Plan9 によるコマンドベース MapReduce の検討

中原 健志1　　佐藤 未来子 1　　並木 美太郎 1

1. 背景

近年，分散処理のプログラミングモデルとして MapReduce[1]が広く利用されている. MapReduce では入力データを複数に分割する Map 処理，及び分割されたデータをネットワークで接続されている複数のマシンで処理し，それらの結果をまとめる Reduce 処理の二つから成り立つ．MapReduce の実装として Hadoop[2]や Apache Spark[3]などが有名である．しかし，これらの実装では処理対象のプログラムを MapReduce 用に書き換える必要があり，既存のプログラムをそのまま利用することが難しい．そこで本研究では，UNIX の「パイプとフィルタ」のモデルを拡張して，複数のリモートマシン上で既存のコマンドを分散実行させるためのコマンドベース MapReduce の実行基盤について述べる．

2. 課題

通常の UNIX のパイプはデータを分割・結合する機能を持たない．また，ネットワークを介した他のマシンとの間ではパイプを利用することが出来ない．その他に，名前空間がマシンごとに独立しており，リモートマシンでプログラムを動作させる場合，利用者は名前空間の違いを気にする必要があり，位置透過性に欠ける．

3. 目標

利用者は行いたい処理内容を粗粒度な単位へ分け，それぞれ 1 入力 1 出力を持つコマンドとして実装する．そしてシェル上で，コマンド間のデータの流れをシェルのパイプ記号を拡張したものを用いて表す．パイプ記号を基にする利点として，パイプを知っている利用者にとって習得が容易である．シェルは記述内容から自動的に必要なリモートマシンに接続・指定されたコマンドを立ち上げ，それぞれのリモートマシンへデータを分割し，処理を行わせ，それぞれの結果を結合させることでコマンドによる MapReduce を実現する．

4. 設計

UNIX の「パイプとフィルタ」に， Map 処理として「データの分割」，および Reduce 処理として「データの結合」の機能を追加する．これにより，複数のリモートマシンでコマンドを実行できるようにする. 本シェルで利用者は通常のシェルの記号に加え，図 1 の記号を用いて行いたい分散処理の内容を記述する．また，利用者はあらかじめ利用可能なリモートマシンをシェルへ登録する．

ローカル記号（コマンド ｜ コマンド…）
　ローカルマシン上でコマンドを実行
リモート記号（｜｜ コマンド）
　一台のリモートマシン上でコマンドを実行
並列記号（[コマンド]オプション）
　登録されている全てのリモートマシン上でコマンドを分散実行

図1　拡張したシェルのパイプ記号

コマンドへのデータの入出力はシェルが自動的に分割・結合する．分割については UNIX のコマンドではテキストに関する処理が多いため，改行ごとに入力データを分割し，各リモートマシンで動作するコマンドへ渡す. 結合方法についてはオプションで，接続されているリモートマシンの処理出力を「ラウンドロビンに結合」「マージソートを行い結合」「処理結果が出力されたものから結合」の三つの中から指定できる. 名前空間についてはローカルマシンとリモートマシン間で共有化させる．これにより，利用者はリモートマシン上でローカルマシン上にあるファイルやコマンドを利用することが可能になる．

4.1 MapReduce の例：単語の出現頻度生成

単語の出現頻度を求める手順は①入力テキストの単語分解，②分解した単語のソート，③単語の出現数の計算，④出現数でのソートから成り立つ．本

1 東京農工大学
Tokyo University of Agriculture and Technology

シェルで①の処理を「**word**」，②④のを「**sort**」，④を「**count**」というコマンドで実装し，②の処理を複数のマシンで MapReduce を用いたい場合，本シェルでは下記のように記述される．

```
word < 入力ファイル  [ sort ]m | count
| sort > 出力ファイル
```

図２　シェルでの単語の出現頻度の記述例

5.　実装

ネットワークを介しても有効なパイプ，及び名前空間の重ね合わせについては分散向け OS である Plan9[4]の持つ分散透過性を用いることで実現した．データの分割・結合については一台のマシン上で行っている．

6.　評価

評価として本シェルを実装した１台のローカルマシンと３台のリモートマシンを用いて，4.1 節で述べた単語の出現頻度を求める処理を行った．入力テキストのファイルサイズと全体の処理が終了するまでの処理時間を計測した．結果を図３に示す．

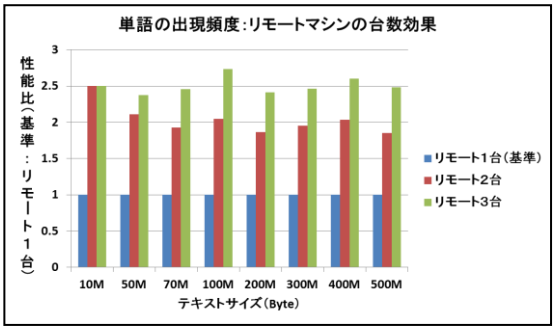

図３　単語の出現頻度による評価

ネットワークを介してほかのマシンの辞書を参照	それぞれのリモートマシン上の辞書を参照
1969 秒	82 秒

表１　辞書参照による位置透過性の評価

位置透過性の評価に，単語の辞書検索を用いた．単語の辞書検索ではローカルマシン上で入力されたテキストファイルを単語に分解，それぞれのリモートマシンで辞書ファイル内に入力された単語があるかを比較し，あった場合のみ出力させる．この時，位置透過性の機能を用いてリモートマシンがネットワークを介してローカルマシンの辞書ファイルを参照する場合と，それぞれのリモートマシン上に辞書ファイルを置き，そちらを参照する場合の全体の処理時間を計測した．結果を表１に示す．

7.　考察

単語の出現頻度を求める評価ではリモートマシンの台数に比例して性能が向上している．しかし，リモートマシン２台では１台の時と比べ，性能比が２倍近くであったのに対して，３台では２．５倍となっている．

また，辞書参照による評価では，位置透過性を用いて，ネットワーク経由で辞書を参照した場合が，リモートマシン上の辞書を参照した場合に比べ，２４倍の処理時間を必要とした．これはネットワークを介して辞書を参照するアクセスコストが高いためと考えられる．

8.　終わりに

評価・考察で判明した問題点に対して，現在下記の設計変更を行っている．

分割・統合の処理を初めのマシンで大まかな分割を行い，次に別の複数のリモートマシンで細かな分割を行う．これによりデータ分割・統合の処理が一台に集中するのを防ぐ．分割・統合の方法についても現状では行単位であるが，この方法については処理対象によって異なるので変更可能にする．またファイルの読み込みについては別のマシンにあるファイルをコマンドが読み込む場合，シェル側で読み込んだファイルをローカルへキャッシュで保存する．これによりリモートマシンで動作しているコマンドが頻繁にある特定のファイルを必要とした場合，ローカルのキャッシュを利用することで読み込みに要している時間を減らす．また，書き込みについては他のリモートマシンとの間でデータの一貫性を保つ必要がある．設計終了後に実装を行い，問題点に対する再評価を行う．

参　考　文　献

[1] Jeffrey Dean and Sanjay Ghemawat．“MapReduce: simplified data processing on large clusters．” In Proceedings of the 6th conference on Symposium on Opearting Systems Design & Implementation - Volume 6 (OSDI'04), Vol. 6. USENIX Association, Berkeley, CA, USA, 10-10. , 2004 ．

[2] Apache Hadoop http://hadoop．apache．org/

[3] Apache Spark https://spark．apache．org/

[4] Rob Pike and Dave Presotto and Ken Thompson and Howard Trickey, Plan 9 from Bell Labs,In Proceedings of the Summer 1990 UKUUG Conference1—9(1990)