様　式　Ｃ－１９、Ｆ－１９－１、Ｚ－１９　（共通）

科学研究費助成事業　　研究成果報告書

平成　２９　年　　６　月　　９　日現在

機関番号：　１５３０１
研究種目：　基盤研究(C)（一般）
研究期間：　2014～2016
課題番号：　２６３３００８６

研究課題名（和文）バグ予測に基づくテストシミュレーション基盤の構築

研究課題名（英文）Development of test simulation framework based on bug prediction

研究代表者
門田　暁人（Monden, Akito）

岡山大学・自然科学研究科・教授

研究者番号：８０３１１７８６

交付決定額（研究期間全体）：（直接経費）　　3,600,000円

研究成果の概要（和文）：ソフトウェアバグ予測に基づくテストの有効性の評価を可能とするために，テスト戦略を資源配分問題として定式化するとともに，この問題を分離凸資源配分問題として解くことが可能であることを示した．また，バグ予測には誤差が含まれることが避けられないことから，バグ予測を行わないテスト戦略との組み合わせ方法の提案・評価を行った．さらに，バグ予測が有効となるコンテキストを明らかにするために，与えられたデータセットに対し，テスト工数削減量の最大値を推定する方法を提案した．以上の成果により，テスト工数削減の効果を見積もることが可能となり，バグ予測技術の採用が促進されると期待される．

研究成果の概要（英文）：To enable evaluation of effectiveness of test strategies based on software bug prediction, we formalized the problem of test strategies as a nonlinear optimization problem; and then, we showed that the problem can be solved as a separable convex problem. Next, as the bug prediction inherently contain errors, we proposed and evaluated a method that combines test strategies not using bug prediction. Then, to clarify the context where bug prediction become effective, we proposed a method to estimate the maximum amount of reducible test effort by bug prediction. From these achievements, practitioners are able to estimate the expected reducible test effort, which will promote employing bug prediction in industry.

研究分野：　ソフトウェア工学

キーワード：　ソフトウェアテスト　ソフトウェア品質保証　ソフトウェアバグ予測　シミュレーション

１．研究開始当初の背景

近年，ソフトウェアの大規模化・短納期化に伴い，ソフトウェアテストの効率化がますます重要となっている．従来，テスト資源（人員，テストケースなど）を重点的に配分すべき，バグの多いモジュール（サブシステム，機能）を特定することを目的として，各モジュールのバグの有無やバグ数を予測するバグ予測研究が盛んに行われてきた．これらの研究では，ソフトウェアに潜在するバグをどの程度正しく予測できるか，という観点から，適合率／再現率，F1 値，ROC Curve などによりバグ予測精度が評価されてきた．しかし，バグを予測することと，テストを効率化することの間には隔たりがあり，テストの効率化への貢献は不明であった．

このために，近年，テスト工数を考慮したバグ予測精度の評価手法が提案されてきた．Mende らの提案では，規模の小さい（すなわちテストの容易な），かつ，潜在バグの多いモジュールのバグ数を正確に予測できた場合ほど精度が良い，と評価する．これによって，テストの効率化の観点から，性能の良い予測モデルを選定することが可能となった．しかし，テストのコストをどの程度削減できるのかについては，依然として不明であった．

２．研究の目的

本研究では，バグ予測に基づくテストの効果（削減可能なコスト）を，シミュレーションにより推定することを可能とすることを目的とする．そのために，テストのモデル化，テスト戦略の定式化および，その解を求めるための方法の確立を行う．また，バグ予測に基づくテストが有効となるコンテキストを明らかにする方法を確立する．

３．研究の方法

バグ予測に基づくテストシミュレーション基盤の確立を目的として，次を実施する．

(1) ソフトウェアテストのモデル化

本研究では，テストに要する工数（コスト）は，実行されるテストケース数に比例すると捉え，テストケース数とバグ発見率との関係をモデル化する．モデル化にあたっては，それぞれのソフトウェアモジュールの特徴がバグ発見率に影響することを考慮する．

(2) 資源配分問題としての定式化とその求解アルゴリズムの開発

総テストケース数，各モジュールの特徴量，バグ予測結果（各モジュールの潜在バグ数の予測値）を入力とし，各モジュールの期待バグ発見数の和を最大化することを目的関数とした定式化を行う．

(3) バグ予測誤差を含むことを前提としたテスト戦略の組み合わせ方法の検討

バグ予測には誤差が含まれるため，(2)で求めたバグ予測に基づく戦略に対し，予測誤差を前提としたリスクヘッジを行う方法について検討する．

(4) バグ予測が有効となるコンテキストを明らかにする方法の開発

バグ予測によるテスト工数削減の効果は，期待されるバグ予測精度，総テストケース数に依存する．ただし，バグの予測精度が高い場合であっても，テスト工数削減の効果が小さい場合がある．例えば，ソフトウェア中にバグがまんべんなく存在する場合である．一般に，バグ予測自体にもコストがかかることから，そのような場合にはバグ予測に基づくテストを行うべきではないといえる．そこで，バグ予測が有効となるようなコンテキストを明らかにするための方法を開発する．

４．研究成果

(1) ソフトウェアテストのモデル化

モジュールの集合を $M=\{m_1,\ldots,m_n\}$，モジュール m_i に割り当てられるテスト工数を t_i，総テスト工数を t_{total} とすると，モジュール m_i の期待発見バグ数 $\hat{H}_i(t_i)$ との関係を次のようにモデル化した．

$$\hat{H}_i(t_i) = a_i[1-\exp(-b_i t_i)], \qquad b_i = b_0/S_i$$

b_i : m_i のバグ発見容易性
a_i : m_i の潜在バグ数
S_i : m_i の複雑さ
b_0 :定数

本モデルでは，指数型ソフトウェア信頼度成長モデルをベースにしており，費やしたテスト工数に応じて指数的にバグ発見率が増加する．ただし，バグ発見容易性は，モジュールの複雑さに反比例する．ここで，S_i は，モジュールの規模，条件分岐数，サイクロマティック数などである．また，定数 b_0 は，実際のソフトウェア開発プロジェクトの実績に基づいて与える．

本モデルにより，次の手順によりソフトウェアテストのシミュレーションを行うことが可能となる．

Step. 1 バグ数の予測：過去の開発実績データから各モジュールのバグ予測数モデル（ランダムフォレスト，重回帰分析など）を構築し，予測対象のプロジェクトのデータに適用することで予測値を得る．

Step 2. テスト戦略に基づく工数割り当て：テスト戦略に基づいて，利用可能な総テスト工数を各モジュールに配分する．

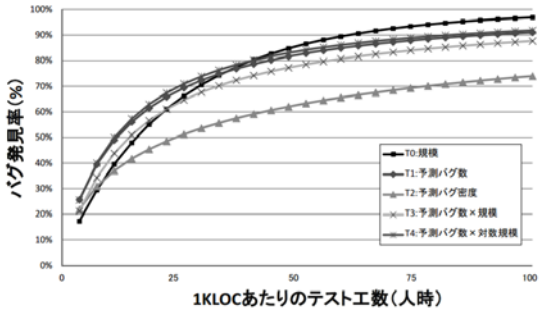
Step 3. 発見できるバグ数の期待値の算出：ソフトウェアテストのモデルに基づいて，Step 2 で割り当てられたテスト工数とテストモジュールの特徴から，発見できるバグ数を推定する．

Step 4. テスト戦略，利用できるテスト工数をそれぞれ変化させて Step 1 から Step 3 を繰り返すことで，様々なテスト戦略の効果を明らかにする．

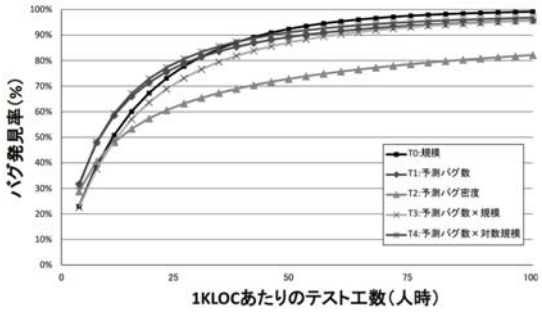
２つのオープンソースソフトウェア（Eclipse，Mylin）の開発データを対象として，本モデルに基づくシミュレーション実験を行った．実験では，次のテスト戦略 T0～T4 の比較を行った．

T0：テスト工数∝モジュール規模
T0’：テスト工数∝モジュールの複雑さ（サイクロマティック数）
T1：テスト工数∝予測バグ数
T2：テスト工数∝予測バグ密度
T3：テスト工数∝予測バグ数×モジュール規模
T4：テスト工数∝予測バグ数×log(モジュール規模)

Eclipse データセットに対する実験結果を下図に示す．図では，図中，戦略 T0’と T0 はほぼ同じ結果であったため省略している．



同様に，Mylin データセットに対する実験結果を下図に示す．



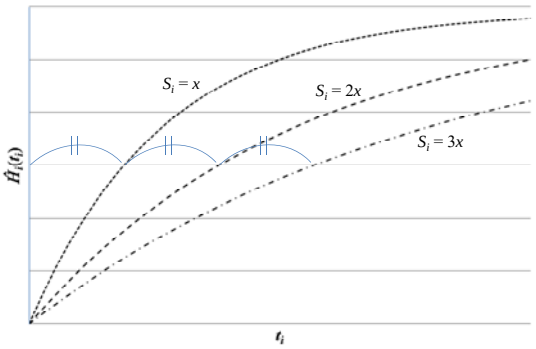
いずれの結果においても，T1 と T4 の効果が高いことが分かった．ただし，利用可能なテスト工数が大きい場合，T0 の効果が高くなっている．

(2) 資源配分問題としての定式化とその求解アルゴリズムの開発

モジュール集合 M の期待発見バグ数を $f(t_1, t_2, \ldots, t_n)$とすると，発見バグ数を最大化するようにテスト工数を配分するという問題は，次式で定式化できる．

$$\text{maximize } f(t_1, t_2, \cdots, t_n) = \sum_{i=1}^{n} \hat{H}_i(t_i)$$
$$\text{subject to} \sum_{i=1}^{n} t_i = t_{total}$$

ここで，$\hat{H}_i(t_i)$は次のグラフで示さえるように，単調増加かつ凸関数である．



従って，本問題は分離凸資源配分問題であり，次に示す greedy 法による求解が可能である．

Step 1: Let $(t_1, \ldots, t_n) = (0, \ldots, 0)$
Step 2: if $\sum_{i=1}^{n} t_i = t_{total}$ then end. $(t_1, \ldots, t_n)$ is the optimal solution.
Step 3: Select i that satisfies $f_i(t_i + 1) > f_j(t_j + 1)$ for all j $(\neq i)$. Let $t_i = t_i + 1$.
Step 4: Goto *Step* 1.

本方法を Mylin データセットに適用した結果を下表に示す．

戦略	必要なテスト工数
T0	1
T1	0.77
T4	0.79
最適戦略	0.73

この表では，バグ予測を用いないテスト戦略である T0 と同数のバグを発見するのに必要なテスト工数を表している．資源配分問題として求解することで，従来戦略である T1，T4 よりも必要なテスト工数を削減できていることが分かる．

(3) バグ予測誤差を含むことを前提としたテスト戦略の組み合わせ方法の検討

バグ予測を行わない場合のテストケース配分方法（戦略１）について検討し，バグ予測に基づくテストケース配分方法（戦略２）との組み合わせ戦略（戦略３）を提案するとともに，その効果について評価を行った．戦略１では，全てのモジュールのバグ密度の期待値は等しいと仮定する．この仮定のもとでの最適なテストケース配分は，全モジュールのテストケース密度を等しくすることである．戦略３では，総テストケースのうちある割合(m とする)については戦略１を採用し，残りの(1-m)のテストケースについては戦略２を採用する．最適な m の値は実験により与えるものとする．

複数のソフトウェアを対象として，戦略１，２，３の効果を比較した結果，いずれのソフトウェア，および，いずれの m を与えた場合においても，バグ発見数の大小は，戦略２＞戦略３＞戦略１となった．したがって，現時点では戦略３よりも戦略２を採用すべきで

あることが分かった.

(4) バグ予測が有効となるコンテキストを明らかにする方法の開発

(2)における $\hat{H}_i(t_i)$を，$\hat{H}_i(t_i) = H_i(t_i)$とする，すなわち，期待発見バグ数を，実際に潜在しているバグ数として与えることで，与えられたテスト工数 t_{total} に対して発見可能な最大バグ数を推定することが可能となる．適用実験を通して，バグがソフトウェア中に偏在していないほど，バグ予測の効果が小さくなることを確認した.

以上の成果より，テスト工数削減の効果を見積もることが可能となり，バグ予測技術の採用が促進されると期待される.

５．主な発表論文等
（研究代表者、研究分担者及び連携研究者には下線）

〔雑誌論文〕（計 8 件）

① Passakorn Phannachitta, Jacky Keung, Akito Monden, Kenichi Matsumoto, A stability assessment of solution adaptation techniques for analogy-based software effort estimation, Empirical Software Engineering, Vol.22, No. 1, pp.474-504, 2017.（査読あり）
② Takashi Watanabe, Akito Monden, Yasutaka Kamei, Shuji Morisaki, Identifying recurring association rules in software defect prediction,″ Proc. 15th International Conference on Computer and Information Science, pp.1-6, 2016. （査読あり）
③ Passakoron Phannachitta, Akito Monden, Jacky Keung, Kenichi Matsumoto, LSA-X: Exploiting productivity factors in linear size adaptation for analogy-based software effort estimation, IEICE Transactions on Information and Systems, Vol.E99-D, No.1, pp. 151-162, 2016. （査読あり）
④ Passakoron Phannachitta, Akito Monden, Jacky Keung, Kenichi Matsumoto, Case consistency: a necessary data quality property for software engineering data sets, Proc. 19th International Conference on Evaluation and Assessment in Software Engineering, No.19, pp.1-10, 2015. （査読あり）
⑤ 門田暁人，テスト戦略の最適化に向けて，ウィンターワークショップ 2015・イン・宜野湾論文集，35-36，2015. （査読あり）
⑥ 中野大輔，門田暁人，亀井靖高，松本健一，バグモジュール予測を用いたテスト工数割り当て戦略のシミュレーション，コンピュータソフトウェア，Vol.31，No.1，pp.118-128，2014. （査読あり）
⑦ 門田暁人，Passakorn Phannachitta，松本健一，ソフトウェア開発データの無矛盾性の評価，ソフトウェア工学の基礎 XXI，189-194，2014.（査読あり）
⑧ Passakoron Phannachitta, Jacky Keung, Akito Monden, Kenichi Matsumoto, Scaling up analogy-based software effort estimation: a comparison of multiple Hadoop implementation schemes, Proc. International Workshop on Innovative Software Development Methodologies and Practices, 65-72, 2014. （査読あり）

〔学会発表〕（計 2 件）

① 松井聖，門田暁人，プロジェクト間バグ予測方法の実験的評価，電子情報通信学会ソフトウェアサイエンス研究会，2017 年 1 月 26～27 日，京都工芸繊維大学.
② 渡部恭史，門田暁人，亀井靖高，森崎修司，再現性のあるアソシエーションルールの選定 ～ソフトウェアバグ予測を題材として～，情報処理学会ソフトウェア工学研究会，2015 年 12 月 15～16 日，JR 博多シティ会議室.

〔図書〕（計 0 件）

〔産業財産権〕

○出願状況（計 0 件）

○取得状況（計 0 件）

〔その他〕
なし

６．研究組織
(1)研究代表者

門田　暁人（MONDEN AKITO）
岡山大学・大学院自然科学研究科・教授
研究者番号：80311786

(2)研究分担者
なし

(3)連携研究者

亀井　靖高（KAMEI YASUTAKA）
九州大学・大学院システム情報科学研究院・准教授
研究者番号：10610222

(4)研究協力者
なし