Programação Paralela e Distribuída
Programação Paralela com MPI – Modelo Fases Paralelas
Exercício Extra– *Odd-Even Transposition Sort*

O objetivo do trabalho é implementar, usando a biblioteca MPI, uma versão paralela usando o modelo fases paralelas do algoritmo de ordenação *Odd-Even Transposition Sort*. Após implementado, o programa deve ser executado no *cluster* para realização das medições de desempenho e, por fim, deve-se gerar o gráfico de *speed up*.

O algoritmo *Odd-Even Transposition Sort* funciona da seguinte forma: "O *Odd-Even Transposition Sort* é um algoritmo paralelo baseado nas etapas de comparação e troca do algoritmo *Bubble Sort*. O algoritmo funciona com duas fases alternadas: *odd* e *even*. Na fase *even*, os processos pares trocam números com seus vizinhos a sua direita, e realizam a ordenação local usando o seu próprio vetor juntamente com os valores trocados com o vizinho. Da mesma forma, na fase *odd*, os processos ímpares trocam números com seus vizinhos a sua direita e também realizam a ordenação local. Cada processo contém uma parte do vetor original com tamanho N/P, onde N é o número de valores do vetor e P o número de processos. A quantidade de valores trocados nas iterações influem consideravelmente no desempenho do algoritmo e podem variar de 1 até N/P. O algoritmo acaba quando, durante uma iteração, não modificarem os seus vetores locais."

```
ordenado = 0;
while(!ordenado) {
   ordenado = 1;
   for(i=1; i<n-1; i += 2)
      if(a[i] > a[i+1]){
        aux = a[i]; a[i] = a[i+1]; a[i+1] = aux;
        ordenado = 0;
      }
   for(i=0; i<n-1; i += 2)
      if(a[i] > a[i+1]){
         aux = a[i]; a[i] = a[i+1]; a[i+1] = aux;
         ordenado = 0;
      }
}
PS: se n for par então primeiro for the que ir até n-1, se n for impar então o
segundo for tem que ir até n-1
```

O programa deverá receber como parâmetros de entrada um número N, representando o número de valores a ser testado e o nome de um arquivo, que contém a lista de valores inteiros a serem ordenados. Deverão ser utilizados 3 casos de teste para realização das medições no *cluster*. A saída que deve ser gerada é a lista ordenada (crescente) dos valores de entrada (1 valor por linha) e também o tempo de execução da aplicação. Devem ainda ser testados os seguintes tamanhos para as trocas de valores entre os processos: 1, (N/P)/2 e N/P.

Os itens para avaliação são:

- implementação da versão paralela do algoritmo (*Odd-Even Transposition Sort*);
- medição dos tempos de execução para as versões sequencial e paralelas (usando 2, 4 e 8 processadores) para os 3 casos de teste;
- cálculo do *speed up* de cada versão paralela para cada caso de teste;
- geração do gráfico de *speed up* para cada versão paralela com os 3 casos de teste;
- otimização do código;
- clareza do código (utilização de comentários e nomes de variáveis adequadas).

PS: A nota do Trabalho 2 será uma composição das notas dos exercícios que serão realizados em aula. **O exercício pode ser feito em duplas.**