Árvore Rubro-negra

SERGIO SIVONEI DE SANT'ANA FILHO GRR20242337
EDUARDO KALUF GRR20241770

April 28, 2025

1 Apresentação

O objetivo deste trablho é a implementação da árvore Red-Black, também conhecida como Rubro-Negra, utilizando a linguagem C, com base no livro "Algoritmos: Teoria e Prática" do H. Cormen Esta é uma árvore binária que utiliza um campo de cor a fim de manter a árvore o mais balanceada possível, fazendo com que as operações feitas nela sejam executadas com tempo $\mathcal{O}(\log n)$

2 Implementação

A implementação foi feita com base nas notas de aula e no próprio livro do Cormen, com modificações a fim de cumprir as especificações do trabalho. A construção da árvore foi feita do zero, sem utilizar funções de uma BST já implementada, focando em deixar o trabalho mais sucinto e didático.

A implementação foi feita com base nas notas de aula e no próprio livro do Cormen, com modificações a fim de cumprir as especificações do trabalho. A construção da árvore foi feita do zero, sem utilizar funções de uma BST já implementada, focando em deixar o trabalho mais sucinto e didático.

Optamos por usar um sentinela que guarda um ponteiro para o nodo raiz e para o nodo NIL. O nodo especial NIL é utilizado para se referir a NULL, sendo então um nó preto (na Red-Black, ponteiros para NULL são tratados como nós pretos), o qual todos os nodos folhas da árvore apontam.

Sobre as funções da Rubro-Negra, todas foram implementadas segundo o livro do H. Cormen e notas de aula. Houveram modificações para atender aos critérios do trabalho, a exemplo função "rb delete", a qual ao se remover a raiz, busca o antecessor (nodo com a chave imediatamente menor que a chave da raiz) para se tornar nova raiz da árvore. No header "rb.h" constam informações mais particulares sobre cada função.

O algoritmo de busca utilizado no código é a busca binária recursiva, pois a Rubro-Negra trata-se de uma árvore binária específica.

Os comandos aceitos são para inserção e remoção de um nodo, representados pelos caracteres "i" e "r" respectivamente, seguindo o formato "*instrução valor".

Como presente nas especificações, a saída é espressada em ordem(it) na saída padrão (stdout), ordenando por valor, nível e cor, respectivamente, e em caso de valores iguais, segue ordem crescente de nível, pois chaves idênticas são inseridas à direita, portanto impressas depois seguindo a ordem (esquerda, raiz, direita).

3 Diretórios e Arquivos

- `rb.h`
  Arquivo header que contém a documentação de cada função utilizada.
- `rb.c`
  Arquivo de implementação das funções da árvore Red-Black.
- `main.c`
  Arquivo responsável por receber os comandos passados pelo terminal e chamar as funções correspondentes, gerenciando o fluxo de dados do trabalho.
- `Makefile`
  Arquivo simples utilizado para compilar e fazer a linkagem do trabalho.
  - `make` → gera o executável `myrb`
  - `make clean` → remove os arquivos gerados pela compilação
- `testes` e `testes.sh`
  `testes.sh` é um script simples que verifica, através dos testes e soluções armazenados no diretório `testes`, se a árvore implementada está apresentando o comportamento esperado.
  O script compila automaticamente o projeto e o executa utilizando o comando `diff`. Sendo assim, caso nada seja impresso no terminal, todos os testes foram bem-sucedidos.

4 Conclusão

Com todo o processo foi possível verificar a maneira como uma arvore red black pode ser implementada, ponderando sua eficiencia e praticidade de uso