5H-6

PCクラスタ環境下におけるシェルスクリプトの半自動並列化手法の提案と評価

†前田　直人　‡水谷　泰治
†大阪工業大学　大学院情報科学研究科
‡大阪工業大学　情報科学部　情報システム学科

1 はじめに

シェルスクリプトは UNIX 系の OS で広く利用されている標準的なスクリプト言語である．特に近年，シェルスクリプトで大規模な処理が行われている．例えば文献[2]では，大学のプログラミング演習において学生の提出したプログラムの採点をシェルスクリプトで行っている．このスクリプトを高速に採点できれば演習中に迅速に適切な指導ができるため，スクリプトの高速化が望まれている．

一方，シェルスクリプトの高速化の技術として，マルチコアプロセッサ上でシェルスクリプトを並列化する手法が提案されている[3]．しかし，近年のマルチコアプロセッサでは 4〜8 コアが主流であり大規模な処理の並列化に向いていているとは言えない．

また，著者らは文献[1]で複数の PC を用いてシェルスクリプトを並列処理する手法を提案している．しかし，文献[1]では単純なダミープログラムしか評価を行っておらず，実アプリケーションでの評価を行っていない．

そこで本研究では，実アプリケーションを用いて文献[1]の手法の性能を評価する．また，この過程で文献[1]の手法ではリダイレクト先のファイル名に変数展開を含む場合に対応できないことがわかったため，その点の改善を行った．

2 シェルスクリプトの並列化手法

for文の前後をユーザが特殊コメントで囲み，並列処理用のスクリプト群を自動生成する．そのスクリプトを実行することで，for文の繰り返し1回を1つのタスクとしてマスタ・ワーカ法(MW法)を用いて並列処理を行う．詳細は文献[1]を参考されたい．

図1にファイルリダイレクトを考慮した変換を示す．図1に示すように並列化するfor文内のコマンドでリダイレクトする場合，各タスクを識別するidをリダイレクト先のファイル名の後尾に付加する．そして，分割ファイルとして別々にリダイレクトする(A)．同時に，各タスクのリダイレクト先のファイル名の元々の名前を記録する(B)．その後，(B)で記録した各名前について，その名前を先頭にもつ分割ファイル群を結合する．これによりファイル名に変数展開を用いるようなリダイレクションに対応した．

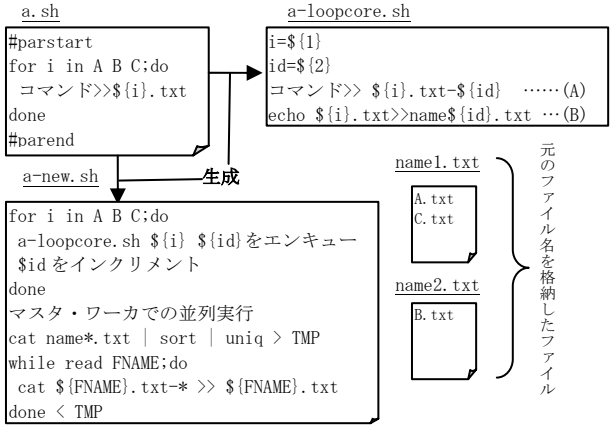

図1　ファイルリダイレクトを考慮した変換

3 性能評価

3.1 実験環境

実験環境としてLinuxOSのPCクラスタを使用した．PCクラスタの各ノードでは，NISによってアカウントを一元管理し，NFSによってホームディレクトリで共有している．並列化するスクリプトは文献[2]を参考に作成したプログラムであり，プログラミング演習にて学生の作成した課題プログラムを採点するプログラム(test.sh) である．また，比較対象として4秒のsleepコマンドを繰り返すプログラム(sleep.sh) を用いた．

```
#!/bin/sh
#parstart
for StudentNo in 学生番号のリスト;do
   for File in 課題ファイルのリスト;do
      while read Keyword #キーワード読込;do
         #ファイルにキーワードがないとき キーワード >> ログ
      #エラーテキストがないならコンパイル
      while read Input #実行ファイルに入力するデータ読込;do
         #実行.エラーのとき 入力データ >> ログ;break
         #正解と違うとき 正解 >> ログ;break
#parend          (紙面の都合上 do に対する done コマンドは省略)
```

図2　採点用プログラムの概要(test.sh)

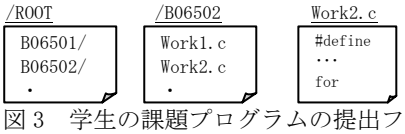

図3　学生の課題プログラムの提出フォルダ

図2は採点用プログラムの概要である．図3のように格納されている各学生のプログラムを参

照し，キーワードの検索，コンパイル，出力の検査を行い，課題の提出物として不都合があればエラーメッセージをログに記録する．

3.2　結果と考察

2章に述べた方法により，逐次実行した場合と同等のファイルが生成されたことを確認した．以降は性能について述べる．

図4にtest.shとsleep.shの速度向上率を示す．縦軸は速度向上率，横軸はワーカ数である．test.shとsleep.shの速度向上率は，ワーカ22台においてそれぞれ7.0倍と14.6倍であり，理想値(ワーカ数と同値)の1/3と2/3程度である．この原因は並列化によるオーバヘッドによるものである．ここで本手法におけるオーバヘッドとは，タスク生成，リダイレクション結果の結合，PC間の通信である．実際ワーカ数22台においてtest.shとsleep.shのオーバヘッドを計測した結果，それぞれ全実行時間の23%と19%を占めており，この比率は無視できないことがわかった．

一方，図4よりtest.shの速度向上率はsleep.shの約半分であることがわかる．この原因を調べるために，各ワーカ数において1タスクの平均実行時間を計測した．その詳細を図5に示す．図5より，ワーカ数が増加するに従い，test.shの1タスクの実行時間が増加していくことがわかった．これより，test.sh全体の実行時間が長くなり，sleep.shよりも低い速度向上率になったといえる．

この原因としてNFS上でのファイル操作が考えられる．このことを確認するため，あらかじめ各PCにtest.shで使用するファイル群をローカルディスク(/tmp)に格納し，tmpディレクトリ上のみでtest.shと同等の処理を行った(test2.sh)．図6はその結果である．縦軸は実行時間，横軸はワーカ数である．test2.shはtest.shに比べて性能が向上した．この結果より，図4でtest.shの性能が低い要因として，各PCでファイル操作を行う際，NFSサーバにアクセスが集中して処理を制限していたと考えられる．そのため，NFS上でのファイル操作が無いtest2.shやsleep.shでは，タスクの実行時間に影響がなかったと考えられる．よって，ファイルサーバの性能を向上させることで処理時間を短縮できる見込みがある．

4 まとめ

本研究では，文献[1]の手法をファイル名に変数展開を含むリダイレクトに対応できるようにした．また，実験の結果，実アプリケーションにおいてワーカ22台を用いて7.0倍の速度向上率を得ることがわかり，一定の有用性はあると我々は考える．この速度向上率は理想値の1/3程度であり，必ずしも良い値とは言えない．しかし，NFSのファイルサーバの性能を向上させることで性能をより向上できる余地があると考える．

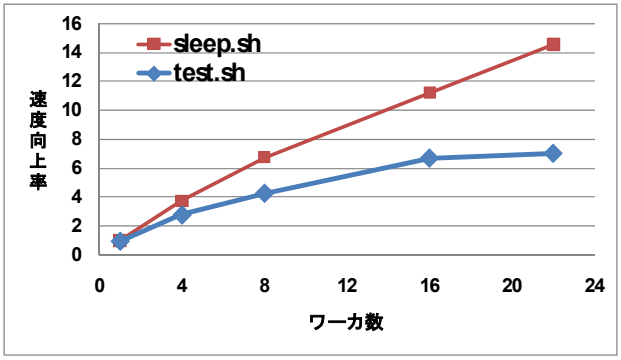

図4　手法の性能評価

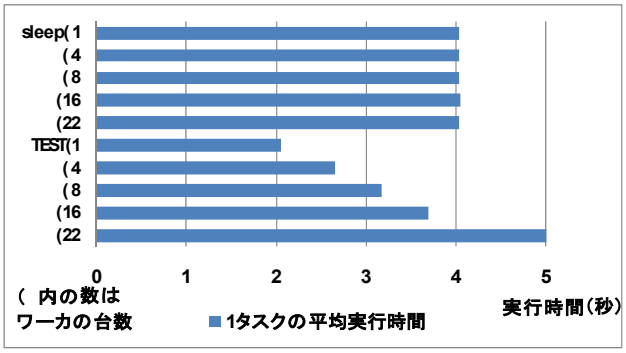

図5　1タスクあたりの平均実行時間

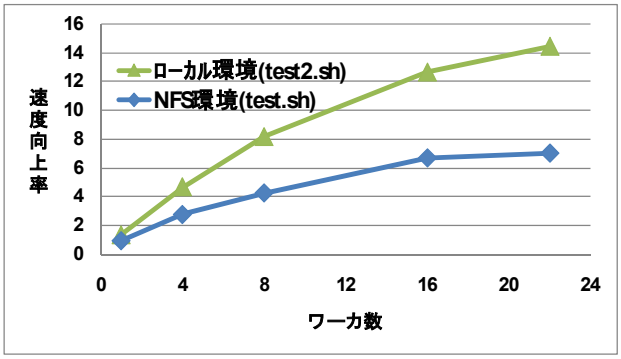

図6　ローカル環境での性能評価

謝辞　本研究の一部は，科学研究費補助金基盤研究(B)(21300011)の補助による．

参考文献

[1]菅圭佑他“繰り返しに着目したシェルスクリプトの半他自動並列化手法の提案”．第15回 電子情報通信学会 関西支部 学生会研究発表会，p. 79，(2010)

[2] 内藤広志他“プログラミング演習の進捗モニタリングシステムの評価”．第7回情報科学技術フォーラム講演論文集(FIT2008)，pp. 319-320，(2008)

[3] 杉田秀他“マルチコア・SMTプロセッサ上におけるシェルスクリプト高速化手法”．情報処理学会研究報告，2007-ARC-172，pp. 73-78，(2007)