Universidade Federal de Minas Gerais
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Exercício de Programação 2 (EP2) - Programação Paralela

1. Comunicação Coletiva

Uma comunicação é denominada coletiva se todos os processos de um determinado grupo necessitam executá-la, com argumentos compatíveis, para que a mesma tenha efeito.

2. Enunciado

Deve-se implementar, utilizando-se MPI, uma árvore de redução para a soma em paralelo que simule tantos processos quanto elementos a serem somados. Dessa forma, cada processo MPI deverá simular diverso processos "virtuais". Essa estratégia tem como objetivo simular uma redução envolvendo um número grande de processos.

2.1. Fases da Implementação

O exercício é dividido em três etapas:

2.1.1. Distribuição dos Elementos

Os elementos a serem somados serão passados pela entrada padrão (*stdin*). A partir daí, deverá ser feita uma distribuição dos elementos a serem somados entre os processos.

2.1.2. Simulação da Árvore de Redução

Para configurar a árvore de redução, um dos operandos de cada soma deverá ser originado de outro processo. Dessa forma, os processos deverão trocar mensagens entre si somando esses valores recebidos àqueles já de posse do processo, até que cada um possua somente um elemento (Figura 1).

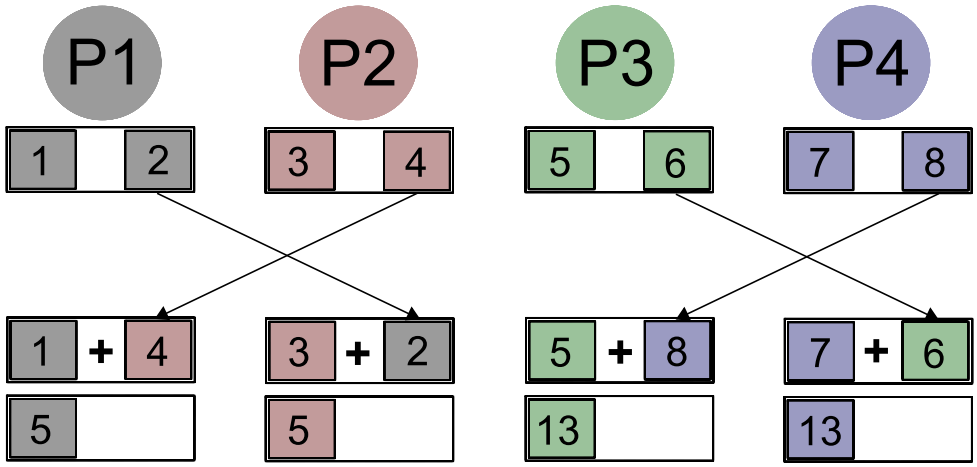

Figure 1. Exemplo do passo de simulação.

2.1.3. Árvore de Redução

Na árvore de redução, cada processo possui um único elemento e os mesmos deverão comunicar-se entre si de forma a acumular a soma em algum processo final (Figura 2).

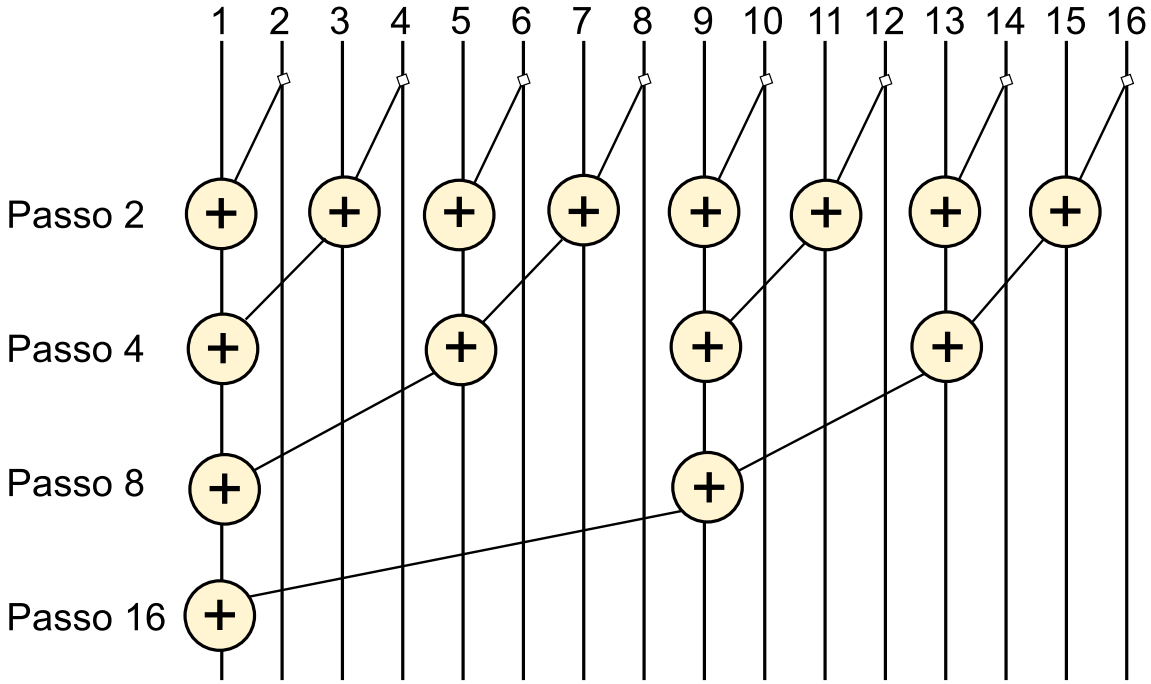

Figure 2. Exemplo de árvore de redução.

2.2. Restrições da Implementação

Deverão ser utilizadas **SOMENTE** funções `MPI_Send()` e `MPI_Recv()`. Não deverão ser utilizadas funções de comunicação coletiva (`MPI_Reduce()`, por exemplo).

2.3. Entrada do Programa

O programa deverá ser executado via *mpiexec/mpirun* passando como parâmetro o número de processos a serem utilizados. Após a chamada, serão passados, no início da execução, uma string s que representará os tipos de saída ("*sum*" para imprimir somente a soma, "*time*"para imprimir somente o tempo e "*all*" para imprimir a soma na primeira linha e o tempo na segunda linha), um inteiro n com a quantidade de números a serem somados e n números v (tipo *float*) representando os valores a serem somados.

2.4. Saída do Programa

O programa deverá imprimir a soma com 2 casas decimais e o tempo em milissegundos da Simulação da Árvore de Redução e da Árvore de Redução. O resultado deverá ser impresso na saída padrão (stdout) no seguinte o formato, de acordo com a string s de entrada:

- "sum": Uma linha contendo a soma com duas casas decimais.
- "$time$": Uma linha contendo o tempo de execução em milissegundos.
- "all": A primeira linha contendo a soma com duas casas decimais e e a segunda linha com o tempo de execução em milissegundos.

Após a exibição, acrescente uma quebra de linha - `printf("\n")` - na saída.

3. Exemplos de Execução

A Tabela 1 possui alguns exemplos de execução, de acordo com os parâmetros exigidos na entrada e saída do programa.

Table 1. Tabela com exemplos de execução

Nr	Exemplo de entrada	Exemplo de Saída
01	mpiexec -n 2 prog sum 8 0.0 4.4 4.8 8.15 15.16 16.23 23.42 42.42	114.88
01	mpiexec -n 4 prog time 8 0.1 1.2 3.5 8.13 21.34 55.89 14.42 33.37	0
01	mpiexec -n 4 prog all 4 2.7 18.28 18.28 45.9	85.16 0
04	mpiexec -n 32 prog < dados.txt	1.61 142857

4. Observação

Implementar a árvore de redução para elementos e processos que não necessariamente sejam potência de 2.