数理解析研究所講究録
第1559巻 2007年 1-14

適応的核密度推定による最適ソフトウェア若化スケジューリング

林坂 弘一郎[†], 石丸 雅章[††], 土肥 正[‡]
Koichiro Rinsaka[†], Masaaki Ishimaru[††] and Tadashi Dohi[‡]

[†] 神戸学院大学経営学部
[††] 広島大学工学部第2類
[‡] 広島大学大学院工学研究科情報工学専攻

要旨— 本稿では，定常状態におけるアベイラビリティを最大にするソフトウェアシステムの若化スケジュールを決定するためのモデルに対して，少数の障害発生時間データしか得ることができない状況下や障害発生時間データの分散が大きいような状況下において推定精度を向上させるための統計アルゴリズムを提案する．具体的には，得られた障害発生時間データから適応的核密度推定により障害発生時間の密度関数及び総試験時間変換の推定量をノンパラメトリックに推定する．これにより障害発生時間データから直接最適ソフトウェア若化スケジュールを推定する枠組みを提案する．シミュレーション実験により，提案アルゴリズムは障害発生時間データの分散が大きいときに，従来のアルゴリズムと比較して最適若化スケジュールの推定生後が向上することを示す．

キーワード— ソフトウェア若化，セミマルコフ過程，総試験時間変換，適応的核密度推定，定常アベイラビリティ

1. はじめに

近年のコンピュータシステムの爆発的な普及により，一旦システムに障害が発生すると社会的に大きな影響を及ぼすことは周知の通りである．特に，コンピュータのシステムダウンによる障害はハードウェア障害よりもむしろソフトウェア障害に起因することが多く，ソフトウェアシステムの信頼性を向上させることは最重要課題となっている．ソフトウェア障害は，(i) ソフトウェアプログラムに含まれる固有フォールトによる障害と (ii) ソフトウェアシステムの経年劣化による障害とに大別される．特に後者は，ソフトウェアの運用期間が経過するにつれてソフトウェアシステムの内部構造が変化することにより生じる障害を意味する．このような現象はソフトウェアエージング (software aging) と呼ばれ，オペレーティングシステム，ミッドウェアシステム，通信アプリケーションの動作時において頻繁に観測されている [1–3].

ソフトウェアエージングによる障害は一過性の障害であることが多い [3]．すなわち，障害が発生した後，若干異なる内容（データ，環境）でシステムをリトライすることにより，あたかも障害が発生していなかったかのような操作可能状況に復帰する可能性がある．反面，このような一過性の障害は，ソフトウェアシステムのソースコード上で原因を特定することが極めて困難であることから，その対処法について数多くの研究がなされてきた．特に，ソフトウェア若化 (software rejuvenation) と呼ばれる方策は，ソフトウェアエージングによる一過性の障害を予防するための有効な方法として認識されており，ソフトウェアシステムの稼働を一時的に停止し，その内部構造を浄化した後にシステムを再稼働する一連の予防保全手続きを意味する [4–11]．ここで，ソフトウェアシステムの内部構造の浄化とは，ガーベジコレクションやオペレーティングシステムにおけるカーネルテーブルの洗浄，データ構造の初期化などを示す．アプリケーションシステムの運用上，極端ではあるが最も頻繁に行われているソフトウェア若化の一例として，ハードウェアリブートが挙げられる．

ここで問題となるのは，どのようなタイミングでシステムを予防的に若化を実施するかにある．Huang ら [4] はソフトウェアシステムの時間的挙動を，正常稼働状態，障害発生可能な状態，障害の発生状態，ソフトウェア若化状態の 4 状態をもつ連続時間マルコフ連鎖で記述し，ランダムな若化スケジュールのもとでシステムのアンアベイラビリティや定常状態における運用費用を評価している．Dohi ら [6,7] は Huang ら [4] のモデルをセミマルコフモデルに拡張し，更に障害発生時間データから直接最適な若化スケジュールを推定する統計的手法を開発している．また，Dohi ら [8] や土肥ら [9] では，総期待割引費用とコスト有効性と呼ばれる評価規範を導入し，文献 [6,7] とは異なる視点からソフトウェア若化スケジュールを決定している．ここで，文献 [6–9] において，具体的には，観測された障害発生時間データから，経験分布関数に基づいた標準総試験時間統計量を定義することによりソフトウェア若化スケジュールをノンパラメトリックに推定するアルゴリズムを提案している．この推定アルゴリズムにより，20 個程度のデータが得られれば，推定値は真の最適解に収束することが示された．

更に，文献 [11] では文献 [6,7] と同様の問題について核密度推定 [14–17] に基づいた推定アルゴリズムを提案し，少数の障害発生時間データしか得ることができない状況で，従来のアルゴリズム [8] よりも推定精度を向上させた．文献 [11] では核密度推定に基づいた推定アルゴリズムを提案し，その重要な設計パラメータであるウィンドウ幅の決定基準に対して尤度交差基準 [18,19] を採用した．しかしながら文献 [11] で採用された方法では，推定において重要な設計パラメータであるウィンドウ幅は全ての観測データにおいて一定であるため，裾の長い密度関数から発生したデータに対しては必ずしも正確に密度関数を推定できるとは限らないという問題がある．データが密に観測されている周辺で良好に機能するようなウィンドウ幅が選択された場合，データが疎な裾付近では滑らかな密度関数を描くことができない．一方で裾付近で良好に機能するウインドウ幅を選択した場合には，データ密に観測される周辺では過度に密度関数が平滑化されてしまい分布の特性が消えてしまうこととなる．このように裾の長いデータに対しては可変ウィンドウ幅を考えることが有効であると考えられる．

本稿では多様な障害特性を持つデータに対して最適ソフトウェア若化スケジュールの推定精度を向上させることを目的として，固定ウィンドウ幅と可変ウィンドウ幅両者の特徴を持った適応的核密度推定 (adaptive kernel estimation) [17] に基づいたノンパラメトリック推定アルゴリズムを提案する．また，シミュレーション実験において提案アルゴリズムの漸近収束性を調べ，従来のアルゴリズム [6,7,11] と比較することにより，障害発生時間データの分散が大きいときに最適若化スケジュールの推定精度が向上することを示す．

2. セミマルコフモデル

ここでは，以下の四つの状態をもつセミマルコフモデルについて考える．

状態 0： 正常稼働状態

状態 1： 劣化状態

状態 2： 障害発生状態

状態 3： 予防保全状態

ここで，状態 1 はメモリ漏れ量があるしきい値を超えたり，システムが正常稼働状態と比べて不安定な状態に陥ることを意味している．本モデルにおいて，ソフトウェアシステムは確率的に正常稼働状態から劣化状態へと移行し，状態 0 から状態 1 へと推移する時間 Z の確率分布関数を $\Pr\{Z \leq t\} = F_0(t)$，平均を μ_0 (> 0) とする．システムが障害発生可能な状態に推移すると，ソフトウェア若化を行うか否かの決定をするものとす

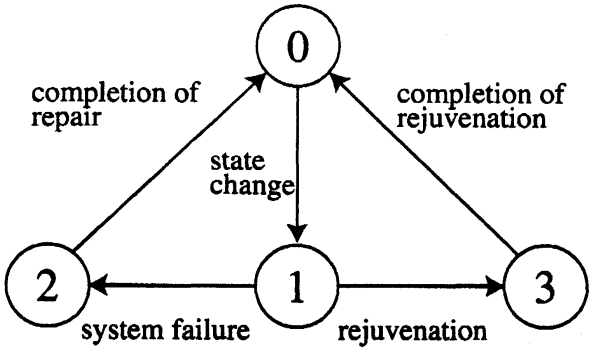

図 1: モデル 1 の状態推移図

る．システムが障害発生可能な状態から具体的に障害に至るまでの時間は，非負で連続な確率変数 X によって記述され，その確率分布関数を $\Pr\{X \leq t\} = F_f(t)$，平均を λ_f (> 0) とする．一旦障害が発生すると，回復動作が直ちに開始される．ここで回復動作とは，障害が発生した後に事後的にシステムを若化することを意味する．回復動作に要する時間 Y もまた非負の連続形確率変数であり，その分布関数を $\Pr\{Y \leq t\} = F_a(t)$，平均を μ_a (> 0) とする．回復動作が完了すると，システムの障害発生率は正常稼働状態における初期状態まで復旧される．

一方，ソフトウェアの予防的な若化は，システムが障害発生可能な状態に推移した後の T 時間経過後になされるものとする．ここで，T は非負で連続な確率変数であり，その確率分布関数を $F_r(t)$，平均を t_0 (> 0) とする．システム障害が発生する前に時刻 T が経過した場合は，直ちに予防的に若化を開始し，そのシステムオーバーヘッド V に対する確率分布関数を $\Pr\{V \leq t\} = F_c(t)$，平均を μ_c (> 0) とする．修理の場合と同様に，予防的若化が完了すると，システムの障害発生率は正常稼働状態における初期状態まで復旧される．このモデルは状態 0 から再度状態 0 に戻るまでの時間間隔を周期にもつセミマルコフ過程であり，すべての状態が再生点 (regeneration points) となることから，推移確率を求めるために通常のマルコフ解析の手法が適用可能である [12]．システムの状態推移図を図 1 に示す．特に障害発生可能な状態から予防的に若化を実施するまでの時間 T が一定である (periodic rejuvenation) とすれば，確率分布関数 $F_r(t)$ を次のようなユニット関数

$$F_r(t) = U(t - t_0) = \begin{cases} 1 & \text{if } t \geq t_0 \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

に置き換えればよい．

Dohi ら [6,7] の結果を直接用いることによって定常アベイラビリティは

$$A(t_0) = \frac{\mu_0 + \int_0^{t_0} \overline{F}_f(t)dt}{\mu_0 + \mu_a F_f(t_0) + \mu_c \overline{F}_f(t_0) + \int_0^{t_0} \overline{F}_f(t)dt} \tag{2}$$

のように求めることができる．ここで，$\overline{F}_f(\cdot) = 1 - F_f(\cdot)$ である．

3. 標準総試験時間変換によるソフトウェア最適若化スケジュールの導出

ここでは，定常アベイラビリティを最大にする最適ソフトウェア若化スケジュールをグラフ上で導出する．今，障害発生時間分布 $F_f(t)$ に対する標準総試験時間変換 (scaled total time on test transform) を次のように定義

する [13].

$$\phi(p) = \frac{1}{\lambda_f} \int_0^{F_f^{-1}(p)} \overline{F}_f(t)dt. \tag{3}$$

ここで，$F_f(t)$ は絶対連続かつ非減少関数であるので，その逆関数

$$F_f^{-1}(p) = \inf\{t_0; F_f(t_0) \geq p\}, \quad 0 \leq p \leq 1 \tag{4}$$

が必ず存在する．よく知られた結果として，確率分布関数 $F_f(t)$ が IFR (DFR) であるための必要かつ十分条件は，関数 $\phi(p)$ が $p \in [0,1]$ に関して凹（凸）関数であることである．式 (2) で与えられる定常アベイラビリティを $F_f(t)$ の標準総試験時間変換を用いて書き換えることにより，以下の結果が得られる [6,7].

定理 1 定常アベイラビリティ $A(t_0)$ を最大にする最適ソフトウェア若化スケジュールを求める問題は，以下の問題の解 p^* $(0 \leq p^* \leq 1)$ を求めることと等価である．

$$\max_{0 \leq p \leq 1} \frac{\phi(p) + \alpha}{p + \eta}. \tag{5}$$

ただし，

$$\alpha = \mu_0/\lambda_f, \tag{6}$$

$$\eta = \mu_c/(\mu_a - \mu_c). \tag{7}$$

定理 1 から，障害発生時間分布 $F_f(t)$ が既知であるならば，最適ソフトウェア若化スケジュールは $t_0^* = F_f^{-1}(p^*)$ によって求めることができる．ここで，$p^*(0 \leq p^* \leq 1)$ は 2 次元平面上で点 $(-\eta, -\alpha) \in (-\infty, 0) \times (-\infty, 0)$ から曲線 $(p, \phi(p)) \in [0,1] \times [0,1]$ に引いた線分のうち，最も大きい傾斜を示す曲線上の点に対する x 座標値 p^* によって与えられる．

4. 適応的核密度推定アルゴリズム

前章で示した経験分布に基づいた推定アルゴリズムによると，20 個程度の障害発生時間データが得られれば推定値は真の最適解に収束することが示されている．しかし，実用上の問題点として，ソフトウェアシステムの運用期間中に多くの障害発生時間データを取得するのはきわめて困難である．ここでは，少数の障害発生時間データしか得ることができないような状況を想定し，データの分散が大きく，分布の裾が長い場合においても最適なソフトウェア若化スケジュールを高精度に推定することを考え，適応的核密度推定に基づいた推定アルゴリズムを提案する．

4.1. 核密度推定アルゴリズム

観測された障害発生時間データ $x_1, x_2, \cdots, x_n$ が確率密度関数 f_f からの標本であると仮定する．今，核密度推定量 (kernel density estimator) を次式によって定義する [14–17].

$$\hat{f}_{f_{n,k}}(y) = \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{y - x_i}{h}\right). \tag{8}$$

ここで，h (> 0) はウィンドウ幅と呼ばれる．関数 $K(\cdot)$ は核関数 (kernel function) と呼ばれ，次式を満足するように決定される．

$$\int_{-\infty}^{\infty} K(t)dt = 1, \quad \int_{-\infty}^{\infty} tK(t)dt = 0, \quad \int_{-\infty}^{\infty} t^2 K(t)dt = r^2 \neq 0. \tag{9}$$

表 1: 核関数の例 [17]

Kernel	$K(t)$
Rectangular	$\frac{1}{2}$ for $\|t\| < 1$, 0 otherwise
Gaussian	$\frac{1}{\sqrt{2\pi}} e^{-(1/2)t^2}$
Triangular	$1 - \|t\|$ for $\|t\| < 1$, 0 otherwise
Biweight	$\frac{15}{16}\left(1-t^2\right)^2$ for $\|t\| < 1$, 0 otherwise
Epanechnikov	$\frac{3}{4}\left(1-\frac{1}{5}t^2\right)/\sqrt{5}$ for $\|t\| < \sqrt{5}$, 0 otherwise

表 1 には代表的な核関数を示す．通常，$K(\cdot)$ には対称な密度関数が選択される．核密度推定においては，得られた障害発生時間データの各観測点について核関数によって核を作成し，それらの核をの和を取ることによって密度関数を推定する．ウインドウ幅 h は核の幅を，核関数は核の形状を決定する．

ウィンドウ幅 h は核密度推定において最も重要な設計パラメータのひとつである．観測された各障害発生時間データ上に置かれる核の幅を決定するためのパラメータであり，スムージングパラメータや帯域幅などとも呼ばれることがある．ウィンドウ幅が大きすぎる場合には推定すべき密度関数の特性がぼやけてしまい，小さすぎる場合には特に分布の裾付近においてノイズの影響を強く受けるため適切な値を設定する必要がある．

ウィンドウ幅を決定するためのひとつの方法は平均積分二乗誤差 (mean integrated squares error), MISE を最小にするような h を選択する方法である．MISE は次式により定義される．

$$\mathrm{MISE}(\tilde{f}_f) = E\int_{-\infty}^{\infty}\left\{\tilde{f}_f(x) - f_f(x)\right\}^2 dx. \tag{10}$$

ここで $\tilde{f}_f$ は確率密度関数 f_f に基づく核密度推定量を意味する．式 (10) は以下のように近似可能である．

$$\begin{aligned}
\mathrm{MISE}(\tilde{f}_f) &= \int_{-\infty}^{\infty} E\left\{\tilde{f}_f(x) - f_f(x)\right\}^2 dx \\
&= \int_{-\infty}^{\infty}\left\{E\tilde{f}_f(x) - f_f(x)\right\}^2 dx + \int_{-\infty}^{\infty} \mathrm{Var}\tilde{f}_f(x)dx \\
&\approx \frac{1}{4}h^4 r^2 \int_{-\infty}^{\infty} f_f''(x)^2 dx + \frac{1}{nh}\int_{-\infty}^{\infty} K(t)^2 dt.
\end{aligned} \tag{11}$$

式 (11) を最小化するような h_{ideal} は，次式より求めることができる．

$$h_{ideal} = r^{-2/5}\left\{\int_{-\infty}^{\infty} K(t)^2 dt\right\}^{1/5}\left\{\int_{-\infty}^{\infty} f_f''(x)^2 dx\right\}^{-1/5} n^{-1/5}. \tag{12}$$

上式を解析的に求めるための簡便なアプローチは，式 (12) の項 $\int f_f''(x)^2 dx$ に対して密度関数 ϕ，分散 σ^2 の正規分布を仮定することである．これにより式 (12) の項 $\int f_f''(x)^2 dx$ は

$$\int_{-\infty}^{\infty} f_f''(x)^2 dx = \sigma^{-5}\int_{-\infty}^{\infty}\phi''(x)^2 dx = \frac{3}{8}\phi^{-1/2}\sigma^{-5} \tag{13}$$

と書き換えることができる．いま，核関数 $K(\cdot)$ に Gaussian 核関数

$$K(t) = \frac{1}{\sqrt{2\pi}} e^{-(1/2)t^2} \tag{14}$$

を仮定するなら，最適なウィンドウ幅 h_{ideal} は次式によって求めることが可能となる．

$$\begin{aligned} h_{ideal} &= (4\pi)^{-1/10}\frac{3}{8}\pi^{-1/2}\sigma n^{-1/5} \\ &= \left(\frac{4}{3}\right)^{1/5}\sigma n^{-1/5} \approx 1.06\sigma n^{-1/5}. \end{aligned} \tag{15}$$

式 (15) より明らかに観測データ数が増加するに従って最適なウィンドウ幅 h_{ideal} は小さくなる．また上記のアプローチを採用した場合には尤度交差基準 [11] を採用した場合と比較して最適なウィンドウ幅を求めるための計算コストを大幅に削減することが可能となる．

4.2. 適応的核密度推定アルゴリズム

核密度推定アルゴリズムでは全ての観測データに対して同じウィンドウ幅を用いた．しかしながら，データの分散が大きく，裾の長い密度関数から発生したデータに対する推定精度に問題があることは前述したとおりである．これに対し本稿では次の手順により密度関数を推定する適応的核密度推定アルゴリズムを提案する．

まず，全ての $i\ (=1,\cdots,n)$ について $f_{f_{n,p}}(x_i) > 0$ を満たすように，核密度推定によってパイロット推定値 $f_{f_{n,p}}(t)$ を求める．次に，$f_{f_{n,p}}(x_i)$ の幾何平均 g を

$$\log g = n^{-1}\sum_{i=1}^{n}\log f_{f_{n,k}}(x_i) \tag{16}$$

として求め，i 番目の観測データ x_i に対応するウィンドウ幅を調整するためのパラメータ δ_i を

$$\delta_i = \{f_{f_{n,p}}(x_i)/g\}^{-\beta} \tag{17}$$

によって定義する．この δ_i を用いて次式によって適応的核密度推定量を定義する．

$$f_{f_{n,a}}(y) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{h\delta_i}K\left(\frac{y-x_i}{h\delta_i}\right). \tag{18}$$

ここで，$\beta\ (0 \le \beta \le 1)$ は感度パラメータと呼ばれ，0.5 と設定するのが望ましいとされている [17]．式 (18) で与えられる適応的核密度推定量の下では，観測データ x_i に置かれた核のウィンドウ幅は δ_i に比例する．すなわち，周辺が密な観測データに対してはウィンドウ幅を小さくするように調整することで核を高くし，周辺が疎な観測データに対してはウィンドウ幅を大きく調整することで核を低く平らにする．これにより，分布の裾付近の推定精度を向上させることが可能となる．また $\beta = 0$ としたとき，式 (18) の適応的核密度推定量は式 (8) の核密度推定量と一致する．

4.3. 最適若化スケジューリング

ここでは観測データを用いて推定した密度関数から式 (2) で与えられる定常アベイラビリティを最大にするような最適若化スケジュールを導出することを考える．今，障害発生時間分布の推定量 $\hat{F}_f(t) = \int_0^t \hat{f}_f(s)ds$ の標準総試験時間変換を次式のように定義する．

$$\phi_{\mathrm{AKD}}(p) = \frac{n}{\sum_{j=1}^{n}x_j}\int_0^{\hat{F}_f^{-1}(p)}\overline{\hat{F}}_f(t)dt. \tag{19}$$

また，障害発生時間分布の推定量 $\hat{F}_f(t)$ は絶対連続かつ非減少関数であるので，その逆関数

$$\hat{F}_f^{-1}(p) = \inf\left\{t_0;\ \hat{F}_f(t_0) \ge p\right\},\quad 0 \le p \le 1 \tag{20}$$

が常に存在する．推定量 $(p, \phi_{\mathrm{AKD}}(p))$ は $(p, \phi(p))$ のノンパラメトリック推定量であるので，定理 1 の結果を直接適用することにより，最適ソフトウェア若化スケジュールに関する以下の定理が得られる．

定理 2 障害発生時間に対する n (>0) 個の完全データ $x_1, x_2, \cdots, x_n$ が観測されているものとする．定常アベイラビリティ $A(t_0)$ を最大にする最適ソフトウェア若化スケジュールのノンパラメトリック推定量 $\hat{t}_0^*$ を求める問題は以下の問題の解 $p^*(0 \leq p^* \leq 1)$ を求めることと等価である．

$$\max_{0 \leq p \leq 1} \frac{\phi_{\mathrm{AKD}}(p) + \hat{\alpha}_n}{p + \eta} \tag{21}$$

ただし，

$$\hat{\alpha}_n = \mu_0 / \hat{\lambda}_{fn}, \tag{22}$$

$$\eta = \mu_c / (\mu_a + \mu_c) \tag{23}$$

である．

定理 2 から，障害発生時間分布 $F_f(t)$ が未知の場合においても障害発生時間データが得られるならば，最適ソフトウェア若化スケジュールは $\hat{t}_0^* = \hat{F}_f^{-1}(p^*)$ によって求めることができる．なお，定理 2 の幾何学的解釈は定理 1 と同様である．

5. シミュレーション実験

ここでは本稿で提案した適応的核密度推定に基づいたノンパラメトリック推定アルゴリズムの漸近的性質とその推定精度について考察を行う．比較対象は従来の経験分布 [6,7]，核密度推定 [11] に基づいたアルゴリズムである．文献 [6,7,11]，及び本稿で示した定理 2 の結果は，いずれも統計的に十分な数の障害発生時間データが与えられれば，最適ソフトウェア若化スケジュールの推定量は（未知ではあるが）真の最適解に収束する．しかしながら，実用上の問題点として，ソフトウェアシステムの運用期間中に十分な数の障害発生時間データを取得することは極めて困難である．よって，それぞれのアルゴリズムがどれくらいの障害発生時間データ数で機能しするかを考察する．更に，本稿で提案したアルゴリズムを用いることで障害発生時間データの分散が大きく，分布の裾が長い場合において最適若化スケジュールを高精度に推定可能であることを示す．

以下では障害発生時間がワイブル分布

$$F_f(t) = 1 - e^{-\left(\frac{t}{\theta}\right)^{\gamma}} \tag{24}$$

に従うものと仮定する．ここで，形状パラメータ $\gamma = 2.0$，尺度パラメータ $\theta = 160.0$ とする．また，その他のパラメータとして $\mu_0 = 24.0(h)$, $\mu_a = 1.0(h)$, $\mu_c = 0.25(h)$ と設定した．以上のようなパラメータ設定のもとで $t_0^* = 72.3(h)$，$A(t_0^*) = 0.995626$ が得られた．

図 2 及び 3 には 3 種類の推定アルゴリズムに対して，最適ソフトウェア若化スケジュールの推定値とその定常アベイラビリティ推定値に関する漸近的性質を示す．なお，図中の Adaptive kernek, Kernel density, Empirical distribution はそれぞれ本稿で提案した適応的核密度推定，文献 [11] の核密度推定，文献 [6,7] の経験分布による推定値を意味している．また，Real optimal は真の最適解を示している．これらの図より，3 種類の推定アルゴリズムに対して，障害発生時間データ数が 20 付近で真の最適解に漸近的に収束している様子を読み取ることができる．

次に同様のシミュレーション実験を 500 回ずつ繰り返すことにより得られた推定値の平均値，中央値，標準偏差，及び相対絶対誤差平均 (Relative Absolute Error Average, RAEA_{t_0}, RAEA_A) を表 2 から表 9 に示す．

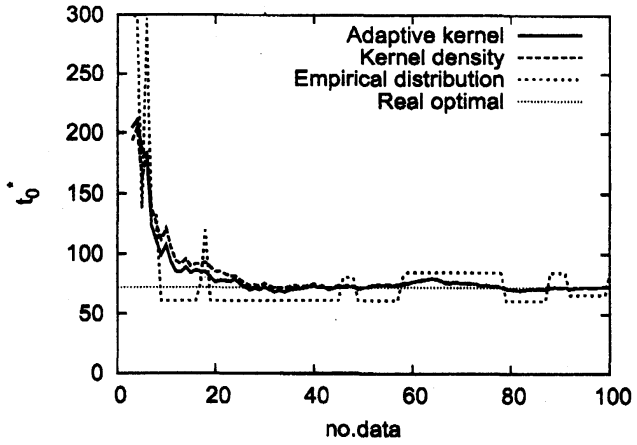

図 2: 最適ソフトウェア若化スケジュールの推定値に関する漸近的性質

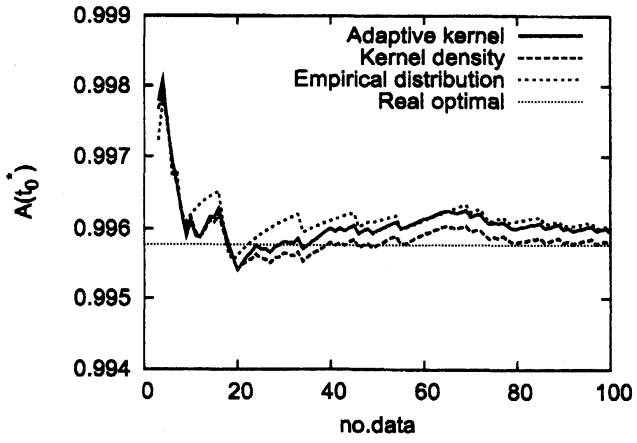

図 3: 定常アベイラビリティの推定値に関する漸近的性質

表 2: $\hat{t}_0^*$ に関する統計量 ($\gamma = 1.5$).

アルゴリズム	n	平均値	中央値	標準偏差	RAEA_{t_0}
適応的	10	**106.6**	**90.6**	78.1	0.478
核密度	30	**88.6**	**83.4**	28.6	**0.216**
	100	**79.5**	76.2	15.3	0.142
核密度	10	109.9	96.3	**60.6**	**0.472**
	30	97.1	92.1	**26.8**	0.249
	100	88.6	85.7	**14.4**	**0.134**
経験分布	10	123.2	104.6	79.9	0.704
	30	103.9	89.4	58.8	0.487
	100	89.3	**83.8**	32.4	0.305
真の最適解		82.3			

表 3: $\hat{t}_0^*$ に関する統計量 ($\gamma = 2.0$).

アルゴリズム	n	平均値	中央値	標準偏差	RAEA_{t_0}
適応的	10	**88.1**	**83.4**	36.5	**0.313**
核密度	30	**80.5**	**79.1**	14.7	**0.171**
	100	74.8	**73.7**	8.3	**0.088**
核密度	10	89.6	84.7	**30.0**	0.315
	30	83.2	82.0	**14.2**	0.186
	100	77.8	76.8	**8.1**	0.102
経験分布	10	100.0	91.7	46.2	0.514
	30	84.7	81.9	27.2	0.306
	100	**73.8**	69.5	18.0	0.194
真の最適解		72.3			

ここで，ワイブル障害発生時間分布の形状パラメータは $\gamma = 1.5, 2.0, 4.0, 6.0$ としている．また，RAEA_{t_0} 及び RAEA_A はそれぞれ

$$\mathrm{RAEA}_{t_0} = \frac{1}{m t_0^*} \sum_{j=1}^{m} \left| \hat{t}_{0_j}^* - t_0^* \right|, \tag{25}$$

$$\mathrm{RAEA}_A = \frac{1}{m A(t_0^*)} \sum_{j=1}^{m} \left| A(\hat{t}_{0_j}^*) - A(t_0^*) \right| \tag{26}$$

によって定義される．ここで，$m = 500$ はシミュレーション実験回数，$\hat{t}_{0_j}^*$ は j 番目のシミュレーションランにおける最適若化スケジュールの推定値である．

表 2 から表 5 において，最適若化スケジュール推定値 $\hat{t}_0^*$ の平均値および中央値について着目すると，$\gamma = 1.5$, 2.0 の場合，すなわち障害発生時間分布の分散が大きく，分布の裾が長い場合に適応的核密度推定の結果がよくなっていることが読み取れる．反対に $\gamma = 4.0, 6.0$ のように分散が小さくなると核密度推定の平均値，中央値の結果がよくなっている．次に最適若化スケジュール推定値 $\hat{t}_0^*$ の標準偏差に着目すると，本稿で提案する適

表 4: $\hat{t}_0^*$ に関する統計量 ($\gamma = 4.0$).

アルゴリズム	n	平均値	中央値	標準偏差	RAEA_{t_0}
適応的	10	90.9	90.6	15.0	0.149
核密度	30	89.2	89.0	9.2	0.092
	100	87.5	87.5	4.9	0.051
核密度	10	**86.6**	**85.9**	**13.8**	**0.128**
	30	**85.2**	**84.6**	**8.7**	**0.081**
	100	**84.3**	**84.3**	**4.9**	**0.048**
経験分布	10	100.3	99.8	21.3	0.244
	30	92.3	91.5	14.9	0.152
	100	88.2	88.2	9.7	0.096
真の最適解		85.4			

表 5: $\hat{t}_0^*$ に関する統計量 ($\gamma = 6.0$).

アルゴリズム	n	平均値	中央値	標準偏差	RAEA_{t_0}
適応的	10	100.1	100.2	**13.0**	0.107
核密度	30	99.7	99.8	**8.0**	**0.067**
	100	99.3	99.5	**4.3**	**0.039**
核密度	10	**96.5**	**96.0**	**13.0**	**0.106**
	30	**95.7**	**95.9**	8.1	0.068
	100	**96.0**	**96.3**	4.6	0.041
経験分布	10	111.3	111.6	16.4	0.179
	30	104.0	104.2	12.3	0.116
	100	100.1	99.9	7.9	0.069
真の最適解		97.6			

表 6: $A(\hat{t}_0^*)$ に関する統計量 ($\gamma = 1.5$).

アルゴリズム	n	平均値	中央値	標準偏差	RAEA_A
適応的	10	0.995602	0.995754	0.001030	0.001011
核密度	30	0.995447	0.995478	0.000608	0.000644
	100	0.995303	0.995302	0.000340	0.000392
核密度	10	0.995584	0.995682	**0.000959**	**0.000950**
	30	0.995432	0.995454	**0.000544**	**0.000592**
	100	0.995280	0.995280	**0.000306**	0.000358
経験分布	10	**0.995337**	**0.995513**	0.001244	0.001061
	30	**0.995308**	**0.995383**	0.000750	0.000672
	100	**0.995139**	**0.995158**	0.000401	**0.000353**
真の最適解		0.994981			

表 7: $A(\hat{t}_0^*)$ に関する統計量 ($\gamma = 2.0$).

アルゴリズム	n	平均値	中央値	標準偏差	RAEA_A
適応的	10	0.995962	0.996057	0.000874	0.000727
核密度	30	0.995903	0.995932	0.000523	0.000432
	100	0.995864	0.995883	0.000282	0.000242
核密度	10	**0.995850**	**0.995895**	**0.000831**	**0.000668**
	30	**0.995758**	**0.995783**	**0.000489**	**0.000382**
	100	**0.995716**	**0.995729**	**0.000267**	**0.000221**
経験分布	10	0.995908	0.996050	0.000983	0.000785
	30	0.995975	0.996046	0.000597	0.000517
	100	0.995898	0.995923	0.000324	0.000284
真の最適解		0.995779			

表 8: $A(\hat{t}_0^*)$ に関する統計量 ($\gamma = 4.0$).

アルゴリズム	n	平均値	中央値	標準偏差	RAEA_A
適応的	10	**0.997070**	**0.997119**	0.000544	0.000423
核密度	30	0.997078	**0.997092**	0.000315	**0.000254**
	100	**0.997115**	**0.997115**	**0.000164**	**0.000134**
核密度	10	0.997004	0.997068	0.000585	0.000470
	30	0.997000	0.997032	0.000336	0.000289
	100	0.997045	0.997045	0.000176	0.000166
経験分布	10	0.997023	0.997097	**0.000497**	**0.000397**
	30	**0.997193**	0.997224	**0.000313**	0.000257
	100	0.997218	0.997220	0.000178	0.000152
真の最適解		0.997153			

表 9: $A(\hat{t}_0^*)$ に関する統計量 ($\gamma = 6.0$).

アルゴリズム	n	平均値	中央値	標準偏差	RAEA_A
適応的	10	**0.997564**	0.997615	0.000378	**0.000299**
核密度	30	0.997557	0.997582	0.000220	0.000181
	100	**0.997586**	0.997586	**0.000118**	**0.000100**
核密度	10	0.997563	**0.997628**	0.000407	0.000324
	30	0.997548	0.997570	0.000233	0.000195
	100	0.997571	0.997566	0.000126	0.000111
経験分布	10	0.997431	0.997476	**0.000342**	**0.000299**
	30	**0.997623**	**0.997643**	**0.000213**	**0.000174**
	100	**0.997666**	**0.997665**	0.000125	0.000105
真の最適解		0.997626			

応的核密度推定アルゴリズムによる推定値は核密度推定と比較して分散が大きくなる傾向にあることがわかる．相対絶対誤差平均については適応的核密度推定と核密度推定がほぼ二分する結果となった．したがって，推定値の分散は若干大きくなるものの障害発生時間データの分散が大きい場合において従来のアルゴリズムと比較して早い段階で真の最適解に収束するため，提案アルゴリズムが有効であることが示された．

更に表6から表9において，定常アベイラビリティ推定値 $A(\hat{t}_0^*)$ の統計量について着目すると，障害発生時間分布の分散が小さいときに核密度推定の推定精度が高く，分散が大きくなると適応的核密度推定の推定精度が高くなることが読み取れる．これは最適若化スケジュール推定値の傾向とは反対の結果である．

6. まとめと今後の課題

本稿では，文献 [6,7,11] において扱われたソフトウェアシステムの若化スケジュールを決定するためのモデルに対して，少数の障害発生時間データしか得ることができない状況下やデータの分散が大きいような状況下において推定精度を向上させるための統計アルゴリズムを提案した．具体的には，得られたデータから適応的核密度推定により障害発生時間の密度関数及び総試験時間変換の推定量をノンパラメトリックに推定した．これにより少数のデータから直接最適ソフトウェア若化スケジュールを推定する枠組みを提案した．シミュレーション実験により検証した結果，障害発生時間データの分散が大きい場合において従来のアルゴリズムと比較して早い段階で真の最適解に収束するため，提案アルゴリズムは有効であることが示された．

本稿では，定常状態におけるアベイラビリティを評価規範とするソフトウェアシステムの若化スケジュールを取り扱った．今後の課題として，文献 [8,9] で考えられた総期待割引費用やコスト有効性を評価規範とするような若化スケジュールに対しても本稿で提案した推定アルゴリズムを適用し，有効性を検証する．また，障害発生時間データをオンラインで取得しながら適応的に若化スケジュールを決定する管理ソフトウェアの開発を行う予定である．

謝辞: 本研究の一部は科学研究費基盤研究 (B)(1) Grant No. 16310116 (2004–2006)，科学研究費若手研究 (B) Grant No. 18710145 (2006–2007) の助成の下で行われたものである．

参考文献

[1] E. Adams, "Optimizing preventive service of the software products," IBM J. Research and Development, vol.28, pp.2–14, 1984.

[2] V. Castelli, R.E. Harper, P. Heidelberger, S.W. Hunter, K.S. Trivedi, V. Vaidyanathan, and W.P. Zeggert, "Proactive management of software aging," IBM J. Research & Development, vol.45, pp.311–332, 2001.

[3] J. Gray and D.P. Siewiorek, "High-availability computer systems," IEEE Comput., vol.9, pp.39–48, 1991.

[4] Y. Huang, C. Kintala, N. Kolettis, and N.D. Funton, "Software rejuvenation: Analysis, module and applications," Proc. 25th IEEE Int'l Symp. Fault Tolerant Computing, pp.381–390, IEEE Computer Society Press, Los Alamitos, CA, 1995.

[5] K.S. Trivedi, K. Vaidyanathan, and K. Goševa-Popstojanova, "Modeling and analysis of software aging and rejuvenation," Proc. 33rd Annual Simulation Symp. pp.270–279, IEEE Computer Society Press, Los Alamitos, CA, 2000.

[6] T. Dohi, K. Goševa-Popstojanova, and K.S. Trivedi, "Analysis of software cost models with rejuvenation," Proc. 5th IEEE Int'l Symp. High Assurance Systems Engineering, pp.25–34, IEEE Computer Society Press, Los Alamitos, CA, 2000.

[7] T. Dohi, K. Goševa-Popstojanova, and K.S. Trivedi, "Statistical non-parametric algorithms t estimate the optimal software rejuvenation schedule," Proc. 2000 Pacific Rim Int'l Symp. on Dependable Computing, pp.77–84, IEEE Computer Society Press, Los Alamitos, CA, 2000.

[8] T. Dohi, T. Danjou, and H. Okamura, "Optimal software rejuvenation policy with discounting," Proc. 2001 Pacific Rim Int'l Sympo. on Dependable Computing, pp.87–94, IEEE Computer Society Press, Los Alamitos, CA, 2001.

[9] 土肥　正，海生直人，尾崎俊治，"コスト有効性に基づいた通信ソフトウェアシステムに対する予防保全スケジュールの決定，" 信学論 (A)，vol.J85-A, no.2, pp.197–206, Feb. 2002.

[10] K. Rinsaka and T. Dohi, "Behavioral analysis of a fault-tolerant software system with rejuvenation," IEICE Transactions on Information and Systems, vol.E88-D, no.12, pp.2681–2690, 2005.

[11] K. Rinsaka and T. Dohi, "Estimating the optimal software rejuvenation schedule with small sample data," Advanced Reliability Modeling II, pp. 443–450, World Scientific, Singapore, 2006.

[12] S. Osaki, Applied Stochastic System Modeling, Springer-Verlag, Berlin, 1992.

[13] R.E. Barlow and R. Campo, "Total time on test processes and applications to failure data," Reliability and Fault Tree Analysis, ed. R.E. Barlow, J. Fussell and N.D. Singpurwalla, pp.451–481, SIAM, Philadelphia, 1975.

[14] E. Parzen, "On the estimation of a probability density function and the mode," Annals of Mathematical Statistics, vol.33, pp.1065–1076, 1962.

[15] M. Rosenblatt, "Remarks on some nonparametric estimates of a density function," Annals of Mathematical Statistics vol.27, pp.832–837, 1956.

[16] T. Cacoullos, "Estimation of a multivariate density," Annals of the Institute of Statistical Mathematics, vol.18, pp.178–189, 1966.

[17] B.W. Silverman, Density Estimation for Statistics and Data Analysis, Chapman and Hall, London, 1986.

[18] R.P.W. Duin, "On the choice of smoothing parameters for Parzen estimators of probability density functions," IEEE Trnas. Comput., vol.C-25, pp.1175–1179, Nov. 1976.

[19] J.D.F. Habbema, J. Hermans, and K. van der Broek, "A stepwise discrimination program using density estimation," Compstat, ed. G. Bruckman, pp.100–110, Physica Verlag, Vienna, 1974.