UNIVERSIDADE POSITIVO

Pós-Graduação – Big Data
Disciplina: MapReduce e Spark
Prof. Alessandro Binhara

Por: ***Cristiane Fagundes***
Tiyomi Nakaba

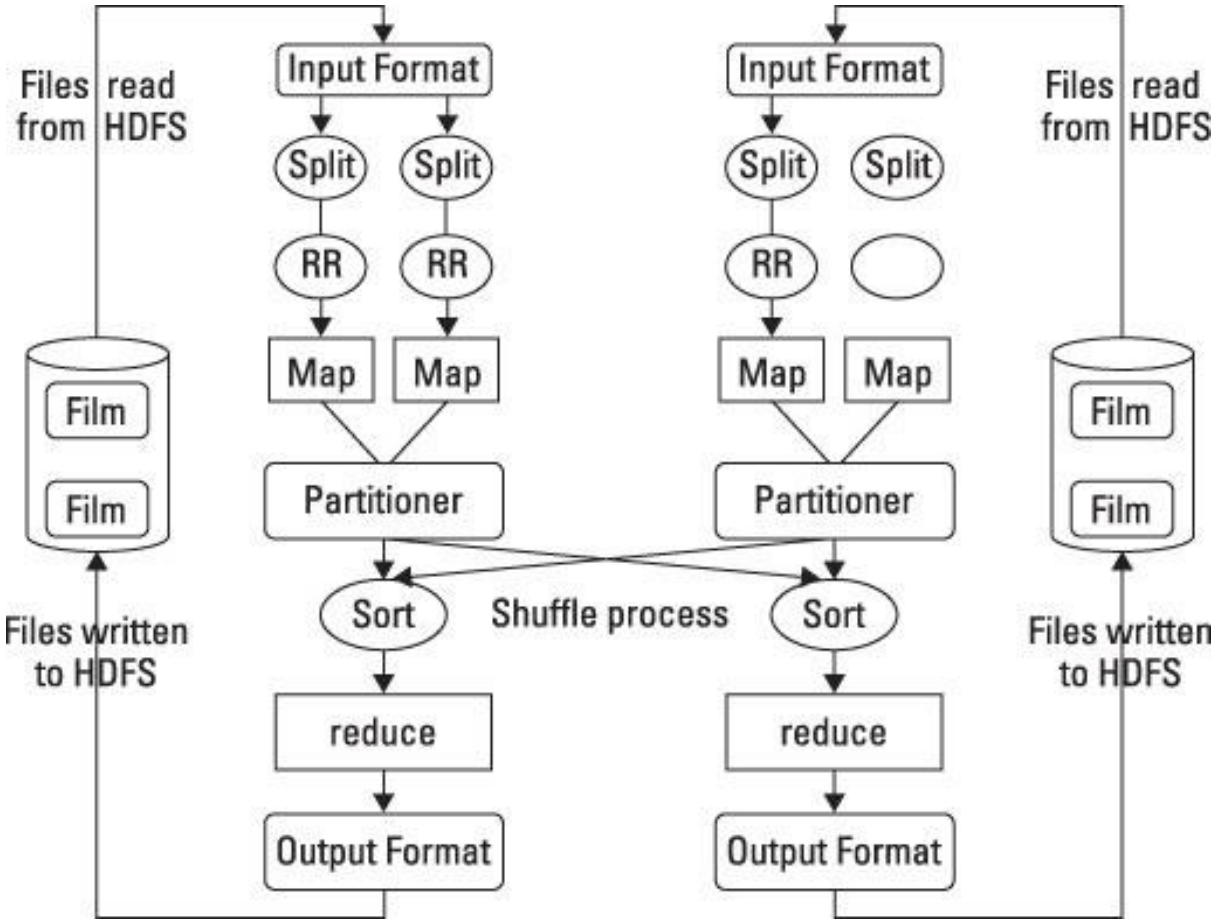
Uma tradução do material disponibilizado em www.dummies.com sobre o algoritmo MapReduce no Apache Hadoop: *http://www.dummies.com/programming/big-data/hadoop/hadoop-mapreduce-for-big-data/*

HADOOP MAPREDUCE PARA BIG DATA

Para entender completamente as capacidades do Hadoop MapReduce, é importante diferenciar o algoritmo *MapReduce* da sua implementação. Hadoop MapReduce é a implementação do algoritmo desenvolvido e mantido pela plataforma Apache Hadoop.

É útil pensar sobre esta implementação como um motor MapReduce, porque isso é exatamente como ele funciona. Você fornece entrada (combustível), o mecanismo converte a entrada em saída de forma rápida e eficiente, e você obtém as respostas de que precisa.

Hadoop MapReduce inclui vários estágios, cada um com um conjunto importante de operações que ajudam a atingir o objetivo de obter as respostas que você precisa extrair de grandes dados. O processo começa com uma solicitação do usuário para executar um programa *MapReduce* e continua até que os resultados sejam gravados de volta para o HDFS.



HDFS e MapReduce executam seu trabalho em nós em um cluster hospedado em racks de servidores de comodidade. Para simplificar a discussão, o diagrama mostra apenas dois nós.

OBTENHA O BIG DATA CORRETAMENTE

Quando um cliente solicita um programa *MapReduce* para executar, a primeira etapa é localizar e ler o arquivo de entrada contendo os dados brutos. O formato do arquivo é completamente arbitrário, mas os dados devem ser convertidos em algo que o programa possa processar. Estas são as funções *InputFormat*, do Formato de Entrada, e *RecorderReader*, de Leitor de Registro. O *InputFormat* decide de que maneira o arquivo será quebrado em pedaços usando uma função chamada *InputSplit*, Divisão de Entrada.

Em seguida, atribui um *RecorderReader* para transformar os dados brutos em dados prontos para o mapeamento. Vários tipos de *RecorderReader* são fornecidos com o Hadoop, oferecendo uma ampla variedade de opções de conversão. Este recurso é uma das maneiras pelas quais o Hadoop gerencia a enorme variedade de tipos de dados encontrados em grandes problemas de dados.

INICIE O MAPEAMENTO DO BIG DATA

Agora seus dados estão em um formato aceitável para mapear. Para cada par de entradas, uma instância distinta de *Map* é chamada para processar os dados. Mas o que ele faz com a saída processada e como você pode acompanhar isso?
A função *Map* tem duas capacidades adicionais para resolver essas questões. Porque o *Map* e o *Reduce* precisam trabalhar em conjunto. Isto porque para processar seus dados, o programa precisa coletar a saída dos mapeadores independentes e passá-los para os redutores. Essa tarefa é executada por um *OutputCollector* (Coletor de Saída). A função *Reporter* fornece informações sobre o andamento das tarefas de mapeamento, se foram concluídas ou quando estarão prontas.

Todo este trabalho é realizado em múltiplos nós no cluster Hadoop simultaneamente. Você pode ter casos em que a saída de determinados processos de mapeamento precisa ser acumulada antes que os redutores possam ser executados. Ainda, há casos em que os resultados intermediários devem ser processados antes da redução.

Além disso, parte dessa saída pode estar em um nó diferente do nó onde os redutores para essa saída específica serão executados. O agrupar e arrastar de resultados intermediários são realizados por um particionador e um classificador. As tarefas de mapeamento entregarão os resultados para uma partição específica que servirão de entrada para as tarefas de redução.

Depois que todas as tarefas do mapeamento estiverem concluídas, os resultados intermediários são agrupados em partição e então ocorre o arrastamento, classificando a saída para proporcionar um processamento otimizado de redução.

REDUZA E COMBINE O BIG DATA

Para cada par de saída, a função *Reduce* é chamada para executar sua tarefa. Da mesma forma que *MapReduce* reúne sua saída enquanto todas as tarefas estão processando. O comando *Reduce* não pode iniciar o processamento até que todo o mapeamento esteja concluído. A saída do *Reduce* também é uma chave e um valor. Embora isso seja necessário para o *Reduce* fazer seu trabalho, ele pode não ser o formato de saída mais eficaz para sua aplicação.

O Hadoop fornece um recurso para Formato de Saída, o *OutputFormat*, ele funciona como o *InputFormat*. O *OutputFormat* pega o par chave-valor e organiza a saída para gravação em HDFS. A última tarefa é realmente escrever os dados para o HDFS. Isso é executado pelo Gravador, *RecorderWriter*, e ele funciona como o RecorderReader, só que no sentido contrário. O comando *OutputFormat*, Formato de Saída, libera o resultado no HDFS no formato requerido pelo programa.

Em versões anteriores do Hadoop, a coordenação de todas essas atividades era gerenciada por um planejador de tarefas que era rudimentar, mas à medida que a

interação de atividades foi sofrendo mudanças e crescendo, era claro que uma abordagem diferente era necessária. A principal deficiência era a falta do gerenciamento de recursos. A versão mais recente do Hadoop absorve essa capacidade.

Hadoop MapReduce é o coração do sistema Hadoop. Ele fornece todos os recursos necessários para quebrar grandes dados em blocos gerenciáveis, processar os dados em paralelo em seu cluster distribuído e, em seguida, disponibilizar os dados para consumo do usuário ou processamento adicional. E tudo isso funciona de uma forma altamente resiliente, tolerante a falhas, e isto é apenas o começo.

Fonte:
Por Judith Hurwitz, Alan Nugent, Fern Halper, Marcia Kaufman
http://www.dummies.com/programming/big-data/hadoop/hadoop-mapreduce-for-big-data/