比例ハザードモデルに基づいたソフトウェア最適リリース問題

西尾泰彦*（02401815），土肥正（01307065）
広島大学大学院工学研究科

1. まえがき

ソフトウェア開発工程の最終段階であるテスト工程において，ソフトウェア製品の信頼性を評価することは，安定した品質の製品を市場もしくはユーザに提供するために必要不可欠である．従来までに，テスト工程で得られたソフトウェアフォールトデータ（時間データ，個数データ）に基づいて，ソフトウェア信頼度成長モデル (Software Reliability Growth Model, 以下では SRGM) と呼ばれる確率モデルを用いて当該製品の信頼性を定量的に評価する試みがなされてきた [1]．しかしながら，これまでに数多く提案されてきた SRGM はソフトウェアフォールトデータのみから推定される Black Box モデルであり，ソフトウェア製品やテスト環境を特徴づけるソフトウェアメトリクスの影響が考慮されることはあまりなされていなかった．

最近筆者ら [2] は，ソフトウェアメトリクスとしてフォールト発見労力と計算機処理時間が観測されている場合に，Cox 比例ハザードモデル [3] をソフトウェアの信頼性評価に適用し，メトリクスとしての環境データがソフトウェア故障発生メカニズムに影響を与えることを示している．実際，ソフトウェアフォールトが発見される時間間隔（ソフトウェア故障発生時間間隔）は独立であるが必ずしも同一ではなく，テスト期間が経過するにつれて，信頼度成長現象により（確率順序の意味で）大きくなることが知られている．よって，このような性質をもつデータに比例ハザード性を仮定して Cox モデル解析法を適用することは有意義であり，従来の SRGM の欠点を補う方法として注目を集めている．例えば Pham [4] は，古典的な Jelinski and Moranda モデル [5] を比例ハザード性の観点から拡張したモデルである EPJM モデル（enhanced proportional Jelinski and Moranda model）を提案している．

本稿では，上述の比例ハザードモデルをソフトウェア最適リリース問題に適用し，環境データの影響を考慮した場合の最適リリース方策について考察する．ソフトウェア最適リリース問題 (例えば [6]) は，ソフトウェア製品をテスト工程から運用段階に移行する時期を決定する問題であり，ソフトウェアの開発工程を管理する上で極めて重要な問題である．ここでは特に，比例ハザードモデルにおける基本ハザード関数 (baseline hazard) に Jelinski and Moranda モデル [5]，Moranda モデル [7], Xie モデル [8] を仮定した場合において，総期待ソフトウェア費用を最小にする最適リリース方策を導出し，環境データがリリース時期と費用関数に与える影響を定量的に調査する．

2. 比例ハザードモデル

2.1 記号と仮定

ここでは Cox 比例ハザードモデルについて述べる．いま，$n\ (\geq 1)$ 個のソフトウェアフォールト発見時間間隔データ $x_i\ (i=1,\cdots,n)$ が観測されており，それを小さい順に並べかえた順序統計量を $0=t_0<t_1<\cdots<t_i<\cdots<t_n$ とする．以下のような記号を定義する．

$$\boldsymbol{Z_i} = (z_{i1}, z_{i2}, \cdots, z_{ij}, \cdots, z_{iq})^{\mathbf{T}}, \tag{1}$$
$$\boldsymbol{\beta} = (\beta_1, \beta_2, \cdots, \beta_j, \cdots, \beta_q)^{\mathbf{T}}. \tag{2}$$

ここで，$\boldsymbol{Z_i}$ は時刻 $t_i\ \ (i=1,\cdots,n)$ における $q\ (\geq 1)$ 種類の環境データ $z_{ij}\ (j=1,\cdots,q)$ を要素にもつ共変量ベクトル，$\boldsymbol{\beta}$ は環境データ $\boldsymbol{Z_i}$ の係数ベクトルであり未知パラメータである．β_j $(j=1,\cdots,q)$ の絶対値が大きくなるにつれ，環境データ z_{ij} がハザード関数に大きな影響を及ぼすようなモデル化を実現するために，Cox [3] は時刻 t_i におけるハザード関数を以下のように記述することを提案している．

$$\begin{aligned} h(t_i, \boldsymbol{Z_i}) &= h_0(t_i)\exp(\boldsymbol{Z_i^{\mathbf{T}}}\boldsymbol{\beta}) \\ &= h_0(t_i)\exp(z_{i1}\beta_1+\cdots+z_{ip}\beta_q). \end{aligned} \tag{3}$$

ただし，式 (3) の $h_0(t_i)$ は基本ハザード関数と呼ばれ，時間変数のみに依存する関数である．

2.2 比例ハザード SRGM

Pham [4] は式 (3) の基本ハザード関数に　Jelinski and Moranda モデル [5]:

$$h_0(t_i) = h_{\mathrm{JM}}(i) = \phi[N-(i-1)], \quad (i=1,\cdots,N) \tag{4}$$

を仮定し，環境データを考慮した比例ハザード SRGM を提案している．ここで，$\phi\ (>0)$ は単位テスト時間当たりのフォールトの発見率であり，$N\ (>0)$ はテスト開始前にソフトウェア内に潜在するフォールト数を表す．本稿ではさらに，以下に示す 2 種類のモデルについても考察する．

比例ハザード Moranda モデル:

$$h_0(t_i) = h_{\mathrm{M}}(i) = D\cdot c^{i-1}, \quad (i=1,2,\cdots). \tag{5}$$

ここで，

$D(>0)$ ：最初に発見されるソフトウェアフォールトに対する初期ハザード率

$c\ (0<c<1)$ ：ハザード率の減少係数

である．

比例ハザード Xie モデル:

$$h_0(t_i) = h_{\mathrm{X}}(i) = \phi[N-(i-1)]^{\alpha}, \quad (i=1,\cdots,N). \tag{6}$$

ここで，

$\alpha\ (\leq 1)$ ：残存フォールト数の減少係数

である．

時刻 t_i における信頼度関数を

$$S(t_i, \boldsymbol{Z_i}) = \exp(-h_j(i)t_i)^{\exp(\boldsymbol{Z_i^{\mathbf{T}}}\boldsymbol{\beta})} \tag{7}$$

とおくと $(j=\mathrm{JM},\mathrm{M},\mathrm{X})$，ソフトウェアフォールト発見時間間隔 X_i の確率密度関数は

$$f(t_i, \boldsymbol{Z_i}) = h(t_i, \boldsymbol{Z_i})S(t_i, \boldsymbol{Z_i}) \tag{8}$$

となる．これより，EPJM モデルにおける尤度関数は

$$L(N, \phi, \boldsymbol{\beta}) = \prod_{i=1}^{n} f(t_i, \boldsymbol{Z_i}) \tag{9}$$

によって与えられる．通常，比例ハザードモデルにおける回帰係数 $\boldsymbol{\beta}$ は部分尤度 (partial likelihood) に基づいて推定される．しかしながら，SRGM では基本ハザード関数にも未知パラメータが含まれるため，全尤度に基づいた未知パラメータの推定が必要となる．よってここでは，式 (9) における対数尤度関数 $\log L(N,\phi,\boldsymbol{\beta})$ を最大にするパラメータ $\hat{\boldsymbol{\beta}}, \hat{N}, \hat{\phi}$ を求めることを考える．例えば EPJM モデルでは，

同時尤度方程式 $\partial \log L/\partial\phi = 0,\ \partial \log L/\partial N = 0,\ \partial \log L/\partial \boldsymbol{\beta} = 0$ を数値的に解くことにより，各パラメータの最尤推定値を求めることが出来る．

3 ソフトウェア最適リリース問題への応用

次に，比例ハザードモデルに基づいたソフトウェア最適リリース問題について議論する．文献 [9] に従って，以下のような費用パラメータを定義する．

c_1 (>0) ：テスト段階において発見されたフォールト 1 個を除去するのにかかる費用

c_2 $(>c_1)$ ：運用段階において発見されたフォールト 1 個を除去するのにかかる費用

c_3 (>0) ：単位時間あたりのテスト費用

この時，総期待ソフトウェア費用 $C(n)$ は，発見フォールト数 n $(=0,1,2,\cdots,N)$ の関数として以下のように定義される．

$$C(n) = c_1 n + c_2(N-n) + \sum_{i=1}^{n} c_3 \mathrm{E}[X_i]. \tag{10}$$

但し，$\mathrm{E}[X_i]$ $(i=1,\cdots,N)$ は i 番目のフォールト発見時間間隔 X_i の期待値であり，ソフトウェア製品は n 個のフォールトを発見・除去した時点 $\tau = \sum_{i=1}^{n} X_i$ で運用段階に移行するものと仮定している．

定理: 総期待ソフトウェア費用 $C(n)$ が n $(=0,1,\cdots,N)$ の狭義凸関数となるための必要条件は，$h_j(n)\exp(\boldsymbol{Z}_{\boldsymbol{n}}^{\mathbf{T}}\boldsymbol{\beta}) > h_j(n+1)\exp(\boldsymbol{Z}_{\boldsymbol{n+1}}^{\mathbf{T}}\boldsymbol{\beta})$ $(j=\mathrm{JM,M,X})$　が全ての n $(=0,1,\cdots,N)$ について成立することである．

これより，例えば EPJM モデルでは　$h_{\mathrm{JM}}(n) = \phi[N-n+1]$ であるので，$\boldsymbol{Z}_{\boldsymbol{n}}^{\mathbf{T}}\boldsymbol{\beta} > \boldsymbol{Z}_{\boldsymbol{n+1}}^{\mathbf{T}}\boldsymbol{\beta}$ であれば $C(n)$ は n $(=0,1,\cdots,N)$ の狭義凸関数となり，総期待ソフトウェア費用を最小にする有限で非零な最適解 n^* が存在する．この時の総期待テスト期間は $\mathrm{E}[\tau^*] = \sum_{i=1}^{n^*} \mathrm{E}[X_i]$ となる．ここで，

$$\mathrm{E}[X_i] = \frac{1}{\phi[N-i+1]\exp(\boldsymbol{Z}_{\boldsymbol{i}}^{\mathbf{T}}\boldsymbol{\beta})} \tag{11}$$

である．もちろん，上述の定理で与えられた条件が満たされているかどうかをチェックするためには，未知パラメータの最尤推定値を求めることが必要となる．特に，$\beta_j = 0$ $(j=1,\cdots,q)$ の時 EPJM モデルは Jelinski and Moranda モデル [5] に帰着されるため，上述の結果は文献 [9] の結果に一致する．

4 数値例

ここでは，環境データと回帰係数が最適リリース時期に及ぼす影響について考察する．表 1～表 3 は，Jelinski and Moranda モデル，Moranda モデル，Xie モデルを仮定した比例ハザードモデルにおいて，$\exp(\boldsymbol{Z}_{\boldsymbol{i}}^{\mathbf{T}}\boldsymbol{\beta}) = \exp(\sum_{j=1}^{q} z_{ij}\beta_j)$ における $\sum_{j=1}^{q} z_{ij}\beta_j$ $(i=1,2,\cdots,n)$ の値を変化させた場合の最適リリース方策 n^*，期待テスト期間 $\mathrm{E}[\tau^*]$，総期待ソフトウェア費用 $C(n^*)$ の値を示したものである．使用された費用パラメータは，それぞれ $c_1 = 1$ [\$/fault]，$c_2 = 2$ [\$/fault]，$c_3 = 1\times10^{-5}$ [\$/time] である．表中の $\sum_{j=1}^{q} z_{ij}\beta_j = 0$ $(i=1,2,\cdots,n)$ は環境データを考慮しない従来の SRGM の場合に対応している．

表 1～表 3 の結果より，$\sum_{j=1}^{q} z_{ij}\beta_j$ の値が大きくなるにつれて，最適リリース方策 n^* の値が大きくなり，逆に総期待ソフトウェア費用 $C(n^*)$ の値が小さくなる傾向にあることがわかる．しかしながら，$\sum_{j=1}^{q} z_{ij}\beta_j$ が単調に増加したとしても，期待ソフトウェアテスト期間 $\mathrm{E}[\tau^*]$ は最初のうちは増加するが，いずれ減少傾向に転じることが確認された．すなわち，関数 $\mathrm{E}[\tau^*]$ は $\sum_{j=1}^{q} z_{ij}\beta_j$ の単峰関数となり，環境データの値によっては期待ソフトウェアテスト期間が最大になる場合が存在する．以上のことから，$\sum_{j=1}^{q} z_{ij}\beta_j$ の値の大きさ，すなわちソフトウェアテスト環境を特徴づける外部環境の変化によって，費用の意味で最適なソフトウェアリリース計画は影響を及ぼされることが示される．

表 1　比例ハザード Jelinski and Moranda モデルにおける最適リリース方策 ($\hat{\phi} = 0.00001$, $\hat{N} = 51.5$).

$\sum_{j=1}^{q} z_{ij}\beta_j$	n^*	$\mathrm{E}[\tau^*]$	$C(n^*)$
-2	25	13154.0	79.3829
-1	26	13829.0	79.3714
0	43	107231.0	71.6755
1	49	77147.1	62.5322
2	50	40129.9	57.3997

表 2　比例ハザード Moranda モデルにおける最適リリース方策 ($\hat{D} = 2$, $\hat{c} = 0.9$).

$\sum_{j=1}^{q} z_{ij}\beta_j$	n^*	$\mathrm{E}[\tau^*]$	$C(n^*)$
-2	50	19172.4	55.9818
-1	50	15794.7	55.6032
0	50	14552.1	55.4639
1	50	14095.0	55.4127
2	50	13926.9	55.3929

表 3 比例ハザード Xie モデルにおける最適リリース方策 ($\hat{\lambda} = 0.00001$, $\hat{N} = 51.5$, $\hat{\alpha} = 1.5$).

$\sum_{j=1}^{q} z_{ij}\beta_j$	n^*	T^*	$C(n^*)$
-2	35	73563.9	76.2850
-1	44	82957.3	68.2241
0	48	57613.3	61.5366
1	50	36642.2	57.2260
2	50	22221.5	55.4288

5 むすび

本稿では，比例ハザードモデルを適用したソフトウェア最適リリース問題について議論し，その理論的な枠組みについて言及した．これにより，ソフトウェアメトリクスとソフトウェア製品のテスト計画の立案を関連づけることが可能になった．ここでは発見フォールト数 n に着目した最適リリース問題について考察を行ったが，連続変数であるテスト実行時間を決定変数とする最適リリース問題 [10] も同様に定式化することが可能である．

参考文献

[1] 山田 茂，ソフトウェア信頼性モデル–基礎と応用，日科技連 (1994).

[2] 西尾泰彦，土肥 正，尾崎俊治，"比例ハザードモデルに基づいたソフトウェア製品の信頼性評価"，(投稿中).

[3] D. R. Cox, "Regression models and life tables", *J. Roy. Statist. Soc.*, **B-34**, pp. 187–220 (1972).

[4] H. Pham, *Software Reliability*, Springer-Verlag, Singapore (1999).

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

[6] T. Dohi, Y. Nishio and S. Osaki, "Optimal software release scheduling based on artificial neural networks," *Annals of Software Engineering*, **8**, pp. 167–185 (1999).

[7] P. B. Moranda, "Event-altered rate models for general reliability analysis", *IEEE Trans. Reliab.*, **R-28**, pp. 376–381 (1979).

[8] M. Xie, "On a generalization of J-M model", *Proc. Reliability '89*, pp. 5 Ba/3/1–5 Ba/3/7 (1989).

[9] 山田 茂，一森哲男，増山博，"時間計測信頼性モデルに基づくソフトウェアの最適リリース問題"，電子情報通信学会論文誌 (A), **J73-A**, pp. 1117–1122 (1990).

[10] H. S. Koch and P. Kubat, "Optimal release time of computer software", *IEEE Trans. Software Eng.*, **SE-9**, pp. 323–327 (1983).