Escalabilidade da multiplicação de matrizes

Usando o mecanismo de paralelização de aplicações com OpenMP, investigar a escalabilidade do programa de multiplicação de matrizes. Para tanto, deve-se ajustar o código para a paralização de apenas um dos loops do cálculo de cada vez.

```
for (i=0; i < nlin; i++)                              // loop externo
   for (j=0; j < ncol; j++) {.                        // loop intermediário
      c[i][j] = 0;
      // problema: somas parciais de c[i][j]....
      for (k=0; k < dim; k++).                        // loop interno
         c[i][j]= c[i][j] + a[i][k]*b[k][j];
   }
```

Para cada forma de paralelização, investigar os tempos envolvidos e como muda o resultado da aplicação quando são utilizados diferentes números de threads (1, 2, 4, 8, 16, 32, 64, 128).

Por fim, deve-se submeter a esta tarefa um arquivo com o código, os resultados dos testes e comentários sobre as estratégias e os ganhos de desempenho obtidos.

Como medir os tempos de execução? Vejam os exemplos (limits/consumo.c), ou usem to utilitário time (1) na ativação do programa.

O trabalho pode ser feito em duplas.

```
-----------------------------
/*
** PPD / DC/UFSCar - Helio
** Programa : multiplicacao de matrizes
** Objetivo: paralelizacao om OpenMP
*/

#include math.h
#include stdlib.h
#include string.h
#include stdio.h
#include unistd.h
#include time.h

float *A, *B, *C;

int
main(int argc, char *argv[])
{
  int lin_a,col_a,lin_b,col_b,lin_c,col_c;
  int i,j,k;
```

```
  printf("Linhas A: ");   scanf("%d",&lin_a);
  printf("Colunas A / Linhas B: "); scanf("%d",&col_a);
  lin_b=col_a;
  printf("Colunas B: ");  scanf("%d",&col_b);
  printf("\n");
  lin_c=lin_a;
  col_c=col_b;

  // Alocacao dinamica das matrizes, com linhas em sequencia
  A = (float *)malloc(lin_a*col_a*sizeof(float));
  B = (float *)malloc(lin_b*col_b*sizeof(float));
  C = (float *)malloc(lin_c*col_c*sizeof(float));

  // Atribucao de valores iniciais as matrizes
  srandom(time(NULL));

  for(i=0; i < lin_a * col_a; i++)
    A[i]=(float)rand() / (float)RAND_MAX;

  for(i=0; i < lin_b * col_b; i++)
    B[i]=(float)rand() / (float)RAND_MAX;

  // calculo da multiplicacao

  // Qual/quais loop(s) paralelizar? Vale a pena paralelizar todos?
  // Qual é o efeito de fazer um parallel for em cada um dos fors abaixo?
  // É necessários sincronizar alguma operação, garantindo exclusão mútua?

  for(i=0; i < lin_c; i++)

    for(j=0; j < col_c; j++) {

      C[i*col_c+j]=0;

      for(k=0; k < col_a; k++)
        C[i*col_c+j] = C[i*col_c+j] + A[i*col_a+k] * B[k*col_b+j];
    }

  return(0);
}
```