柿元 健, 門田 暁人, 亀井 靖高, まつ本 真佑, 松本 健一, ``Fault-Prone モジュール判別におけるF1値とソフトウェア信頼性の関係," 岸知二, 野田夏子(編), ソフトウェア工学の基礎XIV 日本ソフトウェア科学会 FOSE2007, pp.75-83, November 2007.

Fault-prone モジュール判別における F1 値とソフトウェア信頼性の関係

Relationship between Software Reliability and F1 Value of Fault-prone Module Detection

柿元 健*　門田 暁人†　亀井 靖高‡　松本 真佑§　松本 健一¶

あらまし　ソフトウェアモジュールから計測したメトリクス値（SLOC, サイクロマティック数など）に基づいて，fault の有無を推定するモデル（fault-prone モジュール判別モデル）が従来研究されてきたが，それらが開発現場にどの程度役立つのかについての議論はほとんどなされていない．本稿では，SRGM に基づいて，fault の有無，テスト工数，fault 発見率の関係をモデル化し，fault 発見率モデルを提案する．そして，構築した fault 発見率モデルを用いて，fault-prone モジュール判別モデルの評価指標である F1 値に着目し，F1 値の変化がテスト効率や信頼性の向上にどのような影響を及ぼすかを明らかにする．

1　はじめに

ソフトウェアモジュールから計測したメトリクス値（SLOC，サイクロマティック数，継承の深さなど）に基づいて，欠陥（fault）の有無を推定するモデル（fault-prone モジュール判別モデル）が従来盛んに研究されてきた [8] [12]．モデル化の方法としては，線形判別分析，ロジスティック回帰分析，ニューラルネット，分類木などがよく用いられている [3] [7] [10]．さらに，モデルの性能を向上させるために，オーバー/アンダーサンプリング [6] や外れ値除去 [9] が併用されたり，SVM(Support Vector Machine) などの新しいモデルも提案されている [5].

その一方で，これらの fault-prone モジュール判別モデルが，開発現場にどの程度役立つのかについての議論はほとんどなされていない．開発現場での想定される利用方法は，fault を含むと判別されたモジュールにより多くのテスト工数を割り当てる（含まないと判別されたモジュールには少しのテスト工数しか割り当てない）ことで，(1) 信頼性の確保と (2) 無駄なテスト工数の削減の両立を目指すものである．ABB Inc. では実際にこのような利用を試み [8]，効果のあることを定性的に示している．ただし，fault の有無を判別すること，テスト工数を割り当てること，及び，信頼性を確保することの定量的な関係は従来明らかにされていない．そのため，たとえ精度のよい判別モデルが得られたとしても，その有用性の程度は不明であり，判別モデルが現場で採用されにくい一因となっている．

本稿では，fault-prone モジュール判別モデルの評価指標として広く採用されている F1 値（適合率と再現率の調和平均）[4] [10] に着目し，F1 値の変化がテスト効率や信頼性の向上にどのような影響を及ぼすかを明らかにすることを目的とする．そのために，fault-prone モジュールに対するテスト工数の割り当てに関する仮定を整理し，fault の有無，テスト工数，及び，fault 発見率の関係をモデル化する（fault 発見率モデルと呼ぶ）．また，F1 値と信頼性（ソフトウェア全体での fault 発見率）の関係を明らかにするために，様々な条件の下でシミュレーションを行う．シミュレーションでは，(1)fault-prone と判別したモジュールに割り当てるテスト工数の

*Takeshi Kakimoto, 奈良先端科学技術大学院大学 情報科学研究科

†Akito Monden, 奈良先端科学技術大学院大学 情報科学研究科

‡Yasutaka Kamei, 奈良先端科学技術大学院大学 情報科学研究科

§Shinsuke Matsumoto, 奈良先端科学技術大学院大学 情報科学研究科

¶Ken-ichi Matsumoto, 奈良先端科学技術大学院大学 情報科学研究科

倍率を変化させた場合，(2)F1 値を固定して各種条件を変動させた場合，(3) 各種条件を固定して F1 値を変化させた場合，の 3 つのケースを実施した．

以降，2 章では，構築した fault 発見率モデルについて説明する．3 章では，構築したモデルを用いたシミュレーションとその結果について述べ，4 章では F1 値とテストの効率化や信頼性向上の関係について議論する．最後に 5 章で本稿のまとめと今後の課題について述べる．

2 Fault 発見率モデルの構築

2.1 テスト工数の割り当てに関する仮定

Fault-prone モジュールの判別結果に対し，開発現場では，次のポリシーに基づいてテスト工数の割り当てを行うと仮定する．

- Fault を含むと判別されたモジュール（fault-prone 判別モジュール）により多くのテスト工数を割り当てる．逆に，fault を含まないと判別されたモジュール (non-fault-prone 判別モジュール) には少しのテスト工数しか割り当てない．
- Fault-prone 判別モジュールには，non-fault-prone 判別モジュールの r 倍のテスト工数が割り当てられる．ここで，r は 1 以上の実数である．

この仮定の下では，fault-prone 判別の精度が悪いほど，実際には fault の無いモジュールにより多くのテスト工数を割り当てたり，fault が存在するモジュールに少ししか工数を割り当てないこととなり，信頼性の低下や工数の増大を招く．

現実には，fault を含む確率（期待値）に応じて段階的に工数を割り当てることも考えられるが，議論が複雑になるため，本稿では，fault の有無は 2 値判別されるものとし，$1 : r$ の比でテスト工数を割り当てることとした．

2.2 1 個のモジュールに対する fault 発見率モデル

前節で決定されたテスト工数に基づいて，ある確率で fault が発見されることになる．一般に，モジュールに fault が含まれている場合，より多くのテスト工数を費やすほど fault の発見率は高まる．ただし，fault が含まれていない場合は，費やしたテスト工数に関わらず fault は発見されない．ここでは，モジュール中の fault の有無，テスト工数，fault 発見率の関係をモデル化する（fault 発見率モデルと呼ぶ）．

従来，fault 発見率に関するモデルとしては，ソフトウェア信頼度成長モデル（SRGM: Software Reliability Growth Model）[13] が一般に知られている．SRGM では，プロジェクト全体で時間 t までに発見される期待 fault 数を求めることができる．本稿では，モデル化にあたって，SRGM とは異なり，時間を取り扱わず，また，期待 fault 数ではなく fault の有無のみを取り扱う．期待 fault 数でなく falut 数を取り扱うため，fault が発見される確率（期待値）を求めることとする．fault 発見率は，全 fault 数に対する発見 fault 数の割合であり，fault 発見率は fault 数に置き換え可能であると言える．また，時間 t の代わりに，テスト工数を用いることとする．テスト工数は，要員数と時間の積であり，時間 t に関する変数であるテスト工数は時間 t と置き換え可能であると言える．

本稿では，1 個のモジュールごとの fault 発見率モデルとして，比較的単純なモデルである指数関数モデルを採用する．このモデルは，fault 数と発見率，時間と工数といった違いはあるものの，SRGM でも採用されている [2]．あるモジュールに含まれる fault のうち発見される fault の割合を式 (1) で定義する．

$$d(E) = 1 - e^{-ZE} \tag{1}$$

ここで，$d(E)$ はテスト工数 E を割り当てたときの fault 発見率．Z は単位工数あたりに fault を発見する割合を決定する定数である．この指数関数モデルでは，fault は費やしたコスト（テスト工数）に対して一様の確率で発見されることとなる．

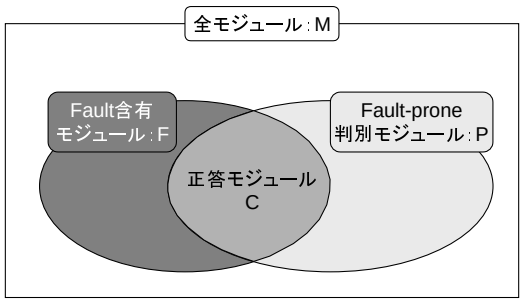

図 1　モジュールの関係

2.3　全モジュールに対する fault 発見率モデル

式 (1) のモデルは，1 個のモジュールにおける fault 発見率を表したモデルである．このモデルを，全モジュールの fault 発見率のモデルに拡張する．

本稿では，fault 数ではなく fault の有無を扱うため，実際に fault が含まれるモジュール（fault 含有モジュール）の数がモデルで重要な要素となる．そのため，複数モジュールにまたがる fault の場合，fault に影響する各モジュールを全て fault 含有モジュールとする．

全モジュール数を M，fault 含有モジュールの数を F，fault-prone モジュールと判別したモジュール（fault-prone 判別モジュール）の数を P，fault-prone モジュールと判別し実際に fault を含むモジュールであったモジュール（正答モジュール）の数を C とする．それぞれを集合と考えると，それぞれの関係は図 1 のようになる．

Fault-prone 判別モジュール 1 個あたりにテスト工数 E_F を費やし，non-fault-prone 判別モジュール 1 つあたりにテスト工数 E_{NF} を費やすとすると，fault が発見される確率は，fault 含有モジュールの内，fault-prone 判別モジュール（図 1 の C）では $d(E_F)$，fault 含有モジュールの内，non-fault-prone モジュールと判別したモジュール（図 1 の $F - C$）では $d(E_{NF})$，実際に fault を含まないモジュール（図 1 の $M - F$）では 0 となる．

従って，fault 含有モジュールに含まれる全ての fault に対して，

- fault 含有モジュールの内，fault-prone モジュールと判別した割合 (Recall) のモジュールでは，$d(E_F)$ の割合で fault が発見される．
- fault 含有モジュールの内，non-fault-prone モジュールと判別した割合 (1-Recall) のモジュールでは，$d(E_{NF})$ の割合で fault が発見される．

そこで，全モジュールの fault 発見率 D は式（2）で表される．

$$\begin{aligned} D &= d(E_F) \times Recall + d(E_{NF}) \times (1 - Recall) \\ &= \left(1 - e^{-Z \times E_F}\right) \times Recall + \left(1 - e^{-Z \times E_{NF}}\right) \times (1 - Recall) \end{aligned} \tag{2}$$

式 (2) の E_F，E_{NF}，Recall について，詳細な式を算出する．

全テスト工数を E_{all}，fault-prone 判別モジュールには non-fault-prone 判別モジュールの r 倍のテスト工数が重点的に割り当てられるとすると，式 (3) と式 (4) が成り立つ．

$$E_{all} = E_F \times P + E_{NF} \times (M - P) \tag{3}$$

$$E_F = E_{NF} \times r \tag{4}$$

E_F，E_{NF} を求めるため，式 (3) と式 (4) の連立方程式を E_F と E_{NF} について解くと

$$E_F = \frac{E_{all} \times r}{P(r-1)+M} \tag{5}$$

$$E_{NF} = \frac{E_{all}}{P(r-1)+M} \tag{6}$$

が得られる．

また，Recall は式 (7) で表される．

$$Recall = \frac{C}{F} \tag{7}$$

式 (2) に，式 (5)，式 (6) および，式 (7) を代入すると，式 (8) が得られる．

$$D = \left(1 - e^{-\frac{E_{all} \times r}{P(r-1)+M} Z}\right) \times \frac{C}{F} + \left(1 - e^{-\frac{E_{all}}{P(r-1)+M} Z}\right) \times \frac{F-C}{F} \tag{8}$$

また，F1 値は図 1 の記号を用いると式 (9) で表される．

$$F1 = \frac{2C}{F+P} \tag{9}$$

従って，式 (8) を F1 値を用いて表すと式 (10) のようになる．

$$D = \left(1 - e^{-\frac{E_{all} \times r}{P(r-1)+M} Z}\right) \times \left(\frac{2}{F1} - \frac{P}{C}\right) + \left(1 - e^{-\frac{E_{all}}{P(r-1)+M} Z}\right) \times \left(1 - \frac{2}{F1} + \frac{P}{C}\right) \tag{10}$$

本稿では，式 (8)，および，式 (10) の fault 発見率モデルを使用してシミュレーションを行う．

構築した fault 発見率モデルでは，全テスト工数 E_all，もしくは，単位工数あたりに fault を発見する割合を決定する定数 Z が増加すると fault 発見率 D は $1-e^{-x}$（x は E_{all} または Z）で増加する．また，全モジュール数 M，もしくは，判別モジュール数 P が増加すると fault 発見率 D は $1-e^{-1/x}$（x は M または P）で減少する．それ以外のパラメータは他のパラメータの値によって複合的に fault 発見率は変化する．

3 シミュレーション

3.1 シミュレーションの設定

本稿のシミュレーションは，fault-prone モジュール判別モデルの評価指標の 1 つである F1 値の変化が，fault 発見率にどのような影響を及ぼすかを明らかにするために行う．シミュレーションは式 (8) のモデルに入力する値を変化させながら fault 発見率を算出することで行う．

Fault-prone モジュール判別モデルの代表的な評価指標としては，適合率（Precision），再現率（Recall），F1 値が挙げられる [4]．これらの評価指標のうち，適合率と再現率はトレードオフの関係である．そこで，本稿では，それらの調和平均の値である F1 値を採用することとした．

モデルに入力する値は，現実のプロジェクトの状況を反映したものであることが望ましい．本稿では，NASA/WVU で公開されているモジュールに関するデータ [11] や，情報処理推進機構 (IPA) ソフトウェア・エンジニアリング・センターで収集されているテスト工数に関するデータ [1] を参考に，次のように値を決めた．(なお，本稿のシミュレーションでは，特に明記しない場合にはこれらの値を入力として用いる．)

- 全モジュール数：M=1000 (個)
- 全テスト工数：E_{all}=1500 (人時)
- 定数：Z=0.5
- Fault-prone 判別モジュールに割り当てるテスト工数の倍率：r=3

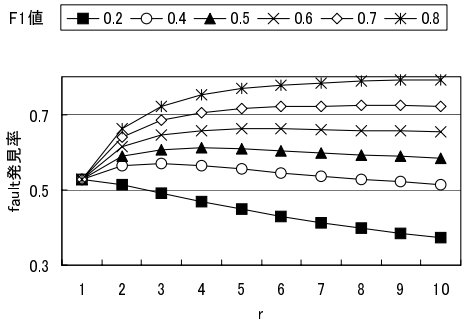

図 2　Fault-prone 判別モジュールに割り当てるテスト工数の倍率の変化

全モジュール数 M は値を大きくするとソフトウェア全体での fault 発見率は低下し，全テスト工数 E_{all}，および，定数 Z は値を大きくすると fault 発見率は向上する．これらの値の変化は，シミュレーション結果のグラフにおいて縦軸の値が変化するだけで，グラフの形状は変化しない．従って，全モジュール数 M，全テスト工数 E_{all}，定数 Z については今回のシミュレーションでは定数とした．r については，次のシミュレーション 1 で述べる．

3.2　シミュレーション 1：fault-prone 判別モジュールに割り当てるテスト工数の倍率 r を変化させた場合

3.2.1　概要

シミュレーション 1 では，fault-prone 判別モジュールに重点的にテスト工数を割り当てたときに，ソフトウェアの信頼性，すなわちソフトウェア全体の fault 発見率が，F1 値によってどのように変化するかを明らかにする．そのために，F1 値ごとに non-fault-prone 判別モジュールに対する fault-prone 判別モジュールに割り当てるテスト工数の倍率 r を変化させて，fault 発見率の変化を確かめた．

シミュレーションでは，fault モジュール含有率を 0.2，fault-prone 判別モジュール率を 0.2 とし，正答モジュール数は F1 値によって変化させた．

3.2.2　結果

シミュレーションの結果を図 2 に示す．シミュレーションの結果，F1 値が高いときには，fault-prone 判別モジュールにテスト工数を多く割り当てることで，発見される fault は増加するが，F1 値が低いときには，fault-prone 判別モジュールにテスト工数を多く割り当てると，発見される fault が減少している．また，境目となる F1 値付近（シミュレーションの結果では 0.5 付近）では，fault-prone 判別モジュールに対して，ある程度重点的にテスト工数を割り当てることで発見される fault 数は増加するが，割り当てすぎると発見される fault 数が減少している．これは，fault-prone 判別モジュールに対してテスト工数を割り当てすぎることで，non-fault-prone 判別モジュールに割り当てるテスト工数が減少し，fault 含有モジュールのうち non-fault-prone と判別されたモジュールに含まれる fault の発見率の低下が，正答モジュールに含まれる fault の発見率の上昇を上回ったためである．

3.3　シミュレーション 2：F1 値を固定して他の条件を変動させた場合

3.3.1　概要

シミュレーション 2 では，F1 値を固定し，fault 含有モジュール数や fault-prone 判別モジュール数を変化させることで fault 発見率がどのように変化するかを明らかにする．そのために，F1 値を同じ値に保ったまま，fault-prone 判別モジュール

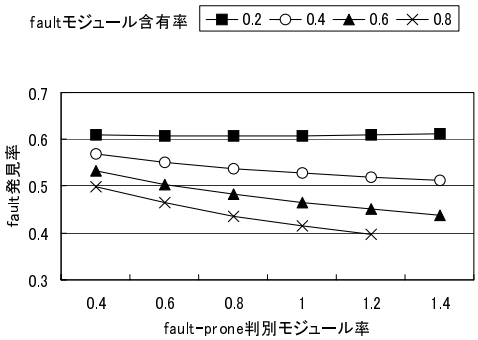

図 3　F1 値固定時の fault 発見率

数の fault 含有モジュールに対する割合（fault-prone 判別モジュール率），および，fault 含有モジュールの全モジュールに対する割合（fault-prone 判別モジュール含有率）を変化させて，fault 発見率の変化を確かめた．

このシミュレーションは，ロジスティック回帰分析などの fault-prone モジュール判別モデルによって，各モジュールが fault を含む確率（期待値）を求め，一定の閾値で fault あり／なしを判別するケースを想定している．このケースでは，閾値を変化させることで，fault-prone と判定するモジュールの割合（fault-prone 判別モジュール率）を変化させることができる．

シミュレーションでは，F1 値を 0.5 に固定した．そして，F1 値が 0.5 となるように，fault 含有モジュール数，fault-prone 判別モジュール数，正答モジュール数を変化させた．

3.3.2　結果

シミュレーションの結果を図 3 に示す．グラフ中のプロットが存在しない箇所は，正答モジュール数が fault モジュール含有率や fault-prone 判別モジュール数を超える場合である．シミュレーションの結果，同じ F1 値をとる場合でも，fault-prone 判別モジュール率や，fault モジュール含有率が異なると fault 発見率は大きく異なっている．

Fault モジュール含有率が高い場合には，fault-prone 判別モジュール率を高くするほど同じ F1 値でも発見される fault は減少している．しかし，fault モジュール含有率が低い場合には，発見される fault に大きな差は見られない．また，fault モジュール含有率が低い場合には，fault-prone 判別モジュール率を高くすると，最初は発見される fault は減少するが，fault-prone 判別モジュール率をより高くすることで発見される fault が増加している．これは，fault-prone 判別モジュール率を高くすることで，適合率は低下するが再現率が向上し，より多くの fault 含有モジュールに対して fault-prone 判別モジュールに費やされるテスト工数が割り当てられることになるためである．

3.4　シミュレーション 3：条件を固定して F1 値を変化させた場合

3.4.1　概要

シミュレーション 3 では，fault 含有モジュール数や fault-prone 判別モジュール数を固定し，F1 値を変化させることで fault 発見率がどのように変化するかを明らかにする．そのために，fault-prone 判別モジュール率，および，fault モジュール含有率ごとに F1 値の値を変化させて fault 発見率の変化を確かめた．

シミュレーションでは，正答モジュール数を F1 値ごとに変化させた．

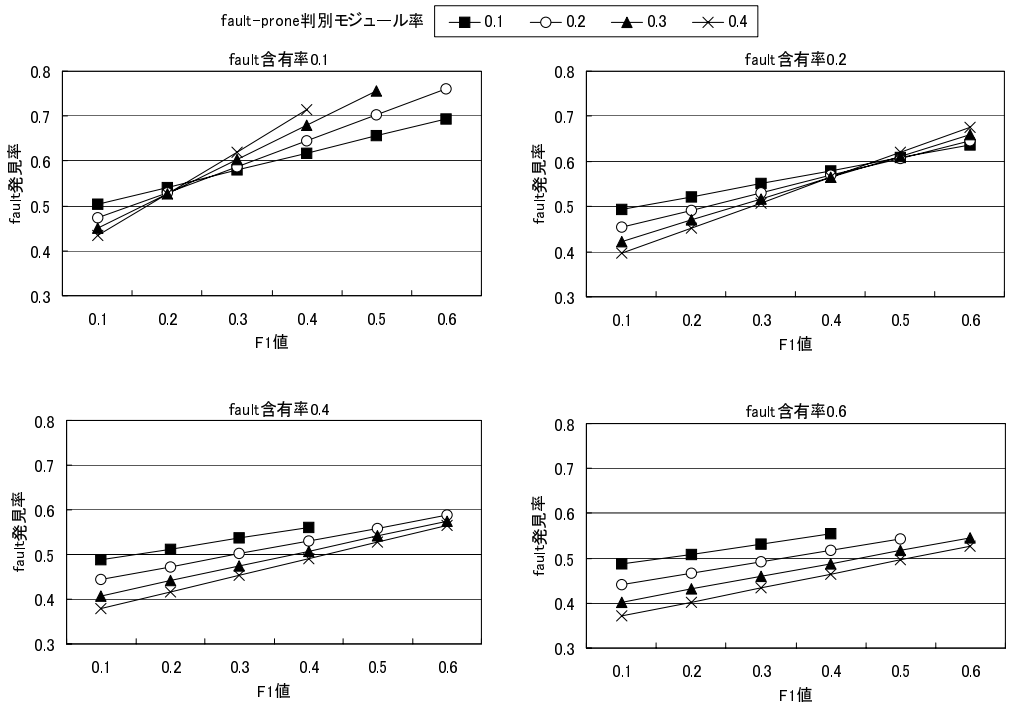

図 4　F1 値を変化させたときの fault-prone 判別モジュール率ごとの fault 発見率

3.4.2　結果

シミュレーションの結果を図 4 に示す．グラフ中のプロットが存在しない箇所は，正答モジュール数が fault モジュール含有率や fault-prone 判別モジュール数を超える場合である．シミュレーションの結果，fault-prone 判別モジュール率を高くするほど，F1 値の変化に伴う fault 発見率の変化は大きく，fault-prone 判別モジュール率が低いほど，F1 値の変化に伴う fault 発見率の変化は小さい．また，F1 値が小さいときには fault-prone 判別モジュール率が低い方が fault 発見率は小さいが，F1 値が増加するほど fault-prone 判別モジュール率による差は小さくなり，ある F1 値を境目に，fault-prone 判別モジュール率が高い方が fault 発見率が高くなる．境目となる F1 値の値は，fault モジュール含有率の上昇に伴って大きくなる．

4　考察

シミュレーション 1 の結果から，fault-prone モジュール判別モデルの精度（F1 値）が低いと予想される場合には，fault-prone 判別モジュールと non-fault-prone 判別モジュールに均等にテスト工数を割り当てることで（ソフトウェア全体における）fault 発見率を高くでき（すなわち，fault-prone モジュール判別を行わない方が良い），予想される F1 値が高くなるにつれて，fault-prone 判別モジュールにテスト工数を重点的に割り当てることで fault 発見率が高くできると言える．

シミュレーション 2 の結果から，fault モジュール含有率が低いと予想される場合には fault-prone 判別モジュール率を高くする（fault-prone と判別するモジュール数を多くする）ことで fault 発見率を高くでき，予想される fault モジュール含有率が増加するにつれて fault-prone 判別モジュール率を低くする（fault-prone と判別するモジュール数を少なくする）ことで fault 発見率を高くできると言える．

シミュレーション 3 の結果から，F1 値が低いと予想される場合には，再現率を重視して fault-prone 判別モジュール率を低くすることで，fault 発見率を高くできると言え，予想される F1 値が増加するにつれて，適合率を無視して fault-prone 判別モジュール率を高くすることで，fault 発見率を高くできると言える．また，fault モジュール含有率に関してはシミュレーション 2 と同様の結果が得られている．

これらのシミュレーションの結果から，プロジェクト全体の信頼性（fault 発見率）は，F1 値や，fault モジュール含有率，fault-prone 判別モジュール率によって決定されると言える．従って，テストの効率化や信頼性の向上のためには，fault モジュール含有率，fault-prone 判別モジュール率を考慮したうえで F1 値を上昇させる必要があると考えられる．

ただし，fault-prone モジュール判別モデルの比較などにおいては，fault-prone 判別モジュール率を一定とすることは難しく，また，fault-prone 判別モジュール率の変化に伴って判別精度も変化する可能性が高い．そのため，fault-prone 判別モジュール率を固定することも，fault-prone モジュール判別モデルの性能を制限する可能性がある．fault モジュール含有率や fault-prone 判別モジュール率が判明していれば，本稿で用いたような何らかのモデル式等によって fault 発見率等の性能が得られる．従って，fault-prone モジュール判別モデルの評価などにおいては，F1 値だけでなく，fault モジュール含有率や fault-prone 判別モジュール率を F1 値と併記することが望ましいと言える．

また，本稿のシミュレーションで得られた結果に基づいたテストの効率化や信頼性の向上が行えると考えられる．予想される fault モジュール含有率や F1 値に基づいて，fault-prone 判別モジュールに割り当てるテスト工数の割合を変化させたり，fault-prone モジュール判別モデルをチューニングすることで fault-prone 判別モジュール率を変化させることで，fault 発見率の増加が見込めると考えられる．

5 おわりに

本稿では，fault-prone モジュール判別において F1 値の変化がテスト効率や信頼性の向上にどのような影響を及ぼすかを明らかにするために，fault の有無，テスト工数，fault 発見率をモデル化し，F1 値と信頼性の関係をシミュレーションで明らかにした．シミュレーションの結果，プロジェクト全体の信頼性（fault 発見率）は，F1 値や，fault モジュール含有率，fault-prone 判別モジュール率によって決定されることが判明した．

本稿では，比較的簡単なモデルである指数関数モデルを基に fault 発見率モデルを構築した．今後，テスト工数指数関数以外のモデルの構築や F1 値以外の評価指標との関係についても検討していきたい．また，fault 発見率モデルの妥当性を示すため，実際のソフトウェア開発プロジェクトにおけるテスト工程の結果を用いて実証実験を行うことが今後の課題である．

謝辞 本研究の一部は，文部科学省「e-Society 基盤ソフトウェアの総合開発」の委託に基づいて行われた．

参考文献

[1] 独立行政法人 情報処理推進機構 ソフトウェア・エンジニアリング・センター，ソフトウェア開発データ白書２００６ ～IT 企業 1400 プロジェクトの定量データで示す開発の実態～，日経 BP 社，東京，2006.

[2] A.L. Goel and K. Okumoto, ”Time-dependent errordetection rate model for software reliability and other performance measures,” IEEE Trans. Reliability, Vol.28, No.3, pp.206-211, 1979.

[3] A.R. Gray and S.G. MacDonell, ”Software metrics data analysis–exploring the relative performance of some commonly used modeling techniques,” Empirical Softw. Eng., Vol.4, No.4, pp.297–316, 1999.

[4] J.L. Herlocker, J.A. Konstan, L.G. Terveen and J.T. Riedl, "Evaluating collaborative filtering recommender systems," ACM Trans. Information Systems, Vol.22, No.1, pp.5–53, 2004.
[5] Y. Kamei, A. Monden and K. Matsumoto, "Empirical evaluation of svm-based software reliability model," Proc. of the 5th ACM-IEEE Intl. Symposium on Empirical Softw. Eng. (ISESE2006), Vol.2, pp.39–41, Rio de Janeiro, Brasil, 2006.
[6] 亀井 靖高, 松本 真佑, 柿元 健, 門田 暁人, 松本 健一, "Fault-Prone モジュール判別におけるサンプリング法適用の効果," 情報処理学会論文誌, Vol.48, No.8, pp.2651-2662, 2007.
[7] T.M. Khoshgoftaar, and E.B Allen, "Modeling software quality with classification trees," Recent Advances in Reliability and Quality Eng., World Scientific, pp.247–270, Singapore, 1999.
[8] P.L. Li, J. Herbsleb, M. Shaw and B. Robinson, "Experiences and results from initiating field defect prediction and product test prioritization efforts at ABB Inc," Proc. 28th Int'l Conf. on Softw. Eng. (ICSE'06), pp.413–422, Shanghai, China, 2006.
[9] S. Matsumoto, Y. Kamei, A. Monden and K. Matsumoto, "Comparison of outlier detection methods on fault-proneness models," Proc of the 1st Intl. Symposium on Empirical Softw. Eng. and Measurement (ESEM2007), (to appear).
[10] J.C. Munson and T.M. Khoshgoftaar, "The detection of fault-prone programs," IEEE Trans. Softw. Eng., Vol.18, No.5, pp.423–433, 1992.
[11] NASA IV&V Facility, Metrics Data Program, http://mdp.ivv.nasa.gov/
[12] N. Ohlsson and H. Alberg, "Predicting fault-prone software modules in telephone switches," IEEE Trans. Softw. Eng., Vol.22, No.12, pp.886–894, 1996.
[13] C.V. Ramamoorthy and F.B. Bastani, "Software reliability-status and perspective," IEEE Trans. Softw. Eng., Vol.8, No.4, pp.354–371 1982.