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

ソフトウェアシステムにおける動的若化スケジュールの決定

衛藤 浩幸, 土肥 正 (01307065)
広島大学大学院工学研究科

1. はじめに

近年，多くのソフトウェアシステムにおいてエージング (aging) と呼ばれる現象が観測されており，劣化状態から安定した動作状態にソフトウェアを若化 (rejuvenate) するタイミングを決定することは耐故障ソフトウェアシステムを運用する上で必要不可欠となっている．Garg *et al.* [3] は，サーバタイプのソフトウェアを解析するための待ち行列モデルについて考察している．Bobbio *et al.* [1] はメモリエージングなどの現象を累積ショック過程として捉えることにより，ソフトウェア若化スケジュールを決定するための静的モデルを提案している．Dohi *et al.* [2] は単純なセミマルコフモデルに基づいて，期待費用及びアベイラビリティの観点から若化スケジュールを推定するための統計アルゴリズムを開発している．

一方，ソフトウェアの動作を観測しながら動的に若化を実施する動的若化スケジュール問題についてはあまり考察されていない．Pfening *et al.* [4] は，システム動作時間と処理待ちトランザクション数を制御変数としたサーバシステムの挙動をマルコフ決定過程によって記述し，期待費用関数を最小にする最適若化スケジュールの動的決定問題について考察をしている．しかしながら，エージングによる劣化度合いを動作時間だけで測ることは常に現実的であるとは言えず，メモリエージングであればメモリ消費量などソフトウェアの状態を考慮した若化スケジューリングを実施すべきである．そこで本稿では，サーバの劣化が最終的にシステム障害に至るという仮定の下で，処理待ちトランザクション数とサーバの劣化状態を制御変数として捉えることで，ソフトウェア若化スケジュールの決定問題をセミマルコフ決定過程として定式化する．ここで考察するモデルと文献 [4] の相違点は，(i) 評価規範として定常期待費用を導入している (ii) システムのサービス率がサーバの状態に依存している，等である．ある物理的に妥当な仮定の下で，制御限界タイプの若化スケジュールの最適性について証明する．

2. モデルの記述

サーバタイプのソフトウェアシステムを考える．トランザクションはパラメータ $\lambda\,(>0)$ の指数分布に従って到着し，各トランザクションの処理時間はサーバの状態 $k\,(=0,1,2,\cdots)$ に依存した独立で同一に分布する非負の連続形確率変数 (処理率: $\mu_k\ (>0)$) であり，その密度関数と分布関数を $g_k(x)$ および $G_k(x)$ とする．サーバの推移可能な状態は $k=0$ (正常) から $k=s+1$ (システムダウン) までの $s+2$ 状態とし，サーバの状態推移は連続時間マルコフ連鎖によって記述される．すなわち，状態 i から状態 j への推移率を γ_{ij} $(i,j=0,\cdots,s+1,\ i\neq j)$ とすれば，$\sum_{j=0}^{s+1}\gamma_{ij}=\Gamma$ (定数) となる．

サーバがダウン ($s+1$ 状態に推移) すれば直ちに回復動作が実施され，回復動作が完了するまでに要するオーバーヘッドの確率分布を $H_2(x)$ (平均: $1/\chi\ (>0)$) とする．サーバがダウンしなければ，(i) トランザクション処理を完了する (ii) アイドル状態で新しいトランザクションが到着する (iii) アイドル状態でサーバの状態が変化する，のいずれかの時点で，予防的にソフトウェア若化を実施するかシステム動作を継続するかの選択を行う．若化を実施するために必要なオーバーヘッドの確率分布関数を $H_1(x)$ (平均: $1/\omega\ (>0)$) とし，$x_1\ (>0)$ と $x_2\ (>0)$ を，それぞれ若化と回復動作に要する単位時間当たりの費用とする．障害が発生したり若化を実施すると，システム内で未処理のトランザクションは全て失われる．また，回復動作中もしくは若化中に新たに到着するトランザクションは全てシステムから拒否されるものと仮定する．これらの仮定は，すべて関連の文献 [3], [4] においてなされたものと同様である．失われた各トランザクションに対しては $\alpha\ (>0)$ 単位の費用が発生するものとする．

3. セミマルコフ決定過程

システムの状態は待ち行列長 $i\,(\geq 0)$ とサーバの状態 $k\,(0\leq k\leq s+1)$ により記述される．状態 (i,k) $(k\leq s)$ を観測する時点において，行動 1 (ソフトウェア若化) と行動 2 (処理継続) のいずれかを実施する．また，障害が発生した場合 (すなわち状態が $(i,s+1)$ のとき)，行動 3 (回復動作) を選択する．$P_i(x)$ を時間 x の間に i 個のトランザクションが到着する確率，$T_{kl}(x)$ を x 時間経過後にサーバーの状態が k から l に推移する確率とする．$Q^{(\delta)}((i,k),(j,l))$ を行動 $\delta\ (=1,2,3)$ の下で状態 (i,k) から状態 (j,l) へ推移する確率として定義すれば，それぞれ以下のように導出することができる．

(i) $$Q^{(1)}\Big((i,k),(0,0)\Big)=\int_0^\infty dH_1(t)=1.$$

ここで，このときの期待推移時間を h_1 とする．

(ii)

$$Q^{(2)}\Big((i,k),(j,l)\Big)=\begin{cases}\int_0^\infty P_{j-i+1}(t)T_{kl}(t)g_k(t)dt & (i\geq 1, l\leq s)\\ \int_0^\infty P_{j-i}(t)\overline{G}_k(t)dT_{ks+1}(t) & (i\geq 1, l=s+1)\\ \int_0^\infty \gamma_{kl}e^{-(\Gamma+\lambda)t}dt & (i=0, j=0)\\ \int_0^\infty \lambda e^{-(\Gamma+\lambda)t}dt & (i=0, j=1, k=l).\end{cases}$$

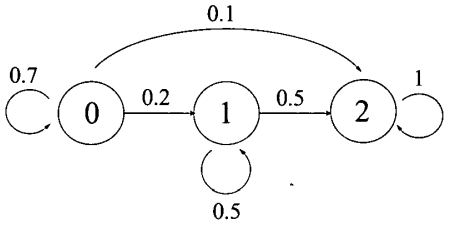

図 1: Transition diagram of server state.

ここで，a_k，b_k，c_k，d はそれぞれの場合に対応する期待推移時間である．

(iii) $Q^{(3)}\Big((i,s+1),(0,0)\Big) = \displaystyle\int_0^\infty dH_2(t) = 1.$

ここで，このときの期待推移時間を h_2 とする．

4. 最適性方程式

次のような費用関数を定義する

$V(i,k)$: 状態 (i,k) における相対値関数

$M(i,k)$: 状態 (i,k) で行動 1 が選択され，以降最適に振る舞った場合の期待費用

$W(i,k)$: 状態 (i,k) で行動 2 が選択され，以降最適に振る舞った場合の期待費用

z: 定常状態における単位時間当たりの期待費用の最小値.

以上の準備より，最適性 (ベルマン) 方程式は以下のようになる．

$$
\begin{aligned}
V(i,k) &= \min[M(i,k), W(i,k)], \\
V(i,s+1) &= \alpha i + x_2 h_2 + \alpha\lambda h_2 + V(0,0) - z h_2, \\
W(0,k) &= \frac{1}{\lambda+\Gamma}\Big[\sum_{l=0}^{s+1}\gamma_{kl}V(0,l) + \lambda V(1,k)\Big] \\
&\quad -z(c_k+d), \\
M(i,k) &= \alpha i + x_1 h_1 + \alpha\lambda h_1 + V(0,0) - z h_1, \\
W(i,k) &= \sum_{j=0}^{\infty}\sum_{l=0}^{s}\int_0^\infty P_j(t)T_{kl}(t) \\
&\quad \times V(i+j-1,l)g_k(t)dt \\
&\quad + \sum_{j=0}^{\infty}\int_0^\infty P_j(t)\overline{G}_k(t)V(i+j,s+1) \\
&\quad \times dT_{k\,s+1}(t) - z(a_k+b_k).
\end{aligned}
$$

上記の最適性方程式を解くには，Data Transmission Method に基づいたセミマルコフ決定過程に対する Value Iteration Method を適用すればよい [5].

いま，以下の仮定を設定する.

(A-1) 任意の m に対して $\sum_{l=m}^{s+1}\gamma_{kl}$ は k について単調増加

(A-2) 任意の x に対して $\overline{H_2}(x) > \overline{H_1}(x)$

(A-3) 任意の k に対して $g_k(x)/\overline{G}_k(x)$ は k について単調減少

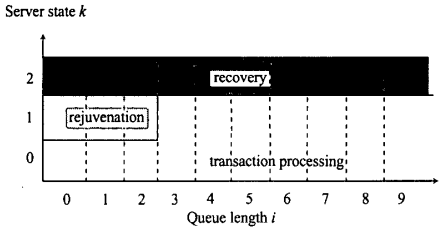

図 2: Decision Map: $\lambda = 0.2$, $\mu_0 = 0.3$, $\mu_1 = 0.25$, $x_1 = 1.2$, $x_2 = 2$, α=1, $\omega = 0.3$, $\chi = 0.2$.

(A-4) 任意の x に対して $\lambda \le g_k(x)/\overline{G}_k(x)$.

仮定 (A-1) はサーバが劣化するとさらに劣化しやすい状態に推移することを意味している．(A-2) は回復動作に要する時間は若化に要する時間よりも確率的に大きいことを，(A-3) はサーバが劣化するにつれトランザクションの処理率が低下することを，(A-4) はトランザクションの処理率は常に到着率よりも大きいことを，それぞれ示唆している．以上の仮定の下で，制御限界タイプの若化スケジュールが最適となることを示す.

定理: 仮定 (A-1)-(A-4) の下で，任意の $k \le l$ と $j \ge i$ に対し $D(i,l) = 2$ ならば $D(j,k) = 2$ となるような最適な若化スケジュール (i^*, k^*) が存在する

5. 数値例

図 1 に示すように，サーバの状態が 3 状態 (0: 正常，1: 劣化，2: システムダウン) 吸収マルコフ連鎖によって記述されるものとする．図 2 において，最適若化タイミングを決定するための Decision Map を示す．これは Value Iteration Method によって最適性方程式を解くことで得られ，各選択時点において若化を行うかトランザクション処理の継続を行うかについての動的な意思決定指標を与えるものである..

参考文献

[1] Bobbio, A., Sereno, M. and Anglano, C. (2001), Fine grained software degradation models for optimal rejuvenation policies, *Performance Evaluation*, **46**, 45–62.

[2] Dohi, T., Goševa-Popstojanova, K. and Trivedi, K. S. (2001), Estimating software rejuvenation schedule in high assurance systems, *The Computer Journal*, **44**, 473–485.

[3] Garg, S., Pfening, S., Puliafito, A., Telek, M. and Trivedi, K. S. (1998). Analysis of preventive maintenance in transactions based software systems, *IEEE Trans. on Computers*, **47**, 96–107.

[4] Pfening, S., Garg, S., Puliafito, A., Telek, M. and Trivedi, K. S. (1996), Optimal rejuvenation for tolerating soft failure, *Performance Evaluation*, **27/28**, 491–506.

[5] Tijms, H. C. (1994), *Stochastic Models: An Algorithmic Approach*, John Wiley & Sons, Chichester.