Iniciado em	segunda-feira, 20 nov. 2023, 14:06
Estado	Finalizada
Concluída em	segunda-feira, 20 nov. 2023, 15:35
Tempo empregado	1 hora 29 minutos
Avaliar	**5,19** de um máximo de 10,00(**51,9**%)

Questão 1

Completo

Atingiu 0,20 de 2,50

O objetivo do código a seguir é fazer a impressão do texto repetidas vezes, em função da constante MAX e da variável number, que é calculada a cada iteração do while.

```c
#include <sys/time.h>
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
#define MAX 1000
int main (void) {
        char texto_base[] = "abcdefghijklmnopqrstuvwxyz 1234567890 ABCDEFGHIJKLMNOPQRSTUVWXYZ";
        /* a variavel indice aponta para o primeiro caracter do texto */
        int    *indice=(int *) malloc (sizeof(int));
        *indice = 0;
        struct timeval tv;
        int number, tmp_index, i, cont=0;
        while(cont<MAX) {
                gettimeofday (&tv, NULL );
                number = ((tv.tv_usec / 47) % 3) + 1;
                tmp_index = *indice;
                for (i = 0; i < number; i++ )
                        if( ! (tmp_index + i > sizeof(texto_base)) )
                                fprintf(stderr, "%c", texto_base[tmp_index + i]);
                *indice = tmp_index + i;
                if (tmp_index + i > sizeof(texto_base)) {
                        fprintf(stderr, "\n");
                        *indice = 0;
                } /* fim-if */
                cont++;
        } /* fim-while */
        printf("\n");
        return 0;
} /* fim-main */
```

Crie uma versão MPI deste código com apenas três processos, de modo que a lógica original se mantenha, mas a impressão do texto ocorra de modo colaborativo (com divisão de trabalho mais igual possível) entre esses três processos.

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/time.h>
#include <mpi.h>

#define MAX 1000
#define TEXTO_SIZE 67 // Tamanho do texto base

int main(int argc, char *argv[]) {
  int rank, size;
  char texto_base[] = "abcdefghijklmnopqrstuvwxyz 1234567890 ABCDEFGHIJKLMNOPQRSTUVWXYZ";
  struct timeval tv;

  MPI_Init(&argc, &argv);
  MPI_Comm_rank(MPI_COMM_WORLD, &rank);
  MPI_Comm_size(MPI_COMM_WORLD, &size);

  if (size != 3) {
    if (rank == 0)
      printf("Este programa requer exatamente 3 processos.\n");
    MPI_Finalize();
    return 1;
  }
```

```
  int *indice = (int *)malloc(sizeof(int));
  *indice = 0;

  int local_max = MAX / size; // Quantidade de iteracoes para cada processo

  int number, tmp_index, i, cont = 0;
  while (cont < local_max) {
    gettimeofday(&tv, NULL);
    number = ((tv.tv_usec / 47) % 3) + 1;
    tmp_index = *indice;

    for (i = 0; i < number; i++) {
      if (!(tmp_index + i >= TEXTO_SIZE)) {
        fprintf(stderr, "%c", texto_base[tmp_index + i]);
      }
    }

    *indice = tmp_index + i;

    if (tmp_index + i >= TEXTO_SIZE) {
      fprintf(stderr, "\n");
      *indice = 0;
    }

    cont++;
  }
  printf("\n");
  free(indice);
  MPI_Finalize();
  return 0;
}
```

Comentário:

Questão 2

Incorreto

Atingiu 0,00 de 0,84

Analise as afirmativas e, a seguir, marque a alternativa correta:

I - Mecanismos que provêem interoperabilidade entre sistemas distintos pressupõem o uso de um sistema de mensageria e um protocolo de comunicação. Sem esses recursos não há como realizar a interoperabilidade citada

II - No grpc as aplicações que usam protobuf enviam dados em formato binário. Por isso, essas aplicações tem tempo de processamento melhor do que aplicações grpc que fazem uso de formatos como o JSON

III - Num diálogo http/2, se o cliente fizer uma solicitação de recurso para o servidor, este último pode enviar não só o recurso solicitado, mas vários outros associados (sem uma solicitação explícita) na mesma conexão . Essa característica difere o http/2 do http/1.1.

a. Apenas I está correta

b. Apenas II está correta

c. Nenhuma das alternativas satisfaz as afirmativas apresentadas

d. Apenas II e III estão corretas

e. Apenas III está correta ✖

Sua resposta está incorreta.

I - correto. No contexto citado, a interoperabilidade pressupõe um sistema de mensageria e um protocolo de comunicação entre as partes comunicantes.

II - correto.

III - correto.

A resposta correta é:
Nenhuma das alternativas satisfaz as afirmativas apresentadas

Questão 3

Correto

Atingiu 0,83 de 0,83

O código a seguir foi compilado com OpenMP e o binário tem o nome t1. Com base no código fonte, analise as afirmações e marque V para as verdadeiras e F para as falsas.

```
1   #include <stdio.h>
2   #include <omp.h>
3
4   int main(int argc, char *argv[]) {
5
6           int i=0;
7           #pragma omp parallel
8           {
9                   if (omp_get_thread_num() == 1)
10                          i=i+10;
11          }
12          printf("i=%d\n", i);
13          return 0;
14  }
```

I - A execução com o comando **OMP_NUM_THREADS=4 t1** vai imprimir o valor 40

II - Se a linha 9 for suprimida, o binário equivalente acionado com o comando **OMP_NUM_THREADS=3 t1** imprimirá sempre o valor 30

III - Se na linha 7 for acrescentada a declaração **private(i)** e houver supressão da linha 9, o binário equivalente acionado com o comando OMP_NUM_THREADS=6 t1, o programa vai imprimir 60

a. Apenas a afirmativa I está correta

b. Apenas as afirmativas II e III estão corretas

c. Apenas as afirmativas I e III estão corretas

d. Apenas as afirmativas I e II estão corretas

e. Nenhuma das opções apresentadas corresponde às afirmativas apresentadas ✔

Sua resposta está correta.

A resposta correta é:
Nenhuma das opções apresentadas corresponde às afirmativas apresentadas

Questão 4

Completo

Atingiu 2,50 de 2,50

O objetivo do código a seguir é fazer a impressão do texto repetidas vezes, em função da constante MAX e da variável number, que é calculada a cada iteração do while.

```c
#include <sys/time.h>
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
#define MAX 1000
int main (void) {
        char texto_base[] = "abcdefghijklmnopqrstuvwxyz 1234567890 ABCDEFGHIJKLMNOPQRSTUVWXYZ";
        /* a variavel indice aponta para o primeiro caracter do texto */
        int    *indice=(int *) malloc (sizeof(int));
        *indice = 0;
        struct timeval tv;
        int number, tmp_index, i, cont=0;
        while(cont<MAX) {
                gettimeofday (&tv, NULL );
                number = ((tv.tv_usec / 47) % 3) + 1;
                tmp_index = *indice;
                for (i = 0; i < number; i++ )
                        if( ! (tmp_index + i > sizeof(texto_base)) )
                                fprintf(stderr, "%c", texto_base[tmp_index + i]);
                *indice = tmp_index + i;
                if (tmp_index + i > sizeof(texto_base)) {
                        fprintf(stderr, "\n");
                        *indice = 0;
                } /* fim-if */
                cont++;
        } /* fim-while */
        printf("\n");
        return 0;
} /* fim-main */
```

Crie uma versão OpenMP deste código com apenas três threads, de modo que a lógica original se mantenha, mas a impressão do texto ocorra de modo colaborativo (com divisão de trabalho mais igual possível) entre as threads.

```c
#include <sys/time.h>
#include <stdio.h>
#include <stdlib.h>
#include <omp.h>

#define MAX 1000

int main(void) {
  char texto_base[] = "abcdefghijklmnopqrstuvwxyz 1234567890 ABCDEFGHIJKLMNOPQRSTUVWXYZ";
  int *indice = (int *)malloc(sizeof(int));
  *indice = 0;
  struct timeval tv;
  int number, tmp_index, i, cont = 0;

  #pragma omp parallel for num_threads(3) private(tv, number, tmp_index, i)
  for (cont = 0; cont < MAX; cont++) {
    #pragma omp critical
    {
      gettimeofday(&tv, NULL);
      number = ((tv.tv_usec / 47) % 3) + 1;
      tmp_index = *indice;
      for (i = 0; i < number; i++)
        if (!(tmp_index + i > sizeof(texto_base)))
```

```
            fprintf(stderr, "%c", texto_base[tmp_index + i]);
        *indice = tmp_index + i;
        if (tmp_index + i > sizeof(texto_base)) {
          fprintf(stderr, "\n");
          *indice = 0;
        }
      }
    }

    printf("\n");
    free(indice);
    return 0;
}
```

```
#include <sys/time.h>
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
#include <omp.h>
#define  MAX 1000
int         *indice;
char texto_base[] = "abcdefghijklmnopqrstuvwxyz 1234567890 ABCDEFGHIJKLMNOPQRSTUVWXYZ";
void imprimeTexto(void) {
     int tmp_index, i;
     struct timeval tv;
     int number;
     gettimeofday( &tv, NULL );
     number = ((tv.tv_usec / 47) % 3) + 1;

     tmp_index = *indice;
     for(i = 0; i < number; i++ )
          if( ! (tmp_index + i > sizeof(texto_base)) ) {
               fprintf(stderr, "%c", texto_base[tmp_index + i]);
               usleep(1);
          } /* fim-if */
     *indice = tmp_index + i;
     if( tmp_index + i > sizeof(texto_base) ) {
          fprintf(stderr, "\n");
          *indice = 0;
     } /* fim-if */
} /* fim-imprimeTexto */
int main() {
     indice = (int *) malloc(sizeof(int));
     *indice = 0;
     int cont=0;

     #pragma omp parallel num_threads(3)
     {
          printf("Thread %d iniciada...\n", omp_get_thread_num());
          sleep(1);
```

```
        /* Entrando no loop principal */
        while(cont<MAX) {
            #pragma omp critical
            imprimeTexto();
            #pragma omp atomic
            cont++;
        } /* fim-while */
    } /* fim-pragma */
    printf("\n");
    return 0;
} /* fim-main */
```

Comentário:

Questão 5

Correto

Atingiu 0,83 de 0,83

O código a seguir foi compilado com OpenMP e o binário tem o nome t1. Com base no código fonte, analise as afirmações e marque V para as verdadeiras e F para as falsas.

```
1   include <stdio.h>
2   #include <omp.h>
3   #include <string.h>
4   #define MAX 100
5
6   int main(int argc, char *argv[]) {
7
8           #pragma omp parallel
9           {
10                  int soma=0;
11                  #pragma omp for
12                  for (int i=0;i<MAX;i++) {
13                    soma +=omp_get_num_threads()*i;
14                  } /* fim-for */
15                  printf("Thread[%d] iterou %d vezes\n",
16                                  omp_get_thread_num(), soma);
17          } /* fim omp parallel */
18          return 0;
19  }
```

I - A execução com o comando **OMP_NUM_THREADS=4 t1** vai imprimir que cada thread foi executada 25 vezes
II - Se este programa for acionado tendo a variável OMP_NUM_THREADS um valor maior do que o número de núcleos da máquina, apenas as threads equivalentes ao número de núcleos serão criadas
III - Se o programa for executado numa máquina com 10 núcleos de processamento e a variável OMP_NUM_THREADS estiver com valor igual a 20, o programa não será ativado

a. Todas as afirmativas estão erradas ✔
b. Apenas a afirmativa I está correta
c. Apenas a afirmativa II está correta
d. Apenas a afirmativa III está correta
e. Apenas as afirmativas II e III estão corretas

Sua resposta está correta.

A resposta correta é:
Todas as afirmativas estão erradas

Questão 6

Incorreto

Atingiu 0,00 de 0,83

Julgue as afirmações abaixo:
I - Sockets UDP, por serem não orientados à conexão, permitem a comunicação persistente entre processos cliente e servidor
II - Sincronicidade é uma das funcionalidades atendidas pela biblioteca MPI, uma vez que esta garante a entrega da mensagem no receptor, mesmo que o processo destinatário não esteja executando
III - Brokers como Kafka e RabbitMQ são interessantes para viabilizar comunicação persistente entre processos

a. Apenas a afirmação I está correta
b. Apenas as afirmações I e III estão corretas
c. Todas as afirmações estão corretas
d. Apenas a afirmação III está correta
e. Apenas as afirmações II e III estão corretas ✖

Sua resposta está incorreta.

A resposta correta é:
Apenas a afirmação III está correta

Questão 7

Correto

Atingiu 0,83 de 0,83

Analise o código a seguir e responda o que se segue

```
1   #include <stdio.h>
2   #include <omp.h>
3   int main(){
4       int tid=0, nthreads=0;
5       printf("\nRegião serial (thread única)\n\n");
6       #pragma omp parallel
7       {
8           tid      = omp_get_thread_num();
9           nthreads = omp_get_num_threads();
10          printf("Região paralela (thread %d de %d threads)\n", tid, nthreads);
11      } /*fim-pragma */
12      printf("\nRegião serial (thread única)\n\n");
13      #pragma omp parallel num_threads(4)
14      {
15          tid = omp_get_thread_num();
16          nthreads = omp_get_num_threads();
17          printf("Região paralela (thread %d de %d threads)\n", tid, nthreads);
18      } /* fim-pragma */
19      printf("\nRegião serial (thread única)\n\n");
20      return 0;
21  } /* fim-main */
```

1. Se OMP_NUM_THREADS=6, na segunda região paralela desse código (linhas 13 a 18), serão geradas 10 threads e, portanto, 10 impressões (linha 17)
2. Se a linha 15 for movida para ficar fora da região paralela (entre as linhas 11 e 13), esse código passa a ser não compilável, pois não é possível saber o número de threads em uma região serial do código
3. Esse código é mais apropriado para funcionar em arquiteturas UMA (Uniform Memory Access) ou de memória compartilhada do que em arquiteturas NUMA (Non Uniform Memory Access)

a. Apenas a terceira afirmação está correta ✔

b. Apenas a segunda e a terceira afirmação está correta

c. Apenas a primeira e a terceira afirmação está correta

d. Nenhuma das alternativas apresentadas é válida

e. Apenas a primeira afirmação está correta

Sua resposta está correta.

A resposta correta é:
Apenas a terceira afirmação está correta

Questão 8

Incorreto

Atingiu 0,00 de 0,84

Julgue as afirmações abaixo e marque a alternativa correta:

I - RDDs (***Resilient Distributed Datasets***) são estruturas tipadas do Spark que podem ser alteradas por comandos python ou R

II - As transformações (***Transformations***) são implementadas no Spark em modo ***lazy***, ou seja, são executadas posteriormente, apenas quando é instanciado uma ação (***Action***), visando melhoria de performance

III - O Spark é mais rápido do que o Hadoop/Map-Reduce, porque os estágios de execução são implementados com uso intensivo de memória ao invés de uso de disco (memória secundária).

a. Apenas as alternativas I e II estão corretas

b. Apenas as alternativas I e III estão corretas

c. Apenas as alternativas II e III estão corretas

d. Todas as alternativas estão corretas

e. Apenas a alternativa II está correta ✖

Sua resposta está incorreta.

A resposta correta é:
Apenas as alternativas II e III estão corretas