リポジトリマイニング研究への高速化手法適用に向けた検討

山 下 一 寛[†1] 亀 井 靖 高[†1] 鵜 林 尚 靖[†1]

本発表では，リポジトリマイニングへの高速化手法の適用の初期実験の結果を報告する．

A Study to Apply Processing Techniques for High Speed to Mining Software Repositories

Kazuhiro Yamashita ,[†1] Yasutaka Kamei [†1]
and Naoyasu Ubayashi[†1]

In this paper, we report an initial work to apply high speed processing techniques to MSR.

1. はじめに

我々の研究グループでは，リポジトリマイニング（MSR）分野への高速化手法の適用として，HadoopやGPGPUを用いたソフトウェアメトリクスの計測方法について提案してきた[1)3)]．これらの研究の発展として，高速化手法をメトリクス計測に限定することなく，リポジトリの取得，データの計測，データの解析といったMSRに必要な工程全体に適用する予定である．本発表では，工程全体のうち、どの部分にボトルネックがあるかを調査する．

MSRの実装対象には，バグ予測を選択した．その理由は、1）この10年で100本以上の論文が発表され注目されている研究分野であること，2）近年，予測に用いられるデータセットの数や種類が少ないこと（多くの論文が6個以下のケーススタディしか行っていない）が問題視されており[2)]，今後大規模なデータセットを用いた研究が盛んになると予想され高速化の必要性があると考えられるためである．

2. 実装対象

2.1 実験概要

本研究では，類似のプロジェクト同士はバグ予測の予測精度においても類似性を持つ（つまり，類似プロジェクト間の予測では，予測精度が類似していないプロジェクト間の予測精度よりも高い）のではないかという仮説を検証するため，大規模なデータセットを用いてクラスタリングを行い，それぞれのクラスタ毎のバグ予測の結果の傾向を調べる．本実験では，類似性を求める際に重回帰モデルにおける各変数の係数を用いた．

2.2 データセット

本研究では，GitHub上に公開されているプロジェクトの中からまず100個以上のソースコードファイルを含む7,000プロジェクトを取り出し，そのうちランダムに60プロジェクトを用いて実験を行った．ここで，ソースコードファイルの数が100個以上のプロジェクトを選択したのは，十分に開発の行われていないプロジェクトを除くためである．

2.3 実験方法

本研究の実験手順は以下の6つに分けられる（図1）．

1）リポジトリをcloneする．

2）説明変数として，ファイル毎のコミット数，著者数およびLOCを，目的変数としてバグの有無を計測する．

3）予測モデルの作成では，計測したメトリクスを用いて重回帰モデル（予測モデル）を構築する．

4）プロジェクトのクラスタリングでは，予測モデルの各変数の係数を用いてEMアルゴリズムによってクラスタリングを行う．

5）バグ予測では，各予測モデルを用いて，予測モデル構築に用いた以外のすべてのプロジェクトを予測する．つまり，合計で3,540（=60*59）個の予測結果を得る．

6）予測精度の計測では，3,540個の結果からそれぞれPrecisionを計算し，クラスタごとに分類する．

2.4 結果

本研究の結果について，実行時間，および、バグ予

†1 九州大学
Kyushu University, Fukuoka, Japan

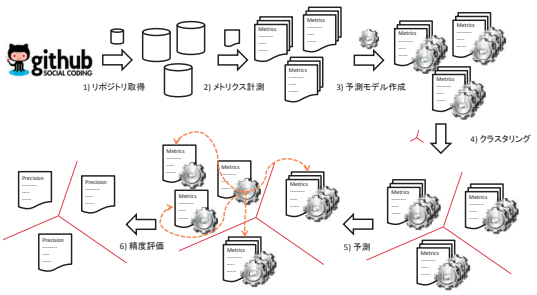

図 1　実験手順

表 1　各工程の所要時間 ⋆1

工程	1	2	3	4	5	6
所要時間	×	×	◎	◎	◎	◎

測によって得られた結果の 2 つの観点から述べる．

実行時間：実行時間を表 1 に示す．工程の番号は図 1 の番号と対応する．今回の実験では，工程 2 までは 7,000 件のプロジェクトを対象とし，工程 3 以降は 60 件のプロジェクトを対象としている．

今回の実装で実行時間の大きい部分は 2 箇所あった．まず，工程 1 である．リポジトリ 1 つ当たりにかかる時間は，数十秒から 2，3 分程度であるが今回 7,000 件のリポジトリを用いたので実行時間が大きくなった．

もう一方は，工程 2 である．今回，プロセスメトリクスの計測には Perl を用い，プロダクトメトリクスの計測には Understand というツールを用いた．プロセスメトリクスの計測では，全履歴から計測するのに 5，6 時間程度を要した．プロダクトメトリクスの計測では，通常であれば 1 プロジェクト当たり 2，3 分で計測が完了するが，いくつかのプロジェクトを計測する際に，数時間かかっても計測が終わらないことがあった．

その他の工程については，データセット数が 60 件であるためか，実行時間は 10 分以内に収まっている．

バグ予測：図 2 に，クラスタ 9 を予測した実験結果を示す．クラスタ 9 では，同じクラスタから予測した場合が最も精度が高くなっている．しかし，このような傾向はクラスタ全体を通しては見られなかった．

3. 考　　察

本研究の結果について，高速化，実験テーマの 2 つの観点から考察する．

高速化：プロセスメトリクス計測に関しては，我々がこれまで取り組んできた高速化手法を適用でき，リポジトリの clone，プロダクトメトリクス計測に関しては，並列化することによって実行時間を減らすことが出来ると考える．

⋆1 ◎は 30 分以内，○は 1 時間以内，△は 12 時間未満，×は 12 時間以上を示す．

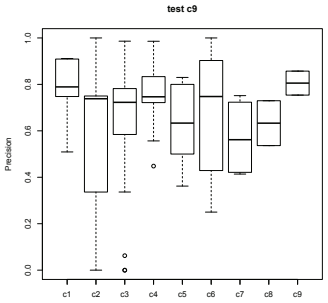

図 2　クラスタ 9 をテストデータとする Precision

また，予測・評価の工程では R 言語を用いて実装を行った．今回のデータセット数・計算ではさほど時間がかからなかったものの，データセット数の増加や，計算の複雑化よって上記の 2 つの工程ほどではないものの，実行時間が増大する可能性がある．そこで，R で Hadoop を利用する RHadoop や GPU を利用する Gputool，Rpud などを利用することを検討している．

バグ予測：本実験では，60 件のプロジェクトを用い，予測モデルの係数を用いてクラスタリングをし，予測精度を求めた．結果，我々の仮説を証明するような結果は得られなかったが，プロジェクト数，クラスタリングの手法・対象については，どのように設定するのが最適であるかを検討する余地があると考える．

謝辞　本研究の一部は，JST CREST「ポストペタスケール時代のスーパーコンピューティング向けソフトウェア開発環境」による助成を受けた．

参　考　文　献

1) Nagano, R., Nakamura, H., Kamei, Y., Adams, B., Hisazumi, K., Ubayashi, N. and Fukuda, A.: Using the GPGPU for scaling up Mining Software Repositories, *ICSE*, pp.1435–1436 (2012).
2) Shihab, E.: An Exploration of Challenges Limiting Pragmatic Software Defect Prediction, *PhD Thesis, School of Computing, Queen's University, Kingston, Ontario, Canada* (2012).
3) 大坂　陽，山下一寛，亀井靖高，鵜林尚靖：リポジトリマイニングに対する Hadoop の導入に向けた性能評価，情報処理学会ソフトウェアエンジニアリングシンポジウム 2013 (SES 2013).