2001年度日本オペレーションズ・リサーチ学会
秋季研究発表会

ソフトウエアのテストプロセスモデル

（株）タトウ　高橋暁史　TAKAHASHI Akifumi
01102990　神奈川大学　*紀　一誠　KINO Issei

1. はじめに

開発された直後のソフトウエア製品には多数のバグが含まれており、これらのバグをテストしながら取り除いていく。何回かのテスト工程を繰り返したのち、様々な品質評価指標を調べ、品質が確保されたと判断された後出荷される。評価指標の1つにバグ残存率というものがある。これは、テスト工程で取りきれなかったバグがどの程度の割合で残っているかを示す指標である。本稿では、1回のテスト工程で残存してしまうバグの割合を確率変数とし、その分布を仮定することにより、テスト回数 n とバグ残存分布との関係を導く。

2. モデル

はじめに B_0 個（ある定数）のバグがあったとする。1回目のテスト後にはバグ数は B_1（確率変数）個に減る。このときのバグ残存率を表す確率変数を

$$X_1 = \frac{B_1}{B_0}$$

と定義する。テストを n 回繰り返すものとする。B_i を i 回目のテスト終了後に残存しているバグ数とし、i 回目のテストにおけるバグ残存率を示す確率変数

$$X_i = \frac{B_i}{B_{i-1}}, (i = 1, 2, \cdots, n)$$

と定義する。n 回テストを繰り返した後のバグ残存率を確率変数 Z_n とすれば、次のような関係となる。

$$Z_n = \frac{B_n}{B_0} = X_1 X_2 \cdots X_n$$

ここで、$X_1, X_2, \cdots, X_n$ は互いに独立に同一の分布に従うものとし、その確立密度関数は次の形とする。

$$f(x) = \frac{m+1}{m}(1 - x^m) \tag{1}$$

3. 解析

3.1 方針

Z_n の分布を求めることが本稿の目標である。Z_n は確率変数の積であるので、その対数をとったものを考える。そのため、確率変数

$$S_n = Y_1 + Y_2 + \cdots + Y_n$$

を考える。ただし、$Y_i = -\log X_i, (i = 1, 2, \cdots, n)$ とし、対数の底は e とする。確率変数 $Y_1, Y_2, \cdots, Y_n$ も互いに独立に同一の分布にしたがう。このとき、

$$\log Z_n = -(Y_1 + Y_2 + \cdots + Y_n) = -S_n$$

であるから、

$$Z_n = e^{-S_n}$$

となる。S_n の分布関数を

$$F(x) = P(S_n \leq x)$$

とすれば、$\{S_n \leq x \iff \log Z_n \geq -x\}$ であるから、

$$F(x) = P(\log Z_n \geq -x) = P(Z_n \geq e^{-x})$$

となる。従って、

$$\frac{d}{dx}P(Z_n \leq e^{-x}) = -\frac{d}{dx}F(x) = -f(x)$$

となる。Z_n の密度関数を $\phi(y)$ とすれば、変数変換 $y = e^{-x}$ を用いて、

$$\phi(y) = \frac{d}{dy}P(Z_n \leq y) = \frac{1}{y}f(-\log y) \tag{2}$$

となる。この関係を用いて、S_n 分布 $f(x)$ が分かれば Z_n の分布 $\phi(y)$ を知る事ができる。このため、本稿ではまず S_n のラプラス変換形を求める。次にその逆ラプラス変換を行なうことにより $f(x)$ を求め、これを用いて Z_n の分布を導く。

3.2 S_n のラプラス変換形

Y_i のラプラス変換は、$i=1,2,\ldots,n$ について、

$$E(e^{-sY_i}) = E(e^{s\log X_i}) = E(X_i^s)$$

となる。ここで (1) を用いて、

$$\begin{aligned} E(X_i^s) &= \int_0^1 x^s f(x)\,dx \\ &= \frac{m+1}{(s+1)(s+m+1)} \end{aligned}$$

であり、

$$E(e^{-sS_n}) = \prod_{i=1}^{n} E(e^{-sY_i})$$

より、S_n のラプラス変換は次のようになる。

$$E(e^{-sSn}) = \frac{(m+1)^n}{(s+1)^n(s+m+1)^n}$$

3.3 ラプラス逆変換

$E(e^{-sSn})$ を部分分数展開すると、

$$\begin{aligned} E(e^{-sSn}) &= (m+1)^n[\frac{\alpha_0}{(s+1)^n} + \frac{\alpha_1}{(s+1)^{n-1}} + \cdots \\ &\quad + \frac{\alpha_{n-1}}{s+1} + \frac{\beta_0}{(s+m+1)^n} \\ &\quad + \frac{\beta_1}{(s+m+1)^{n-1}} + \cdots + \frac{\beta_{n-1}}{s+m+1}] \end{aligned}$$

となる。ここで、α_j は、

$$\alpha_j = \frac{1}{j!}\frac{d^j}{ds^j}(s+1)^n E(e^{-sS_n})\Big|_{s=-1}$$

と定める事ができ、β_j も同様に得られる。その結果、

$$\begin{aligned} E(e^{-sSn}) &= \Big(\frac{m+1}{m}\Big)^n \sum_{k=0}^{n-1} \frac{(-1)^k[n]_k}{k!m^k}\{\frac{1}{(s+1)^{n-k}} \\ &\quad + (-1)^{n+k}\frac{1}{(s+m+1)^{n-k}}\} \end{aligned}$$

となる。ここで、$[n]_k = n(n+1)\cdots(n+k-1)$ で、$k \geq 1, [n]_0 = 1$ とする。したがって、アーラン分布の逆変換形を利用し、$E(e^{-sSn})$ のラプラス逆変換形 $f(x)$ は、

$$\begin{aligned} f(x) &= \Big(\frac{m+1}{m}\Big)^n \sum_{k=0}^{n-1} \frac{(-1)^k[n]_k}{k!m^k}\frac{x^{n-k-1}}{(n-k-1)!} \\ &\quad \{e^{-x} + (-1)^{n+k}e^{-(m+1)x}\} \end{aligned}$$

となる。したがって (2) を用いて、Z_n の密度関数は

$$\begin{aligned} \phi(y) &= \frac{1}{y}\Big(\frac{m+1}{m}\Big)^n (-1)^{n-1}\sum_{k=0}^{n-1}\frac{[n]_k}{k!m^k} \\ &\quad \frac{(\log y)^{n-1}}{(n-k-1)!}\{y + (-1)^{n+k}y^{m+1}\} \end{aligned}$$

という形に得られる。

4. 結果

Mathematica を用いた数値計算を示す。

$m = 0.1$ の場合を例にとり、$E(Z_n)$ (平均バグ残存率) および α% 以上バグが残存している確率 $P(Z_n \geq \alpha) = \int_\alpha^1 \phi(y)\,dy$ とテスト回数 n の関係を次に示す。数値は % 表示である。

n	$E(Z_n)$	$\alpha = 10\%$	$\alpha = 5\%$	$\alpha = 1\%$
1	26.19	69.43	82.05	95.31
2	6.85	22.41	38.40	70.95
3	1.80	3.65	9.87	35.36
4	0.47	0.36	1.52	11.59
5	0.12	0.023	0.16	2.60

これらの結果から、バグの残存率を α としたときに、何回くらいテストを繰り返せばその値を実現できるのか、あるいは、n 回テストを行った後にバグの残存率が α を超える確率はどの程度になるかなどを定量的に評価できることが示された。

5. おわりに

実際にソフト開発を行う時にあらかじめ何回のテストでバグの残存率が基準値に達するか分かっていれば便利である。実用的にするには、今回は $f(x)$ を便宜上仮定したが、どんな $f(x)$ の場合でも計算できるようにする必要がある。

参考文献

[1] クラインロック著、手塚慶一訳『待ち行列システム理論』(上) ((株) マグロウヒル好学社、1979 年)

[2] Stephen Wolfram 原著者『Mathematica ブック』(株式会社トッパン、1998 年)

[3] W. フェラー原著者、国沢清典訳『確率論とその応用 II』上 (株式会社紀伊国屋書店、1969 年)