UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
DEPARTAMENTO DE ENGENHARIA DE COMPUTAÇÃO E AUTOMAÇÃO
Programação Concorrente e Distribuida
Aluno: Igor Macedo Silva

As tabelas deste documento mostram as métricas de desempenho para alguns testes e comparações entre algoritmo paralelos distribuidos implementados em MPI. Todos os tempos mostrados nas tabelas são a média de resultados de 40 execuções do programa em estudo. Com os tempos obtidos, foram calculadas as métricas de speedup e eficiência de acordo com as fórmulas estudadas no livro Introduction to Parallel Programming de Peter Pacheco.

A Tabela 1 de Desempenho mostra os tempos e métricas de desempenho obtidas a partir da execução do algoritmo de integração numérica pelo método do trapézio utilizando a implementação nativa do MPI_Reduce. Neste caso, apenas a execução da função MPI_Reduce está sendo cronometrada.

É possível notar que os tempos de execução da função aumentam a medida que o número de processos cresce. Isso acontece pela característica de funcionamento da própria função, já que mais processos implica necessariamente em mais comunicação e troca de informacão entre os processos, justificando o aumento do tempo.

Como consequência desse aumento, vemos que o speedup e a eficiência decrescem rapidamente também.

Tabela 1 de Desempenho - MPI_Reduce

Time	1	2	4	8	16
2^20	9.54E-08	7.03E-07	2.40E-06	4.26E-05	1.68E-04
2^21	7.75E-08	8.46E-07	2.86E-06	7.36E-05	3.75E-05
2^22	1.19E-07	6.08E-07	1.96E-06	8.14E-06	6.84E-05
2^23	1.31E-07	6.44E-07	2.19E-06	9.82E-06	8.68E-05
2^24	4.29E-07	8.46E-07	2.21E-06	3.61E-06	6.27E-05

speedup	1	2	4	8	16
2^20	100.00%	13.56%	3.97%	0.22%	0.06%
2^21	100.00%	9.15%	2.71%	0.11%	0.21%
2^22	100.00%	19.61%	6.08%	1.46%	0.17%
2^23	100.00%	20.37%	5.98%	1.33%	0.15%
2^24	100.00%	50.70%	19.41%	11.90%	0.68%

eficiency	1	2	4	8	16
2^20	100.00%	6.78%	0.99%	0.03%	0.00%
2^21	100.00%	4.58%	0.68%	0.01%	0.01%
2^22	100.00%	9.80%	1.52%	0.18%	0.01%
2^23	100.00%	10.19%	1.49%	0.17%	0.01%
2^24	100.00%	25.35%	4.85%	1.49%	0.04%

A Tabela 2 de Desempenho mostra os tempos e métricas de desempenho obtidas a partir da execução do algoritmo de integração numérica pelo método do trapézio utilizando a implementação uma implementação de soma em árvore ponto a ponto. Neste caso, essa implementação da soma está sendo cronometrada.

Assim como na Tabela anterior, vemos que os tempos de execução crescem a medida que o número de processos aumenta, como esperado, já que o número de mensagens também cresce em conjunto. E como resultado, temos que o tanto o speedup quanto a eficiência apresentam um comportamento pouco convencional, mas também esperado, pois as tarefas não são feitas de forma completamente paralela, para as quais essas métricas foram criadas.

Comparando os tempos mostrados na Tabela 1 com os da Tabela 2, o MPI_Reduce possui maior desempenho quando o número de processos é igual a 1 ou 2, possui desempenho semelhante à implementação de soma em árvore quando o número de processos é 4, e possui desempenho um pouco abaixo da nossa implementação à medida que o número de processos cresce em 8 e 16. Para justificar esse ganho, supomos que o nosso algoritmo, apesar de não ser ótimo, é uma das implementações mais simples possíveis e diretas, o que favorece quando existem um maior número de processos para enviar os dados.

Tabela 2 de Desempenho – Adição em Árvore

Time	1	2	4	8	16
2^20	2.03E-07	1.03E-06	2.00E-06	5.71E-06	6.84E-05
2^21	2.21E-07	1.00E-06	2.26E-06	1.05E-05	2.64E-04
2^22	1.49E-07	1.31E-06	1.72E-06	3.62E-05	8.00E-05
2^23	1.79E-07	1.19E-06	1.76E-06	2.23E-05	2.00E-04
2^24	2.98E-07	7.75E-07	1.81E-06	4.68E-05	3.32E-04

speedup	1	2	4	8	16
2^20	100.00%	19.65%	10.12%	3.55%	0.30%
2^21	100.00%	22.02%	9.76%	2.11%	0.08%
2^22	100.00%	11.42%	8.65%	0.41%	0.19%
2^23	100.00%	15.00%	10.14%	0.80%	0.09%
2^24	100.00%	38.46%	16.45%	0.64%	0.09%

eficiency	1	2	4	8	16
2^20	100.00%	9.83%	2.53%	0.44%	0.02%
2^21	100.00%	11.01%	2.44%	0.26%	0.01%
2^22	100.00%	5.71%	2.16%	0.05%	0.01%
2^23	100.00%	7.50%	2.53%	0.10%	0.01%
2^24	100.00%	19.23%	4.11%	0.08%	0.01%

A Tabela 3 de Desempenho mostra os tempos e métricas de desempenho obtidas a partir da execução do algoritmo de multiplicação de uma matriz por um vetor de forma a mutilplicar linha com coluna. Neste caso, apenas a implementação da multiplicação está sendo cronometrada.

Analisando os tempos obtidos, vemos que para o número de processos igual a 1,2 e 4, o tempo tende a cair, porém ao chegar em 8 e 16 esses tempos crescem novamente, sendo essa uma tendência que acontece independentemente do tamanho do problema. O speedup segue essa tendência de crescimento, porém com o número de processos igual a 16 cai drasticamente por conta do aumento do tempo de execução. Observando a eficiência, ela cai rapidamente quando o tamanho do problema é menor, mas melhora a medida que o tamanho do problema aumenta, sendo essa uma tendência que vale a pena ser melhor investigada para entender o quão bom pode se tornar.

Vemos que o speedup é próximo do linear, seguindo a diagonal principal, para p= 1,2,4,

mas perdemos esse comportamento em p=8 e 16.

Vemos que em um momento, temos uma eficiência maior que 100%. Isso pode ter sido ocasionado pela superação de recursos limitantes, mas também pode ter acontecido pelo melhor gerenciamento do processo se este estiver competindo com outros processos no sistema operacional.

Tabela 3 de Desempenho – Mutiplicação em Linhas

time		1	2	4	8	16
	128	5.89E-05	4.46E-05	3.84E-05	2.86E-05	1.55E-04
	256	2.38E-04	1.20E-04	1.11E-04	9.44E-05	3.73E-04
	512	8.96E-04	7.28E-04	3.11E-04	2.88E-04	5.31E-04
	1024	3.48E-03	1.92E-03	9.25E-04	1.19E-03	1.61E-03
	2048	1.42E-02	6.92E-03	3.76E-03	3.93E-003	8.56E-03

speedup		1	2	4	8	16
	128	100.00%	132.00%	153.61%	205.96%	37.97%
	256	100.00%	198.76%	214.89%	251.83%	63.85%
	512	100.00%	123.09%	287.76%	311.11%	168.90%
	1024	100.00%	180.73%	376.20%	292.41%	216.54%
	2048	100.00%	205.35%	378.25%	361.67%	166.03%

efficiency		1	2	4	8	16
	128	100.00%	66.00%	38.40%	25.74%	2.37%
	256	100.00%	99.38%	53.72%	31.48%	3.99%
	512	100.00%	61.54%	71.94%	38.89%	10.56%
	1024	100.00%	90.36%	94.05%	36.55%	13.53%
	2048	100.00%	102.68%	94.56%	45.21%	10.38%

A Tabela 4 de Desempenho mostra os tempos e métricas de desempenho obtidas a partir da execução do algoritmo de multiplicação de uma matriz por um vetor de forma a mutilplicar colunas com os elementos do vetor e uma posterior redução em soma. Neste caso, apenas a implementação da multiplicação e a redução está sendo cronometrada.

Ao observar a tabela vemos que os tempos tem um comportamento semelhante aos da tabela anterior, por diminuiem em p= 1,2,4, mas crescem em p=8,16. E, como esperado, para um mesmo p, e maior tamanho do problema, o tempo de execução também é maior. Observando o speedup, vemos que ele inicia bem, próximo do linear, mas se distancia do speedup linear a medida que o tamanho do problema e p são dobrados.

Considerando a eficiência, vemos que ela melhora e consegue se manter mais constante quando o tamanho do problema é maior e, além disso, apresentou em alguns momentos valores maiores que 100%, que podem ter explicações como comentados nos resultados da Tabela 3.

Tabela 4 de Desempenho – Mutiplicação em Colunas

time		1	2	4	8	16
	128	1.52E-04	9.05E-05	4.86E-05	9.50E-05	1.21E-04
	256	5.82E-04	3.02E-04	2.43E-04	3.02E-04	2.67E-04
	512	2.32E-03	1.26E-03	1.02E-03	9.62E-04	8.40E-04
	1024	1.32E-02	4.98E-03	2.82E-03	2.84E-03	4.18E-03
	2048	5.12E-02	2.68E-02	1.37E-02	1.14E-002	1.57E-02

speedup		1	2	4	8	16
	128	100.00%	168.31%	313.08%	160.20%	125.40%
	256	100.00%	192.37%	239.33%	192.95%	218.08%
	512	100.00%	184.60%	226.39%	240.86%	275.90%
	1024	100.00%	265.59%	469.05%	464.54%	316.48%
	2048	100.00%	190.84%	374.63%	447.67%	325.23%

efficiency		1	2	4	8	16
	128	100.00%	84.15%	78.27%	20.03%	7.84%
	256	100.00%	96.19%	59.83%	24.12%	13.63%
	512	100.00%	92.30%	56.60%	30.11%	17.24%
	1024	100.00%	132.80%	117.26%	58.07%	19.78%
	2048	100.00%	95.42%	93.66%	55.96%	20.33%