UNIVERSIDADE FEDERAL DE PELOTAS
Ciência da Computação
Disciplina: IPPD
Prova 1

Instruções:

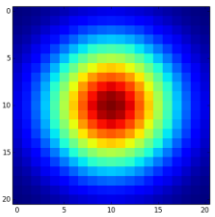
1. **Respostas:**
   a. As questões teóricas devem ser respondidas nos arquivos "questão_*n*.odt" (onde *n* corresponde ao número da questão).
   b. As questões práticas devem ser respondidas de forma isolada na respectiva pasta.
   c. As questões práticas são avaliadas considerando a exatidão e desempenho.
   d. Deve ser entregue uma versão executável de cada questão prática.
   e. Código que não compila não é avaliado.
2. **Quando tiver concluído a prova:**
   a. Gere um pacote "*aluno*.zip" (substitua "aluno" pelo seu nome) contendo todos os arquivos mais o executável de cada questão prática.
   b. Avise o professor que você está pronto para postar o arquivo na sua conta do moodle.
3. Somente é permitido o uso de código fornecido pelo professor.
4. Não conecte ao computador qualquer tipo de dispositivo!
5. Desligue o celular.
6. Não respeitar as instruções resultará em nota zero.
7. **SALVE CONSTANTEMENTE SEU TRABALHO!**

Questões Teóricas

Questão 1 (20%): Qual a diferença entre divisão de tarefas e divisão de dados? Dos três modelos de programação que vimos, como cada um se comporta com os dois tipos de divisão?
Questão 2 (25%): O que é o método de redução? Para que serve?
Questão 3 (25%): No livro "An Introduction to Parallel Programming" no capitulo 2 são discutidos vários exemplos de configurações de sistemas paralelos e suas vantagens e desvantagens. É discutida a principal dificuldade de hardware que leva a um limitador ao sistema de memória compartilhada. Diga qual é esse limitador e justifique.

Questões Práticas

Questão 4 (30%): Utilizando OpenMP, faça um programa que aplique um kernel sobre uma matriz repedidas vezes. Esse tipo de operação é utilizado para diversos cálculos, por exemplo, filtros em imagens e dissipação de calor em sólidos como na figura a baixo. Para cada **célula** da matriz de entrada (A) o kernel é centralizado sobre (amarelo) e seus pesos são aplicados sobre os valores. O novo valor da célula é a soma dos valores resultantes (B) da aplicação dos pesos do kernel. Esse processo é repetido (C) até que a energia seja totalmente dissipada ou se deseje parar a simulação. Geralmente o kernel não é aplicado na borda da matriz, como nos exemplos, para evitar o tratamento especial, faça isso. Faça um código que solicite um tamanho de matriz e comece a matriz toda com valores zero. Após, solicite as coordenadas de uma célula e um valor de entrada, veja A. Também solicite quantas repetições são desejadas. Essas entradas podem estar hardcoded, não precisa usar scanf, mas faça de forma que se possa alterar facilmente esses valores. Por fim, a cada passo, imprima a matriz (A,B e C). Toda a implementação deve ser o mais paralela e eficiente possível. Leia atentamente as instruções e figuras antes de pedir ajuda.



Matriz:

0	0	0	0	0
0	0	0	0	0
0	0	5	0	0
0	0	0	0	0
0	0	0	0	0

A

0	0	0	0	0
0	0.5	0.5	0.5	0
0	0.5	1	0.5	0
0	0.5	0.5	0.5	0
0	0	0	0	0

B

0	0	0	0	0
0	0.3	0.4	0.3	0
0	0.4	0.6	0.4	0
0	0.3	0.4	0.3	0
0	0	0	0	0

C

Kernel:

0.1	0.1	0.1
0.1	0.2	0.1
0.1	0.1	0.1