UNIVERSIDADE FEDERAL DO PARÁ
INSTITUTO DE CIÊNCIAS EXATAS E NATURAIS
FACULDADE DE CIÊNCIA DA COMPUTAÇÃO

questão 14 da lista 5 de matemática discreta

BELÉM/PA
2023

CHRISTIAN AMARILDO AMORIM MORAIS

questão 14 da lista 5 de matemática discreta

Análise assintótica a busca binária iterativa e recursiva

Trabalho apresentado como requisito parcial para a avaliação 3 de matemática discreta, pela Universidade Federal do Pará, ministrada pelo professor Nelson Cruz.

BELÉM/PA
2023

1 INTRODUÇÃO

O objetivo deste trabalho é explorar e analisar a eficiência dos algoritmos de busca binária, tanto na implementação iterativa quanto recursiva. A busca binária é um algoritmo fundamental em ciência da computação, utilizado para encontrar a posição de um elemento em um conjunto ordenado de dados. A análise de desempenho dessas implementações é crucial para compreender como o tempo de execução varia em relação ao tamanho do conjunto de dados.

O trabalho se debruça em solucionar a seguinte questão presente na lista 5 da matéria de matemática discreta:

“14. Escreva um algoritmo (em pseudocódigo) que realize busca binária de forma iterativa e o implemente numa linguagem de programação a sua escolha. Construa um gráfico mostrando a relação do valor de entrada x tempo de execução do algoritmo. Considerando uma análise assintótica em pior caso, explique se o desempenho do algoritmo é superior, inferior ou igual ao do algoritmo que implementa busca binária de forma recursiva. “

2 DESENVOLVIMENTO

1. Pseudocódigo para Busca Binária Iterativa:

“

```
Função buscaBinariaIterativa(arr, x):
    início = 0
    fim = tamanho(arr) - 1

    enquanto início <= fim:
        meio = (início + fim) / 2

        se arr[meio] é igual a x:
            retorne meio
        senão, se arr[meio] < x:
            início = meio + 1
        senão:
            fim = meio - 1

    retorne -1
```

“

2. Pseudocódigo para Busca Binária recursiva:

```
Função busca_binaria_recursiva(array, valor, inicio, fim):
    Se inicio > fim:
        Retorne -1  # Elemento não encontrado
    Senão:
        Meio = (inicio + fim) / 2
        Se array[meio] == valor:
            Retorne meio  # Elemento encontrado
        Se array[meio] > valor:
            Retorne busca_binaria_recursiva(array, valor, inicio, meio - 1)
        Senão:
            Retorne busca_binaria_recursiva(array, valor, meio + 1, fim)  # I
```

3. Análise de Desempenho:**:**

A busca binária, tanto iterativa quanto recursiva, possui uma complexidade de tempo de O(log n) no pior caso, onde n é o tamanho do array. A busca binária é mais eficiente do que uma busca linear (O(n)) para arrays grandes.

Ambos os algoritmos, iterativo e recursivo, têm a mesma complexidade assintótica. A escolha entre eles muitas vezes depende das preferências do programador ou da situação específica do problema. A implementação iterativa pode ser mais eficiente em termos de uso de memória, pois evita o acúmulo de chamadas recursivas. Contudo, a busca binária recursiva é frequentemente considerada mais elegante e fácil de entender.

2. Gráfico de Desempenho e implementação em python:

```python
import timeit
import matplotlib.pyplot as plt

def busca_binaria_iterativa(A, x):
    inicio = 0
    fim = len(A) - 1
    while inicio <= fim:
        meio = (inicio + fim) // 2
        if A[meio] == x:
            return meio
        elif A[meio] < x:
            inicio = meio + 1
        else:
            fim = meio - 1
    return -1

def busca_binaria_recursiva(A, x, inicio, fim):
    if inicio > fim:
        return -1
    meio = (inicio + fim) // 2
    if A[meio] == x:
        return meio
    elif A[meio] < x:
        return busca_binaria_recursiva(A, x, meio + 1, fim)
    else:
        return busca_binaria_recursiva(A, x, inicio, meio - 1)

tempos_iterativa = []
tempos_recursiva = []
tamanhos_entrada = [10, 50, 100, 500, 1000]

for tamanho in tamanhos_entrada:
    lista_ordenada = list(range(1, tamanho + 1))
    elemento_a_buscar = len(lista_ordenada) - 1

    # Busca binária iterativa
    tempo_iterativo = timeit.timeit(
        lambda: busca_binaria_iterativa(lista_ordenada, elemento_a_buscar),
        number=10000
    )
    tempos_iterativa.append(tempo_iterativo)

    # Busca binária recursiva
    tempo_recursivo = timeit.timeit(
        lambda: busca_binaria_recursiva(lista_ordenada, elemento_a_buscar, 0, len(lista_ordenada) - 1),
        number=10000
    )
    tempos_recursiva.append(tempo_recursivo)

plt.plot(tamanhos_entrada, tempos_iterativa, label='Iterativa')
plt.plot(tamanhos_entrada, tempos_recursiva, label='Recursiva')
plt.title('Busca binária iterativa vs. recursiva')
plt.ylabel('Tempo médio de execução (s)')
plt.xlabel('Tamanho da entrada (n)')
plt.legend()
plt.show()
```

imagem do codigo

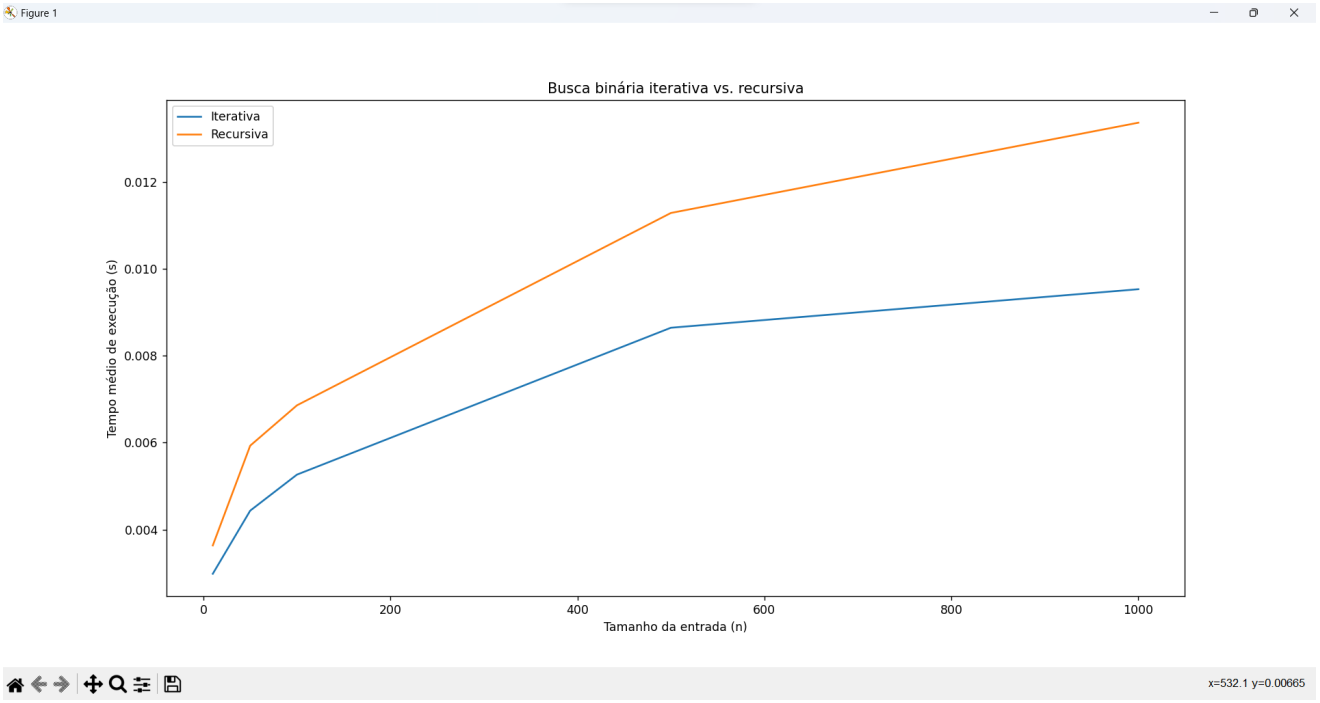

imagem do gráfico

3 CONCLUSÃO

Os resultados do experimento demonstram que, em geral, a versão iterativa da busca binária apresenta um desempenho ligeiramente superior à versão recursiva. Isso pode ser atribuído às chamadas funções adicionais e à sobrecarga associada à implementação recursiva. Para conjuntos de dados maiores, a diferença de desempenho torna-se mais evidente, favorecendo a abordagem iterativa.