**UNIVERSIDADE DO EXTREMO SUL CATARINENSE - UNESC**

CURSO DE CIÊNCIA DA COMPUTAÇÃO

AUGUSTO SAVI

DIVIDIR PARA CONQUISTAR

CRICIÚMA, MAIO 2019

**SUMÁRIO**

[1 INTRODUÇÃO](#_toc247)

[2 OBJETIVO](#_toc252)

[3 CONCEITO](#_toc268)

[4 EXPLICAÇÃO SOBRE O MÉTODO](#_toc280)

5 [EXEMPLO PRATICO](#_toc280)

[REFERÊNCIAS](#_toc309)

# 1 INTRODUÇÃO

O trabalho presente tem como objetivo explicar o método dividir para conquistar,aplicado a algoritmos. Apresentando os objetivos, conceitos, aplicações e exemplos práticos.

# 2 OBJETIVO

Esse método tem como objetivo ter uma maior eficiência em solucionar

grandes problemas,usando menos tempo para solucionar o problema e acessa a esses dados.

# 3 CONCEITO

Esta técnica consiste em dividir um grande problema em problemas menores até que o problema possa ser resolvido, com isso a solução do grande problema e dada através da combinação da solução dos problemas menores.

Problemas que utilizam esta técnica podem ter ganhos em maquinas com múltiplos processadores, pois cada um dos problemas menores pode ser solucionado por um thread.

# 4 EXPLICAÇÃO DO MÉTODO

O método dividir para conquistar soluciona o problema através de três etapas:

1- **Divisão:** o problema maior é dividido em problemas menores e os problemas menores obtidos são novamente divididos sucessivamente de maneira recursiva.

2- **Conquista**: o resultado do problema é calculado quando o problema é pequeno o suficiente.

3-**Combinação**: o resultado dos problemas menores são combinados até que seja obtida a solução do problema maior.

# 5 EXEMPLO PRATICO

Os dois algoritmos mais conhecidos que utilizam esse método são o Merge sort e o Quicksort

MergeSort código em Python:

**def** mergeSort(lista):

**if** len(lista) > 1:

meio = len(lista)//2

*#também é valido: meio = int(len(lista)/2)*

listaDaEsquerda = lista[:meio]

listaDaDireita = lista[meio:]

mergeSort(listaDaEsquerda)

mergeSort(listaDaDireita)

i = 0

j = 0

k = 0

**while** i < len(listaDaEsquerda) **and** j < len(listaDaDireita):

**if** listaDaEsquerda[i] < listaDaDireita[j]:

lista[k]=listaDaEsquerda[i]

i += 1

**else**:

lista[k]=listaDaDireita[j]

j += 1

k += 1

**while** i < len(listaDaEsquerda):

lista[k]=listaDaEsquerda[i]

i += 1

k += 1

**while** j < len(listaDaDireita):

lista[k]=listaDaDireita[j]

j += 1

k += 1

Porem como o algoritmo *Merge Sort* usa a recursividade, há um alto consumo de memória e tempo de execução, tornando esta técnica não muito eficiente em alguns problemas.

Quicksort código em Python:

**import** **random**

2

3 **class** **Quick**(object):

4 **def** particao(self, a, ini, fim):

5 pivo = a[fim-1]

6 start = ini

7 end = ini

8 **for** i **in** range(ini,fim):

9 **if** a[i] > pivo:

10 end += 1

11 **else**:

12 end += 1

13 start += 1

14 aux = a[start-1]

15 a[start-1] = a[i]

16 a[i] = aux

17 **return** start-1

18

19 **def** quickSort(self, a, ini, fim):

20 **if** ini < fim:

21 pp = self.randparticao(a, ini, fim)

22 self.quickSort(a, ini, pp)

23 self.quickSort(a, pp+1,fim)

24 **return** a

25

26 **def** randparticao(self,a,ini,fim):

27 rand = random.randrange(ini,fim)

28 aux = a[fim-1]

29 a[fim-1] = a[rand]

30 a[rand] = aux

31 **return** self.particao(a,ini,fim)

32

33 a = [8,5,12,55,3,7,82,44,35,25,41,29,17]

34 **print** (a)

35 q = Quick()

36 **print** (q.quickSort(a,0,len(a)))

Uma desvantagem do Quicksort é que ele não pode ser facilmente adaptado para operar em listas encadeadas ou em listas grandes em Network-Attached-Storage

# REFERÊNCIAS

QUICKSORT. In: WIKIPÉDIA, a enciclopédia livre. Flórida: Wikimedia Foundation, 2019. Disponívelem:<https://pt.wikipedia.org/w/index.php?title=Quicksort&oldid=54954189>. Acesso em: 26 abr. 2019.

MERGE SORT. In: WIKIPÉDIA, a enciclopédia livre. Flórida: Wikimedia Foundation, 2019.Disponívelem:<https://pt.wikipedia.org/w/index.php?title=Merge\_sort&oldid=54573364>. Acesso em: 23 mar. 2019.

DIVISÃO E CONQUISTA. In: WIKIPÉDIA, a enciclopédia livre. Flórida: Wikimedia Foundation, 2018. Disponível em: <https://pt.wikipedia.org/w/index.php?title=Divis%C3%A3o\_e\_conquista&oldid=53459631>. Acesso em: 27 out. 2018.