Estruturas de Dados 2023/1 - Trabalho Final

Análise de Sentimento

Julianne Emanuelle Martins Carrazzoni (00264558)
Danilo de Brito Cardoso Oliveira (00338135)

Este trabalho possui a finalidade de comparar aplicações com diferentes árvores (principalmente árvores balanceadas para pelo menos uma das aplicações) que recebem um arquivo com sentenças opinativas e um léxico de opinião e atribui um escore de polaridade às sentenças.

Nosso objetivo inicial, foi fazer a comparação entre AVL vs. ABP.

Porém tivemos alguns problemas, apenas conseguimos implementar para a AVL e não conseguimos fazer a comparação.

Abaixo segue nossos principais problemas:

Fizemos este caso de teste, em menor escala de dados, e funciona como o esperado:

entrada:

```
voce eh bonito e fofo , mas eh chato .
voce eh legal, mas eh timido .
```

léxico:

```
bonito 2.0
legal 1.0
fofo 3.0
chato -1.0
feio -2.0
timido -1.0
```

saída:

```
4,00 voce eh bonito e fofo , mas eh chato .
0,00 voce eh legal, mas eh timido .
```

Prints na linha de comando para teste:

```
 palavra: bonito
escore: 2,00

 palavra: legal
escore: 1,00

 palavra: fofo
escore: 3,00

 palavra: chato
escore: -1,00

 palavra: feio
escore: -2,00
```

```
palavra: voce, score: 0,00
palavra: eh, score: 0,00
palavra: bonito, score: 2,00
palavra: e, score: 0,00
palavra: fofo, score: 3,00
palavra: ,, score: 0,00
palavra: mas, score: 0,00
palavra: eh, score: 0,00
palavra: chato, score: -1,00
palavra: ., score: 0,00
palavra: voce, score: 0,00
palavra: eh, score: 0,00
palavra: legal,, score: 0,00
palavra: mas, score: 0,00
palavra: eh, score: 0,00
palavra: timido, score: 0,00
palavra: ., score: 0,00

Arquivo .\zpp.txt gerado com sucesso.
Tempo: 230,00000 ms
```

Porém, a aplicação não está obtendo as casas decimais após a vírgula do seu escore.

Tentando seguir esta lógica para obter a AVL com as palavras e os escores vindo do arquivo léxico:

```
pNodoA *Arvore_AVL;

Arvore_AVL = cria_arvore();

int *ok;
int nivel = 0;
float escore;

while(!(feof(lexico)))
{

    if(fgets(linha,sizeof(linha),lexico) != NULL)
    {
        char lex[100];

        // Usar sscanf para ler a palavra e o valor diretamente da linha
        if (sscanf(linha, "%s %f", lex, &escore) == 2)
        {
            Arvore_AVL = InsereAVL(Arvore_AVL, lex, escore, &ok);
            //Imprimir a palavra e o escore
            //printf("\n palavra: %s\nescore: %.2f\n", lex, escore);
        }
    }

}
```

Tentamos várias alternativas mas nenhuma obteve o escore com suas casas decimais depois da vírgula.

Árvore Utilizada - AVL

Para a inserção na AVL foram feitas as seguinte modificações para inserir o léxico na AVL:

```
// Função para inserir o lexico na avl
pNodoA* InsereAVL(pNodoA* a, char* palavra, float escore, int* ok) {

    //enquanto a nao for null
    if (a == NULL) {

        a = (pNodoA*)malloc(sizeof(pNodoA)); //aloca
        strcpy(a->palavra, palavra); //pega palavra passada nada main
        a->escore = escore;
        a->esq = NULL;
        a->dir = NULL;
        a->FB = 0;
        *ok = 1;
        a->altura = 1;

        //stricmp compara duas palavras, e ve qual eh maior, sem case sentive
    }else if (stricmp(a->palavra, palavra) > 0)     //nova palavra eh menor q a palavra (raiz), entao aloca na esquerda
    {
        a->esq = InsereAVL(a->esq, palavra, escore, ok);

        if(*ok)
        {
            switch (a->FB)
            {
            case -1:
                a->FB = 0;



} else {

    a->dir = InsereAVL(a->dir, palavra, escore, ok);

    if(*ok)
    {
        switch(a->FB)
        {
        case 1:
            a->FB = 0;
            *ok = 0;
            break;
```

As rotações seguem as mesmas da inserção normal em avl. As modificações foram feitas nos critérios de inserção na esquerda e na direita.

Para consulta na AVL foi modificada da seguinte forma:

```
// Função para buscar o escore de polaridade de uma palavra na AVL
float consultaAVL (pNodoA* a, char* chave)
{
    if( a == NULL)
    {
        return 0.00;

    }else if(stricmp(a->palavra, chave) == 0)
    {
        return a->escore;
    }else if (stricmp(a->palavra, chave) > 0){
        consultaAVL(a->esq, chave);
    }else{
        consultaAVL(a->dir, chave);
    }

}
```

Esta função é importante pois ela é responsável por obter o escore da palavra na sentença.

```
struct TNodoA
{
    char palavra[50];
    float escore;
    int FB;
    struct TNodoA *esq;
    struct TNodoA *dir;
    int altura;
};
typedef struct TNodoA pNodoA;
```

Estrutura usada para a avl.