Instituto Tecnológico de Aeronáutica

CE-265: Processamento Paralelo

Exercício 11: Modelo computacional para Big Data

Bruno Benjamim Bertucci - Turma 23.2

Tarefa 1

A primeira década do século XXI foi marcada por um acelerado desenvolvimento tecnológico, sobretudo na área de computação, impulsionada pelo avanço da internet. Com isso, a coleta e o processamento de grandes quantidades de dados se tornou cada vez uma necessidade maior, criando, dessa forma, a demanda por métodos escaláveis, eficientes, e de fácil uso para essas tarefas.

Por mais que houvessem métodos já existentes para distribuir computações complexas entre múltiplas unidades de processamento, como paralelismo de *threads* com *OpenMP* e de processos com *MPI*, essas soluções exigem um grande esforço por parte do programador, muitas vezes necessitando que mudanças dramáticas fossem feitas no algoritmo para possibilitar o processamento paralelo.

Nesse contexto, havia uma demanda considerável por uma solução de distribuição da computação que desse conta, de forma autônoma, da separação das tarefas, da atribuição equilibrada delas às unidades de processamento disponíveis, do gerenciamento de falhas nessas unidades sem perda de dados, e da concentração dos resultados de cada computação em um resultado final.

Foi nesse cenário que, em 2004, os pesquisadores Jeffrey Dean e Sanjay Ghemawat, da empresa *Google*, apresentaram o modelo de computação distribuída denominado *MapReduce*. O modelo essencialmente permite a distribuição de um processamento grande em processos menores e paralelos, enviando-os via conexão de internet a um determinado número de computadores disponíveis para realizarem a computação.

O modelo, que já estava sendo testado internamente na empresa, foi dado como um sucesso, pois satisfez às demandas citadas anteriormente. Primeiramente, era fácil de adaptar um programa para funcionar com o modelo, necessitando essencialmente a criação de duas funções, uma para preparar a partição da computação, na etapa denominada *Map*, e outra para concentrar os dados processados, na etapa denominada *Reduce*.

Além disso, o modelo lida automaticamente com tolerância de falhas. São criadas várias cópias de um mesmo bloco de dados, de tal forma que, caso uma das máquinas que recebem dados falharem, seja por defeitos da máquina ou de conectividade, há outras cópias distribuídas em outras máquinas, prontas para executarem o processamento que foi perdido. Os detalhes minuciosos da paralelização, como a manutenção do equilíbrio do trabalho enviado às máquinas, também são atribuições autônomas de uma instância especial do programa, denominada *master*, facilitando o trabalho do programador.

Finalmente, esse modelo, por distribuir trabalhos para computadores independentes via internet, é facilmente escalável, permitindo o emprego de milhares de máquinas simultaneamente, o que foi muito pertinente para as crescentes demandas de processamento de dados em massa da época. Por esses motivos, o modelo foi um marco importante no desenvolvimento da computação distribuída.

Tarefa 2

O processamento de uma multiplicação de duas matrizes pode ser adaptado ao modelo *MapReduce* usando o seguinte procedimento:

Primeiramente, as matrizes, tomando como exemplo de tamanhos 3x3, vai ser dividida nas combinações possíveis das linhas da primeira com as colunas da segunda, ou seja, a operação *Map* será feita da seguinte forma:

$$\texttt{map:} \quad (k_1, (M_1, M_2)) \rightarrow \texttt{list}(k_2, (l_1, c_2))$$

Onde M_1, M_2 são as matrizes, e $l1, c2$ são pares formados por uma linha da matriz 1 e uma coluna da matriz 2.

Após a distribuição desses pares serem feitos, cada unidade de processamento irá realizar a multiplicação dos elementos da sua linha pelos elementos da sua coluna, concentrando essas multiplicações em uma soma, cujo valor será equivalente a um elemento da matriz multiplicada. Por exemplo, para a linha 1 e coluna 2 do exemplo adotado, temos que

$$(k2, (l_1, c_2)) \rightarrow (k2, v_2)$$

Onde

$$v_2 = l_1[0] * c_2[0] + l_1[1] * c_2[1] + l_1[2] * c_2[2]$$

Ao final, o conjunto desses resultados calculados formam a matriz multiplicada, como ilustrado abaixo:

$$\texttt{reduce:} \quad (k_2, v_2) \rightarrow list(v_2) = M_3$$

Dessa forma, é possível aproveitar-se do modelo *MapReduce* para acelerar, através da computação distribuída, o cálculo de multiplicação de matrizes.

Em vez de enviar somente uma linha e uma coluna para cada máquina calcular um único elemento da matriz multiplicada, também é possível enviar conjuntos consecutivos de linhas e de colunas, possibilitando o cálculo de múltiplos elementos vizinhos em cada máquina do sistema de computação distribuído.