Optimal Release Policies for Hyper-Geometric Distribution Software Reliability Growth Model with Scheduled Delivery Time[1]

Rong-Huei Hou, Ing-Yi Chen[2], Yi-Ping Chang[3], and Sy-Yen Kuo
Department of Electrical Engineering
National Taiwan University
Taipei, Taiwan, R.O.C.
Email: sykuo@cc.ee.ntu.edu.tw.

Abstract

The Hyper-Geometric Distribution software reliability growth Model (HGDM) has been used for estimating the number of initial faults in a software program. Another important problem in the software development process is to determine when to stop testing and release the software. In this paper, we investigate the optimal release policies minimizing the total expected software cost with a scheduled software delivery time for the HGDM. The total expected software cost includes the penalty cost which should be paid by the manufacturer if the software is delivered after the scheduled delivery time. The main result is that the optimal release time can be determined and shown to be finite. Numerical examples illustrating the optimal software release problem are also presented.

1. Introduction

In recent years, computer systems have been widely applied for the control of many complex and critical systems, such as nuclear reactors, space shuttles, chemical plants, etc. The breakdown of a computer system, caused by software faults, may result in tremendous damage for social life. Therefore, it is very important to develop a highly reliable software system, and software reliability is one of the key issues in the software product development. In the literature, many Software Reliability Growth Models (SRGMs) have been developed (e.g., Goel and Okumoto [1], Musa and Okumoto [2], Ohba [3], Yamada *et al.* [4] and Tohma *et al.* [5]). The SRGMs are usually used to estimate the number of remaining faults and the software reliability. However, another important problem is to determine when to stop testing and release the software.

In general, the longer the software is tested, the more reliable it will be. However, the delay in the test period will cause additional cost and the late release for operational use will also lead to the users' dissatisfaction. Therefore, it is important to determine when to stop testing and release the software from the cost–benefit viewpoint. Such decision problem is called an optimum software release problem, and it has been studied by Okumoto and Goel [6], Koch and Kubat [7], Yamada *et al.* [8], Kapur and Garg [9], Ross [10], etc. In [6], Okumoto and Goel addressed the optimum release problem considering the cost–benefit of the software. However, the model excludes penalty cost due to delay for a scheduled time. It is usually assumed that the additional penalty cost should be paid if a software is released after the scheduled delivery time. Therefore, the optimal release policies minimizing the total expected software cost with a scheduled software delivery time have been investigated [7–9].

The Hyper–Geometric Distribution software reliability growth Model (HGDM) for estimating the number of initial faults has been proposed and studied by Tohma *et al.* [11–12] and Jocoby *et al.* [13–14]. In their papers, they usually assume the learning curve of the HGDM is linear. However, the linear learning curve assumption seems not realistic in some applications. Traditionally, the exponential or the S–shaped curve is commonly used to interpret human learning process. Therefore, the HGDM with the exponential or the S-shaped learning curve has been proposed by Hou *et al.* [15].

In this paper, we will discuss the optimal software release policies minimizing the total expected software cost with a scheduled delivery time for the HGDM

[1]This research was supported by the National Science Council, Taiwan, R.O.C., under Grants NSC 82–0408–E002–021 and NSC 82–0404–E033–049–T.

[2]Department of Electric Engineering, Chung Yuan University, Chungli, Taiwan, R.O.C.

[3]Department of Business Mathematics, Soochow University, Taipei, Taiwan, R.O.C.

with various learning curves. A brief review of the HGDM with various learning curves is given in Section 2. The software cost model including penalty cost is introduced in Section 3. The optimal software release policies with the scheduled delivery time are discussed in Section 4. Numerical results are presented for illustration in Section 5 followed by the conclusions in Section 6.

2. HGDM with Various Leaning Curves

In this section, we will briefly review the Hyper–Geometric Distribution software reliability growth Model (HGDM) [14] and the concept of applying various learning curves to the HGDM [15].

2.1. HGDM

In the software development process, programmers first debug their products by themselves. After this preliminary debugging, the products will be passed to test workers for further test-and-debugging. In general, a program is assumed to have m initial faults when the test–and-debugging stage begins. The collection of test operations performed in a day or a week is called a *"test instance"*. Test instances are denoted by t_i, $i = 1, 2, \ldots, n$ in accordance with the order of applying them. The *"sensitivity factor"*, w_i, represents how many faults are discovered by the application of the test instance t_i. Especially, each fault will be classified into one of the two categories, **newly discovered** faults or **rediscovered** faults. Note that the number of newly detected faults at the application of the i^{th} test instance is not necessarily equal to w_i.

Considering the application of t_i, let C_{i-1} be the number of faults already detected so far by $t_1, t_2, \ldots, t_{i-1}$ and N_i be the number of faults newly detected by t_i. Then, some of the w_i faults may be those that already counted in C_{i-1}, and the remaining w_i faults account for the newly detected faults. With the assumption that new faults will not be inserted into the program while correction is being performed, the conditional probability $Prob(N_i = x_i \mid m, w_i, C_{i-1})$ can be formulated as

$$Prob(N_i = x_i \mid m, w_i, C_{i-1}) = \frac{\binom{m - C_{i-1}}{x_i}\binom{C_{i-1}}{w_i - x_i}}{\binom{m}{w_i}}$$

where $C_{i-1} = \sum_{k=1}^{i-1} x_k$, $C_0 = 0$, and x_k is an observed instance of N_k. The expected value of C_i denoted by EC_i is [14]

$$EC_i = m\left[1 - \prod_{j=1}^{i}(1 - p_j)\right], \qquad (1)$$

where $p_i = w_i/m$.

In general, w_i is usually defined as follows [14]:

$$\begin{aligned} w_i &= mu_i(ai + b), \text{ with} && (2) \\ u_i &= \{\text{number of testers}(i) \text{ or computer time}(i) \text{ or number of test items}(i)\}, \end{aligned}$$

where $a > 0$ and i represents the i^{th} test instance.

2.2. HGDM with Exponential or Logistic Learning Factor

The HGDM usually assumes that the learning curve of testers' skill involved in w_i is linear [14]. However, the assumption of the linear learning curve seems not realistic in some applications. Traditionally, the exponential or the S–shaped curve is commonly used to interpret human learning process. Therefore, we consider two w_i functions based on the exponential and the S-shaped learning curve, respectively [15].

The first w_i function is

$$w_i = mp_{LT}(1 - e^{-ai}),\ a > 0,\ 0 < p_{LT} \le 1, \qquad (3)$$

which is called the *"exponential learning factor"*. The second w_i function is

$$w_i = mp_{LT}\frac{1}{1 + be^{-ai}},\ a > 0,\ b > 0,\ 0 < p_{LT} \le 1, \ (4)$$

which is called the *"logistic learning factor"*.

In this paper, we will discuss the optimal software release policies minimizing the total expected software cost with a scheduled software delivery time for the HGDM with the exponential and the logistic learning factors, respectively.

3. Software Cost Model

Okumoto and Goel [6] have investigated the software optimum release policies from the cost–benefit viewpoint. The software cost model proposed by Okumoto and Goel [6] consists of the cost of testing the software before release, the cost of fixing faults during the testing phase, and the cost of fixing faults during the operational phase. However, the model excludes penalty cost due to delay for a scheduled time. In general, the additional penalty cost should be paid if a software is released after the scheduled delivery time. Considering the effect of cost penalty, the optimal release policies minimizing the total expected software cost with a scheduled software delivery time have been investigated [7-9]. The penalty cost function for the HGDM can be written as

$$C_D(i) = \begin{cases} 0, & i < D; \\ c_4 + c_5 g(i - D), & i \geq D; \end{cases} \qquad (5)$$

where
c_4, c_5 are nonnegative real numbers;
D is the scheduled software delivery time ($D \geq 1$);
$C_D(i)$ is the penalty cost function due to delay for the scheduled software release time.

Besides, it is reasonable that the longer delay the software is delivered, the more penalty cost should be paid by the manufacturer. Therefore, $g(i)$ is usually assumed to satisfy the following three conditions A1–A3.

A1: $g(0) = 0$.

A2: $g(i)$ is increasing in i.

A3: $g(i+1) + g(i-1) \geq 2g(i)$ for all $i \geq 1$.

For example, $g(i)$ can be assumed to be proportional and exponential to the time interval between the scheduled software delivery time and the software release time as follows [8], respectively.

(i) $g(i) = i$;

(ii) $g(i) = i^h, \ h > 1$;

(iii) $g(i) = e^{hi} - 1, \ h > 0$.

Including penalty cost, the total expected software cost for the HGDM is given by

$$Cost(i) = c_1 EC_i + c_2(m - EC_i) + c_3 i + C_D(i), \quad i = 0, 1, 2, \cdots, \qquad (6)$$

where
$Cost(i)$ is the total expected cost when the software is released at the i^{th} test instance time;
c_1 is the cost of fixing a fault during the testing phase;
c_2 is the cost of fixing a fault during the operational phase ($c_2 > c_1$);
c_3 is the cost of per unit time of software testing.

In the following section, we will determine the optimal release time I^* of i minimizing the total expected software cost Eq.(6) and prove that I^* is finite.

4. Optimal Release Policies with Scheduled Delivery Time

To determine the optimal release time for a software system with a scheduled delivery time seems very interesting. In this section, we will discuss such optimal software release policies for the HGDM with the exponential and the logistic learning factors, respectively.

Since $Cost(i)$ is the evaluation criterion, the optimum release problem is to find an optimum release time I^* of i which minimizes Eq.(6). Let

$$\begin{cases} f(0) = (c_2 - c_1)m; \\ f(i) = (c_2 - c_1)m\prod_{j=1}^{i}(1 - p_j) + c_3 i + C_D(i), \quad i = 1, 2, \cdots, \end{cases} \qquad (7)$$

and then minimizing Eq.(6) is equivalent to minimize Eq.(7). For convenience, let

$$C^*(i) = \frac{c_3 + C_D(i) - C_D(i-1)}{(c_2 - c_1)m}, \ i = 1, 2, \cdots, \qquad (8)$$

and

$$\begin{cases} \delta(1) = p_1; \\ \delta(i) = p_i \prod_{j=1}^{i-1}(1 - p_j), \quad i = 2, 3, \cdots. \end{cases} \qquad (9)$$

Since

$$\begin{cases} f(1) - f(0) = c_3 - (c_2 - c_1)mp_1 + C_D(1); \\ f(i+1) - f(i) = c_3 - (c_2 - c_1)mp_{i+1}\prod_{j=1}^{i}(1 - p_j) \\ \qquad + C_D(i+1) - C_D(i), \quad i = 1, 2, \cdots. \end{cases}$$

the following Lemma can be obtained.

Lemma 1. Assume $c_2 > c_1 > 0$ and $c_3 > 0$. For $i = 0, 1, 2, \cdots$, we have:

(i) if $C^*(i+1) > \delta(i+1)$, then $f(i+1) > f(i)$;

(ii) if $C^*(i+1) < \delta(i+1)$, then $f(i+1) < f(i)$;

(iii) if $C^*(i+1) = \delta(i+1)$, then $f(i+1) = f(i)$;

where $C^*(i)$ and $\delta(i)$ are defined in Eq.(8) and Eq.(9), respectively. □

From Eq.(5) and by simple calculation, the following Lemma can be obtained.

Lemma 2. Suppose that $g(i)$ satisfies the conditions A1–A3, we have

(i) $C_D(i+1) - C_D(i) \geq C_D(i) - C_D(i-1)$ for $i = 1, 2, \cdots, D-1, D+1, \cdots$;

(ii) $C_D(D+1) - C_D(D) > C_D(D) - C_D(D-1)$ if $c_5 g(1) > c_4$;

(iii) $C_D(D+1) - C_D(D) = C_D(D) - C_D(D-1)$ if $c_5 g(1) = c_4$;

(iv) $C_D(D+1) - C_D(D) < C_D(D) - C_D(D-1)$ if $c_5 g(1) < c_4$. □

To determine the optimal release time I^*, we need some properties of $\delta(i)$. In the following, the value of $\delta(i+1) - \delta(i)$ will be evaluated, and then some properties of $\delta(i)$ can be obtained.

For convenience, let

$$\Delta(i) = p_{i+1} - p_i - p_{i+1}p_i \ , \quad i = 1, 2, \cdots , \tag{10}$$

and then

$$\begin{cases} \delta(2) - \delta(1) = \Delta(1); \\ \delta(i+1) - \delta(i) = \Delta(i) \prod_{j=1}^{i-1}(1-p_j), i = 2, 3, \cdots . \end{cases} \tag{11}$$

Define

$$I = inf\{i \geq 1 : \Delta(i) < 0\}. \tag{12}$$

For the exponential and logistic learning factors, we have the following Lemmas, respectively .

Lemma 3. If $p_i = p_{LT}(1 - e^{-ai})$, then:

(i) $\delta(i) \leq \delta(i+1)$ for $1 \leq i \leq I - 1$;

(ii) $\delta(i) > \delta(i+1)$ for $i \geq I$;

where I is defined in Eq.(12).

Proof: Since $p_i = p_{LT}(1 - e^{-ai})$, $\Delta(i)$ is decreasing in i [15] and $\lim_{i\to\infty} \Delta(i) = -p_{LT}^2 < 0$. It means that I is finite and unique. From Eq.(11), this Lemma can be proved. □

Lemma 4. If $p_i = p_{LT}/(1 + be^{-ai})$, then:

(i) $\Delta(i)$ is decreasing for $i \geq I$;

(ii) $\delta(i) \leq \delta(i+1)$ for $1 \leq i \leq I-1$ and $\delta(i) > \delta(i+1)$ for $i \geq I$;

where I is defined in Eq.(12).

Proof: From Eq.(10), we have

$$\Delta(i) = \frac{p_{LT}e^{-a(i+1)}(be^a - b - p_{LT}e^{a(i+1)})}{(1 + be^{-a(i+1)})(1 + be^{-ai})}.$$

Therefore,

$$\Delta(i) < 0 \text{ if and only if } be^a - b - p_{LT}e^{a(i+1)} < 0. \tag{13}$$

Besides, the value of $\Delta(i+1) - \Delta(i)$ is equal to

$$p_{LT}\frac{D(i)}{(1 + be^{-a(i+2)})(1 + be^{-a(i+1)})(1 + be^{-ai})},$$

where $D(i)$ is equal to

$$b(e^{-a}-1)e^{-2a(i+1)}\{b - be^a + (p_{LT}-1)e^{a(i+1)} + (p_{LT}+1)(e^{a(i+2)}\}.$$

Since $a > 0$, i.e., $(e^{-a} - 1) < 0$, we have

$D(i) < 0$ if and only if

$$b - be^a + (p_{LT} - 1)e^{a(i+1)} + (p_{LT} + 1)e^{a(i+2)} > 0.$$

Suppose that $\Delta(i') < 0$, by Eq.(13), we have

$$\begin{aligned} & b - be^a + (p_{LT} - 1)e^{a(i'+1)} + (p_{LT} + 1)e^{a(i'+2)} \\ = & -(be^a - b - p_{LT}e^{a(i'+1)}) + e^{a(i'+1)}((p_{LT}+1)e^a - 1) > 0. \end{aligned}$$

That is, $\Delta(i'+1) < \Delta(i')$. Therefore, by the definition of I, $\Delta(i)$ is decreasing for $i \geq I$. Besides, by Eq.(11), we have $\delta(i) \leq \delta(i+1)$ for $1 \leq i \leq I - 1$ and $\delta(i) > \delta(i+1)$ for $i \geq I$. □

For convenience, we define

$$I_f = inf\{i \geq I : \ \delta(i+1) \leq C^*(i+1)\}, \tag{14}$$

and

$$S = \{D-1 \leq i \leq I-1 : \delta(i) > C^*(i) \text{ and } \delta(i+1) \leq C^*(i+1)\}. \tag{15}$$

If $p_i = p_{LT}(1 - e^{-ai})$ or $p_i = p_{LT}/(1 + be^{-ai})$, then $\delta(i)$ is decreasing in i, $\lim_{i\to\infty} \delta(i) = 0$, and $C^*(i+1) \geq c_3/[(c_2 - c_1)m] > 0$ for all $i \geq I$. It is obvious that I_f is finite. In the following main Theorem, we will determine the optimal release time I^* and show that it is finite.

Theorem 1. Suppose that $g(i)$ satisfies the conditions A1–A3. If $p_i = p_{LT}(1-e^{-ai})$ or $p_i = p_{LT}/(1+be^{-ai})$, we have

(I) if $c_5g(1) \geq c_4$
 (i) if $D > I$, then $I^* \in \{0, I_f\}$;
 (ii) if $D \leq I$, then $I^* \in S \cup \{0, I_f\}$;

(II) if $c_5g(1) < c_4$
 (i) if $D > I$, then $I^* \in \{0, I_f, D\}$;
 (ii) if $D \leq I$, then $I^* \in S \cup \{0, I_f, D\}$;

where I, I_f, and S are defined in Eq.(12), Eq.(14), Eq.(15), respectively.

Proof: Consider the following two cases for the relationship between $c_5g(1)$ and c_4.

Case (I): $c_5g(1) \geq c_4$.

From Eq.(5), we have

$$C^*(1) = \cdots = C^*(D-1) \leq C^*(D) \leq C^*(D+1) \leq \cdots \leq C^*(\infty). \tag{16}$$

If $D > I$ and $C^*(I) \geq \delta(I)$, by Lemma 2 and Lemma 3, we have $C^*(i) > \delta(i)$ for $i = 1, 2, \cdots, I-1, I+1, \cdots$. By Lemma 1, we have $f(0) < f(1) < \cdots < f(I+1) \leq f(I) < f(I+1) < \cdots$. That is, the cost is minimum at $i = 0$. Therefore, $I^* = 0$.

If $D > I$ and $\delta(I) > C^*(I) \geq \delta(1)$, by Lemma 2, Lemma 3, and Eq.(16), there exists two finite and unique integers I_A ($I_A < I$) and I_f such that

$$\begin{aligned} &C^*(i) \geq \delta(i) \text{ for } i = 1, 2, \cdots, I_A; \\ &C^*(i) < \delta(i) \text{ for } i = I_A + 1, I_A + 2, \cdots, I_f; \\ &C^*(i) \geq \delta(i) \text{ for } i = I_f + 1, I_f + 2, \cdots. \end{aligned}$$

Therefore, by Lemma 1, we have

$$\begin{aligned} &f(0) \leq f(1) \leq \cdots \leq f(I_A); \\ &f(I_A) > f(I_A + 1) > \cdots > f(I_f); \\ &f(I_f) \leq f(I_f + 1) \leq \cdots. \end{aligned}$$

This means if $f(0) \leq f(I_f)$, then $I^* = 0$; otherwise, $I^* = I_f$. That is $I^* \in \{0, I_f\}$.

If $D > I$ and $C^*(I) < \delta(1)$, by Lemma 2, Lemma 3, and Eq.(16), we have

$$\begin{aligned} &C^*(i) < \delta(i) \text{ for } i = 1, 2, \cdots, I_f; \\ &C^*(i) \geq \delta(i) \text{ for } i = I_f + 1, I_f + 2, \cdots. \end{aligned}$$

Therefore, by Lemma 1, we have

$$\begin{aligned} &f(0) > f(1) > \cdots > f(I_f); \\ &f(I_f) \leq f(I_f + 1) \leq \cdots. \end{aligned}$$

Therefore, $I^* = I_f$. By the above argument and simple arrangement, (i) of (I) of Theorem 1 holds.

If $D \leq I$, there may exist some integers k's $\in \{D - 1, D - 2, \cdots, I - 1\}$ such that

$$\delta(k) > C^*(k) \text{ and } \delta(k + 1) \leq C^*(k + 1).$$

Let S as defined in Eq.(15), and then $I^* \in S$.

Applying the same argument in the case $D > I$, the remaining results of the case $D \leq I$ can be obtained. Moreover, by simple arrangement, (ii) of (I) of Theorem 1 holds.

Case (II): $c_5 g(1) < c_4$.

From Eq.(5), we have

$$\begin{aligned} &C^*(1) = C^*(2) = \cdots = C^*(D - 1) \leq C^*(D); \\ &C^*(D) > C^*(D + 1); \\ &C^*(D + 1) \leq C^*(D + 2) \leq \cdots. \end{aligned}$$

Applying the same argument in Case (I), then (II) of theorem 1 can be obtained. □

In the following, some special cases of $C_D(i)$ are discussed, respectively.

Corollary 1. Assume $g(i) = i$. If $p_i = p_{LT}(1 - e^{-ai})$ or $p_i = p_{LT}/(1 + be^{-ai})$, then $I^* \in \{0, I_f, D - 1\}$.

Proof: If $g(i) = i$, we have

$$C_D(i + 1) - C_D(i) = \begin{cases} 0, & i = 0, 1, \cdots, D - 2; \\ c_4, & i = D - 1; \\ c_5, & i = D, D + 1, \cdots. \end{cases}$$

Hence, if $D \leq I$, by the definition of S, we have

$$S = \begin{cases} D - 1, & \text{if } \delta(D-1) > C^*(D-1) \text{ and } \delta(D) \leq C^*(D); \\ \emptyset, & \text{otherwise.} \end{cases}$$

By the same argument of Theorem 1, Corollary 1 can be obtained. □

In some applications, the penalty cost due to delay for the scheduled delivery time may be negligible. Therefore, we can consider the case $c_4 = c_5 = 0$.

Corollary 2. Assume that $c_4 = 0$, $c_5 = 0$, and $g(i)$ satisfies the conditions A1–A3. If $p_i = p_{LT}(1 - e^{-ai})$ or $p_i = p_{LT}/(1 + be^{-ai})$, then $I^* \in \{0, I_f\}$. That is, if $f(I^*) < f(0)$ then $I^* = I_f$; otherwise, $I^* = 0$. □

In fact, if $c_4 = c_5 = 0$, then our cost model coincides with the cost model proposed by Okumoto and Goel [6]. On the contrary, in some applications, the penalty cost due to delay for the scheduled delivery time may be tremendous. Therefore, we can consider the case $c_5 \to \infty$.

Corollary 3. Assume that $c_5 \to \infty$ and $g(i)$ satisfies the conditions A1–A3. If $p_i = p_{LT}(1 - e^{-ai})$ or $p_i = p_{LT}/(1 + be^{-ai})$, we have

(I) if $D > I$, then $I^* \in \{0, D\}$;

(II) if $D \leq I$, then $I^* \in \{0, D - 1, D\}$. □

In this section, we discuss the procedure of determining the optimal software release time I^* for the HGDM with the exponential and the logistic learning factors, respectively. Especially, Theorem 1 shows that I^* is finite. Besides, some special cases of $C_D(i)$ are also discussed in above Corollaries.

5. Numerical Examples

In this section, we will use numerical examples to illustrate the optimal release policies for the HGDM with the exponential and the logistic learning factors, respectively. The data used in this analysis are the test-and-debug data of a software system [16]. The parameters of $Cost(i)$ are cited from [6] as: $c_1 = 1\$$ per fault, $c_2 = 5\$$ per fault, $c_3 = 10\$$ per week, and

$c_4 = 10\$$. The various values of the software scheduled delivery time D are assumed to be 10 and 20. The various values of c_5 in the penalty cost function $C_D(i)$ are assumed to be 1, 5, 10, 20, and 40. For simplicity, the following three cases of $g(i)$ are considered.

(i) $g(i) = i$;

(ii) $g(i) = i^2$;

(iii) $g(i) = e^i - 1$.

5.1. Exponential Learning Factor

The least squared estimate of the parameters of the HGDM with the exponential learning factor are $\widehat{m} = 2300.8$, $\widehat{a} = 0.1206$, and $\widehat{p_{LT}} = 0.1602$ [15]. Tables 1, 2, and 3 shows relationships between the parameter c_5 in the penalty cost function, the scheduled delivery time D, the optimal release time I^*, and the total expected software cost $Cost(I^*)$. The $g(i)$ functions are assumed to be $g(i) = i$, $g(i) = i^2$, and $g(i) = e^i - 1$ corresponding to Tables 1, 2, and 3, respectively. Besides, assuming $g(i) = i$, the total expected costs for the various values of c_5 are shown in Fig. 1 (where $D = 10$) and Fig. 2 (where $D = 20$), respectively.

5.2. Logistic Learning Factor

The least squared estimate of the parameters of the HGDM with the logistic learning factor are $\widehat{m} = 2313.4$, $\widehat{a} = 0.3362$, $\widehat{b} = 5.5395$, and $\widehat{p_{LT}} = 0.1363$ [15]. Tables 4, 5, and 6 shows relationships between c_5, D, I^*, and $Cost(I^*)$. The $g(i)$ functions are assumed to be $g(i) = i$, $g(i) = i^2$, and $g(i) = e^i - 1$ corresponding to Tables 4, 5, and 6, respectively. Besides, assuming $g(i) = i$, the total expected costs for the various values of c_5 are shown in Fig. 3 (where $D = 10$) and Fig. 4 (where $D = 20$), respectively.

From Tables 1–6, the following two phenomena can be observed.

(i) The more the penalty cost increases, the more the optimal release time I^* decreases.

(ii) The more c_5 increases, the more total expected cost $Cost(I^*)$ increases but the optimal release time I^* decreases.

(iii) The optimal release times for the exponential learning factor and the logistic learning factor are very close. That is, there is no significant difference between using the exponential learning factor and the logistic learning factor to determine the optimal release time.

6. Conclusions

We have discussed the optimal release policies for the HGDM with the exponential and the logistic learning factors, respectively. The total expected software cost with a scheduled software delivery time is used as the criterion for determining the optimal release time. The finiteness of the optimal release time I^* has also been proved. Besides, Corollary 2 shows that if $c_4 = c_5 = 0$ (the penalty cost due to delay for the scheduled delivery time can not be occurred), then the optimal release time belongs to $\{0, I_f\}$. On the contrary, Corollary 3 shows that if $c_5 \to \infty$, then the optimal release time belongs to $\{0, D-1, D\}$.

Table 1. I^* and $Cost(I^*)$ for the HGDM with the exponential learning factor when $g(i) = i$.

	$D = 10$		$D = 20$	
c_5	I^*	$Cost(I^*)$	I^*	$Cost(I^*)$
1	37	2766.75	37	2756.75
5	35	2868.99	35	2818.99
10	33	2988.06	33	2888.06
20	31	3205.81	31	3005.81
40	28	3585.15	28	3185.15

Table 2. I^* and $Cost(I^*)$ for the HGDM with the exponential learning factor when $g(i) = i^2$.

	$D = 10$		$D = 20$	
c_5	I^*	$Cost(I^*)$	I^*	$Cost(I^*)$
1	28	3198.15	30	2906.40
5	22	3991.97	26	3134.46
10	19	4506.71	24	3245.33
20	16	5083.09	23	3350.65
40	14	5622.46	21	3431.65

Table 3. I^* and $Cost(I^*)$ for the HGDM with the exponential learning factor when $g(i) = e^i - 1$.

	$D = 10$		$D = 20$	
c_5	I^*	$Cost(I^*)$	I^*	$Cost(I^*)$
1	16	4765.52	24	3138.93
5	14	5250.45	23	3266.07
10	14	5518.44	22	3335.86
20	13	5734.45	22	3399.75
40	12	6021.14	21	3460.39

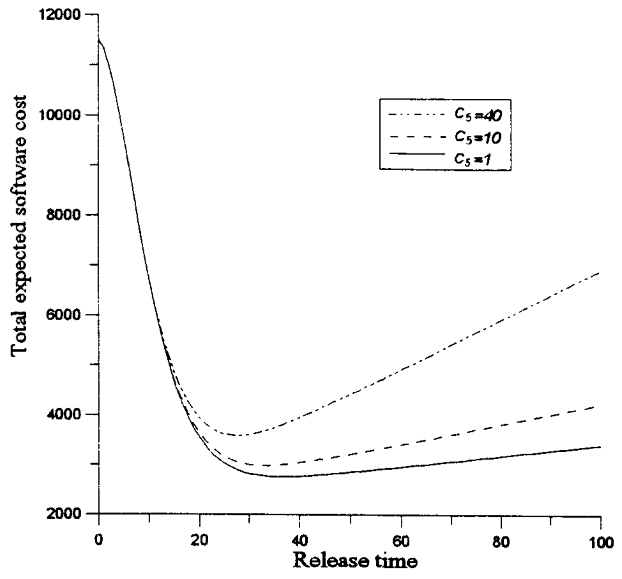

Fig 1. Dependence of c_5 on the total expected cost for the exponential learning factor with D=10.

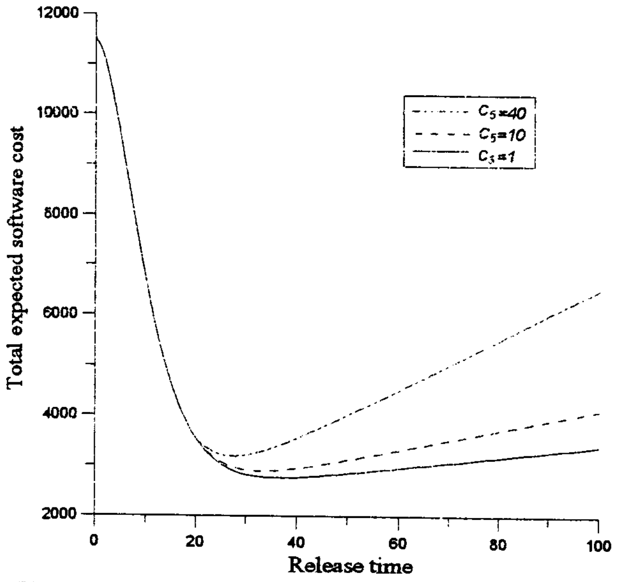

Fig 2. Dependence of c_5 on the total expected cost for the exponential learning factor with D=20.

Table 4. I^* and $Cost(I^*)$ for the HGDM with the logistic learning factor when $g(i)=i$.

	$D=10$		$D=20$	
c_5	I^*	$Cost(I^*)$	I^*	$Cost(I^*)$
1	38	2808.73	38	2798.73
5	36	2917.06	36	2867.06
10	34	3042.36	34	2942.36
20	31	3269.07	31	3069.07
40	28	3658.09	28	3258.10

Table 5. I^* and $Cost(I^*)$ for the HGDM with the logistic learning factor when $g(i)=i^2$.

	$D=10$		$D=20$	
c_5	I^*	$Cost(I^*)$	I^*	$Cost(I^*)$
1	28	3262.10	31	2970.07
5	22	4068.72	26	3211.99
10	19	4570.41	24	3324.54
20	16	5126.67	22	3428.72
40	14	5657.58	21	3505.31

Table 6. I^* and $Cost(I^*)$ for the HGDM with the logistic learning factor when $g(i)=e^i-1$.

	$D=10$		$D=20$	
c_5	I^*	$Cost(I^*)$	I^*	$Cost(I^*)$
1	16	4809.10	24	3218.14
5	14	5285.57	23	3344.65
10	14	5553.56	22	3412.61
20	13	5770.29	22	3476.50
40	12	6062.60	21	3534.04

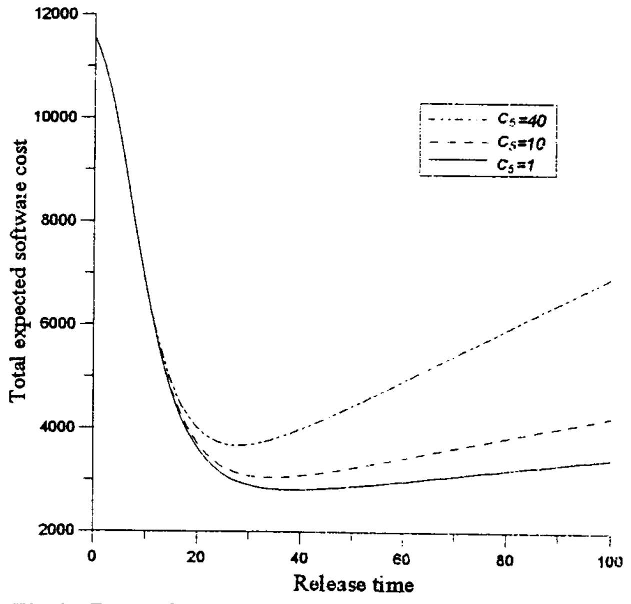

Fig 3. Dependence of c_5 on the total expected cost for the logistic learning factor with D=10.

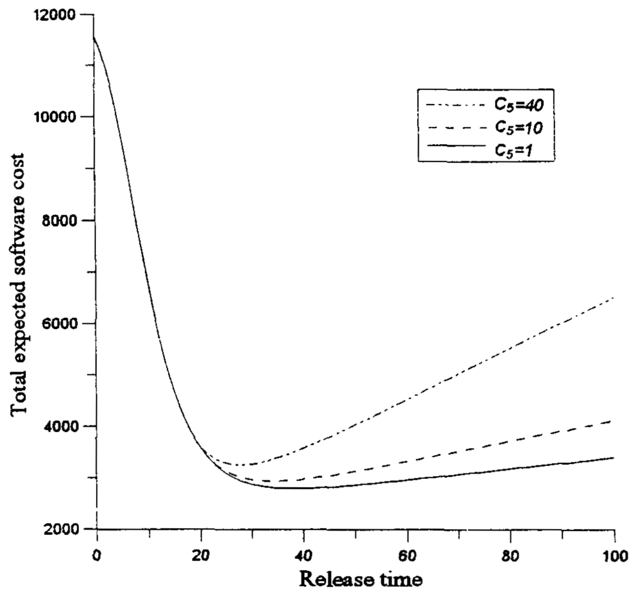

Fig 4. Dependence of c_5 on the total expected cost for the logistic learning factor with D=20.

References

[1] A. L. Goel and K. Okumoto, "Time-Dependent Error Detection Rate Model for Software Reliability and Other Performance Measures", *IEEE Trans. Reliability*, Vol. R-28, No. 3, pp. 206–211, August 1979.

[2] J. D. Musa and K. Okumoto, "A Logarithmic Poisson Execution Time Model for Software Reliability Measurement", *Proc. 7th Int. Conf. Software Engineering*, pp. 230–238, 1984.

[3] M. Ohba, "Software Reliability Analysis Models", *IBM J. Res. Develop.*, Vol. 28, No. 4, pp. 428–443, July 1984.

[4] S. Yamada, M. Ohba, and S. Osaki, "S-Shaped Software Reliability Growth Models and their Applications", *IEEE Trans. Reliability*, Vol. R-33, No. 4, pp. 289–292, October 1984.

[5] Y. Tohma, K. Tokunaga, S. Nagase, and Y. Murata, "Structural Approach to the Estimation of the Number of Residual Software Faults Based on the Hyper-Geometric Distribution", *IEEE Trans. Software Engineering*, Vol. 15, No. 3, pp. 345–355, March 1989.

[6] K. Okumoto and A. L. Goel, "Optimum Release Time for Software Systems Based on Reliability and Cost Criteria", *J. System Software*, Vol. 1, pp. 315–318, 1980.

[7] H. S. Koch and P. Kubat, " Optimal Release Time of Computer Software", *IEEE Trans. Software Engineering*, Vol. SE-9, No.3, pp. 323–327, 1983.

[8] S. Yamada, H. Narihisa, and S. Osaki, "Optimum Release Policies for a Software System with a Scheduled Delivery Time", *Int. J. Systems Science*, Vol. 15, pp. 905–914, 1984.

[9] P. K. Kapur and R. B. Garg, "Cost-reliability Optimum Release Policies for a Software System under Penalty Cost", *Int. J. Systems Science*, Vol. 20, pp. 2547–2562, 1989.

[10] S. M. Ross, "Software Reliability: The Stopping Problem", *IEEE Trans. Software Engineering*, Vol. SE-11, No.12, pp. 1472–1476, 1985.

[11] Y. Tohma, H. Yamano, M. Ohba, and R. Jacoby, "Parameter Estimation of the Hyper-Geometric Distribution Model for Real Test/Debug Data", *Proc. Int. Conf. Software Reliabiity*, pp. 28–34, 1991.

[12] Y. Tohma, H. Yamano, M. Ohba, and R. Jacoby, "The Estimation of Parameters of the Hypergeometric Distribution and Its Application to the Software Reliability Growth Model", *IEEE Trans. Software Engineering*, Vol. SE-17, No. 5, pp. 483–489, 1991.

[13] R. Jacoby and Y. Tohma, "The Hyper-Geometric Distribution Software Reliability Growth Model (HGDM): Precise Formulation and Applicability", *Proc. COMPSAC90*, Chicago, pp. 13–19, 1990.

[14] R. Jacoby and Y. Tohma. "Parameter Value Computation by Least Square Method and Evaluation of Software Availability and Reliability at Service-Operation by the Hyper-Geometric Distribution Software Reliability Growth Model (HGDM)", *Proc. 13th Int. Conf. Software Engineering*, pp. 226–237, 1991.

[15] R. H. Hou, S. Y. Kuo, and Y. P. Chang, "Applying Various Learning Curves to Hyper-Geometric Distribution Software Reliability Growth Model", to appear in *Proc. Int. Symposium on Software Reliability Engineering*, Monterey, Californa, Nov. 6-9, 1994.

[16] A. L. Goal, "Software Reliability Modeling and Estimation Technique", Final Technical Report RADC-TR-82-263, 1983.