a todas as informações gerais da árvore desenhada;
- barra de menu: disponibiliza os recursos de inserção, exclusão e localização de um nó na estrutura da árvore, além de recursos para visualizar os níveis da árvore graficamente, preencher a árvore com valores aleatórios e limpar a área de desenho;
- área de desenho: realiza a apresentação do desenho que representará a estrutura da árvore criada pelo usuário.

A área de desenho foi criada utilizando o elemento gráfico SVG, e para criar os elementos círculos que representam os nós, as arestas que são as ligações entre os nós e seus rótulos foi utilizado o *framework* SVG. Com o *framework* svg.js foi possível criar os elementos geométricos e todas as animações realizadas sobre a estrutura da árvore.

Na Figura 2 é possível observar a área de desenho com a estrutura de uma árvore binária de busca. Toda a interação do usuário com o aplicativo é realizada através do teclado e mouse. Para realizar as três operações básicas, inserir, excluir e localizar um determinado nó, o usuário deve informar o valor no campo nó, selecionar uma das opções: “Novo nó”, “Excluir nó” ou “Localizar nó” e por fim, clicar no botão “Executar ação”.

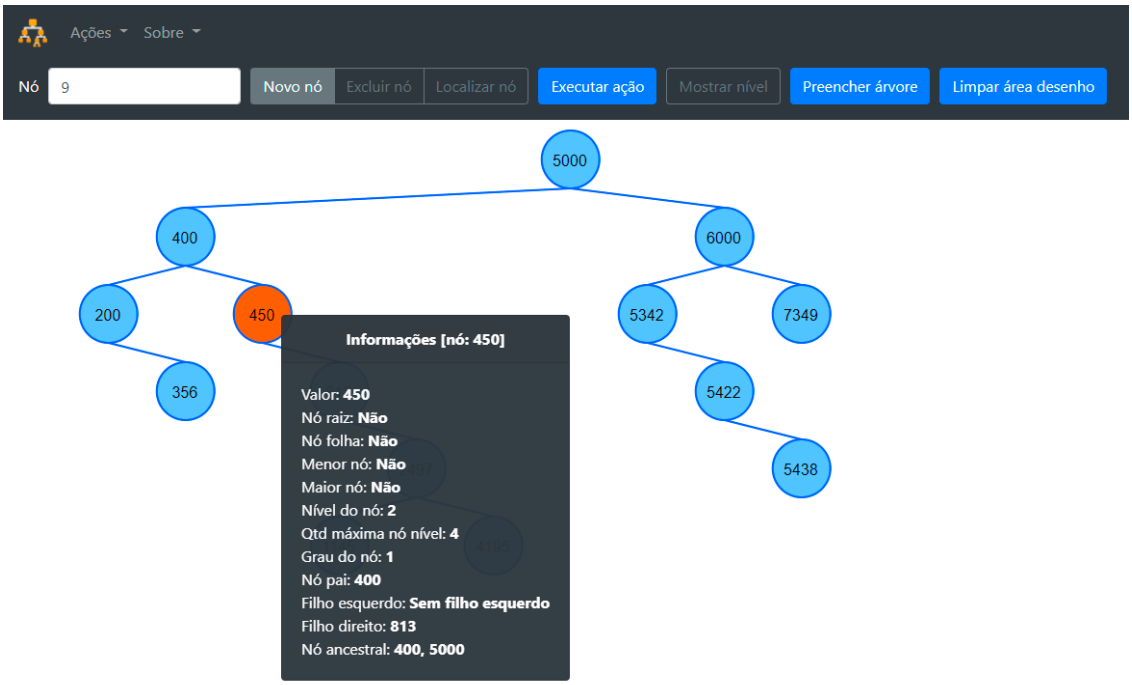

Figura 2. Área de desenho

Na figura 2 é possível ainda observar uma árvore desenhada. O nó da árvore é representado pela forma geométrica círculo com um rótulo contendo seu respectivo valor. As arestas são linhas comuns e são utilizadas para fazerem as ligações entre o nó pai e seus nós filhos. Ao pressionar o botão esquerdo no mouse sobre um determinado nó será apresentado ao usuário uma caixa de texto contendo informações relacionadas ao nó selecionado. Assim, com o clique com o botão esquerdo do mouse sobre o nó é apresentada a caixa de texto contendo todas as informações do nó respectivo: valor, nó raiz, nó folha, menor nó, maior nó, dentre outras informações pertinentes. As informações apresentadas em cada descrição do menu suspenso são extraídas da árvore criada em código Javascript.

Figura 3. Menu suspenso

O usuário terá acesso também à um menu suspenso, clicando com o botão direito do mouse sobre o nó na área de desenho. A Figura 3 apresenta o menu acionado que disponibiliza opções que permitem interagir com o nó. Este menu contém as opções informação do nó (que permite obter informações do nó semelhantes ao mostrado na Figura 2), excluir nós, obter nós ancestrais, descendentes, descendentes esquerdos e direitos, além de voltar às cores padrões iniciais da árvore, permitindo assim a interação com o nó.

A implementação da estrutura da árvore se dá na linguagem JavaScript e utiliza-se da característica recursiva da linguagem, como pode-se ver na Figura 4 que apresenta o código da função `recuperarNo`.

```
94
95      /*recupera o nó na estrutura da árvore binária ****/
96      this.recuperarNo = function (valor) {
97          var no = this.raiz;
98
99          var funcaoLocalizarNo = function (no) {
100             if (!no) return false;
101
102             if (valor === no.valor) {
103                 return no;
104             } else if (valor > no.valor) {
105                 return funcaoLocalizarNo(no.filho_direito);
106             } else if (valor < no.valor) {
107                 return funcaoLocalizarNo(no.filho_esquerdo);
108             }
109         };
110
111         //chama a função recursiva
112         return funcaoLocalizarNo(no);
113     };
```

Figura 4. Função `recuperarNo`

A dificuldade encontrada foi posicionar toda a estrutura da árvore na área de desenho mantendo-a exposta de forma completa, centralizada e visível por inteiro mesmo após sucessivas ações de inserção e remoção de elementos. A figura 5, a seguir, apresenta trecho da função `criarNoGraficamente` que, nas linhas 2 a 5, verifica se o nível do nó é igual a zero. Sendo igual a zero trata-se do nó raiz, então sua posição x será 1/4 da largura da tela e y será `posicaoInicialYNo`. Definida a posição x e y do nó, na linha 8 é chamada a função `criarAnimacaoMovimentoPadrao` para posicionar o nó no local correto dentro da área de desenho.

ISSN: 2447-0767

```
1  //verifica o nível do nó. Caso seja o primeiro nível então é o nó raiz
2  if (bst.nivelNo(no.valor) == 0) {
3      var widthArea = document.getElementById('areadesenho').clientWidth;
4      no.posX = widthArea / 4;
5      no.posY = this.posicaoInicialYNo;
6
7      //cria a animação de movimento do nó
8      this.criarAnimacaoMovimentacaoPadrao(no, no.posX, no.posY, true);
9  } else if (no.no_pai && no.no_pai.filho_esquerdo && no.no_pai.filho_esquerdo.valor == no.valor) {
10     no.posX = no.no_pai.posX - this.distanciaPadraoHorizontal;
11     no.posY = no.no_pai.posY + this.distanciaPadraoVertical;
12
13     //indentifica qual o caminho que o novo nó deverá pecorrer até
14     //chegar na posição que ficará
15     var caminho = this.getCaminhoNo(no.valor);
16     //cria a animação de movimento do nó
17     //this.criarAnimacaoMovimentacaoCaminhoNo(no, caminho);
18     this.criarAnimacaoMovimentacaoPadrao(no, no.posX, no.posY, false);
19
20     this.verificaPosicaoNoArvoreDireitaEsquerda(no);
21 } else if (no.no_pai && no.no_pai.filho_direito && no.no_pai.filho_direito.valor == no.valor) {
22     no.posX = no.no_pai.posX + this.distanciaPadraoHorizontal;
23     no.posY = no.no_pai.posY + this.distanciaPadraoVertical;
24
25     //indentifica qual o caminho que o novo nó deverá pecorrer até
26     //chegar na posição que ficará
27     var caminho = this.getCaminhoNo(no.valor);
28     //cria a animação de movimento do nó
29     //this.criarAnimacaoMovimentacaoCaminhoNo(no, caminho);
30     this.criarAnimacaoMovimentacaoPadrao(no, no.posX, no.posY, false);
31
32     this.verificaPosicaoNoArvoreDireitaEsquerda(no);
33 }
34
35 //verifica se o menor nó está dentro da área de desenho
36 //em outras palavras, verifica se o X no menor nó e maior que 0;
37 this.verificaLimiteEsquerda();
```

Figura 5. Trecho da função `criarNoGraficamente`

A Figura 5 apresenta, ainda, que, para casos que o nó não seja a raiz, é verificado se o elemento deve ser colocado do lado esquerdo ou direito do seu nó pai. Se, por exemplo, o nó for adicionado ao lado esquerdo, para que seja feito o posicionamento correto é atribuída a posição x do nó pai menos a distância padrão horizontal da variável `distanciaPadraoHorizontal` para a posição x do nó filho, como mostra a linha 10, e posição y do nó pai mais a distância padrão vertical da variável `distanciaPadraoVertical` para a posição y do nó filho, linha 11. Por fim linha 37 é chamada a função `verificaLimiteEsquerdo` para verificar se existe algum elemento fora da margem esquerda da área de desenho.

5. Considerações Finais

Neste trabalho foi proposta e desenvolvida uma aplicação web que permite desenhar árvores binárias de busca e obter informações sobre o desenho criado. O desenvolvimento da aplicação foi motivado pela ausência de uma ferramenta local que pudesse auxiliar o professor da disciplina Estruturas de Dados no ensino de árvore binária de busca e, principalmente, pela dificuldade que geralmente os alunos da disciplina possuem em entender e visualizar o conceito desta estrutura. Com isso o professor terá a oportunidade de dar ênfase ao ensino de maneira mais dinâmica, intuitiva e lúdica. O fato de ser uma ferramenta desenvolvida na própria instituição permitirá que adequações sejam feitas para que se adeque às necessidades da disciplina bem como ocorram inserções de novas funcionalidades.

No desenvolvimento da aplicação muita atenção foi dada à criação da rotina de

posicionamento dos nós na área de desenho especialmente na forma como proceder para organizar os nós da árvore na área de desenho de modo que não ocorressem sobreposições de nós e cruzamentos de arestas. Outro ponto que demandou muita atenção foi a exclusão de nó com dois filhos, isto devido à dificuldade de reorganizar a árvore de modo a permanecer uma árvore binária de busca, junto à dificuldade do reposicionamento dos nós e arestas na área de desenho. Também houve especial cuidado com a apresentação animada dos percursos pré-ordem, em-ordem e pós-ordem, pois para a animação do percurso acontecer de forma devida foi necessário primeiro mapear todo o caminho, ida e volta, que o círculo gráfico deveria percorrer na árvore e o momento que deveria registrar a marcação do nó de acordo com o tipo de busca solicitado pelo usuário.

Para trabalhos futuros, novas funcionalidades podem ser adicionadas à aplicação para assim melhorar a experiência do professor e aluno, como por exemplo: sistema de login adicionado à possibilidade de salvar a árvore criada pelo aluno no banco de dados e recursos para recarregá-la quando necessário; aperfeiçoar a função responsável por organizar os nós na área de desenho; criar novas animações melhorando a experiência do professor e do aluno na aplicação.

Referências Bibliográficas

Horowitz, Ellis; Sahni, Sartaj. *Fundamentos de estruturas de dados*. Rio de Janeiro: Campus, 1984.

Lafore, Robert. *Aprenda em 24 horas estrutura de dados e algoritmos*. Rio de Janeiro: Campus, 1999.

Lopes, Arthur Vargas. *Estruturas de dados*: Para a Construção de Software. – Canoas: Ed. ULBRA, 1999.

Szwarcfiter, Jayme Luiz; Markenzon, Lilian. *Estruturas de dados e seus algoritmos*. Rio de Janeiro: LTC, 1994.

Tenenbaum, Aaron Ai; Langsam , Yedidyah; Augenstein, Moshe J. *Estruturas de Dados Usando C*. São Paulo: McGraw-Hill, In, 1995.

Veloso, Paulo et al. *Estruturas de dados*. Rio de Janeiro: Campus, 1986.

ISSN: 2447-0767