日本オペレーションズ・リサーチ学会
2004年秋季研究発表会

ソフトウェア開発プロジェクトにおける離散最適テスト／保全方策

林坂弘一郎 (01800035)，土肥正 (01307065)
広島大学大学院工学研究科情報工学専攻

1. はじめに

ソフトウェアテスト工程で発見されたフォールト数に関するデータを分析し，ソフトウェアの信頼性を定量的に評価するとともに，運用段階に移行するための最適な時期を決定することは必要不可欠である．このような問題は最適リリース問題と呼ばれ，開発プロジェクトの管理者にとってプロジェクトの成否を左右する重要な問題である．

実際のテスト工程ではソフトウェア内に潜在する全てのフォールトを発見・除去することが極めて困難であるため，リリース後にソフトウェアフォールトに起因する障害が発生することを避けることは容易ではない．多くの場合，開発管理者はユーザとの間で結ばれた保守・保証契約に基づき，リリース後に発生するであろうソフトウェア障害の原因究明を行い，フォールトの発見・除去を行う必要がある．リリース後の運用段階において保全を実施するために，ソフトウェア開発管理者はプロジェクトチームの維持を継続しなければならない反面，運用段階における管理費用を削減し，人的資源を有効に活用することが要求される．この意味において，プロジェクトチームを維持継続する期間を決定する問題はリリース時刻を決定することと同様に重要であるが，これまでにあまり考えられることはなかった．

Kimura ら [1] は，ソフトウェア保証期間が確率変数である場合を想定し，最適リリース時刻を決定する問題を考えている．Dohi ら [2] はテスト工程におけるデバッグ過程が非同次ポアソン過程 (NHPP) によって記述されるという仮定の下で，最適保証期間を決定する問題を定式化している．

本稿では，文献 [3] によって提案された運用段階における連続時間環境下の信頼性評価モデルと同様な方法に基づき，離散時間環境に対してテスト段階と運用段階におけるソフトウェア実行環境の違いを運用プロファイルとしてモデル化する．次に離散時間 NHPP に基づいた総期待ソフトウェア費用を最小化する最適テスト期間（リリース時刻）を解析的に導出する．また，プロジェクトチームの維持継続を完了する時点を計画保守限界 (planned maintenance limit) と呼ぶこととし，総期待ソフトウェア費用を最小にする最適計画保守限界を決定する．最終的に数値例において，テスト期間及び計画保守限界の同時最適解を求める．

2. モデルの記述

ソフトウェアフォールトの発見過程を記述するために次の仮定を設定する．

(仮定 **A**) ソフトウェアのテスト中にフォールトが発見された場合，瞬間的に修正・除去される．

(仮定 **B**) プログラム中に含まれる初期フォールト数 N_0 は平均 ω (>0) のポアソン分布に従う．

(仮定 **C**) ソフトウェアフォールトは各々独立かつ時間に関してランダムに発見され，各ソフトウェアフォールトの発見時刻は確率分布 $P(i)=\sum_{k=1}^{i} p_k$ に従う離散確率変数によって記述される．

以上の仮定により，時刻 i までに検出・修正されるソフトウェアフォールトの累積数を N_i, $i=0,1,2,\cdots$ で表すと，N_i の確率関数 (p.m.f.) は

$$\Pr\{N_i=m\}=\frac{\{\omega P(i)\}^m e^{-\omega P(i)}}{m!} \quad m=0,1,2,\cdots \quad (1)$$

によって表現される．これにより，確率過程 N_i は平均値関数 $\omega P(i)$ をもつ離散時間 NHPP となる．

ソフトウェアの開発工程において，時刻 0 でテストを開始し，n_0 (≥ 0) 時間のテストを行った後，当該製品をユーザもしくは市場にリリースすることを考える．ソフトウェア製品のライフサイクル n_L (>0) は既知であるとする．開発管理者は保守契約に基づき，リリース直後からソフトウェア製品のライフサイクル終了までの期間 $(n_0, n_0+n_L]$ に発生したソフトウェア障害に対する保守費用を負担する．リリース直後から n_W $(0\leq n_W\leq n_L)$ の期間（すなわち，時刻 n_0+n_W まで），開発管理者はプロジェクトチームの維持を継続する．以降では n_W を計画保守限界と呼ぶこととする．最終的に時刻 n_0+n_W において，運用段階での管理費用を削減するためにプロジェクトチームを解散する．

テスト期間中に発生するフォールト 1 個当りのデバッグ費用を c_0 (>0)，計画保守限界以前に発見されたフォールト 1 個当りのデバッグ費用を c_W (>0) とし，計画保守限界以降に発見されたフォールト 1 個当りのデバッグ費用を c_L (>0) で表す．また，単位時間当りテスト費用を k_0 (>0)，単位時間当りプロジェクトチーム維持継続費用を k_W (>0) とする．

3. 総期待ソフトウェア費用の定式化

一般に，ソフトウェアの運用環境とテスト段階におけるデバッグ環境は異なると考えられており，これらの相違点はハードウェアにおける加速寿命試験環境と通常操作環境の違いに類似している．ここでも文献 [3] と同様に，リリース後の運用環境に対する相対的な厳しさ表すパラメータ a (>0) を環境係数と呼び，ソフトウェアの運用プロファイルを単一のパラメータによってモデル化する．テスト段階と運用段階における時間が単純な比例関係にあると考える．ここで，$a=1$ の場合は運用環境とテスト環境はほぼ同様であることを意味し，$a>1$ $(a<1)$ は運用環境がテスト環境よりも厳しい（緩い）

ことを示している．以上のような仮定の下で，運用段階における計画保守限界以前の期間に発見・除去されるソフトウェアフォールト数は

$$\Pr\{N_{n_0+n_W} - N_{n_0} = m\} = \frac{\{\omega\{P(n_0+[an_W]) - P(n_0)\}\}^m}{m!} \times e^{-\omega\{P(n_0+[an_W])-P(n_0)\}} \quad (2)$$

となる．但し，$[\cdot]$ は Gauss 記号である．

同様に，計画保守限界以降のソフトウェアフォールト発見過程の振る舞いは

$$\Pr\{N_{n_0+n_L} - N_{n_0+n_W} = m\} = \frac{\{\omega\{P(n_0+[an_L]) - P(n_0+[an_W])\}\}^m}{m!} \times e^{-\omega\{P(n_0+[an_L])-P(n_0+[an_W])\}} \quad (3)$$

によって表現される．

以上のことから，総期待ソフトウェア費用 $C(n_0, n_W)$ は次式によって定式化される．

$$\begin{aligned} C(n_0, n_W) &= k_0 n_0 + c_0\omega P(n_0) + k_W n_W \\ &\quad + c_W\omega\{P(n_0+[an_W]) - P(n_0)\} \\ &\quad + c_L\omega\{P(n_0+[an_L]) - P(n_0+[an_W])\}. \end{aligned} \quad (4)$$

4. 最適テスト期間及び最適計画保守限界の決定

ここでは総期待ソフトウェア費用を最小化する最適テスト期間 n_0^* 及び最適計画保守限界 n_W^* についての解析結果を示す．各々のソフトウェアフォールトが発見されるまでの時間がパラメータ b (>0) の幾何分布に従うとし，環境係数 a は正の整数とする．また，次の仮定を設定する．

(A-I) $c_L > c_W > c_0$,

(A-II) $c_W\{1-(1-b)^{an_L}\} > c_0$,

(A-III) $c_W\{1-(1-b)^{an_W}\} + c_L\{(1-b)^{an_W} - (1-b)^{an_L}\} > c_0$.

更に，関数 $Q(n_W) = k_0 + \omega b\{c_0 - c_W(1-(1-b)^{an_W}) - c_L((1-b)^{an_W} - (1-b)^{an_L})\}$ を定義する．このとき，最適テスト期間（リリース時刻）に関する次の定理を得る．

定理 1: ソフトウェアのフォールト発見時間分布がパラメータ b (>0) の幾何分布に従う場合，仮定 (A-I)〜(A-III) の下で総期待ソフトウェア費用を最小にする最適ソフトウェアテスト期間（リリース時刻）は以下のように与えられる．

1. $Q(n_W) < 0$ のとき，総期待ソフトウェア費用 $C(n_0, n_W)$ を最小にする n_0^* $(0 < n_0^* < \infty)$ が少なくとも一つ，せいぜい二つ存在する．
2. $Q(n_W) \geq 0$ のとき，$n_0^* = 0$ となる．すなわち，運用段階においてユーザによる受理テストのみによってソフトウェアのデバッグを実施することが最適となる．

次に，最適計画保守限界に関する定理を示す．

定理 2: ソフトウェアのフォールト発見時間分布がパラメータ b (>0) の幾何分布に従う場合，仮定 (A-I) の下で総期待ソフトウェア費用を最小にする最適計画保守限界は以下のように与えられる．

表 1: 環境係数の変化に対する最適テスト期間及び最適計画保守限界．

a	n_0^{**}	n_W^{**}	$C(n_0^{**}, n_W^{**})$
0.50	58	0	108.57
0.75	47	16	108.53
1.00	44	17	108.48
1.25	43	16	108.44
1.50	43	14	108.41
2.00	42	12	108.38
3.00	41	9	108.33

1. $k_W \geq (c_L - c_W)\omega\{1-(1-b)^a\}(1-b)^{n_0}$ のとき，$n_W^* = 0$ となる．
2. $k_W < (c_L - c_W)\omega\{1-(1-b)^a\}(1-b)^{n_0}$ かつ $k_W > (c_L - c_W)\omega\{1-(1-b)^a\}(1-b)^{n_0+a(n_L-1)}$ のとき，総期待ソフトウェア費用を最小にする n_W^* $(0 < n_W^* < n_L)$ が少なくとも一つ，せいぜい二つ存在する．
3. $k_W \leq (c_L - c_W)\omega\{1-(1-b)^a\}(1-b)^{n_0+a(n_L-1)}$ のとき，$n_W^* = n_L$ となる．すなわち，ソフトウェアのライフサイクルが終了するまで保守を継続することが最適となる．

5. 数値例

ここでは実際の開発工程で観測されたソフトウェアフォールト発見時刻データ（データ数 86）を用い，単一のフォールト発見時刻がパラメータ b の幾何分布に従うとした場合の数値例を示す．モデルパラメータの推定値は，$(\hat{\omega}, \hat{b}) = (107.3, 0.1557)$ となった．なお，その他のパラメータを $k_0 = 0.02$，$k_W = 0.01$，$c_0 = 1$，$c_W = 2$，$c_L = 20$，$n_L = 100$ とした．表 1 で，環境係数 a がテスト期間及び計画保守限界の同時最適解に与える影響を示す．a が大きくなる，すなわち運用環境がテスト環境に比べてより厳しくなるにつれて，最適テスト期間は単調に減少することが読みとれる．しかし，最適計画保守限界の単調性は認められない．また，$C(n_0^{**}, n_W^{**})$ は単調に減少していることも確認できる．

参考文献

[1] M. Kimura, T. Toyota, and S. Yamada, "Economic analysis of software release problems with warranty cost and reliability requirement," *Reliab. Eng. & Sys. Safe.*, **66** 49–55 (1999).

[2] T. Dohi, H. Okamura, N. Kaio and S. Osaki, "The age-dependent optimal warranty policy and its application to software maintenance contract," *Proc. 5th Int'l Conf. on Probab. Safe. Assess. and Mgmt.* (S. Kondo and K. Furuta, eds.), **4** 2547–2552, University Academy Press Inc. (2000).

[3] 岡村 寛之, 土肥 正, 尾崎 俊治, "運用段階におけるソフトウェア製品の信頼性評価法 – 加速寿命試験モデルの提案 –," 信学論 (A), **J83–A** 294–301 (2000).