

# Prova 2

Nome: Jhoanat Helersen , Aluno de Sonda

Matrícula: 20200000880

\* CAPÍTULO 3: 2, 4, 6, 8, 13, 16, 17, 19, 20,  
22, 23, 27, 28 (13 QUESTÕES)

TOTAL: 13 DE 15

3.2) MODIFIQUE A REGRAS TRAPEZOIDAL  
 DE FORMA QUE ELA ESTIME CORRETAMENTE  
 A INTEGRAL MESMA QUE COMM-SZ NÃO  
 divide n uniformemente. (Você ainda pode assumir  
 QUE  $n \geq comm\_sz$ ) 3 \* Local-n é AGREGAÇÃO de TAREFAS

Q3.2.c - competitive-and-distributed-programming - Visual Studio Code

Arquivo Editar Seleção Ver Acessar Executar Terminal Ajuda

Q3.2.c x 2\_prova > Q3.2.c > ...

```

67  /* Length of each process' interval of
68  * integration = local_n*h. So my interval
69  * starts at: */
70 // se o processo atual é menor que o resto de n por quantidades de pr
71 if (my_rank < (n%comm_sz)){
72     // soma + 1 a quantidade de trapezios locais
73     //seta o valor de local_a e local_b
74     local_n = local_n+1;
75     local_a = a + my_rank*local_n*h;
76     local_b = local_a + local_n*h;
77
78 }else {
79     local_a = a + my_rank*local_n*h + (n%comm_sz)*h;
80     local_b = local_a + local_n*h;
81 }
82 local_int = Trap(local_a, local_b, local_n, h);
83

```

PROBLEMAS SAÍDA TERMINAL CONSOLE DE DEPURAÇÃO zsh +

```

competitive-and-distributed-programming/2_prova on ✘ master [?] took 13s
>mpixexec -n 2 ./Q3.2
Enter a, b, and n
0
14
3
With n = 3 trapezoids, our estimate
of the integral from 0.000000 to 14.000000 = 9.654814814814818e+02
competitive-and-distributed-programming/2_prova on ✘ master [?] took 15s
>

```

PARA O CÁLCULO DO TRAPEZIO COM  
 O VALOR  $n \% comm\_sz \neq 0$  DEVE SER CERTO. PRECESSIONAMOS  
 QUE OS PROCESSOS APENAS COM MÚLTIPLO  
 MÍNIMO DA TRAPEZOIDAL QUANDO O "comm\_sz" NÃO  
 É DIVISÍVEL POR "n".  $n \% comm\_sz$  DIFERENTE  
 REDISTRIBUIR OS PROCESSOS INICIANDO DO 0.  
 ALÉM DISSE OS PROCESSOS QUE NÃO RECEBEM ESTADOS.

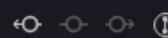
delle Téc AGASTAMENTO.

3.4) MODIFIQUE O PROGRAMA QUE APENAS IMPRIMA UMA LINHA DE SAÍDA DE CADA PROCESSO (MPI - outent.c) PARA QUE A SAÍDA DEJA IMPRESSA NA ORDEM DE CLASSIFICAÇÃO DOS PROCESSOS: A SAÍDA DO PROCESSO "0" É PREMIER, DEPOIS O PROCESSO "1" E ASSIM POR DIANTE

#### Q3.4.c - competitive-and-distributed-programming - Visual Studio Code

○ Editar Seleção Ver Acessar Executar Terminal Ajuda

Q3.4.c x



```
20 if(my_rank == 0) {  
21     printf("Proc %d of %d > Does anyone have a toothpick?\n",  
22         my_rank, comm_sz);  
23     // int MPI_Send(const void *buf, int count, MPI_Datatype datatype, i  
24     //int tag, MPI_Comm comm)  
25     MPI_Send(&send, 1, MPI_INT, my_rank+1, 0, MPI_COMM_WORLD);  
26 }  
27 // recebe do rank anterior e se não for o ultimo envia para proximo  
28 else{  
29     //receb do processo (rank) anterior  
30     //int MPI_Recv(void *buf, int count, MPI_Datatype datatype,  
31     //int source, int tag, MPI_Comm comm, MPI_Status *status)  
32     MPI_Recv(&send, 1, MPI_INT, my_rank-1, 0, MPI_COMM_WORLD, MPI_STATUS)  
33     printf("Proc %d of %d > Does anyone have a toothpick?\n",  
34         my_rank, comm_sz);  
35     // envia para o proximo se não for ultimo processo  
36     if (my_rank < comm_sz-1){  
37         MPI_Send(&send, 1, MPI_INT, my_rank+1, 0, MPI_COMM_WORLD);  
38     }  
39 }
```

PROBLEMAS SAÍDA TERMINAL CONSOLE DE DEPURAÇÃO

zsh +

competitive-and-distributed-programming/2\_prova on master [?] took 15s

> mpiexec -n 2 ./Q3.4

Proc 0 of 2 > Does anyone have a toothpick?

Proc 1 of 2 > Does anyone have a toothpick?

competitive-and-distributed-programming/2\_prova on master [?] took 2s

$n\%_{\text{com\_sz}} =$

3. 6) Se  $\text{com\_sz} = 4$  e  $\text{szpilha}$  que é

é um vetor com  $n = 1^{\text{a}}$  componentes. ~~(n)~~ como

os componentes de  $x$  seriam distribuídos

ENTRE OS PROCESSOS EM UM PROGRAMA QUE

USAVA UMA ~~DISTRIBUIÇÃO~~ EM BLOCOS?  $\rightarrow \text{ex: III}$

NA DISTRIBUIÇÃO EM BLOCOS, O NÚMERO  
" $n$ " DE ELEMENTOS É DIVIDIDO

IGUALMENTE E FORA ENTRE OS  
PROCESSOS, CASO OS RESULTADOS DA DIVISÃO

ENTRE  $n$  e  $\text{com\_sz}$  não tenha resto ( $n\%_{\text{com\_sz}} = 0$ ).

SE TIVER RESTO, PODEMOS DISTRIBUI-LO EM

ordem ENTRE OS PROCESSOS.

CADA PROCESSO VAI RECEBER A

INTIMA de  $n/\text{com\_sz} = 14 = 3$  ( $\text{int}(n/\text{com\_sz})$ ).

O RESTO DA DIVISÃO = 2 ( $n\%_{\text{com\_sz}} = 2$ )

LOGO O PROCESSO 0 e 1 vão receber

4 elementos e 2 e 3 vão ficar com 3:

| rank                     | 0  | 1  | 2 | 3 |
|--------------------------|--|--|---|---|
| $X[0], X[1], X[2], X[3]$ | $X[4], X[5] \times [6, 7] \times [8, 9]$ | $X[10], X[11] \times [12, 13] \times [14, 15]$ |   |   |

3.6) b) como os componentes de x seriam

distribuídos entre os processos em um programa que realiza uma DISTRIBUIÇÃO CÍCLICA?

TABELA 3.9  
CGIZIO

Em uma DISTRIBUIÇÃO cíclica, o NÚMERO "n" de ELEMENTOS É DISTRIBUÍDO UM POR VZ EM CADA PROCESSO Em um DETERMINADO NÚMERO DE CICLOS.

$x[0]$  vai para o PROCESSO 0, e  $x[1]$  vai para o PROCESSO 1, e  $x[2]$  vai para o PROCESSO 2 e  $x[3]$  vai para o PROCESSO 3, FECHANDO UM CICLO. O PRÓXIMO CICLO SEGUE A MESMA LÓGICA.

| RANK | 0                            | 1                            | 2                               | 3                       |
|------|------------------------------|------------------------------|---------------------------------|-------------------------|
|      | $x[0], x[4]$<br>$x[8], x[2]$ | $x[1], x[5]$<br>$x[3], x[7]$ | $x[2], x[6]$<br>$x[10] \otimes$ | $x[3], x[7]$<br>$x[11]$ |
| :    | :                            | :                            | :                               |                         |

2.6) a) como os componentes de X

SERIAM DISTRIBUÍDOS ENTRE OS PROCESSOS

Em uma programa que usava uma distribuição  
CÍCLICA DE BLOCOS com TAMANHO DE Bloco b=?

UMA DISTRIBUIÇÃO BLOCO-CÍCLICA, AN-

UMA JUNGAIS DAS PROPRIEDADES DE BLOCOS

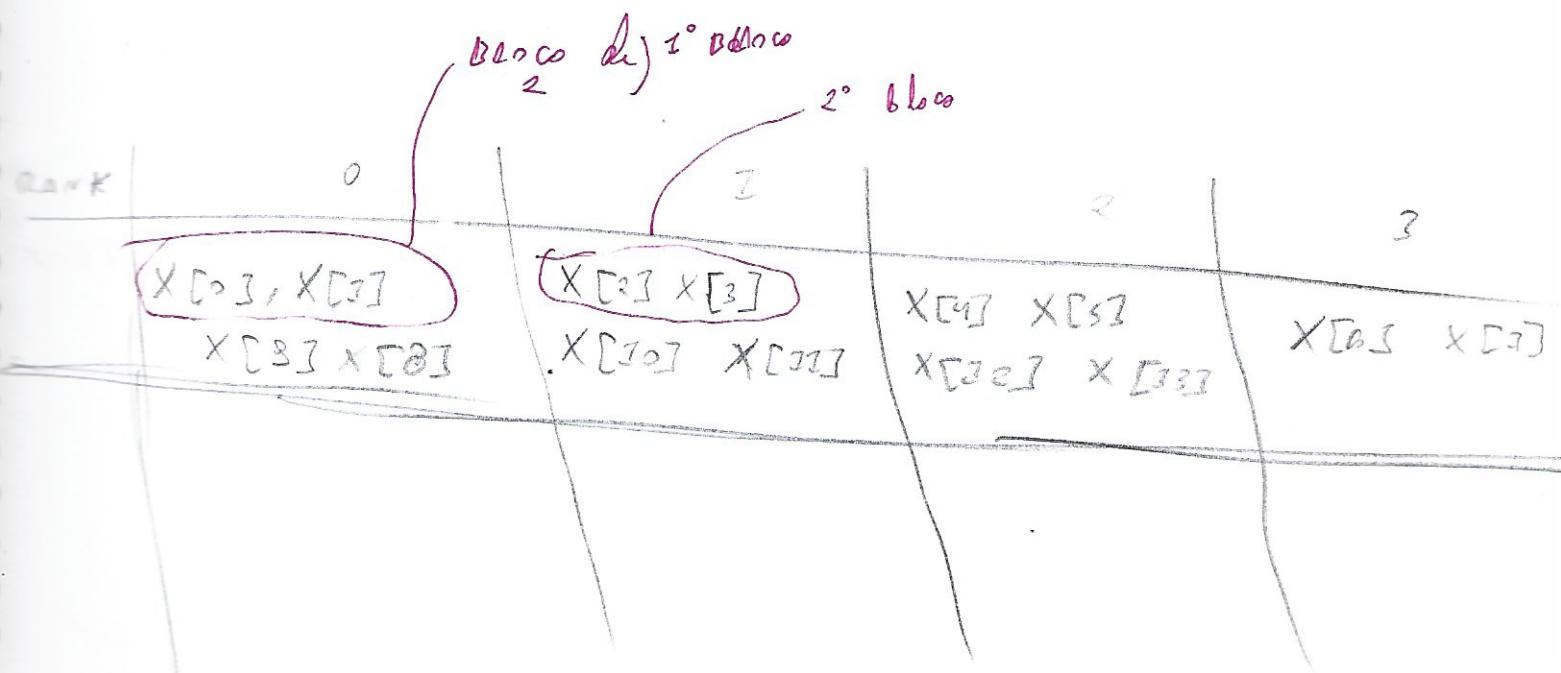
E CÍCLICA. PARA UM DETERMINADO "b"

o vetor "X" DE "n" ELEMENTOS SERIA DIVIDIDO

EM BLOCOS. O NÚMERO DE BLOCOS SERIA

$n/b = 19/2 = 7$ . CASO A DIVISÃO

INTEIRA TIUVESSO RESTO, ELA SERIA DISTRIBUÍDA  
ENTRE OS BLOCOS SIGUINHOS A MESMA ORDEM.



Q3.9) ESCREVA UM PROGRAMA MPI QUE  
IMPLEMENTE A MULTIPLICAÇÃO DE UM  
VETOR POR UM PRODUTO ESCALAR A  
ESCALAR. O USUÁRIO DEVE INSERIR DOIS  
VETORES E UM ESCALAR, TUDO LIDOS PELO  
PROCESSO "0" E DISTRIBUIDOS ENTRE OS  
PROCESSOS. OS RESULTADOS SÃO CALCULADOS  
E COLETADOS NO PROCESSO "0" QUE OS  
IMPRIME. VOCÊ PODE ASSUMIR QUE N  
É ORDEM DOS VETORES, E IGUALMENTE DIVISÍVEL  
POR COMM-SZ.

VAMOS MULTIPLICAR VETOR-1 PELO ESCALAR  
E O RESULTADO FAZENDO O PRODUTO COM

VETOR 2. O VETOR 1 E VETOR 2 SÃO

DIVISÍVEIS COM MPI - SCATTER ENTRE OS  
PROCESSOS E CADA (PEGA O DADO DO VETOR E  
DISSEMINA ENTRE OS PROCESSOS.)

VOCAL-VETOR-1 E LOCAL-VETOR-2 SÃO ATÉ BEM  
PARA TER OS RESULTADOS SOMADO PELO  
MPI - REDUCE.

CÓDIGO PRÓXIMA  
PÁGINA.

Editar Seleção Ver Acessar Executar Terminal Ajuda

Q3.9.c X

2\_prova &gt; C Q3.9.c &gt; ...

```
55     scanf("%d",&vetor2[i]);
56     //MPI_Scatter espalha o vetor1 para os processos como local_
57     MPI_Scatter(vetor1, local_n, MPI_INT,
58     local_vetor1, local_n, MPI_INT, 0, MPI_COMM_WORLD);
59     MPI_Scatter(vetor2, local_n, MPI_INT,
60     local_vetor2, local_n, MPI_INT, 0, MPI_COMM_WORLD);
61     free(vetor1);
62     free(vetor2);
63 }else{
64     MPI_Scatter(vetor1, local_n, MPI_INT,
65     local_vetor1, local_n, MPI_INT, 0, MPI_COMM_WORLD);
66     MPI_Scatter(vetor2, local_n, MPI_INT,
67     local_vetor2, local_n, MPI_INT, 0, MPI_COMM_WORLD);
68 }
```

PROBLEMAS SAÍDA TERMINAL CONSOLE DE DEPURAÇÃO

z5

**competitive-and-distributed-programming/2\_prova** on ↴ master [?] took 28s

&gt;mpieexec -n 2 ./Q3.9

Entre com o tamanho dos vetores

2

Entre com o escalar

8

Digite o primeiro vetor

1

5

Digite o segundo vetor

1

6

0 resultado final: 248

**competitive-and-distributed-programming/2\_prova** on ↴ master [?] took 46s

&gt;

3. 13) MPI scatter & MPI Gather tem a  
LIMITAÇÃO DE QUE CADA PROCESSO DEVE  
ENVIAR EM RECEBER O MESMO NÚMERO DE  
ITENS DE DADOS. QUANDO ESTE NÃO  
for o caso, devemos usar as Funções  
MPI Gather & MPI scatter. VEJA AS  
PÁGINAS DO MANUAL PARA ESSAS FUNÇÕES  
E MODELE AQUELE PROGRAMA DE PRODUTO ESCALAR PARA  
QUE QUE POSSA LIDAR CORRETAMENTE com  
o caso quando "n" não é DIVISÍVEL  
UNIFORMEMENTE por "num\_sz".

CÓDIGO NA PRÓXIMA  
PÁGINA

```
2_prova > c Q3.13.c > main(void)
    int *displs, MPI_Datatype MPI_COMM_WORLD, MPI_Group MPI_COMM_WORLD);
66     geraSendCounts(comm_sz, n, sendCounts, displs);
67
68     //Preenche os vetores e os distribui
69     if(my_rank == 0){
70         printf("Digite o primeiro vetor\n");
71         for(i=0;i<n;i++)
72             scanf("%d",&vetor1[i]);
73         printf("Digite o segundo vetor\n");
74         for(i=0;i<n;i++)
75             scanf("%d",&vetor2[i]);
76     }
77     /*
78     int MPI_Scatterv(const void *sendbuf, const int *sendcounts, const i
79                         MPI_Datatype sendtype, void *recvbuf, int recvcount,
80                         MPI_Datatype recvtype, int root, MPI_Comm comm)
81     sendcounts
82     integer array (of length group size) specifying the number of elemen
83     */
84     MPI_Scatterv(vetor1, sendCounts, displs, MPI_INT,
85                 local_vetor1, local_n, MPI_INT, 0, MPI_COMM_WORLD);
86     MPI_Scatterv(vetor2, sendCounts, displs, MPI_INT,
87                 local_vetor2, local_n, MPI_INT, 0, MPI_COMM_WORLD);
88     free(vetor1);
89     free(vetor2).
```

Q3.13.c - competitive-and-distributed-programming - Visual Studio Code

Arquivo Editar Seleção Ver Acessar Executar Terminal Ajuda



```
c Q3.13.c x
2_prova > c Q3.13.c > main(void)
107
108     int geraSendCounts(int p, int n, int sendCounts[], int displs[]) {
109
110         int local_n = n / p;
111         int i;
112         for (i = 0; i < p; i++) {
113             if(i < (n % p))
114             {
115                 sendCounts[i] = local_n +1;
116                 displs[i] = i*(local_n + 1);
117             }
118             else
119             {
120                 sendCounts[i] = local_n;
121                 displs[i] = i*(local_n) + n % p;
122             }
123         }
124     }
```

3. 16) SUPONHA QUE COMO  $SZ = 8 \times 8$

Vetor  $x = (0, 1, 2, \dots, 75)$  foi distribuído ENTRE OS PROCESSOS USANDO UMA DISTRIBUIÇÃO DE BLOCOS. DESENHE UM DIAGRAMA ILUSTRANDO AS ETAPAS EM UMA IMPLEMENTAÇÃO BORBOLETA DE ALL GATHER DE EX.



3. IF) O TÍPO MPI CONTEÚDO Pode  
SER USADO PARA CONSTUIR UM TÍPO  
DE DADOS DERIVADO DE UMA  
COLEÇÃO DE ELEMENTOS CONSECUTIVOS  
EM MATRIZ. Sua SINTAXE É

```
int MPI_Type_contiguous(  
    int count /* in */,  
    MPI_Datatype old_type /* in */  
    MPI_Datatype * new_type /* out */)
```

MODELAR AS FUNÇÕES READ SECTION  
É BEM VECTORS PARA QUE SEjam USADAS EM TIPOS  
DE DADOS MPI CRIADOS POR UMA  
CHAMADA PARA MPI\_TYPE CONSEGUNTE E  
UM ARGUMENTO DE CONTAGEM DE I  
NAS CHAMADAS PARA MPI\_SCATTER E  
MPI\_GATHER.

Arquivo Editar Seleção Ver Acessar Executar Terminal Ajuda



c Q3.13.c

c Q3.17.c ●

```

2_prova > c Q3.17.c > main(void)
  1  double* local_y;
  2
  3  int m, local_m, n, local_n;
  4  int my_rank, comm_sz;
  5  MPI_Comm comm;
  6  MPI_Datatype type;
  7  MPI_Type_contiguous(local_n, MPI_DOUBLE, &type);
  8  MPI_Type_commit(&type);
  9
 10
 11  MPI_Init(NULL, NULL);
 12  comm = MPI_COMM_WORLD;
 13  MPI_Comm_size(comm, &comm_sz);
 14  MPI_Comm_rank(comm, &my_rank);
 15
 16
 17  Get_dims(&m, &local_m, &n, &local_n, my_rank, comm_sz, comm);
 18  Allocate_arrays(&local_A, &local_x, &local_y, local_m, n, local_n, co
 19  Read_matrix("A", local_A, m, local_m, n, my_rank, comm);
 20
 21 # ifdef DEBUG
 22     Print_matrix("A", local_A, m, local_m, n, my_rank, comm);
 23 # endif
 24 //  Read_vector("x", local_x, n, local_n, my_rank, comm);
 25 //  Read_vector(local_x, n, local_n, my_rank, type, comm);
 26
 27 # ifdef DEBUG
 28     //  Print_vector("x", local_x, n, local_n, my_rank, comm);
 29     Print_vector(local_x, n, local_n, my_rank, type, comm);
 30 # endif
 31
 32
 33
 34
 35
 36

```

Arquivo Editar Seleção Ver Acessar Executar Terminal Ajuda



c Q3.17.c ●

```

2_prova > c Q3.17.c > Read_vector(double [], int, int, int, MPI_Datatype, MPI_Comm)
 284 * ERRORS:    If malloc or temporary storage fails on process 0, the
 285 *               program prints a message and all processes quit
 286 * Notes:
 287 * 1. Communicator should be MPI_COMM_WORLD because of call to
 288 *     Check_for_errors
 289 * 2. local_n should be the same on all processes
 290 */
 291
 292 void Read_vector(double local_vector[], int n, int local_n,
 293 int my_rank, MPI_Datatype type, MPI_Comm comm){      You, seconds ago *
 294     int i;
 295     double *vector = NULL;
 296
 297     if (my_rank ==0){
 298         vector = malloc(n*sizeof(double));
 299         printf("Enter the vector\n");
 300
 301         for (i = 0; i < n; i++)
 302             scanf("%lf", &vector);
 303     }
 304     MPI_Scatter(vector, 1, type, local_vector,1,type,0,comm);
 305     free(vector);
 306 }
 307 // void Read_vector(
 308 //     char      prompt[] /* in */,
 309 //     double   local_vec[] /* out */,

```

3.28) MPI - TYPE - INDEXED Pode ser usada  
PARA CONSTRUIR UM TIPO DE DADOS DERIVADO  
DE ELEMENTOS DE ARRAY ARBITRÁRIO. SUA  
SINTAXE É

Int MPI - TYPE - indexed {

int count /\* in \*/;

int ~~count~~ \*

int array - of - block lengths [ ]

int array - of - block cements [ ]

MPI - DATA TYPE dd MPI - t

MPI - DATA TYPE NEN - MPI - T - P /\* entry \*/;

Ao CONTRARIO DE MPI - TYPE - CREATE - STRUCT,  
OS DESCENDIMENTOS SÃO MEDIDOS EM UNIDADES  
DO ANTES MPI T - NÃO EN BYTES.  
USE O TIPO MPI INDEXADO PARA CRIAR  
UM TIPO DE DADOS DERIVADOS QUE  
CORRESPONDE À PARTE TRIANGULAR SUPERIOR  
DE UMA MATRIZ QUADRADA, POR EXEMPLO  
NA MATRIZ  $4 \times 4$ :

$$\begin{bmatrix} 0 & 1 & 2 & 3 \end{bmatrix}$$

$$\begin{bmatrix} 4 & 5 & 6 & 7 \end{bmatrix}$$

$$\begin{bmatrix} 8 & 9 & 10 & 11 \end{bmatrix}$$

$$\begin{bmatrix} 12 & 13 & 14 & 15 \end{bmatrix}$$

A PARTE TRIANGULAR SUPERIOR SÃO OS ELEMENTOS 0, 1, 2, 3, 5, 6, 7, 10, 11, 15.

O PROCESSO 0 DEVE SER LEIDO EM UMA MATRIZ NN COMOS UMA MATRIZ UNIDIMENSIONAL, CRIAR O TIPO DE DADOS DERIVADO E ENVIAR A PARTE TRIANGULAR SUPERIOR COM UMA UNICA CHAMADA PARA MPI SEND. O PROCESSO 1 DEVE RECEBER A PARTE TRIANGULAR SUPERIOR COM UNICA CHAMADA PARA MPI Rec e DEPOIS IMPRIMIR OS DADOS RECEBIDOS.

CÓDIGO ABAIXO!

```
Q3.19.c - competitive-and-distributed-programming - Visual Studio Code
Arquivo Editar Seleção Ver Acessar Executar Terminal Ajuda
Q3.19.c
2_prova > Q3.19.c > constroiTipoIndexado(double *, int, MPI_Datatype *)
72 */
73 void constroiTipoIndexado(double* matriz, int n,
74 MPI_Datatype* novoTipoConstruido) {
75     int array_tamanhoDosBlocos[n];
76     int array_distancias[n];
77     array_distancias[0] = 0;
78     int i;
79     int j;
80     for (i = 0, j = n; i < n; i++, j--) {
81         // O tamanho dos arrays diminuem -1 a cada linha (triangulo super
82         array_tamanhoDosBlocos[i] = j;
83     }
84     for (i = 1; i < n; i++) {
85         // O deslocamento dele é o deslocamento do elementoAtual - o do
86         array_distancias[i] = i*n + i;
87     }
88     MPI_Type_indexed(n, array_tamanhoDosBlocos, array_distancias, MPI_Datatype);
89     MPI_Type_commit(novoTipoConstruido);
90 }
```

Arquivo Editar Seleção Ver Acessar Executar Terminal Ajuda

C Q3.19.c ●

2\_prova &gt; C Q3.19.c &gt; ...

```
60  /**
61   * Constrói um tipo indexado do triangulo superior
62   * da matriz informada      You, seconds ago • Unc
63   * O tipo construído vem no seguinte formato:
64   *
65   * matriz = [ 0 1 2 ]
66   *           [ 3 4 5 ]
67   *           [ 6 7 8 ]
68   *
69   * novoTipoConstruido = ( [ 0 1 2 ],
70   *                         [     4 5 ],
71   *                         [         8 ] )
72   *
73 */
```

PROBLEMAS

SAÍDA

TERMINAL

CONSOLE DE DEPURAÇÃO

&gt;mpiexec -n 2 ./Q3.19

Digite o tamanho de n:

2

Digite a matriz n x n:

1

0

0

1

Matriz:

1.000000 0.000000

0.000000 1.000000

competitive-and-distributed-programming/2\_prova on ↗ master [!?] took 27s

&gt;

master\* ⌂ ⌂ 0 △ 0 ⌂ Live Share

∅ You, seconds ago Ln 62, Col 4 Espaço

3.20) AS FUNÇÕES MPI PACK E MPI UNPACK

FORNCEM UNA ALTERNATIVA AOS TIPOS DE DADOS DERIVADOS PARA AGRUPAR DADOS.

O MPI PACK COPIA OS DADOS E SEREM ENVIADA OS, UM BLOCO POR VEZ, EM UM buffer FORNECIDO PELA USUÁRIO. O BUFFER PODE ENTÃO SER ENVIADO E RECEBIDO. DEPOIS QUE OS DADOS SÃO RECEBIDOS, MPI UNPACK PODE SER USADO PARA DESCOMPACTA-LOS DO BUFFER DE RECEPÇÃO. A SINTaxe É:

ENT MPI PACKS

```
void * in buf /* in */,  
int In Count /* in */,  
MPI - DATATYPE datatype /* in */,  
Void * Pack buf /* out */,  
int Pack buf Sz /* in */,  
Int * Position /* in/out */,  
MPI Comm /* in */;
```

PODEMOS, PORTANTO, FAZER PARTE OS DADOS DE ENTRADA PARA O PROGRAMA DE REGRAS TRAPEZOIDAL, COM O SEGUINTE:

MPI Pack ( $f_a$ , 1, MPI\_DOUBLE, Pack buf, 500, &written, com )

MPI Pack ( $f_b$ , 1, MPI\_DOUBLE, Pack buf, 500, &written, com )

MPI Pack ( $f_n$ , 1, MPI\_DOUBLE, Pack buf, 500, &written, com )

ESCREVA OUTRA FUNÇÃO GET Input para  
o PROGRAMA DE REGRA TRAPEZOIDAL. ESTE  
DEVE USAR MPI-PACK NO PROCESSO

E MPI - UNPACK nos outros processos  
Porque é melhor?

CÓDIGO sob ABNIXO

Q3.20.c - competitive-and-distributed-programming - Visual Studio Code

←○○→



macos: 2 UTF-8 LF C Ⓜ Go Live Linu

Q3. 22)

CRONOMETRAR NOSSA IMPLEMENTAÇÃO

DA

REGRAS

TRAPEZOIDAL

QUE

USA

MPI - REDUCE

Como

VOCÊ

ESCOLHERÁ

N

E OS NÚMEROS DE TRAPEZOIDAL

Como

OS

TEMPOS

MÍNIMOS

SE

COMPARAM

DOS

TEMPOS

MÉDIOS

E

MÉDIAS NOS?

QUAIS

SÃO

OS

ACELERAMENTOS?

QUAIS

SÃO

AS

EFCIÊNCIAS?

com BASE NOS

DADOS

QUE VOCÊ

COLETOU, VOCÊ

DIREIA

QUE

A

REGRAS

TRAPEZOIDAL E

ESCALAIFI?

MANDANDO

O

CÓDIGO

Seleção Ver Acessar Executar Terminal Ajuda

c Q3.13.c U c Q3.20.c c Q3.22.c X

> c Q3.22.c > ...

```
/* Find out how many processes are being used */
MPI_Comm_size(MPI_COMM_WORLD, &comm_sz);

Get_input(my_rank, comm_sz, &a, &b, &n);
MPI_Barrier(MPI_COMM_WORLD);
start = MPI_Wtime();
h = (b-a)/n; /* h is the same for all processes */
local_n = n/comm_sz; /* So is the number of trapezoids */
/* Length of each process' interval of
 * integration = local_n*h. So my interval
 * starts at: */
local_a = a + my_rank*local_n*h;
local_b = local_a + local_n*h;
local_int = Trap(local_a, local_b, local_n, h);
finish = MPI_Wtime();
loc_elapsed = finish-start;
MPI_Reduce(&loc_elapsed, &tempoMaximo, 1, MPI_DOUBLE, MPI_MAX, 0, MPI_COMM_WORLD);
MPI_Reduce(&loc_elapsed, &tempo_minimo, 1, MPI_DOUBLE, MPI_MIN, 0, MPI_COMM_WORLD);
MPI_Reduce(&loc_elapsed, &tempo_medio, 1, MPI_DOUBLE, MPI_SUM, 0, MPI_COMM_WORLD);
/* Add up the integrals calculated by each process */
MPI_Reduce(&local_int, &total_int, 1, MPI_DOUBLE, MPI_SUM, 0,
           MPI_COMM_WORLD);
/* Print the result */
```

```
competitive-and-distributed-programming/2_prova on þ master [?] took 37s
```

```
>mpiexec -n 1 ./Q3.22
```

```
Enter a, b, and n
```

```
0
```

```
100000
```

```
100000
```

```
With n = 100000 trapezoids, our estimate
```

```
of the integral from 0.000000 to 100000.000000 = 3.333333333500000e+14
```

```
Tempo MÁ nimo gasto = 3.967648e-03
```

```
Tempo MÁximo gasto = 3.967648e-03
```

```
Tempo MÁdio gasto = 3.967648e-03
```

```
Tempo Mediana gasto = 3.967648e-03
```

```
competitive-and-distributed-programming/2_prova on þ master [?] took 15s
```

```
>mpiexec -n 2 ./Q3.22
```

```
Enter a, b, and n
```

```
0
```

```
100000
```

```
100000
```

```
With n = 100000 trapezoids, our estimate
```

```
of the integral from 0.000000 to 100000.000000 = 3.333333333500000e+14
```

```
Tempo MÁ nimo gasto = 1.958824e-03
```

```
Tempo MÁximo gasto = 1.969339e-03
```

```
Tempo MÁdio gasto = 1.964081e-03
```

```
Tempo Mediana gasto = 1.964081e-03
```

```
competitive-and-distributed-programming/2_prova on þ master [?] took 13s
```

```
>
```

Foi usado MPI - WTime()

$n = 2$

| TEMPO     | TEMPO SERIAL                      | TEMPO PARALELO                     |
|-----------|-----------------------------------|------------------------------------|
| MÍNIMO    | $3,96 \times 10^{-3} [\text{ms}]$ | $1,95 \times 10^{-3} [\text{ms}]$  |
| MÁXIMO    | $3,96 \times 10^{-3} [\text{ms}]$ | $1,962 \times 10^{-3} [\text{ms}]$ |
| MÉDIA     | $3,96 \times 10^{-3} [\text{ms}]$ | $1,964 \times 10^{-3} [\text{ms}]$ |
| MEDIANA   | $3,96 \times 10^{-3} [\text{ms}]$ | $1,964 \times 10^{-3} [\text{ms}]$ |
| RESULTADO | $3,33 \times 10^{-4}$             | $3,33 \times 10^{-4}$              |

\* Com. esso agora podemos

obter SPEEDUP e EFFICIENCIA

PELA DEFINIÇÃO

$$S = \text{SPEEDUP} = \frac{T_{\text{SERIAL}}}{T_{\text{PARALELO}}}$$

$$\text{EFEICIENCIA} = \frac{S}{P} = \frac{T_{\text{SERIAL}}}{P \cdot T_{\text{PARALELO}}}$$

ASSIM FAZEMOS A SIGUIENTE  
TABELA!

$$a = 2; b = 100000$$

| n      | SPEEDUP   | EFICIENCIA | RESULTADO             |
|--------|---|------------|-----------------------|
| 10     | $S = \frac{3,35 \times 10^0}{3,42 \times 10^{-1}} = 1,93$     | E = 0,78   | $3,35 \times 10^{24}$ |
| 100    | $S = \frac{5,34 \times 10^0}{2,91 \times 10^{-1}} = 1,84$     | E = 0,87   | $3,33 \times 10^{14}$ |
| 1000   | $S = \frac{4,08 \times 10^{-5}}{2,022 \times 10^{-5}} = 2,02$ | E = 1,01   | $3,33 \times 10^{39}$ |
| 100000 | $S = \frac{3,36 \times 10^{-3}}{2,00 \times 10^{-3}} = 2,00$  | E = 0,93   | $3,33 \times 10^{59}$ |

A ESCALA BIBLIODA

WEI TWPAD - SUPERCOMPUTADOR - MFR

|              |               |            |
|--------------|---------------|------------|
| 9 PROCESSOS  | 120 TRABALHOS | EFICIENCIA |
| 8 PROCESSOS  | 200 TRABALHOS |            |
| 16 PROCESSOS | 400 TRABALHOS |            |
| 32 PROCESSOS | 800 TRABALHOS |            |

PERCEBEU SE QUE COM 100 TRAPEZOS  
TIIVEMOS A PRECISÃO DE TRES CASAS DECIMAS  
possentes. com isso → SPEEDUP E  
EFICIÊNCIA CRESCERAM A Partir  
DESSA QUANTIDADE DE TRAPEZOS.

FOI REALIZADO com VALOR MÍNIMO  
ESSAS MEDIDAS ASSEGURAR MELHOR CASO.

O TRAPEZIO NÃO DEMOSTRAU SER  
ESCALAVEL DEVIDO A EFICIÊNCIA  
NÃO PERMANECER CONSTANTE AUMENTANDO  
A ENTRADA E QUANTIDADE DE PROCESSOS  
PELO MÉTODO EM Comum!

Q3. 23) Embora não saibamos os detalhes internos da implementação do MPI REDUCE, podemos supor que ele usa uma estrutura semelhante à árvore binária que discutimos. Se for esse o caso, esperaríamos que seu tempo de execução crescesse aproximadamente à taxa de  $\log_2^P$ , uma vez que existem cerca de  $\log_2(P)$  níveis na árvore. (Aqui  $P = \text{commsz}$ )

Uma vez que o tempo de execução da regra trapezoidal serial é aproximadamente proporcional a " $n$ ", o número de trapézios, e a regra trapezoidal paralela simplesmente aplica a trapezios

$n/P$  em cada processo, com nossa suposição sobre MPI REDUCE, obtemos uma fórmula para o tempo de execução geral da regra trapezoidal paralela que se parece com algumas constantes  $a$  e  $b$ .

a) USE A FÓRMULA 3.22 E SEU PROGRAMA MATEMÁTICOS (POR EXEMPLO, MATLAB) PARA OBTIR UMA ESTIMATIVA DE MÍNIMOS QUADRADOS DOS VALORES DE " $a$ " E " $b$ "?

| $n^{\circ}$ de trapézios | TEMPO ( $P=2$ )        |
|--------------------------|------------------------|
| 10                       | $5,904 \times 10^{-8}$ |
| 100                      | $7,204 \times 10^{-6}$ |
| 1.000                    | $2,99 \times 10^{-5}$  |
| 10.000                   | $2,20 \times 10^{-9}$  |
| 100.000                  | $2,02 \times 10^{-3}$  |

A Tabela é que a coluna da MATRIZ, é " $n/p$ " (SEGUNDOS COLUNA  $\log_2(P)$ , com os tempos)

Lembrando

$$\begin{bmatrix} n/p & \log_2(P) \\ n/p & \log_2(P) \\ n/p & \log_2(P) \\ n/p & \log_2(P) \\ n/p & \log_2(P) \end{bmatrix}, \{5,904 \cdot 10^{-8}, 7,204 \cdot 10^{-6}, 2,99 \cdot 10^{-5}, 2,20 \cdot 10^{-9}, 2,02 \cdot 10^{-3}\}$$

~~USEI~~ USEI A WOLFRAM ALPH

LeastSquares $\left[\begin{pmatrix} 5 & 1 \\ 50 & 1 \\ 500 & 1 \\ 5000 & 1 \\ 50000 & 1 \end{pmatrix}, \{5.40, 7.20, 24.4, 220, 2023\}\right]$

NATURAL LANGUAGE

MATH INPUT

✖ =

★ √ ∂f (:) √ω aω | ...

### Input

LeastSquares $\left[\begin{pmatrix} 5 & 1 \\ 50 & 1 \\ 500 & 1 \\ 5000 & 1 \\ 50000 & 1 \end{pmatrix}, \{5.4, 7.2, 24.4, 220, 2023\}\right]$

### Result

{0.0403199, 8.00613}

POWERED BY THE WOLFRAM LANGUAGE

Q3.23) b) COMENTE SOBRE A QUALIDADE  
DOS TEMPOS DE EXECUÇÃO PREVISTO  
USANDO A FÓRMULA E OS VALORES  
PARA "a" e "b" CALCULADOS NA Parte (a)

OS RESULTADOS SÃO MUITO BONS, DEVIDO  
A MPI = REDUCE S. J. A IMPLEMENTAÇÃO  
COM FAVOR DE BINÁRIA CONSEGUE MUITO

3. 27) ENCONTRE AS ACCELERAÇÕES

EFEICIENCIAS DO TIPO PARALELO IMPAR-  
PAR. O PROGRAMA obtém acelerações livres?

É ESCALÁVEL? É ALTAMENTE ESCALONAVEL?

**E PRACTICAMENTE ESCALONÁVEL?**

PASSEGRER ADA EX 0

Q3.27.c • competitive and distributed programming - visual Studio Code

Ver Acessar Executar Terminal Ajuda

Q3.27.c •

main(int, char \* [] )

```
MPI_Init(NULL, NULL); > Get_args Aa Ab 2 de 5 ↑ ↓ ≡ ×

/* Get my process rank */
MPI_Comm_rank(MPI_COMM_WORLD, &my_rank);

/* Find out how many processes are being used */
MPI_Comm_size(MPI_COMM_WORLD, &comm_sz);
start = MPI_Wtime();
Get_args(argc, argv, &n, &g_i);
a = (int*) malloc(n*sizeof(int));           itrack, 5 days ago • feat(list)
if (g_i == 'g') {
    Generate_list(a, n);
    Print_list(a, n, "Before sort");
} else {
    Read_list(a, n);
}

Odd_even_sort(a, n);

Print_list(a, n, "After sort");

free(a);
finish = MPI_Wtime();
loc_elapsed = finish-start;
if (my_rank == 0) {
```

Con ISSO MONTAMOS A  
SIGUIENTE TABLA

| n       | SIRFAL                           |                                  | PARALELO                 |
|---------|----------------------------------|----------------------------------|--------------------------|
|         | 1 PROCESADOR<br><del>→ T0J</del> | 2 PROCESADORES<br><del>T0J</del> |                          |
| 100     | $14,73 \times 10^{-6}$           |                                  |                          |
| 1.000   | $135 \times 97 \times 10^{-6}$   |                                  |                          |
| 10.000  | $1375,08 \times 10^{-6}$         |                                  | $665,18 \times 10^{-5}$  |
| 100.000 | $13037,73 \times 10^{-6}$        |                                  | $1923,05 \times 10^{-5}$ |
|         |                                  |                                  | 23630,86 s               |

PERCEBE - SE QUE OS TIPOS ANTES CONSIDERADOS  
2 PROCESSOS, UMA POSSIVEL EXPLICAÇÃO  
DEVIDO A TRAÇA DE MENSAGEM  
ENTRE OS NÚCLEOS DE USAMOS  
USAMOS MAIS PROCESSADORES

\* AGORA VAMOS CALCULAR SPPROS E  
EFICIÊNCIA.

| n       | SPPROS <del>→</del> | EFICIÊNCIA |
|---------|---------------------|------------|
| 100     | 0,45                | 0,23       |
| 1.000   | 0,27                | 0,14       |
| 10.000  | 0,91                | 0,135      |
| 100.000 | 0,80                | 0,44       |

PERCEBER QUE O SPEEDUP  
O PODEROSA  
NÃO É LINEAR DEVIDO ELE  
NÃO CRESCER LEMPARMENTE COM  
O PROBLEMA

Agora VAMOS TESTAR A ESCALA DE UNI  
DE PARA ISSO FOI TESTADO NA  
NPAD

4 PROCESSOS  
100 ELEMENTOS

$$E =$$

8 PROCESSOS  
200 ELEMENTOS

$$E =$$

16 PROCESSOS  
400 ELEMENTOS

$$E =$$

32 PROCESSOS  
800 ELEMENTOS

$$E =$$

NÃO É ESCALAVEL PORQUE A EFICIÊNCIA  
NÃO SE MANTÉVE CONSTANTE.

O PROGRAMA MANTÉM APROXIMADAMENTE  
CONSTANTE A EFICIÊNCIA QUANDO O  
AUMENTAMOS O PROBLEMA E  
MANTEMOS IGUAL O NÚMERO DE  
CÓDIGO DIFERENTES. P. T. C. E. F. R. C. I. K.

DESSA FORMA NÃO É FORTEMENTE  
ESCALÁVEL PORQUE A EFICIÊNCIA  
NAS PERMANECE CONSTANTE COM  
AUMENTO DO PROCESSADORES.

Q16 EXTRA: QUANDO O COMPILADOR  
NÃO É CAPAZ DE VETORIZAR  
AUTOMATICAMENTE, OU VETORIZAR DE  
FORMA INEFICIENTE, O OpenMP PROVÉ  
A diretiva "omp simd", com a qual  
o PROGRAMA PODE INDICAR UM LAGE EXPLÍCI-  
TAMENTE PARA A COMPILAÇÃO VETORIZAR.

NO CÓDIGO ABASTECE A INCLUSÃO  
DA CLÁUSULA "reduction" FUNCIONA  
DE FORMA SEMELHANTE A FLAG  
"-ffast-math", INDICANDO QUE A  
REDUÇÃO NA VARIAVEL sera FEITA  
E DEVE SER FEITA

#pragma omp simd reduction(+:soma)

for (i=0; i < n; i++) {

    x = (i + 0.5) \* h

    soma += q.0 / (1.0 + x\*x)

}

PODE SER NÃO É NECESSÁRIO USAR A CLÁUSULA  
PRIVATE(x) NESTE CASO MAS SERIA ENSO

DEVERIA OMP SIMD POSSER SER COMBINADA

COM A diretiva OMP PARALLEL FOR?

O "SIMD" REALIZA multiplos ITERAÇÕES DO LOOP SEJA EXECUTADA CONCURRENTEMENTE USANDO A INSTRUÇÃO "SIMD"

OU SEJA SINGLE INSTRUCTION MULTIPLE DATA, ASSIM REALIZA VETORIZAÇÃO DE BLOCOS DE ITERAÇÕES DE LOOPS QUE PERTENCEM A MESMA THREADS.

ASSIM, CASO SEJA ADICIONADO "PARALLEL FOR SIMD" STREAM GERA DIFERENTES THREADS, QUE PROCESSA AS OPERAÇÕES VETORIALMENTE EM TEMPOS DINTINTOS, TORNANDO A ATRIBUIÇÃO DE "x" POSSÍVEL DE CONDECAS ARRISCADA. DEVE SE USAR PRIVATE(X) NOSSO CONTEXTO "SIMD", "PARALLEL SIMD" (SEM PRIVATE(X)).