ソフトウェア信頼性評価における最近の話題

広島大学工学部第二類（電気系）
土肥 正

1 はじめに

信頼性技術は，機械工学，土木建築学，電気工学，コンピュータサイエンス等あらゆる工学的諸分野において発達してきた固有技術として認識されており，産業全般における基盤技術として定着しつつある．とりわけ，オペレーションズ・リサーチ (OR) において取り扱われる信頼性理論は，システムの信頼性設計·評価，保全スケジュールの生成等に関連した問題に対して数理的な手法を提供することを指針として，1950 年代から現在に至まで様々な経緯で発展してきた．従来からの OR における信頼性理論では，ハードウェアとしてのシステムが故障するかもしれない不確実な環境を確率的現象として如何にモデル化し，信頼性の観点からどのようにシステムを設計·評価するかに主眼がおかれてきた．

しかしながら，生産技術の飛躍的な向上により，ハードウェアに関してはかなり高品質の製品を供給することが可能となった現在において，最も立ち遅れているのがソフトウェアの信頼性評価であるといっても過言ではない．それは，ソフトウェアは依然として人間による知的生産物であり，ソフトウェアに含まれるフォールト（もしくはバグ）によって誘発されるソフトウェア故障のメカニズムを完全に解明することが極めて困難であるという事実に起因している．しかも，高度情報化社会と呼ばれる現代社会において，大規模かつ複雑なシステムのほとんどがソフトウェアによって制御されている現状を鑑みれば，ソフトウェアの信頼性は今後益々重要視されるものと考えられる．

本稿の目的は，ソフトウェア信頼性評価における話題の中でも特に，最近得られたいくつかの研究成果を確率モデルに焦点をあてて紹介することである．いうまでもなく，ソフトウェア信頼性評価理論はソフトウェア工学の中心的な課題であり，大きく分けて，(i) ソフトウェアの開発過程の最終段階であるテスト工程において採集されたフォールト発見データから，如何にソフトウェア製品の信頼性を定量的に評価するか，(ii) ソフトウェアに残存するフォールトの存在を予め認識した上で，フォールトトレランス（耐故障性）および安全性の観点から如何にシステムを構成するか，という問題に分類される．前者はソフトウェア製品の品質管理技術に対応しており，後者は N-バージョンプログラミングやリカバリブロック等の設計技術に対応している．本稿では特に (i) の問題に焦点を絞り，歴史的な経緯，問題点および最近の成果についてサーベイするものとする．

2 ソフトウェア信頼性評価と確率モデル

2.1 ソフトウェア信頼度成長モデル

ソフトウェアテストは大きく分けて，Black-Box テストと White-Box テストに大別される．前者がプログラムの内部構造や動作に依存することなく要求仕様だけからテストデータを投入するのに対し，後者はプログラムに含まれる論理経路を徹底的に調べ上げることによってフォールトを検出する．もちろん，ひとつのプログラム中に含まれる論理経路の数は天文学的な値であるため，White-Box テストだけを実行することは不可能であり，Black-Box テストに基づいた効果的なテストケースを設計することが極めて重要な問題となる．このような背景から，ソフトウェアに含まれるフォールトを完全に除去することは事実上不可能であり，ソフトウェアの開発工程において製品の信頼性を定量的に評価する必要が生じる．

しかしながら，ソフトウェアがハードウェア製品と大きく異なる点は，ソフトウェアが依然として人間による知的生産物であり，ソフトウェアに含まれるフォールトによって誘発されるソフトウェア故障のメカニズムを完全に解明することが極めて困難であることである．さらに，ソフトウェアのテスト（デバッギング）

には多大な労力と費用がかかるため，同一の製品を複数のデバッグチームによって並列的にテストすることは通常なされないので，製品の信頼性を評価するための十分なデータが得られないのが現状である．いま，テスト時間 $t\,(>0)$ までに $n\,(>0)$ 個のソフトウェアフォールト発見時刻データ $x_1, x_2, \cdots, x_n$ が得られているものとすると，ソフトウェアの信頼性はテストデータセット $D_t = \{n, x_1, x_2, \cdots, x_n, t\}$ を用いて評価されなければならない．よって，現在までにテスト工程におけるフォールト発見過程を表現するために様々な種類のソフトウェア信頼度成長モデルと呼ばれる確率モデルが提案されてきた．これは，テスト時間の経過とともにソフトウェア内に潜在するフォールト数は発見されて減少し，結果としてソフトウェア信頼度が増加する現象を記述するものである．

Jelinski and Moranda [JM-72] が最初の論文を発表して以来，現在までに実に 200 を超えるソフトウェア信頼度成長モデルが提案されている．山田 [Y-91] によれば，ソフトウェア信頼度成長モデルは大きく (i) 時間計測モデル, (ii) 個数計測モデル, (iii) アベイラビリティモデル のように分類できる．いま，i 番目と $i-1$ 番目のフォールトが発見される時間間隔 $\{S_i\}_{i=1}^{\infty}$ は独立な確率変数列であり，S_i のハザード率を $h_i(t)$ と定義する．(i) の時間計測モデルに分類される代表的なモデルとして，

· Jelinski and Moranda モデル [JM-72]: $h_i(t) = \phi\{N_0 - (i-1)\}$, $(\phi > 0,\ 0 < N_0 < \infty, i = 1, 2, \cdots, N_0)$

· Schick and Wolverton モデル [SW-78]: $h_i(t) = \phi\{N_0 - (i-1)\}t$

· Littlewood モデル [L-80]: $h_i(t) = \phi\{N_0 - (i-1)\}, \phi \sim K(a,b) = 1 - \int_s^{\infty} b^a x^{a-1} e^{-bx} dx/\Gamma(a)$

が挙げられる．(ii) の個数計測モデルにおける代表例は，非同次ポアソン過程 (NHPP) に従うモデルであろう．すなわち，時刻 t までに発見された総フォールト数を $\{N(t), t \geq 0\}$ とすると，

$$\Pr\{N(t) = n\} = \frac{[M(t)]^n}{n!} \exp\{-M(t)\} \tag{1}$$

を仮定する．ここで，$M(t) = \mathrm{E}[N(t)]$ であり，$\lim_{t\to\infty} M(t) < \infty$ の場合をタイプ I モデル，$\lim_{t\to\infty} M(t) \to \infty$ の場合をタイプ II モデルと呼ぶ．タイプ I モデルの中でよく知られているものとして，

· 指数型モデル [GO-79]: $M(t) = N_0\{1 - \exp(-\phi t)\}$, $(\phi > 0,\ 0 < N_0 < \infty)$

· 遅延 S 字型モデル [YO-85]: $M(t) = N_0\{1 - [1 + \phi t]\exp(-\phi t)\}$

があり，タイプ II モデルとしては

· 対数型ポアソン実行時間モデル [MIO-87]: $M(t) = (1/\theta)\log\{\lambda\theta t + 1\}$, $(\lambda > 0,\ \theta > 0)$

が有名である．

2.2 問題点

2.1 で紹介したように，ソフトウェアテストにおいてフォールトが出現するふるまいを表現した確率モデルが数多く提案されている一方で，このようなモデルが実際の開発工程において頻繁に利用されているとは限らない．その主な理由として，ソフトウェア製品のフォールト発見時間データとして学術用に公開されているものが極めて少ないため，複数の評価基準を用いて統計的に有為な最良モデルを選定することが困難であった点が挙げられる．さらに，

(a) 文献において提案されたモデルのほとんどは，“直感的な思いつき” や “データの局所的なふるまい” を観測することによって得られたものであり，ソフトウェアに含まれるフォールトの発見事象を物理的もしくは統計的に説明する根拠に乏しい

(b) 例え，ソフトウェアのフォールト発見事象を説明するためのモデルが特定できたとしても，モデルパラメータを合理的に推定するための統計的な手法が完備されていない

等の指摘がなされている．上記の問題点 (a) を解決するためには，従来から提案されてきたソフトウェア信頼度成長モデルを統一的に理解するための一般的な方法論を確立する必要がある．また，問題点 (b) は 90 年代になって主に応用統計学者によって活発に議論され始めた問題であり，モデルを現実の信頼性評価に応用する上で最も重要な課題であるといえる．次節では，主に (a) で挙げた問題点に焦点をあてる．具体的には，従来からよく知られているソフトウェア信頼度成長モデルを異なる観点から統合的に説明するための確率モデルについて述べる．

3 ソフトウェア信頼度成長モデルの一般化

ここでは，**2.1** で紹介した古典的なソフトウェア信頼度成長モデルを特別な場合として包含するような確率モデルについて述べる．Shanthikumar [S-81] や Langberg and Singpurwalla [LS-85] は Jelinski and Moranda モデル [JM-72] が，指数型モデル [GO-72] のような NHPP に基づくソフトウェア信頼度成長モデルを特別な場合として含むことを示している．本節では，さらに数多くのソフトウェア信頼度成長モデルを特殊な場合として説明できるような一般化されたモデルについて紹介する．

3.1 一般化順序統計量モデルと記録値統計量モデル

モデル化のために次のような仮定を設定する．

(A-4-1) テスト中にひとつのソフトウェア故障が発生したとき，その原因となる (ひとつの) フォールトは瞬間的に除去される．

(A-4-2) プログラム中に含まれる初期フォールト数は N_0 (>0) (未知数) である．

(A-4-3) ソフトウェア故障は各々独立かつ時間に関してランダムに発生し，そのハザード率は $h(t)$ $(t \geq 0)$ である．

$\{T_i\}_{i=0}^{N_0} = \{\sum_{j=1}^{i} S_j\}_{i=0}^{N_0}$ をフォールト i が発見されるまでの時間を表す独立で同一の確率変数列とし，それらの確率分布関数を $F(t) = \Pr\{T_i \leq t\}$ とすれば，

$$\Pr\{N(t) = n\} = \binom{N_0}{n} \{F(t)\}^n \{1 - F(t)\}^{N_0 - n} \tag{2}$$

となる．ここで，2 項分布の性質から，

$$\mathrm{E}[N(t)] = N_0 F(t), \quad \mathrm{Var}[N(t)] = N_0 F(t)\{1 - F(t)\} \tag{3}$$

である．Raftery [R-87], Joe [J-89], Kuo and Yang [KY-96] は上述の 2 項分布モデルが NHPP に基づく既存のソフトウェア信頼度成長モデルに帰着されることを示した．例えば，上記仮定 (A-4-2) を

(A-4-2') プログラム中に含まれる初期フォールト数 N_0 (>0) は平均 ω_0 (>0) のポアソン確率変数である

に変更することにより，

$$\Pr\{N(t) = n\} = \frac{[\omega_0 F(t)]^n}{n!} \exp\{-\omega_0 F(t)\} \tag{4}$$

を得る．このことは，仮定 (A-4-2') の下で，確率過程 $\{N(t), t \geq 0\}$ の確率分布関数は平均値関数 $M(t) = \omega_0 F(t)$ をもつ NHPP と等価であることを示唆しており，タイプ I に属するほとんどのソフトウェア信頼度成長モデルを表現することが可能となる．

次に，タイプ II に属する NHPP モデルについて考える．先に述べたフォールト発見時間分布を $F(t) = \Pr\{T_i \leq t\}$ とすると，データ列 $\{T_i\}_{i=0}^{\infty}$ に対する記録値 (record values) $\{\tau_i\}_{i=0}^{\infty}$ は

$$\tau_{k+1} = \min\{i; T_i > T_{\tau_k}\}, \quad \tau_1 = 1, \ k = 1, 2, \cdots \tag{5}$$

のように定義される．よく知られた結果として，$F(t)$ が連続であれば，時刻 t での記録値 $\tau(t)$ は

$$\Pr\{\tau(t) = n\} = \frac{[-\log(1 - F(t))]^n}{n!} \exp\{\log(1 - F(t))\} \tag{6}$$

となる．このことは，記録値 $\{\tau_i\}_{i=0}^{\infty}$ の確率分布関数は平均値関数 $M(t) = -\log(1 - F(t))$ をもつ NHPP と等価であることを意味しており，やはり $F(t)$ の形状によりタイプ II に属する種々のソフトウェア信頼度成長モデルを表現することが可能となる．

3.2 無限サーバ待ち行列モデル

ここでは前節とは若干異なる観点からソフトウェア信頼度成長モデルを説明することを考える．開発されたソフトウェア製品に対して十分な数のテストケースが用意されており，テストケースはパラメータ $\lambda\ (> 0)$ の同次ポアソン過程に従って投入されるものとする．ひとつのテストケース $i\ (= 1, 2, \cdots)$ には X_i 本の論理経路に反応するデータが含まれており，独立で同一の確率分布 $G(t)$ に従う時間で各経路内を探索する．ここで，X_i は確率分布 $P(x)$ に従う非負の離散確率変数であり，$G(t)$ は $t \in [0, \infty)$ で絶対連続かつ非減少の確率分布関数である．投入された X_i 個のデータの中で時刻 t までに探索が終了していない個数（探索中である経路の本数）を発見されたフォールトの数として考えると，これは無限数のサーバを伴う $M^X/G/\infty$ 待ち行列システムと等価である．

いま，時刻 0 でテストが開始され，任意の時刻 t においてソフトウェア内で発見されたフォールトの数を $\{N(t), t \geq 0\}$ とする．このとき，確率過程 $N(t)$ はパラメータ λ，ショックの幅 X_i の複合 NHPP となり，

$$\Pr\{N(t) = k\} = \sum_{n=0}^{\infty} e^{-\lambda t} \frac{(\lambda t)^n}{n!} \Pr\{Z_1 + Z_2 + \cdots + Z_n = k\} \tag{7}$$

となる．ここで，

$$\Pr\{Z_i = j\} \quad = \quad \int_0^t \frac{1}{\lambda t} \sum_{r=j}^{\infty} P(r) \binom{r}{j} (1 - G(t - x))^j G(t - x)^{r-j} d(\lambda x) \tag{8}$$

である．簡単のため，X_i が恒等的に 1 に等しい（すなわち，$M/G/\infty$ システム）とすれば，

$$\Pr\{N(t) = n\} = \frac{[\lambda \int_0^t (1 - G(x))dx]^n}{n!} \exp\left\{-\lambda \int_0^t (1 - G(x))dx\right\} \tag{9}$$

となり，確率分布関数 $G(t)$ の選び方によってタイプ I とタイプ II の両方のモデルを表現することができる．

3.3 Self-Exciting Point Processes による統合化モデル

最近，Chen and Singpurwalla [CS-97] は自己誘発点過程 (Self-Exciting Point Processes) によって，既存のソフトウェア信頼度成長モデルを説明することを試みている．計数過程 $\{N(t), t \geq 0\}$ は時間間隔 $[0, t)$ においてソフトウェアフォールトが発見される個数を表し，$\{T_i\}_{i=0}^{\infty}$ を i 番目の事象が発生する時刻を表す増加列とし，確率過程 $N(t)$ の履歴を $D(t) = \{N(t), T_1, \cdots, T_{N(t)}\}$ によって表現する．

定義 1: $Q(t) \subseteq D(t), t \geq 0$ に対して

$$\Pr\{N(t + h) - N(t) \geq 2 \mid Q(t)\} = \Pr\{N(t + h) - N(t) = 1 \mid Q(t)\} o(h^0) \tag{10}$$

ならば，確率過程 $\{N(t), t \geq 0\}$ は conditional orderliness 特性をもつといわれる．ここで，$\lim_{h\to 0} o(h^i)/h^i = 0, i = 0, 1, 2, \cdots$，である．

定義 2: 確率過程 $\{N(t), t \geq 0\}$ が

(i) $N(0) = 0$,

(ii) $\Pr\{N(t+h) - N(t) = 1 \mid \Lambda(t), D(t)\} = \mathrm{E}[\Lambda(t) \mid D(t)]h + o(h)$,

(iii) $\Pr\{N(t+h) - N(t) \geq 2 \mid \Lambda(t), Q(t)\} = \Pr\{N(t+h) - N(t) = 1 \mid \Lambda(t), Q(t)\}o(h^0)$

を満たすならば，強度関数 $\Lambda(t)$ をもつ自己誘発点過程 (SEPP) であるといわれる．

定義 3: 強度関数 $\Lambda(t)$ をもつ自己誘発点過程 $\{N(t), t \geq 0\}$ が m 次のメモリをもつとは，$\Lambda(t)$ と $D(t)$ に関して，(i) $m \geq 2$: $(\Lambda(t) \mid D(t)) = (\Lambda(t) \mid N(t), T_{N(t)}, S_{N(t)}, \cdots, S_{N(t)-m+2})$, (ii) $m = 1$: $(\Lambda(t) \mid D(t)) = (\Lambda(t) \mid N(t), T_{N(t)})$, (iii) $m = 0$: $(\Lambda(t) \mid D(t)) = (\Lambda(t) \mid N(t))$, (iv) $m \to \infty$: $(\Lambda(t) \mid D(t)) = \Lambda(t)$ のような関係が成立することである．

定理: ソフトウェアのフォールト発見時刻列 $\{T_i\}_{i=0}^{\infty}$ がそれぞれハザード率 $h_i(t)$ をもち，conditional orderliness 特性をもつ計数過程ならば，強度関数 $\Lambda(t \mid D(t)) = h_{N(t)+1}(t - T_{N(t)} \mid D(t))$ の自己誘発点過程となる．

これより，従来から提案されてきたソフトウェア信頼度成長モデルは自己誘発点過程の枠組みにおいて完全に説明される．すなわち，NHPP に基づいた全てのソフトウェア信頼度成長モデルは $-\infty$ 次のメモリをもつ自己誘発点過程（すなわち，NHPP $\subset$ SEPP）であり，Jelinski and Moranda モデル と Schick and Wolverton モデル [SW-78], Littlewood モデル [L-80] は，それぞれ 0 次と 1 次の自己誘発点過程となる．

4 その他の話題

本稿では，ソフトウェア製品の品質ならびに信頼性評価を行うために開発された確率モデルに焦点を絞り，その大まかな経緯から最近の成果までを概説した．ここでは紙面の都合から，**2.2** の問題 (b) にふれることが出来なかったが，モデルパラメータの合理的な推定はソフトウェア信頼度成長モデルによる評価の是非を左右する最重要課題であり，点推定，区間推定，ベイズ推定に関する数多く興味深い結果が報告されている．

ソフトウェア信頼性理論では，**1** で述べた研究領域に加えて，OR における最適化の諸技法が直接応用されている事例が少なくない．例えば，

- ソフトウェア最適出荷問題（動的計画法，最適停止問題，ゲーム理論）
- モジュールやテスト労力の配分問題（非線形計画法，AHP）
- 信頼性予測（時系列解析，ニューラルネットワーク）

等が挙げられるであろう．上述のいくつかの話題に関する詳細は，近々OR 学会誌上で特集が組まれる予定であるので，そちらを参照して頂きたい．また，80 年代後半から新たに展開された Recapture デバッギング法は，ソフトウェアのテスト段階で得られる統計的な情報量を増大させるためには有効であり，このような知見が今後実際のテスト方法に応用されることが期待できる．さらに，テストケースの投入順序問題や運用段階における信頼性評価等，ソフトウェア信頼性における未解決問題が数多く残されていることを付記しておく．

参考文献

[CS-97] Y. Chen and N. D. Singpurwalla, Unification of software reliability models by self-exciting point processes, *Adv. Appl. Prob.*, 29, 337-352 (1997).

[GO-79] A. L. Goel and K. Okumoto, Time-dependent error-detection rate model for software reliability and other performance measures, *IEEE Trans. Reliab.*, R-28, 206-211 (1979).

[J-89] H. Joe, Statistical inference for general-order-statistics and nonhomogeneous-Poisson-process software reliability models, *IEEE Trans. Software Eng.*, 15, 1485-1490 (1989).

[JM-72] Z. Jelinski and P. B. Moranda, Software reliability research, in *Statistical Computer Performance Evaluation*, W. Freiberger (ed.), pp. 465-484, Academic Press, New York (1972).

[KY-96] L. Kuo and T. Y. Yang, Bayesian computation nonhomogeneous Poisson processes in software reliability, *J. Amer. Statist. Assoc.*, 91, 763-773 (1996).

[L-80] B. Littlewood, Theories of software reliability: how good are they and how can they be improved?, *IEEE Trans. Software Eng.*, SE-6, 489-500 (1980).

[LS-85] N. Langberg and N. D. Singpurwalla, Unification of some software reliability models, *SIAM J. Sci. Comput.*, 6, 781-790 (1985).

[MIO-87] J. D. Musa, A. Iannino and K. Okumoto, *Software Reliability, Measurement, Prediction, Application*, McGraw-Hill, New York (1987).

[R-87] A. E. Raftery, Inference and prediction for a general order statistic model with unknown population size, *J. Amer. Statist. Assoc.*, 82, 1163-1168 (1987).

[S-81] J. G. Shanthikumar, A general software reliability model for performance prediction, *Microelectron. Reliab.*, 21, 671-682 (1981).

[SW-78] G. J. Schick and R. Wolverton, An analysis of competing software reliability models, *IEEE Trans. Software Eng.*, SE-4, 104-120 (1978).

[Y-91] 山田茂, ソフトウェアの品質評価に関する考え方と動向, 情報処理, 31, 1189-1202 (1991).

[YO-85] S. Yamada and S. Osaki, Software reliability growth modeling: models and applications, *IEEE Trans. Software Eng.*, SE-11, 1431-1437 (1985).