≪コラム 1 ≫

Hadoopにおける新しいプログラミング

― 並列、分散、関数型～プログラミングの新しい潮流 ―

日本アイ・ビー・エム
システムズ・エンジニアリング株式会社
インテグレーション・アーキテクチャー

石黒 徹 Tohru Ishiguro

【プロフィール】
IBM 入社以来、クライアント・サーバーから Web 関連、SOAP や Web2.0 などの技術関連に従事し、最近ではクラウド関連の技術、リッチ・クライアントやモバイル・クライアント関連の技術に携わっている。

■ 情報爆発時代で求められるアーキテクチャーとは

インターネットの利用率向上に伴いインターネット上のデータ量が飛躍的に増加し、こうした状態を「情報爆発」と表現するようになってきた昨今、膨大なデータの処理に頭を悩ます企業が多くなってきています。

他方、ムーアの法則で高速化の一途をたどってきたコンピューティング・パワーは消費電力や発熱などの問題への配慮から、CPU のクロックアップによりいたずらに高速化するのではなく、CPU のコア数を増やすことでのトータル・パフォーマンス・アップを図るようになってきています。莫大な情報をこうしたマルチコアをベースにした環境で処理をすることは、クラウド・コンピューティングからも要請されており、クラウド・サービスをコア・コンピタンスに据えているネット企業では、並列分散コンピューティングのアーキテクチャーの採用に意欲的です。

■ Hadoop とは

そうした中で Google の論文を基に Yahoo のエンジニアたちにより、オープンソースの検索エンジンである Nutch をスケールさせるためのフレームワークとして Java™ で開発されたのが Hadoop です。Google の論文で MapReduce というアーキテクチャー・パターンが述べられていますが、そこに記されている MapReduce 処理のあらましは、膨大なデータ処理を細分化された Map 処理により並列的に分割実行し、シャッフルやソートを行った後で、Reduce 処理により取りまとめを行うとなっています（図 1）。

MapReduce とは、一定の仕事量をこの Map 処理に分割することで、多数の処理マシンを並列的に実行することにより、スケールアウトを担保するアーキテクチャーです。これにより単一ノードではなし得なかったハードウェア・リソースの限界を超えたデータ処理を行うことが可能となりました。また MapReduce は繰り返した構造を取ることも可能ですので、分析を再帰的に実行できることも特長の 1 つとなります。

並列分散のプログラミング開発と聞くと何やら難解なも

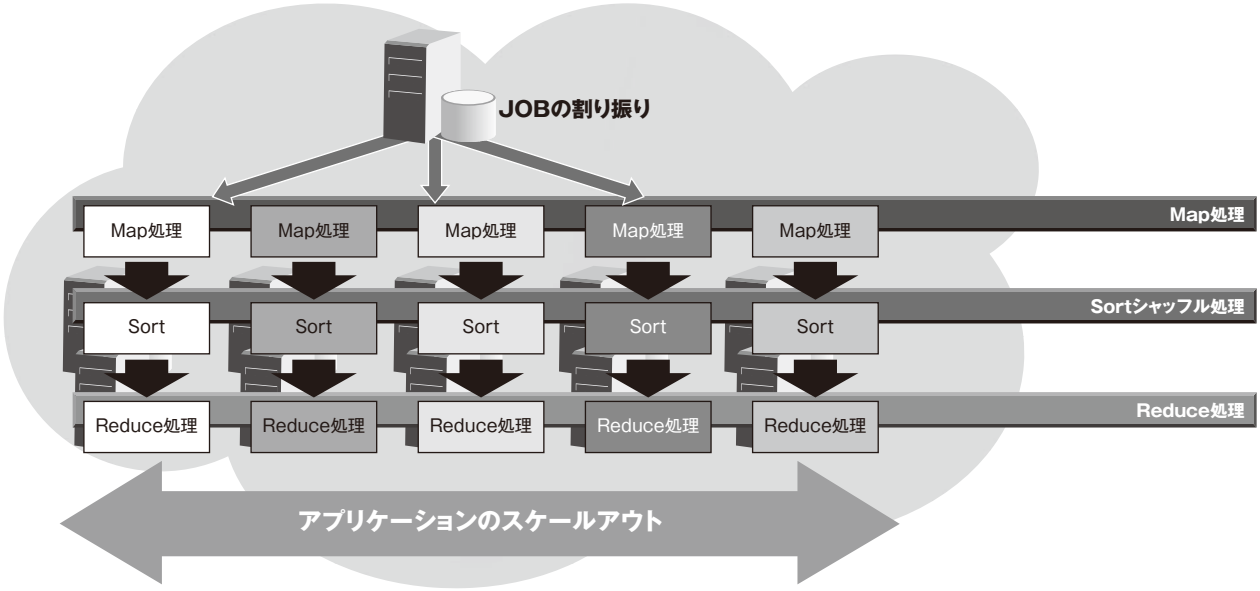

図1. Hadoopのアーキテクチャー

ののように思われるかもしれません。事実マルチスレッドのプログラミングやMPI（Message Passing Interface）における開発、OpenCL開発などの難易度は高いといわれています。そうした中でHadoop自体はJavaで記述されており、Hadoopで動かすMapReduceのアプリケーション自体も原則としてJavaで記述することになっていますので、やはり難易度が高いと思われるかもしれません。しかしJobの並列化やワークロード・マネジメントはフレームワークであるHadoopが行ってくれるため、開発者自身が並列化をそれほど意識しないでも開発することが可能となっています。

さらにHadoopは標準出力のメカニズムを使ったHadoopStreamingという方式もサポートされており、これによりJava以外のRubyやPythonといったスクリプティング言語での開発も可能となっています（図2）。

Hadoopの処理対象のデータや処理後のデータはKey-Valueの形式でHadoopのファイル・システムであるHDFSに格納することになります。これにより膨大なデータは3ノード以上の複数ノードに分散されて格納処理されます。結果としてデータも複数コピーを持って処理がなされるため、フレームワークとしての可能性も確保されています。

Hadoopはデータ形式から個々の処理に至るまで極めてシンプルな構成を取っており、これにより並列化が容易となり、スケールアウトを実現できるのですが、プログラミング・モデルとしても極めてシンプルな構造を取っているため、開発者の学習曲線を比較的低く抑えて習得することが可能です。Javaでの記述ですので言語処理系自身としては関数型言語ではないのですが、Map処理の記述においてのメソッドの実装は、関数型に近い記述を行うことになります。特に並列実行を行うために、ほかのインスタンスとは独立してメソッドは実行されなければならず、また個々の状態を変数として持つことも否定されます。こうした発想はRESTでのサービス開発に相通じるものがあります。

Hadoopの代表的な処理として大規模なログ解析処理がありますが、アプリケーションやシステム、監査など、システムで多数取られているログは、ユーザーの行動分析などにおいて有効とされており、Hadoopを使うことで1台のマシンや単一ノードでは処理しきれないサイズのログを効率的なバッチ処理とすることが可能となります。またほかのデータとの突き合わせや正規表現などのテキスト処理などにおいても、MapReduceの並列分散処理は効果的な威力を発揮します（図3）。

企業内においてソーシャル・ネットワーク・システムの導入などを行った場合、膨大なログが発生しますが、それらを効率よくスケールアウトする環境で処理するにはHadoopのようなフレームワークが必要になるでしょう。

NoSQLなKVSでの入出力、MapReduceによる並列分散によるスケールアウトの実現といったプログラミング・パラダイムを持つHadoop開発はパブリックに公開されるクラウド・サービスの開発の特徴を持つものであり、企業内におけるパブリックとのハイブリッドを考えたサービス・ミックスの下でもこの考え方は有効なものとなるでしょう。これからのクラウド・ネイティブの開発を考えた時、このプログラミング・パラダイムを先行することのメリットは計り知れません。また言語処理系として関数型といわれるものもHaskelやErlang、Scalaなどがあり、このうちJavaとの親和性やオブジェクト指向の特性も持ち合わせるScalaとHadoopを組み合わせて本格的な関数型での開発を考えることも可能となってきています（図2）。

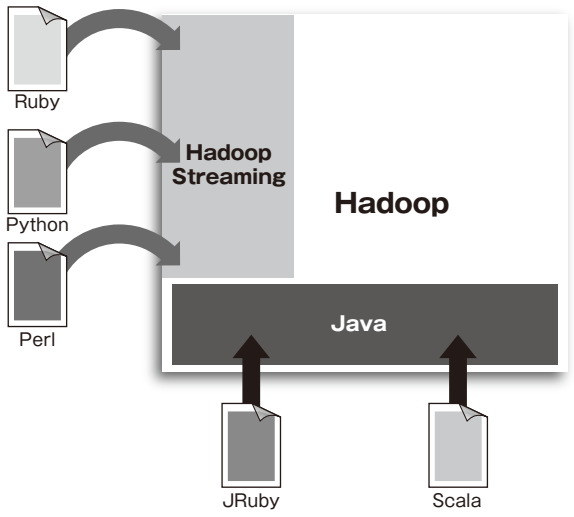

図2. Hadoopと開発言語の関係

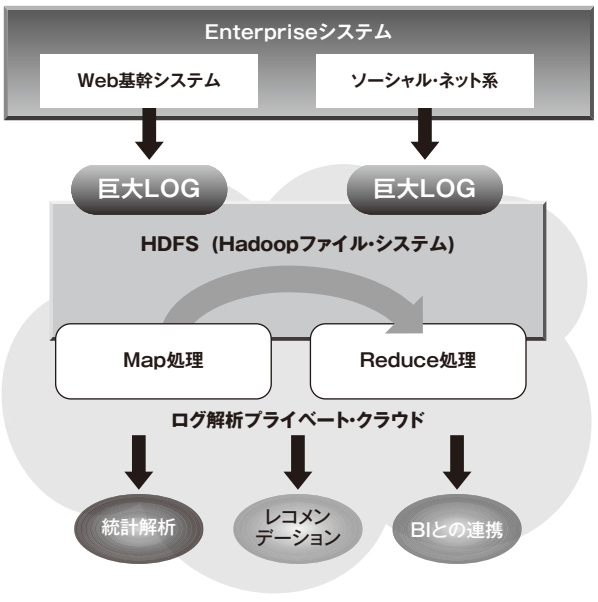

図3. HadoopによるLOG解析