INFLUÊNCIA DO ORDENAMENTO DAS COMUNICAÇÕES ENTRE PROCESSADORES DE UM CLUSTER NA EFICIÊNCIA DA IMPLEMENTAÇÃO PARALELA DE UM CÓDIGO PARA SIMULAÇÃO DE ESCOAMENTO DE FLUIDOS USANDO O MÉTODO DOS ELEMENTOS FINITOS

Carla Trindade Scherer carlatscherer@gmail.com
Orientador: João Ricardo Masuero joao.masuero@ufrgs.br

Centro de Mecânica Aplicada e Computacional - CEMACOM
Curso de Engenharia Civil da Universidade Federal do Rio Grande do Sul, Porto Alegre, Brasil.

Por que computação paralela?

Os modelos numéricos empregados nos problemas atuais de Engenharia implicam na solução de sistemas com milhões de equações, o que demanda uma elevada capacidade computacional e quantidade de memória. A utilização de vários processadores de menor desempenho em conjunto, conhecida como computação paralela, mostra-se como uma solução para este problema, criando uma unidade de alto desempenho, chamada de *cluster*, de forma relativamente simples.

Paralelismo em memória distribuída

No paralelismo em memória distribuída, cada processador tem acesso direto somente à memória a ele alocada, e o acesso aos dados de outros processadores é feito através de uma rede de comunicação, que em geral se constitui em gargalo para o desempenho do processamento.

A forma como é feita a divisão das tarefas para cada processador é decisiva na eficiência do processo, sendo feita geralmente através de técnicas de partição de domínio desenvolvidas com o objetivo de diminuir o número de processadores que se comunicam, a quantidade de dados em comunicação entre eles e a redundância do esforço computacional. Porém, há a possibilidade de haver comunicação de cada um dos processadores com todos os outros, o que cria a necessidade de se desenvolver um algoritmo de comunicação eficiente para todos os casos.

A figura abaixo mostra o escoamento supersônico em torno de um modelo de avião, um dos exemplos utilizados como base para testes de desempenho de algoritmos paralelos no cluster do CEMACOM/UFRGS. A malha de elementos finitos têm 1,5 milhões de elementos tetraédricos e 270 mil nós (1,36 milhões de equações).

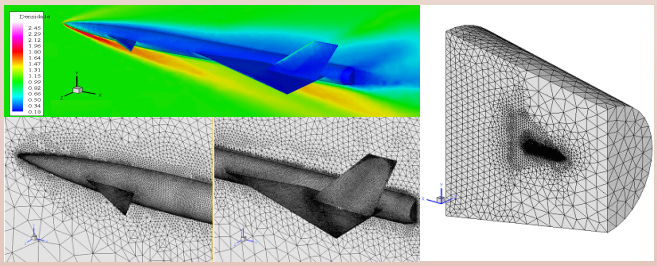

Figura 1: Visão geral do problema de escoamento supersônico em torno de um modelo de avião. Distribuição de densidade e malha utilizada no modelo de Elementos Finitos.

Na tabela abaixo estão os dados de tráfego na rede, em milhares de nós, para a malha dividida sem nenhum tratamento, seguindo apenas a ordenação dos nós na geração da mesma.

n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1		2.8	1.3	3.4	3.8	3.6	4.1	5.5	5.3	5.3	4.4	5.0	5.6	5.8	4.3	4.3
2	3.5		3.2	4.2	5.0	5.8	6.0	5.4	5.0	5.4	6.7	7.3	6.8	6.5	5.2	6.7
3	1.7	3.0		4.8	3.8	2.9	3.1	2.6	2.3	2.4	1.6	1.4	1.7	1.8	4.4	5.0
4	9.8	11.5	4.9		7.3	6.1	5.5	5.0	3.8	3.4	4.4	3.2	2.3	1.6	1.3	1.5
5	10.1	11.9	3.8	7.5		6.7	5.9	5.4	4.3	4.1	4.8	3.7	2.8	2.0	1.2	1.7
6	8.7	13.0	3.5	6.1	6.8		6.2	5.9	4.8	5.0	5.0	3.9	3.1	2.7	1.9	2.1
7	9.3	12.4	3.5	5.5	6.1	6.3		7.0	5.7	5.2	4.7	4.0	3.5	2.8	2.2	2.2
8	12.2	10.3	3.0	5.0	5.3	5.8	7.1		6.5	5.4	5.2	4.7	4.0	3.5	2.6	2.4
9	11.4	9.3	3.0	3.6	3.9	4.6	5.6	6.4		6.2	4.8	4.8	4.6	4.5	3.3	2.3
10	11.3	9.6	2.8	3.1	3.8	4.6	5.0	5.3	6.3		5.2	5.3	5.2	5.3	4.0	2.9
11	9.7	12.9	1.7	4.3	4.6	5.0	4.8	5.4	5.2	5.4		9.8	7.4	5.5	2.7	3.3
12	10.8	12.2	1.6	3.0	3.5	3.7	3.8	4.6	4.7	5.4	9.4		9.8	7.9	3.9	4.0
13	11.4	11.3	1.9	2.2	2.7	2.9	3.3	3.8	4.4	5.0	7.4	9.7		10.3	5.7	4.8
14	11.8	10.7	2.0	1.5	1.9	2.6	2.6	3.5	4.3	5.3	5.6	7.9	10.3		9.0	6.1
15	7.2	7.5	7.3	1.2	1.1	1.8	2.0	2.4	3.0	3.7	2.6	3.7	5.2	7.5		7.4
16	7.9	11.4	5.5	1.6	1.9	2.4	2.7	3.0	2.9	3.5	3.6	4.5	5.6	7.0	8.5	
Nós	137	147	48	54	58	61	64	66	63	66	71	74	72	69	56	52
					Total de nós comunicados entre processadores:										1156	
					Índice de troca de dados:										425%	

Tabela 1: Comunicação de dados entre processadores. Malha sem tratamento.

Nesta configuração de comunicação, onde cada processador troca dados com todos os outros, a ordem das operações de comunicação é fundamental para que não sejam criadas filas de espera no processo de comunicação. O objetivo do desenvolvimento de um algoritmo de comunicação é tentar otimizar este processo, de modo que a comunicação seja feita no menor número de etapas possível.

A comunicação ideal

Para um conjunto de *n* processadores em que cada um deles precisa fazer uma troca de dados com todos os outros, e que somente seja possível a cada processador trocar dados com apenas um outro processador a cada instante, o número mínimo de etapas para completar todo o processo de comunicação é de *n*-1 quando *n* é par, e *n* etapas quando *n* é ímpar.

O Algoritmo

O algoritmo está baseado em uma Tabela de Incrementos, na qual, para cada processador (coluna) a tabela indica para cada etapa (linha) o incremento que deve ser somado ao número do processador para definir o outro processador com o qual será estabelecida uma operação de comunicação.

Sejam *n* processadores. Divide-se *n* por 2, arredondando o resultado R, se necessário, para o número inteiro inferior. A tabela de incrementos apresentará, em cada coluna, números de R a –R para *n* ímpar e valores de R a –(R-1) para *n* par, variando-os dentro deste ciclo de 2 em 2 na linha e de 1 em 1 na coluna, sempre em ordem decrescente (considera-se a seqüência circular, ou seja, -R é seguido por R, R-1, etc.). Quando há um incremento de valor nulo, o mesmo é ignorado, e o processador avança para o incremento seguinte na coluna, ficando em espera para completar a comunicação até a etapa seguinte.

Este processo é melhor compreendido através da observação de um exemplo de tabela de incrementos e, a seguir, o detalhamento das comunicações de cada etapa (processadores sombreados correspondem à estados de espera ou conclusão do processo de comunicação).

n	1	2	3	4	5	6	7
Etapa 1	3	1	-1	-3	2	0	-2
Etapa 2	2	0	-2	3	1	-1	-3
Etapa 3	1	-1	-3	2	0	-2	3
Etapa 4	0	-2	3	1	-1	-3	2
Etapa 5	-1	-3	2	0	-2	3	1
Etapa 6	-2	3	1	-1	-3	2	0
Etapa 7	-3	2	0	-2	3	1	-1

Tabela 2: Tabela de incrementos de comunicação entre 7 processadores.

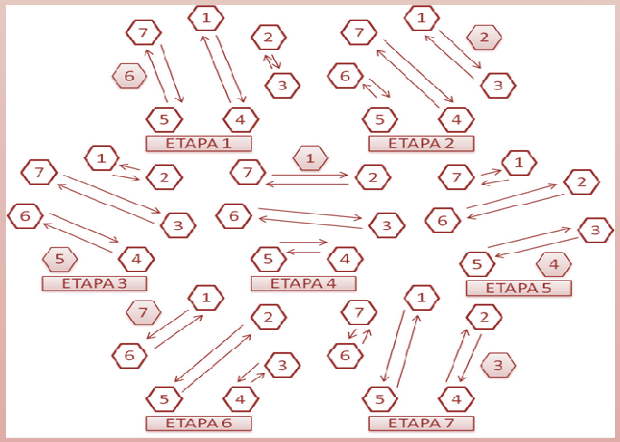

Figura2: Detalhamento de cata etapa de comunicação de um ciclo para 7 processadores.

Conclusões

O algoritmo consegue completar o ciclo de comunicações para qualquer número *n* de processadores sempre em *n* etapas, o que é o número mínimo para *n* impar e têm apenas uma etapa de penalidade para *n* par. Além disso, sua lei de formação é extremamente simples e de fácil implementação computacional. Vale destacar que o ciclo em *n*-1 etapas para *n* par de processadores é possível de ser feito, porém, não se conseguiu estabelecer nenhuma lógica de implementação ou lei de formação.