Probability in the Engineering and Informational Sciences, **36**, 2022, 105–125.
doi:10.1017/S0269964820000352

NASH EQUILIBRIUM STRATEGIES REVISITED IN SOFTWARE RELEASE GAMES

Yasuhiro Saito
Department of Maritime Safety Technology, Japan Coast Guard Academy, 5-1 Wakabacho, Kure Hiroshima 737-0832, Japan
E-mail: yasu-saito@jcga.ac.jp

Tadashi Dohi
Department of Information Engineering, Hiroshima University, 1-4-1 Kagamiyama, Higashihiroshima 739-8527, Japan
E-mail: dohi@hiroshima-u.ac.jp

A software release game was formulated by Zeephongsekul and Chiera [Zeephongsekul, P. & Chiera, C. (1995). Optimal software release policy based on a two-person game of timing. *Journal of Applied Probability* 32: 470–481] and was reconsidered by Dohi *et al.* [Dohi, T., Teraoka, Y., & Osaki, S. (2000). Software release games. *Journal of Optimization Theory and Applications* 105(2): 325–346] in a framework of two-person nonzero-sum games. In this paper, we further point out the faults in the above literature and revisit the Nash equilibrium strategies in the software release games from the viewpoints of both silent and noisy type of games. It is shown that the Nash equilibrium strategies in the silent and noisy of software release games exist under some parametric conditions.

Keywords: Nash equilibrium strategy, noisy-type game, silent-type game, software release decision, software reliability, two-person nonzero-sum games

1. INTRODUCTION

One of the most important issues for a software developer is to determine an economic timing to release a software product with the satisfactory reliability requirement in the market. Since software testing cost is, in general, expensive through the development process, the software developer wishes to reduce the testing period as short as possible. On the other hand, if software testing is not sufficient and software failures caused by software faults occur in the market or the user side, the large amount of fixing and/or penalty costs may incur. Hence, the trade-off relationship between the above two cost components motivates us to seek the best timing to release the software. This problem is called *the software release problem* and has been extensively studied in software engineering [15,21,32]. In most cases, the software release problem is formulated as a minimization of the expected total software cost with some constraints, by taking account of the above trade-off relationship, and can be reduced to a simple deterministic optimization problem.

On the other hand, since the market of commercial software is always competitive, the software development project is rather influenced by an external environment such as the marketing behavior by a competitive developer. In such a situation, it is not enough to determine the software release timing based on only the developer's own cost minimization. Because the competitor may release his or her software in an earlier phase and may occupy the market of the similar software product. Zeephongsekul and Chiera [35] formulated this problem as a silent type of two-person nonzero-sum game of timing [1,14,25,29] and gave a Nash equilibrium strategy for two software developers. However, since their strategy holds in a very limited case, it can be easily shown that the corresponding strategy is not the Nash equilibrium in general cases. Dohi *et al.* [8] pointed out this fatal problem and revisited the original software release game. They further considered two noisy type of software release games as well.

The main purpose of this paper is to show that even the mixed strategies by Dohi *et al.* [8] are not also the Nash equilibrium strategies. We revisit the silent type of Nash equilibrium strategies in [8]. More specifically, under some parametric conditions, we show the existence of the Nash equilibrium strategies of software release game. Also, we revisit the noisy type of Nash equilibrium strategy in [8] in a similar way and derive the Nash equilibrium strategy in the noisy type of software release game.

The remaining part of this paper is organized as follows. Section 2 overviews the related work on software release problems. In Section 3, we formulate a standard software release problem to minimize the expected total software cost and the original silent-type software release game [8,35]. Section 4 gives two counterexamples of the silent type of Nash equilibrium strategies in [8,35]. Section 5 concerns the silent-type software release game and derives the Nash equilibrium strategies revisited and their associated value functions. In Section 6, we give the correct results in the noisy-type software release game where each player can know whether the opponent player has already acted or not. That is, each player can postpone his or her release action until the best timing if the opponent acts earlier and fails in the noisy-type game. Numerical examples are given in Section 7 to investigate the dependence of some model parameters on the game-theoretic software release decisions. Finally, the paper is concluded with some remarks in Section 8.

2. RELATED WORKS

Software release problem was first formulated by Okumoto and Goel [21] to determine the best timing to release the software after testing. Later, Koch and Kubat [15] and Yamada and Osaki [32] formulated different cost minimization problems from the seminal work in [21] and derived the different optimal software release policies. Along with the similar line, Wee [30], Ozekici *et al.* [22], Ho *et al.* [12], Hou *et al.* [13], Pham and Zhang [23], Xie *et al.* [31], Yang *et al.* [33], Chiu *et al.* [3], Sgarbossa and Pham [26], and Boland and Chuivi [2] also considered many software release problems with different cost functions, different constraints, and different stochastic models to describe the software debugging. However, it should be worth mentioning that all the software release problems formulated in the above literature are static problems in nature by minimizing the relevant expected costs with constraints and are categorized into an elementary optimization problem. Zheng [36] formulated the software release problem by means of a Markov decision process and proved that the optimal policy is threshold type.

On the one hand, several mathematical statisticians concerned the similar problems from the viewpoints of statistical perspective. Forman and Singpurwalla [10,11] dealt with the software release problems as hypothesis testing. Ross [24] also proposed a stopping rule of software testing by means of a control chart. Dalal and Mallow [4] formulated the software release problem as a sequential Bayesian decision model. Later, the same authors developed several graphical procedures to decide how much software should be tested before releasing in [5]. Dalal and McIntosh [6] presented a stochastic and economic framework to deal with systems that change as they are tested, and integrated the stopping rule [4] and the graphical approach [5]. Singpurwalla [28] proposed an interesting decision-making approach to maximize the expected utility. In these ways, much attentions have been paid in software engineering, mathematical statistics, and decision theory.

In software engineering perspective, several approaches to treat software release problems have been studied. The recent and most important paradigm shift in software engineering was the spread of open-source software. It is computer software with the source code available with a license in which the copyright holder provides the rights to study, change, and distribute the software to anyone and for any purpose. Since the reduction of software development cost is possible through the effective utilization of open-source software, it has been often used in the actual industry. Liu and Chang [17], Li *et al.* [16], Luo *et al.* [18], Yang *et al.* [34], Singh *et al.* [27], and Zhu and Pham [37] considered multiple release and version up problems of open-source software, which are rather different from the seminal works in [15,21,32]. The direct application of the machine learning technique, such as neural networks, is also becoming popular to predict the number of software bugs in system testing. Dohi *et al.* [7,9] and Momotaz and Dohi [19,20] studied the optimal software release decision via artificial neural networks.

As we have mentioned in Section 1, almost all authors ignored the existence of rival company in the software product market and just considered a single optimization problem to minimize the expected cost or to maximize the expected profit. In that sense, the software release games in [8,35] are quite interesting to understand the competitive marketing behavior by the software developers. Since these authors' results involve theoretical faults in terms of the Nash equilibrium strategies, the challenge in this paper seems to play a significant role in the software release engineering.

3. MODEL DESCRIPTION

The standard software release problem aims at determining the optimal release time which minimizes the expected total software cost. Suppose that there are two software development companies, which are labeled by Player $i = 1, 2$. Define the following notation:

T_{LC}: software life cycle
$N_i(t)$: expected cumulative number of software faults detected/corrected by time t in company i $(= 1, 2)$
$C_i(t)$: expected total software cost when company i $(= 1, 2)$ releases the software at time t
c_{1i}: cost of testing per unit time incurred in the testing phase for company i
c_{2i}: cost of removing a software fault in the testing phase for company i
c_{3i}: cost of removing a software fault in the operational phase for company i.

Without any loss of generality, we assume that $c_{3i} > c_{2i}$ for $i = 1, 2$. This means that the cost of removing a software fault in the operational phase is much more expensive than that in the testing. Under this plausible assumption, the expected total software cost for company i $(= 1, 2)$ is given by

$$C_i(t) = c_{1i}t + c_{2i}N_i(t) + c_{3i}\{N_i(T_{\mathrm{LC}}) - N_i(t)\}, \quad 0 \le t \le T_{\mathrm{LC}},$$

where $N_i(t)$ is differentiable with respect to t and is an increasing function with $N_i(0) = 0$. Suppose that the function $C_i(t)$ is a strictly convex function in t and that there exists a uniquely minimum value in $t = \gamma_i \in (0, T_{\mathrm{LC}})$, where

$$\gamma_i = \left\{ t \geq 0 \,\middle|\, \inf_{0<t<T_{\mathrm{LC}}} C_i(t) \right\}, \quad i = 1, 2$$

for company i. If $C_i(t)$ is a concave function, then it is optimal to continue the software test without release. In this sense, this assumption is trivial.

In the software release game, we assume that there exist only two rival players, Player 1 and Player 2, in the software product market. Each player wishes to maximize the expected total profit by his or her software development project and considers the optimal release timing. To simplify the analysis, we assume that each software product has the almost similar quality on functionalities. This implies that the life cycle of the software (T_{LC}) developed by each player is assumed to be the same. Also, we assume that the success of a monopoly of market by selling the software released to the market is probabilistic, so that the success probability for each player is given by $A_i(t)$ for an arbitrary release time t (> 0) by Player i $(= 1, 2)$. If Player i wins the software release game, then he or she can occupy the market, but the opponent, Player $3 - i$ $(i = 1, 2)$, loses it. Since it is common to consider that the release timing to maximize the success probability is different from the profit-effective timing by maximizing the expected total profit, we assume that the success probability $A_i(t)$ is also a strictly concave function with $A_i(0) = A_i(T_{\mathrm{LC}}) = 0$ and has a uniquely maximum value in $t = \eta_i \in (0, T_{\mathrm{LC}})$, where

$$\eta_i = \left\{ t \geq 0 \,\middle|\, \sup_{0<t<T_{\mathrm{LC}}} A_i(t) \right\}, \tag{1}$$

for Player i [35]. This assumption motivates us to release the software by taking account of the opposite player. It would be appropriate because the success probability in the market increases first and decreases just after the unimodal point is attained.

Let p_i be the net profit by selling the software products by Player i $(= 1, 2)$. Then, the expected total profit for Player i is given by

$$g_i(t) = p_i A_i(t) - C_i(t).$$

We call $g_i(t)$ *the expected total reward*. Note that the function $g_i(t)$ is a strictly concave function because $A_i(t)$ and $C_i(t)$ are strictly concave and convex in t, respectively, so that there exists a unique timing $\tau_i \in (0, T_{\mathrm{LC}})$ satisfying

$$\tau_i = \left\{ t \geq 0 \,\middle|\, \sup_{0<t<T_{\mathrm{LC}}} g_i(t) \right\}, \quad i = 1, 2. \tag{2}$$

Similarly, the expected total profit for Player i when the opponent player releases his or her software earlier and fails in the market is given by

$$\psi_i(t_i, t_{3-i}) = p_i A_i(t_i) \overline{A}_{3-i}(t_{3-i}) - C_i(t_i), \quad i = 1, 2,$$

where $\overline{A}_i(t) = 1 - A_i(t)$.

Let (x, y) denote the pure (non-probabilistic) strategies expressing the release times of software products for Player 1 and Player 2, respectively. Also, let $M_i(x, y)$ be the expected total profit for Player i when Player 1 and Player 2 release at time x and y, respectively. Zeephongsekul and Chiera [35] formulate the expected total profit as follows:

$$M_1(x,y) = \begin{cases} g_1(x) = p_1 A_1(x) - C_1(x), & x < y, \\ \psi_1(x,y) = p_1 \overline{A}_2(y) A_1(x) - C_1(x), & x \geq y, \end{cases} \tag{3}$$

$$M_2(x,y) = \begin{cases} g_2(y) = p_2 A_2(y) - C_2(y), & y < x, \\ \psi_2(y,x) = p_2 \overline{A}_1(x) A_2(y) - C_2(y), & y \geq x. \end{cases} \tag{4}$$

We call the game given in Eqs. (3) and (4) *silent-type game*. In the silent-type game, we assume that each player cannot know whether opponent's action has already been taken or not.

Here, let $(X,Y) \in [0,T_{\mathrm{LC}}] \times [0,T_{\mathrm{LC}}]$ be the sets of pure strategies for each Player i $(= 1,2)$. Define the mixed (probabilistic) strategy for each player:

$$F_1 = F_1(x) = \Pr\{X \leq x\} \in [0,1],$$
$$F_2 = F_2(y) = \Pr\{Y \leq y\} \in [0,1].$$

This implies that Player i $(= 1,2)$ triggers the software release at the random timing with the probability distribution function F_i. If Player i takes the mixed strategy F_i, then the expected total profit for Player i is given by

$$M_i(F_1,F_2) = \int_X \int_Y M_i(x,y)\, dF_1\, dF_2,$$

where

$$\begin{aligned} M_1(x,F_2) &= \int_Y M_1(x,y)\, dF_2, \\ M_2(F_1,y) &= \int_X M_2(x,y)\, dF_1. \end{aligned} \tag{5}$$

The sets of mixed strategies (F_1^*, F_2^*) are called the *mixed equilibrium strategies* or *Nash equilibrium strategies* if they satisfy

$$M_1(F_1^*, F_2^*) \geq M_1(F_1, F_2^*), \tag{6}$$

$$M_2(F_1^*, F_2^*) \geq M_2(F_1^*, F_2). \tag{7}$$

The expected total profit $M_i(F_1^*, F_2^*)$ is called the *value function*. Hence, the challenge here is to find the Nash equilibrium strategies and their associated value functions.

4. TWO COUNTEREXAMPLES

4.1. Counterexample in [35]

For the game in Eqs. (3) and (4), Zeephongsekul and Chiera [35] derived the following mixed strategy:

$$F_1^*(x) = \begin{cases} 0, & 0 \le x < a_1, \\ \displaystyle\int_{a_1}^{x} q_1(t)\,dt, & a_1 \le x \le \mu_1, \\ 1, & \mu_1 < x \le T_{\mathrm{LC}}, \end{cases} \tag{8}$$

$$F_2^*(x) = \begin{cases} 0, & 0 \le y < a_2, \\ \displaystyle\int_{a_2}^{y} q_2(t)dt, & a_2 \le y \le \mu_2, \\ 1, & \mu_2 < y \le T_{\mathrm{LC}}, \end{cases} \tag{9}$$

where

$$\mu_i = \sup\{t > 0 \mid q_i(t) > 0\},$$

a_i $(i = 1, 2)$ is the unique root of the equation:

$$\int_{a_i}^{\mu_i} q_i(t)\,dt = 1,$$

and

$$q_i(t) = \frac{A'_{3-i}(t)\{C_{3-i}(t) + g_{3-i}(a_i)\} - C'_{3-i}(t)A_{3-i}(t)]}{p_{3-i}\{A_{3-i}(t)\}^2 A_i(t)}.$$

Dohi *et al.* [8] gave a counterexample for the above strategy and showed that it holds in an only limited case. If $a_1 \ne a_2$ and/or $\mu_1 \ne \mu_2$, then it is obvious that the mixed strategies in Eqs. (8) and (9) are not optimal. From Eqs. (3) and (4), we have

$$M_1(x, F_2^*) = \begin{cases} p_1 A_1(x) - C_1(x), & 0 \le x < a_2, \\ g_1(a_2), & a_2 \le x \le \mu_2, \\ A_1(x)\dfrac{C_1(\mu_2) + g_1(a_2)}{A_1(\mu_2)} - C_1(x), & \mu_2 < x \le T_{\mathrm{LC}}, \end{cases}$$

$$M_2(F_1^*, y) = \begin{cases} p_2 A_2(y) - C_2(y), & 0 \le y < a_1, \\ g_2(a_1), & a_1 \le y \le \mu_1, \\ A_2(y)\dfrac{C_2(\mu_1) + g_2(a_1)}{A_2(\mu_1)} - C_2(y), & \mu_1 < y \le T_{\mathrm{LC}}. \end{cases}$$

That is,

$$M_1(x, F_2^*) \begin{cases} < g_1(a_2), & 0 \le x < a_2, \\ = g_1(a_2), & a_2 \le x \le \mu_2, \\ < g_1(a_2), & \mu_2 < x \le T_{\mathrm{LC}}, \end{cases}$$

$$M_2(F_1^*, y) \begin{cases} < g_2(a_1), & 0 \le y < a_1, \\ = g_2(a_1), & a_1 \le y \le \mu_1, \\ < g_2(a_1), & \mu_1 < y \le T_{\mathrm{LC}}. \end{cases}$$

If $a_1 < a_2$ and $\mu_1 > \mu_2$, then

$$
\begin{aligned}
M_1(F_1^*, F_2^*) &= \int_{a_1}^{a_2} M_1(x, F_2^*)\, dF_1^* + \int_{a_2}^{\mu_1} M_1(x, F_2^*)\, dF_1^* \\
&< \int_{a_1}^{a_2} g_1(a_2)\, dF_1^* + \int_{a_2}^{\mu_1} g_1(a_2)\, dF_1^*, \\
M_1(F_1^*, F_2^*) &= \int_{a_1}^{\mu_2} M_1(x, F_2^*)\, dF_1^* + \int_{\mu_2}^{\mu_1} M_1(x, F_2^*)\, dF_1^* \\
&< \int_{a_1}^{\mu_2} g_1(a_2)\, dF_1^* + \int_{\mu_2}^{\mu_1} g_1(a_2)\, dF_1^*,
\end{aligned}
$$

and we find that

$$M_1(F_1^*, F_2^*) < g_1(a_2).$$

On the other hand, if $a_2 < a_1$ and $\mu_2 > \mu_1$, then

$$M_2(F_1^*, F_2^*) < g_2(a_1).$$

These facts imply that both players always have to set $a_1 = a_2$ and $\mu_1 = \mu_2$, and that the mixed strategies in Eqs. (8) and (9) are not optimal if $a_1 \neq a_2$ and/or $\mu_1 \neq \mu_2$.

4.2. Counterexample in [8]

Define

$$
\begin{aligned}
\eta &= \min(\eta_1, \eta_2), \\
\dot{a} &= \max(\dot{a}_1, \dot{a}_2),
\end{aligned}
$$

where η_i $(i = 1, 2)$ is already defined in Eq. (1) and $\dot{a}_i$ $(i = 1, 2)$ is the unique root a of the equation:

$$\int_a^\eta f_i(t; a)\, dt = 1, \tag{10}$$

where

$$f_i(t; a) = \frac{A'_{3-i}(t)\{C_{3-i}(t) + g_{3-i}(a)\} - C'_{3-i}(t)A_{3-i}(t)}{p_{3-i}\{A_{3-i}(t)\}^2 A_i(t)}.$$

Dohi *et al.* [8] proposed the following mixed strategy in the silent-type game:

$$F_1^*(x) = \begin{cases} 0, & 0 \le x < \dot{a}, \\ \displaystyle\int_{\dot{a}}^{x} f_1(t; \dot{a})\, dt + \alpha_1^{[\dot{a}, \eta]} I_\eta(x), & \dot{a} \le x \le \eta, \\ 1, & \eta < x < T_{\mathrm{LC}}, \end{cases} \tag{11}$$

$$F_2^*(y) = \begin{cases} 0, & 0 \le y < \dot{a}, \\ \displaystyle\int_{\dot{a}}^{y} f_2(t; \dot{a})\, dt + \alpha_2^{[\dot{a}, \eta]} I_\eta(y), & \dot{a} \le y \le \eta, \\ 1, & \eta < y < T_{\mathrm{LC}}, \end{cases} \tag{12}$$

where

$$I_j(z) = \begin{cases} 1, & z = j, \\ 0, & \text{otherwise}, \end{cases} \tag{13}$$

$$\alpha_i^{[\dot{a},\eta]} = 1 - \int_{\dot{a}}^{\eta} f_i(t;\dot{a})\,dt, \tag{14}$$

and the parameters $\alpha_i^{[\dot{a},\eta]}$ $(i = 1, 2)$ represent the mass parts of Player i's mixed strategy. In their solution, the game value satisfying Eqs. (6) and (7) is given by

$$M_i(F_1^*, F_2^*) = g_i(\dot{a}).$$

We show that the mixed strategies in Eqs. (11) and (12) do not satisfy the Nash inequalities in Eqs. (6) and (7). For Player 1, we consider the case of $\eta < x < T_{\mathrm{LC}}$. From Eq. (12), the probability density function (p.d.f.) for Player 2's mixed strategy is given by

$$f_2^*(y) = \begin{cases} 0, & 0 \le y < \dot{a}, \\ f_2(y;\dot{a}), & \dot{a} \le y < \eta, \\ \alpha_2^{[\dot{a},\eta]}, & y = \eta, \\ 0, & \eta < y < T_{\mathrm{LC}}. \end{cases} \tag{15}$$

Using Eqs. (3) and (15), we obtain

$$\begin{aligned} M_1(x, F_2^*) &= \int_{\dot{a}}^{\eta} \{p_1 \overline{A}_2(y) A_1(x) - C_1(x)\}\, dF_2^* \\ &= p_1 A_1(x) - C_1(x) - p_1 A_1(x) \left\{ \int_{\dot{a}}^{\eta - 0} A_2(y) f_2(y;\dot{a})\,dy + A_2(\eta)\alpha_2^{[\dot{a},\eta]} \right\} \\ &= \frac{C_1(\eta) + g_1(\dot{a})}{A_1(\eta)} A_1(x) - C_1(x) - p_1 A_1(x) A_2(\eta) \alpha_2^{[\dot{a},\eta]}. \end{aligned} \tag{16}$$

Define $G_2(x) = \{C_1(x) + g_1(\dot{a})\}/A_1(x)$. From Eq. (16), we have

$$\frac{g_1(\dot{a}) - M_1(x, F_2^*)}{A_1(x)} = G_2(x) - G_2(\eta) + p_1 A_2(\eta)\alpha_2^{[\dot{a},\eta]}. \tag{17}$$

By differentiating $G_2(x)$ with respect to x, we find

$$\begin{aligned} G_2'(x) &= \frac{-\{A_1'(x)\{C_1(x) + g_1(\dot{a})\} - C_1'(x) A_1(x)\}}{\{A_1(x)\}^2} \\ &= -p_1 A_2(x) f_2(x;\dot{a}). \end{aligned} \tag{18}$$

From Eq. (18), Eq. (17) can be rewritten as follows:

$$\frac{g_1(\dot{a}) - M_1(x, F_2^*)}{p_1 A_1(x)} = -\int_{\eta}^{x} A_2(t) f_2(t;\dot{a})\,dt + A_2(\eta)\alpha_2^{[\dot{a},\eta]}. \tag{19}$$

If $\dot{a} = \dot{a}_2$, then it holds $\alpha_2^{[\dot{a},\eta]} = 0$ from Eqs. (10) and (14). Also, since $f_i(t;\dot{a})$ is the p.d.f., it must hold that $f_i(\eta;\dot{a}) \ge 0$. Hence, in this situation, Eq. (19) is possible to take a negative value. From the above case, for an arbitrary x, it holds that

$$M_1(x, F_2^*) \ge g_1(\dot{a}).$$

The case of Player 2 is similar. Hence, it is shown that the mixed strategy F_i^* in Eqs. (11) and (12) is not always the Nash equilibrium strategy.

5. SILENT-TYPE GAME

We reconsider the solutions of the silent-type game. In Eqs. (5) and (6), the first-order conditions of optimality are given by

$$\frac{\partial}{\partial x} M_1(x, F_2) = 0, \tag{20}$$

$$\frac{\partial}{\partial y} M_2(F_1, y) = 0. \tag{21}$$

We suppose that there exists the first derivative of the function $F_i(t)$ satisfying Eqs. (20) and (21), that is,

$$h_i(t) = \frac{dF_i(t)}{dt}, \quad i = 1, 2.$$

For the function $h_i(t)$, define the parameter $\tilde{\mu}_i$ satisfying

$$\tilde{\mu}_i = \sup\{t > \tilde{a} | h_i(t) > 0\}, \quad i = 1, 2,$$

where

$$\tilde{a} = \max(\tilde{a}_1, \tilde{a}_2), \tag{22}$$

and $\tilde{a}_i$ $(i = 1, 2)$ is the unique root a of the equation:

$$\int_a^\mu h_i(t)\, dt = 1,$$

where

$$\mu = \min(\tilde{\mu}_1, \tilde{\mu}_2). \tag{23}$$

Hence, the function $h_i(t)$ is regarded as a p.d.f.

In the silent-type game, we define the p.d.f. of the Nash equilibrium strategy:

$$f_i^*(t) = \begin{cases} 0, & 0 \le t < \tilde{a}, \\ h_i(t), & \tilde{a} \le t < \mu, \\ \beta_i^{[\tilde{a},\mu]}, & t = \mu, \\ 0, & \mu < t \le T_{\mathrm{LC}}, \end{cases} \tag{24}$$

where

$$\beta_i^{[\tilde{a},\mu]} = 1 - \int_{\tilde{a}}^{\mu} h_i(t)\, dt, \tag{25}$$

and

$$h_i(t) = \frac{A'_{3-i}(t)\{C_{3-i}(t) + g_{3-i}(\tilde{a})\} - C'_{3-i}(t) A_{3-i}(t)}{p_{3-i}\{A_{3-i}(t)\}^2 A_i(t)} \tag{26}$$

is the function that satisfies the first-order conditions of optimality in Eqs. (20) and (21). In the silent-type game, we assume that only one Player i takes the mixed strategy which has the mass part $\beta_i^{[\tilde{a},\mu]}$, that is, $\beta_1^{[\tilde{a},\mu]}\beta_2^{[\tilde{a},\mu]} = 0$.

In the silent-type software release game, we make the following three assumptions:

ASSUMPTION 1: *If $\mu = \tilde{\mu}_{3-i}$, then*

$$A_i(\mu)\beta_i^{[\tilde{a},\mu]} \geq \int_{\mu}^{\tilde{\mu}_i} A_i(t)h_i(t)\,dt, \quad i = 1, 2.$$

ASSUMPTION 2: *For arbitrary t (≥ 0),*

$$C_i(t) + g_i(\tilde{a}) \geq 0, \quad i = 1, 2.$$

ASSUMPTION 3: *For $i = 1, 2$,*

$$\tilde{a} \leq \tau_i,$$

where τ_i is already defined in Eq. (2).

It should be noted that $\beta_i^{[\tilde{a},\mu]}$ $(i = 1, 2)$ are the probabilities that Player i releases the software product at time μ. In Assumption 1, we implicitly assume that the product of $\beta_i^{[\tilde{a},\mu]}$ and the success probability at time μ, $A_i(\mu)$, is greater than the accumulation of $h_i(t)A_i(t)$ in the range $(\mu, \tilde{\mu}_i)$. In Assumption 2, the function $g_i(\tilde{a})$ represents the expected total profit when Player i releases the software product as early as possible. Since the expected total software cost $C_i(t)$ is strictly positive, this is plausible and can be validated intuitively.

If we assume $\tilde{a} > \tau_i$, then each player selects not to release at the best timing τ_i which maximizes the expected total profit in the situation where the opponent player does not release the software yet. This action is not rational, so it can be seen that Assumption 3 is also plausible. This nontrivial case results that no Nash equilibrium strategy may be found. Unfortunately, it is not easy to see that what combination of $p_i, A_i(t)$ and $C_i(t)$ leads to the condition of $\tilde{a} \leq \tau_i$. However, as shown in the numerical examples, the condition can be checked numerically.

LEMMA 1: *For $i = 1, 2$, $h_i(\tilde{a}) \geq 0$.*

PROOF: For Player 1, since the denominator of $h_1(t)$ in Eq. (26) is always positive, it is enough to show the fact that the numerator is also positive when $\tilde{a} \leq \tau_i$. Substituting $t = \tilde{a}$ into the numerator of $h_1(t)$ yields

$$A_2'(\tilde{a})\{C_2(\tilde{a}) + g_2(\tilde{a})\} - C_2'(\tilde{a})A_2(\tilde{a}) = A_2(\tilde{a})g_2'(\tilde{a}) \geq 0.$$

Since the case of Player 2 is similar, under Assumption 3, $h_i(\tilde{a}) \geq 0$. ∎

LEMMA 2: *The function satisfying Eqs. (20) and (21) is given by*

$$f_i^*(t) = \begin{cases} 0, & 0 \leq t < \tilde{a}, \\ h_i(t), & \tilde{a} \leq t < \mu, \\ 0, & \mu \leq t \leq T_{LC}. \end{cases}$$

PROOF: When Player 1 releases at $\tilde{a} \leq x \leq \mu$, we obtain

$$\begin{aligned} M_1(x, F_2^*) &= \int_{\tilde{a}}^{\mu} M_1(x,y)\, dF_2^* \\ &= p_1 A_1(x) - C_1(x) - \int_{\tilde{a}}^{x} \{p_1 A_1(x) A_2(y)\}\, dF_2^*. \end{aligned} \qquad \textbf{(27)}$$

By differentiating Eq. (27) with respect to x and setting it equal to zero, we have

$$\frac{1}{p_1} \cdot \frac{\partial C_1(x)}{\partial x} = \frac{\partial}{\partial x}\left[A_1(x)\left\{1 - \int_{\tilde{a}}^{x} A_2(y)\, dF_2^*\right\}\right]. \qquad \textbf{(28)}$$

Integrating both sides of Eq. (28) yields

$$A_1(x)\left\{1 - \int_{\tilde{a}}^{x} A_2(y)\, dF_2^*\right\} = \frac{C_1(x) + D}{p_1}, \qquad \textbf{(29)}$$

where D is the constant of integration. By substituting $x = \tilde{a}$ into Eq. (29), we have $D = g_1(\tilde{a})$ and

$$h_2(t) = \frac{A_1'(t)\{C_1(t) + g_1(\tilde{a})\} - C_1'(t) A_1(t)}{p_1 \{A_1(t)\}^2 A_2(t)}, \quad \tilde{a} \leq t < \mu.$$

The proof for Player 2 is made in the similar way. ■

THEOREM 1: *The Nash equilibrium strategies for the silent-type software release game are given by*

$$F_1^*(x) = \begin{cases} 0, & 0 \leq x < \tilde{a}, \\ \displaystyle\int_{\tilde{a}}^{x} h_1(t)\, dt + \beta_1^{[\tilde{a},\mu]} I_\mu(x), & \tilde{a} \leq x \leq \mu, \\ 1, & \mu < x \leq T_{LC}, \end{cases} \qquad \textbf{(30)}$$

$$F_2^*(y) = \begin{cases} 0, & 0 \leq y < \tilde{a}, \\ \displaystyle\int_{\tilde{a}}^{y} h_2(t)\, dt + \beta_2^{[\tilde{a},\mu]} I_\mu(y), & \tilde{a} \leq y \leq \mu, \\ 1, & \mu < y \leq T_{LC}, \end{cases} \qquad \textbf{(31)}$$

where $I_\mu(t), \tilde{a}, \mu, \beta_i^{[\tilde{a},\mu]}, h_i(t)$ *are given in Eqs. (13), (22), (23), (25), and (26), respectively.*

THEOREM 2: *The value functions of the silent-type software release game are given by*

$$M_i(F_1^*, F_2^*) = g_i(\tilde{a}), \quad i = 1, 2,$$

so that the Nash equilibrium strategies are given by $F_i^*(t)$ *in Eqs. (30) and (31).*

Proof: For Player 1, we consider four cases: (i) $x < \tilde{a}$, (ii) $\tilde{a} \le x < \mu$, (iii) $x = \mu$, and (iv) $\mu < x \le T_{\mathrm{LC}}$. In Case (i), from Eq. (5) and Assumption 3, we have

$$M_1(x, F_2^*) = p_1 A_1(x) - C_1(x) = g_1(x) < g_1(\tilde{a}).$$

In Case (ii), the function $M_1(x, F_2^*)$ is constant for all x since F_2^* satisfies the first-order condition of optimality. Substituting $x = \tilde{a}$ into Eq. (27) leads to

$$M_1(\tilde{a}, F_2^*) = p_1 A_1(\tilde{a}) - C_1(\tilde{a}) = g_1(\tilde{a}).$$

In Case (iii), using Eqs. (3) and (24), we obtain $M_1(x, F_2^*)$ as follows:

$$\begin{aligned} M_1(x, F_2^*) &= \int_{\tilde{a}}^{\mu} \{p_1 \overline{A}_2(y) A_1(x) - C_1(x)\}\, dF_2^* \\ &= p_1 A_1(x) - C_1(x) - p_1 A_1(x) \left\{ \int_{\tilde{a}}^{\mu - 0} A_2(y) h_2(y)\, dy \right. \\ &\quad \left. + A_2(\mu) \beta_2^{[\tilde{a},\mu]} \right\} \\ &= \frac{c_1(\mu) + g_1(\tilde{a})}{A_1(\mu)} A_1(x) - C_1(x) - p_1 A_1(x) A_2(\mu) \beta_2^{[\tilde{a},\mu]}. \end{aligned} \tag{32}$$

From Eq. (32), it can be seen that

$$M_1(\mu, F_2^*) = g_1(\tilde{a}) - p_1 A_1(\mu) A_2(\mu) \beta_2^{[\tilde{a},\mu]} \le g_1(\tilde{a}).$$

In Case (iv), we have

$$\begin{aligned} M_1(x, F_2^*) &= \int_{\tilde{a}}^{\mu} \{p_1 \overline{A}_2(y) A_1(x) - C_1(x)\}\, dF_2^* \\ &= \frac{c_1(\mu) + g_1(\tilde{a})}{A_1(\mu)} A_1(x) - C_1(x) - p_1 A_1(x) A_2(\mu) \beta_2^{[\tilde{a},\mu]}. \end{aligned} \tag{33}$$

For $G_2(x) = \{C_1(x) + g_1(\tilde{a})\}/A_1(x)$, from Eq. (33), we obtain

$$\frac{g_1(\tilde{a}) - M_1(x, F_2^*)}{A_1(x)} = G_2(x) - G_2(\mu) + p_1 A_2(\mu) \beta_2^{[\tilde{a},\mu]}. \tag{34}$$

By differentiating $G_2(x)$ with respect to x, we find that

$$\begin{aligned} G_2'(x) &= \frac{-\{A_1'(x)\{C_1(x) + g_1(\tilde{a})\} - C_1'(x) A_1(x)\}}{\{A_1(x)\}^2} \\ &= -p_1 A_2(x) h_2(x). \end{aligned} \tag{35}$$

If $\mu = \tilde{\mu}_2$, then $\beta_2^{[\tilde{a},\mu]} = 0$ and $h_2(x) < 0$. From Eqs. (34) and (35), it is seen that $M_1(x, F_2^*) \le g_1(\tilde{a})$. On the other hand, if $\mu = \tilde{\mu}_1$, then $\beta_2^{[\tilde{a},\mu]} > 0$. Based on Assumption 1, we have

$$G_2(x) - G_2(\mu) + p_1 A_2(\mu) \beta_2^{[\tilde{a},\mu]} \ge 0. \tag{36}$$

From Eqs. (34) and (36), it is seen that $M_1(x, F_2^*) \le g_1(\tilde{a})$.

From the above cases (i)–(iv), it is shown for all x that

$$M_1(x, F_2^*) \leq g_1(\tilde{a}).$$

In this game, $\beta_1^{[\tilde{a},\mu]}\beta_2^{[\tilde{a},\mu]} = 0$ always holds by definition of $\tilde{a}$ and $\beta_i^{[\tilde{a},\mu]}$. Hence, the value function is given by

$$\begin{aligned} M_1(F_1^*, F_2^*) &= \int_{\tilde{a}}^{\mu} M_1(x, F_2^*)\, dF_1^* \\ &= g_1(\tilde{a}) - p_1 A_1(\mu) A_2(\mu) \beta_1^{[\tilde{a},\mu]}\beta_2^{[\tilde{a},\mu]} = g_1(\tilde{a}). \end{aligned}$$

The proof for Player 2 is made in the similar way. ■

6. NOISY-TYPE GAME

Next, we consider the case where each player can know whether the opponent player has already released his or her software or not. This type of game is called noisy-type game. Each player can postpone his or her release time until the best timing if the opponent releases earlier and fails in the market, where the best timing means the time when the expected total reward $g_i(t)$ $(i = 1, 2)$ is maximized. For the noisy-type game, let $(x, r_1(y))$ denotes the pure strategy for Player 1, such that Player 1 selects time x and then acts at best timing $r_1(y)$ if Player 2 has acted at time y before time x, or acts at time x if Player 2 has not acted before time x. Similarly, let $(y, r_2(x))$ be the pure strategy for Player 2. Also, let (F_i, r_i) be a mixed strategy such that Player i releases at best timing r_i if Player $3 - i$ has already released and failed under the condition that Player i has not acted yet. Nash equilibrium strategies and value functions can be defined in a similar way to the silent-type game. We define

$$\begin{aligned} &M_1((x, r_1(y)), (y, r_2(x))) \\ &\quad = \begin{cases} p_1 A_1(x) - C_1(x), & x < y, \\ \psi_1(r_1(y), y) = p_1 A_1(r_1(y))\overline{A}_2(y) - C_1(r_1(y)), & x \geq y, \end{cases} \\ &M_2((x, r_1(y)), (y, r_2(x))) \\ &\quad = \begin{cases} p_2 A_2(y) - C_2(y), & y < x, \\ \psi_2(r_2(x), x) = p_2 A_2(r_2(x))\overline{A}_1(x) - C_2(r_2(x)), & y \geq x, \end{cases} \end{aligned} \tag{37}$$

where $M_i((x, r_1(y)), (y, r_2(x)))$ is the expected total profit for Player i and $r_i(t)$ is the best reaction timing for Player i when Player $3 - i$ releases at time t first.

Define

$$\tau = \min(\tau_1, \tau_2).$$

For an arbitrary time $t \in (b_i, \tau]$, let

$$\theta_i(t) = \frac{g_i'(t)}{g_i(t) - \psi_i(\tau_i, t)},$$

where the parameter b_i is the minimum root of the following equation:

$$g_i(b_i) = \psi_i(\tau_i, b_i), \quad i = 1, 2.$$

In the noisy-type software release game, we make the following assumption:

ASSUMPTION 4: *If $\tau = \tau_{3-i}$, then*

$$g_i(\tau_{3-i}) \geq \psi_i(\tau_i, \tau_{3-i}), \quad i = 1, 2.$$

The function $g_i(\tau_{3-i})$ represents the expected total profit for Player i in the situation when he or she releases the software at time τ_{3-i} before the opponent's release. Let $\psi_i(\tau_i, \tau_{3-i})$ denotes the expected total profit for Player i when he or she releases the software at time τ_i after opponent's release at time τ_{3-i}. If $g_i(\tau_{3-i}) < \psi_i(\tau_i, \tau_{3-i})$ holds, then Player $3 - i$ wishes to release at time τ_{3-i} and Player i selects to release software at time τ_i after opponent's release. In this case, the existence of opponent player does not influence the strategies of both players. Hence, we do not consider this trivial case.

LEMMA 3: *The cumulative distribution function satisfying Eqs. (20) and (21) is given by*

$$F_i^*(t) = \begin{cases} 0, & 0 \leq t < b, \\ 1 - \exp\{-\int_b^x \theta_{3-i}(t) dt\}, & b \leq x < c, \\ 1, & c \leq x \leq T_{LC}, \end{cases}$$

where $b = \max(b_1, b_2)$ and c are the arbitrary real numbers satisfying $b < c \leq \tau$.

PROOF: When Player 1 releases at $b \leq x \leq c$, we obtain

$$M_1((x, \tau_1), (F_2^*, \tau_2)) = g_1(\tau_1) \int_b^x dF_2^* - p_1 A_1(\tau_1) \int_b^x A_2(y)\, dF_2^* + g_1(x)[1 - F_2^*(x)]. \tag{38}$$

From the first-order condition of optimality, we have

$$\frac{f_2(x)}{F_2^*(x) - 1} = -\frac{g_1'(x)}{g_1(x) - \psi_1(\tau_1, x)} = -\theta_1(x). \tag{39}$$

Integrating both sides of Eq. (39) yields

$$F_2^*(t) = 1 + D \exp\left\{-\int_b^x \theta_1(t)\, dt\right\}, \quad b \leq t < c, \tag{40}$$

where D is the constant of integration. By substituting $x = b$ into Eq. (40), we find $D = -1$. The proof for Player 2 is made in the similar way. ■

THEOREM 3: *Let $b = b_2$. The Nash equilibrium strategies for the noisy-type software release game are given by $((F_1^*, \tau_1), (F_2^*, \tau_2))$, where*

$$F_1^*(x) = \begin{cases} 0, & 0 \leq x < b, \\ 1, & b \leq x \leq T_{LC}, \end{cases}$$

$$F_2^*(y) = \begin{cases} 0, & 0 \leq y < b, \\ 1 - \exp\left\{-\int_b^y \theta_1(t)\, dt\right\} + \gamma_2^{[b,c]} I_c(y), & b \leq y \leq c, \\ 1, & c < y \leq T_{LC}, \end{cases}$$

$c \leq \tau$ and the parameters $\gamma_i^{[j_1,j_2]}$ $(i = 1, 2)$ are given by

$$\gamma_i^{[j_1,j_2]} = \exp\left\{-\int_{j_1}^{j_2} \theta_{3-i}(t)\, dt\right\}.$$

The value functions of the noisy-type software release game are given by

$$M_i((F_1^*,\tau_1),(F_2^*,\tau_2)) = g_i(b), \quad i=1,2.$$

Proof: For Player 1, we consider four cases: (i) $x<b$, (ii) $b \le x < c$, (iii) $x=c$, and (iv) $c < x \le T_{\mathrm{LC}}$. In Case (i), from Eq. (5) and $b<\tau$, we have

$$M_1((x,\tau_1),(F_2^*,\tau_2)) = g_1(x) < g_1(b).$$

In Case (ii), the function $M_1((x,\tau_1),(F_2^*,\tau_2))$ is constant for all x, since F_2^* satisfies the first-order condition of optimality. Substituting $x=b$ into Eq. (38) leads to

$$M_1((b,\tau_1),(F_2^*,\tau_2)) = g_1(b)[1-F_2^*(b)] = g_1(b).$$

In Case (iii), from Eq. (37), we obtain $M_1((x,\tau_1),(F_2^*,\tau_2))$ as follows:

$$\begin{aligned} M_1((c,\tau_1),(F_2^*,\tau_2)) &= \int_b^{c-0} \{p_1 A_1(\tau_1)\overline{A}_2(y) - C_1(\tau_1)\}\, dF_2^* \\ &\quad + \{p_1 A_1(c)\overline{A}_2(c) - C_1(c)\}\gamma_2^{[b,c]} \\ &= g_1(b) - p_1 A_1(c) A_2(c) \gamma_2^{[b,c]} \le g_1(b). \end{aligned}$$

In Case (iv), under Assumption 4, we get $M_1((x,\tau_1),(F_2^*,\tau_2))$ as follows:

$$\begin{aligned} M_1((x,\tau_1),(F_2^*,\tau_2)) &= \int_b^{c-0} \{p_1 A_1(\tau_1)\overline{A}_2(y) - C_1(\tau_1)\}\, dF_2^* \\ &\quad + \{p_1 A_1(\tau_1)\overline{A}_2(c) - C_1(\tau_1)\}\gamma_2^{[b,c]} \\ &= g_1(b) - \{g_1(c) - \psi_1(\tau_1,c)\}\gamma_2^{[b,c]} \le g_1(b). \end{aligned}$$

From the above cases (i)–(iv), it is shown for all x that

$$M_1((x,\tau_1),(F_2^*,\tau_2)) \le g_1(b).$$

On the other hand, for Player 2, we consider two cases: (i) $y<b$ and (ii) $b \le y < T_{\mathrm{LC}}$. In Case (i) from Eq. (5) and $b<\tau$, we have

$$M_2((F_1^*,\tau_1),(y,\tau_2)) = g_2(y) < g_2(b).$$

In Case (ii), we obtain $M_2((F_1^*,\tau_1),(y,\tau_2))$ as follows:

$$M_2((F_1^*,\tau_1),(y,\tau_2)) = \psi_2(\tau_2,b) = g_2(b).$$

From the above cases (i) and (ii), it is shown for all y that

$$M_2((F_1^*,\tau_1),(y,\tau_2)) \le g_2(b).$$

Therefore, in the noisy-type software release game, the value functions are given by

$$\begin{aligned} M_1((F_1^*,\tau_1),(F_2^*,\tau_2)) &= \int_b^{\tau} M_1((x,\tau_1),(F_2^*,\tau_2))\, dF_1^* \\ &= g_1(b)\int_b^{\tau} dF_1^* = g_1(b), \\ M_2((F_1^*,\tau_1),(F_2^*,\tau_2)) &= g_2(b). \end{aligned}$$

The proof is completed. ∎

Theorem 4: *Let $b = b_2$ and $\tau = \tau_1$. The Nash equilibrium strategies for the noisy-type software release game are also given by $((F_1^*, r_1), (F_2^*, \tau_2))$, where*

$$F_1^*(x) = \begin{cases} 0, & 0 \le x < b, \\ 1, & b \le x \le T_{LC}, \end{cases}$$

$$r_1(y) = \begin{cases} \tau_1, & 0 \le y \le \tau, \\ y, & \tau < y \le T_{LC}, \end{cases}$$

$$F_2^*(y) = \begin{cases} 0, & 0 \le y < b, \\ 1 - \exp\left\{ -\int_b^y \theta_1(t)\,dt \right\}, & b \le y < \tau, \\ 1 - \exp\left\{ -\int_b^\tau \theta_1(t)\,dt \right\} + \gamma_2^{[b,\tau]} I_d(y), & \tau \le y \le d, \\ 1, & d < y \le T_{LC}, \end{cases}$$

and d is an arbitrary real number satisfying $\tau < d \le T_{LC}$. Then, the value functions of the noisy-type software release game are given by

$$M_i((F_1^*, r_1), (F_2^*, \tau_2)) = g_i(b), \quad i = 1, 2.$$

Proof: The expected total profit for Player 1 in the noisy-type game is given by

$$M_1((x, r_1), (F_2^*, \tau_2)) = \begin{cases} g_1(x), & 0 \le x < b, \\ g_1(b), & b \le x \le \tau, \\ g_1(b) - \{g_1(\tau) - g_1(x)\}\gamma_2^{[b,\tau]}, & \tau < x < d, \\ g_1(b) - \{g_1(\tau) - \psi_1(d, d)\}\gamma_2^{[b,\tau]}, & x = d, \\ g_1(b) - \{g_1(\tau) - \psi_1(x, d)\}\gamma_2^{[b,\tau]}, & d < x \le T_{\mathrm{LC}}. \end{cases}$$

On the other hand, the expected total profit for Player 2 is also given by

$$M_2((F_1^*, r_1), (y, \tau_2))) = \begin{cases} g_2(y), & 0 \le y < b, \\ \psi_2(b, b), & y = b, \\ \psi_2(\tau_2, b) = g_2(b), & b < y < d, \\ g_2(b), & d \le y \le T_{\mathrm{LC}}. \end{cases}$$

From the above equations, it is shown for all x and y that

$$M_1((x, r_1), (F_2^*, \tau_2)) \le g_1(b),$$
$$M_2((F_1^*, r_1), (y, \tau_2))) \le g_2(b).$$

The value functions of Player i are given by

$$M_i((F_1^*, r_1), (F_2^*, \tau_2)) = g_i(b), \quad i = 1, 2.$$

The proof is completed. ■

7. NUMERICAL ILLUSTRATIONS

In this section, we give simple numerical examples for both silent-type and noisy-type software release games. Table 1 presents the dependence of the net profit p_i on the value

TABLE 1. Value functions for the silent-type software release game

$p_1 = p_2$	$\tilde{a}_1$ $[\tilde{a}]$	$\tilde{a}_2$	$\tilde{\mu_1}$ $[\mu]$	$\tilde{\mu_2}$	$g_1(\tilde{a})$	$g_2(\tilde{a})$
10,000	65.691	65.250	421.452	421.754	2,584.81	2,497.04
12,000	67.826	67.491	441.908	442.106	3,192.08	3,105.25
14,000	69.496	69.230	457.468	457.606	3,801.65	3,715.47
16,000	70.834	70.616	469.697	469.799	4,412.76	4,327.06
18,000	71.928	71.744	479.560	479.638	5,024.95	4,939.61

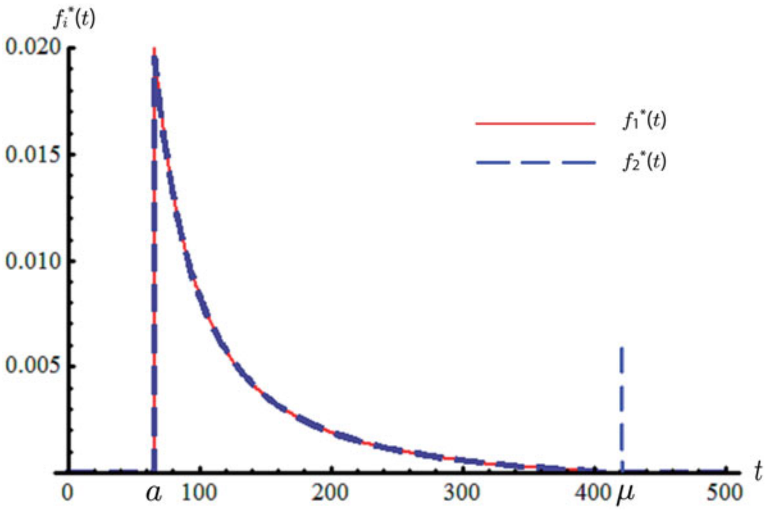

FIGURE 1. Comparison of silent strategies for both players.

function $g_i(\tilde{a})$ $(i = 1, 2)$ in the silent-type software release game, where

$$N_1(t) = 34(1 - e^{-0.06t}), \quad N_2(t) = 50(1 - e^{-0.06t}),$$
$$A_1(t) = A_2(t) = 0.055t^{0.4}(1 - 0.0005t),$$
$$c_{11} = c_{12} = 1, \quad c_{21} = c_{22} = 5,$$
$$c_{31} = c_{32} = 30, \quad T_{\mathrm{LC}} = 2000.$$

From Table 1, we can see that both value functions $g_i(\tilde{a})$ increase monotonously as the net profit $p_1 = p_2$ increases. Since the software of Player 1 includes a little expected initial number of faults, say 34, the value function for Player 1 is greater than that for Player 2. In Figure 1, the behavior of the Nash equilibrium strategies for the silent-type software release game in the case $p_1 = p_2 = 10{,}000$ is shown. From this figure, it is observed that the probability density functions for both players monotonously decrease between $\tilde{a}$ and μ, and that only the probability density function of Player 2 has the mass part at time μ.

Next, we compare the Nash equilibrium strategy of the silent-type software release game

TABLE 2. The supports and value functions of each strategy

	$\tilde{a}$	μ	$g_1(\tilde{a})$	$g_2(\tilde{a})$
Silent-type game	118.52	1,198.13	4,019.2	674.3
	b	τ	$g_1(b)$	$g_2(b)$
Noisy-type game	178.01	1,031.82	12,691 .4	7,826.5

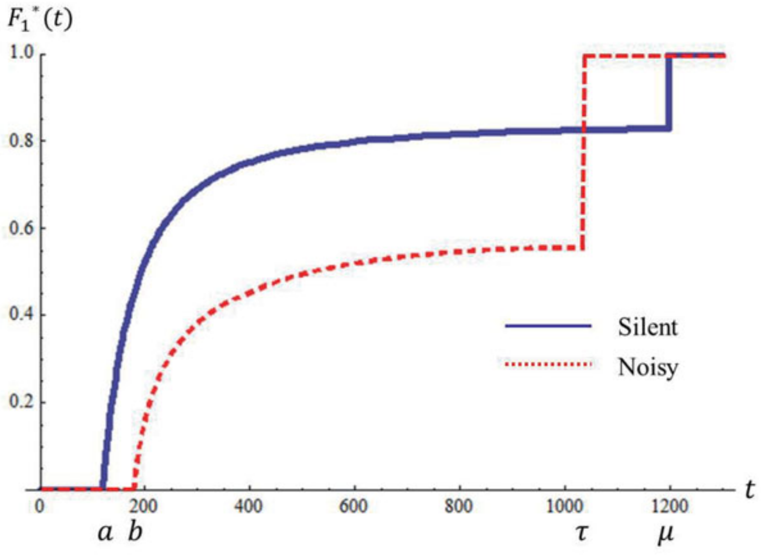

FIGURE 2. The behavior of the Nash equilibrium strategies of Player 1.

with that of the noisy-type software release game, where

$$
\begin{aligned}
&N_1(t) = N_2(t) = 0.256t,\\
&A_1(t) = 0.0055t(1 - 0.0005t),\\
&A_2(t) = 0.0045t(1 - 0.0005t)\\
&p_1 = p_2 = 30,000,\\
&c_{11} = c_{12} = 1, \quad c_{21} = c_{22} = 5,\\
&c_{31} = c_{32} = 30, \quad T_{\mathrm{LC}} = 2000.
\end{aligned}
$$

In Table 2, we calculate the supports and value functions for each strategy. Since each player can know an opponent action and release at advantageous timing in the noisy-type game, the value functions of the noisy-type game are greater than those of the silent-type game. In addition, the success probabilities of both players are influenced by the value functions. Figures 2 and 3 show the behavior of the Nash equilibrium strategies of Player 1 and 2. From these figures, it is observed that the support b of the Nash equilibrium strategy for the noisy-type software release game is shifted to the right compared with the support $\tilde{a}$ of the silent-type software release game, and that both players of the silent-type software release game take the mixed strategies but one player takes the mixed strategy and another player takes the pure strategy in the noisy-type software release game.

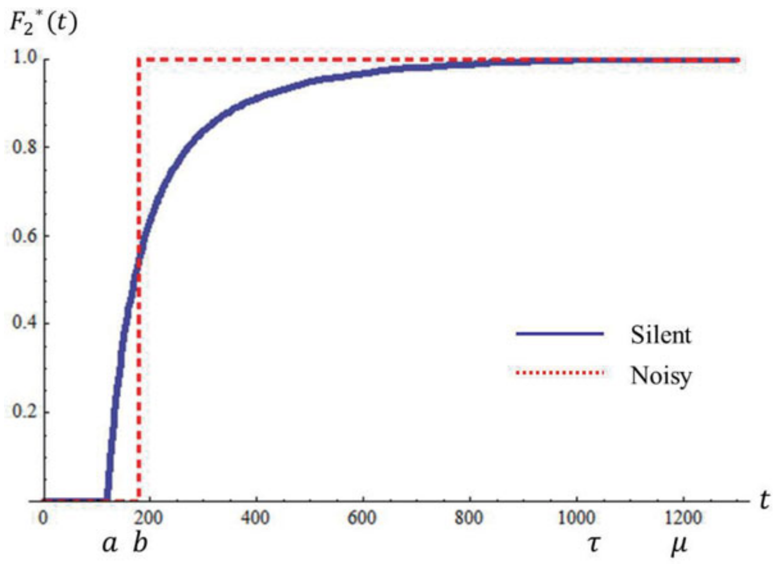

FIGURE 3. The behavior of the Nash equilibrium strategies of Player 2.

TABLE 3. Dependence of the mean release time $E[T_1] = E[T_2]$ on the expected cumulative number of software faults $N_1(t) = N_2(t)$

$N_1(t) = N_2(t)$	$\tilde{a}$	μ	$g_1(\tilde{a}) = g_2(\tilde{a})$	$E[T_1] = E[T_2]$	$\tau_1 = \tau_2$
0.25	97.185	1,039.654	72,834	186.450	1,031.818
0.75	96.944	1,048.348	55,731	161.791	1,031.818
1.25	96.358	1,062.456	38,629	138.504	1,031.818
1.75	96.432	1,090.987	21,527	119.472	1,031.818

Tables 3–6 present the dependence of some parameters on τ_i $(i = 1, 2)$ and $E[T_i]$ $(i = 1, 2)$, where $E[T_i]$ denotes the mean release time which is calculated using the Nash equilibrium strategy and τ_i indicates the best release timing for Player i $(= 1, 2)$ in the situation where there is only one player in the market. In these tables, we set model parameter $\tilde{a}_2(t) = 0.0055t(1 - 0.0005t)$, and other parameters are set as the same values in the case of Table 2. From Tables 3–6, we can see that both players tend to release earlier in the competitive market as the net profit $p_1 = p_2$, the expected cumulative number of software faults $N_1(t) = N_2(t)$ or the cost of removing/fixing a fault in the operational phase, $c_{31} = c_{32}$, increases. On the other hand, each player has to release later to take the best strategies as the cost of testing per unit time $c_{11} = c_{12}$ or the cost of removing/fixing a fault in the testing phase $c_{21} = c_{22}$ increases.

8. CONCLUSION

In this paper, we have revisited the original software release games and corrected the Nash equilibrium strategies in both silent and noisy type of software release games. Under some parametric conditions, the correct Nash equilibrium strategies have been given. Also, some Nash equilibrium strategies have been illustrated in numerical examples. Although we have considered two static software release games in this paper, they will be extended to the

TABLE **4.** Dependence of the mean release time $E[T_1] = E[T_2]$ on the cost of testing per unit time $c_{11} = c_{12}$

$c_{11} = c_{12}$	$\tilde{a}$	μ	$g_1(\tilde{a}) = g_2(\tilde{a})$	$E[T_1] = E[T_2]$	$\tau_1 = \tau_2$
1	97.185	1,267.462	72,834	186.450	1,031.818
3	97.173	1,215.417	70,770	189.068	1,031.818
5	97.153	1,182.621	68,706	191.943	1,031.818
7	97.133	1,159.440	66,643	194.948	1,031.818

TABLE **5.** Dependence of the mean release time $E[T_1] = E[T_2]$ on the cost of removing a fault in the testing phase $c_{21} = c_{22}$

$c_{21} = c_{22}$	$\tilde{a}$	μ	$g_1(\tilde{a}) = g_2(\tilde{a})$	$E[T_1] = E[T_2]$	$\tau_1 = \tau_2$
5	97.185	1,267.462	72,834	186.450	1,031.818
10	97.179	1,231.778	71,544	188.044	1,031.818
15	97.168	1,205.960	70,254	189.770	1,031.818
20	97.155	1,186.079	68,964	191.575	1,031.818

TABLE **6.** Dependence of the mean release time $E[T_1] = E[T_2]$ on the cost of removing a fault in the operational phase $c_{31} = c_{32}$

$c_{31} = c_{32}$	$\tilde{a}$	μ	$g_1(\tilde{a}) = g_2(\tilde{a})$	$E[T_1] = E[T_2]$	$\tau_1 = \tau_2$
30	97.185	1,267.462	72,834	186.450	1,031.818
90	96.786	910.914	58,311	156.366	1,031.818
150	96.034	738.640	43,788	130.803	1,031.818
210	96.598	896.173	29,265	107.460	1,031.818

dynamic version by means of Markov games. Also the software release games under consideration should be reformulated from the standpoint of the Bayesian decision-making in order to treat the Bayesian inference.

References

1. Baston, V.J. & Garnaev, A.Y. (1995). Teraoka-type two-person nonzero-sum silent duel. *Journal of Optimization Theory and Applications* 87: 539–552.
2. Boland, P.J. & Chuivi, N.N. (2007). Optimal times for software release when repair is imperfect. *Statistics & Probability Letters* 77(12): 1176–1184.
3. Chiu, K.C., Ho, J.W., & Huang, Y.S. (2009). Bayesian updating of optimal release time for software systems. *Software Quality Journal* 17: 90–120.
4. Dalal, S.R. & Mallows, C.L. (1988). When should one stop testing software? *Journal of the American Statistical Association* 83: 872–879.
5. Dalal, S.R. & Mallows, C.L. (1994). Some graphical aids for deciding when to stop testing software. *IEEE Journal on Selected Areas in Communications* 8(2): 169–175.
6. Dalal, S.R. & McIntosh, A.A. (1994). When to stop testing for large software systems with changing code. *IEEE Transactions on Software Engineering* 20(4): 318–323.
7. Dohi, T., Nishio, Y., & Osaki, S. (1999). Optimal software release scheduling based on artificial neural networks. *Annals of Software Engineering* 8: 167–185.
8. Dohi, T., Teraoka, Y., & Osaki, S. (2000). Software release games. *Journal of Optimization Theory and Applications* 105(2): 325–346.
9. Dohi, T., Yatsunami, Y., Nishio, Y., & Osaki, S. (2000). The effective smoothing techniques to estimate the optimal software release schedule based on artificial neural network. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences* 105(2): 325–346.

10. Forman, E.H. & Singpurwalla, N.D. (1977). An empirical stopping rule for debugging and testing computer software. *Journal of the American Statistical Association* 72: 750–757.
11. Forman, E.H. & Singpurwalla, N.D. (1979). Optimal time intervals for testing hypotheses on computer software errors. *IEEE Transactions on Reliability* R-28: 250–253.
12. Ho, J.W., Fang, C.C., & Huang, Y.S. (2008). The determination of optimal software release times at different confidence levels with consideration of learning effects. *Software Testing, Verification and Reliability* 18(4): 221–249.
13. Hou, R.H., Kuo, S.-Y., & Chang, Y.-P. (1997). Optimal release times for software systems with scheduled delivery time based on the HGDM. *IEEE Transactions on Computers* 46(2): 216–221.
14. Karlin, S (1959). *Mathematical methods and theory in games, programming, and economics*, vol. 2. Reading: Addison-Wesley.
15. Koch, H.S. & Kubat, P. (1983). Optimal release time for computer software. *IEEE Transactions on Software Engineering* SE-9(3): 323–327.
16. Li, X., Li, Y.F., Xie, M., & Ng, S.H. (2011). Reliability analysis and optimal version-updating for open source software. *Information and Software Technology* 53: 929–936.
17. Liu, C.-T. & Chang, Y.-C. (2007). Optimal planning for open source software updates. *Probability in the Engineering and Informational Sciences* 21: 301–314.
18. Luo, C., Okamura, H., & Dohi, T. (2016). Reliability analysis and optimal version-updating for open source software. *Journal of Risk and Reliability* 230(1): 44–53.
19. Momotaz, B. & Dohi, T. (2018). Optimal stopping time of software system test via artificial neural network with fault count data. *Journal of Quality in Maintenance Engineering* 24(1): 22–36.
20. Momotaz, B. & Dohi, T. (2018). Optimal release time estimation of software system using Box-Cox transformation and neural network. *International Journal of Mathematical, Engineering and Management Sciences* 3(2): 177–194.
21. Okumoto, K. & Goel, A.L. (1980). Optimal release time for software systems based on reliability and cost criteria. *Journal of Systems and Software* 1: 315–318.
22. Ozekici, S., Altinel, I.K., & Angun, E. (2001). A general software testing model involving operational profiles. *Probability in the Engineering and Informational Sciences* 15(4): 519–533.
23. Pham, H. & Zhang, X.M. (1999). A software cost model with warranty and risk costs. *IEEE Transactions on Computers* 48(1): 71–75.
24. Ross, S.M. (1985). Software reliability: the stopping rule problem. *IEEE Transactions on Software Engineering* SE-11: 1472–1476.
25. Saito, Y. & Dohi, T. (2015). Stochastic marksmanship contest games with random termination – survey and application. *Journal of the Operations Research Society of Japan* 58(3): 223–246.
26. Sgarbossa, F. & Pham, H. (2010). A cost analysis of systems subject to random field environments and reliability. *IEEE Transactions on Systems, Man and Cybernetics, Part C* 40(4): 429–437.
27. Singh, V.B., Sharma, M., & Pham, H. (2018). Entropy based software reliability analysis of multi-version open source software. *IEEE Transactions on Software Engineering* 44(12): 1207–1223.
28. Singpurwalla, N.D. (1991). Determining an optimal time interval for testing and debugging software. *IEEE Transactions on Software Engineering* 17(4): 313–319.
29. Teraoka, Y. (1986). Silent-noisy marksmanship contest with random termination. *Journal of Optimization Theory and Applications* 49: 477–487.
30. Wee, N.-S. (1990). Optimal maintenance schedules of computer software. *Probability in the Engineering and Informational Sciences* 4(2): 243–255.
31. Xie, M. & Yang, B. (2003). A study of the effect of imperfect debugging on software development cost. *IEEE Transactions on Software Engineering* 29(5): 471–473.
32. Yamada, S. & Osaki, S. (1985). Cost-reliability optimal release policies for software systems. *IEEE Transactions on Reliability* R-34(5): 422–424.
33. Yang, B., Hu, H., & Jia, L. (2008). A study on uncertainty in software cost and its impact on optimal software release time. *IEEE Transactions on Software Engineering* 34(6): 813–835.
34. Yang, J., Liu, Y., Xie, M., & Zhao, M. (2016). Modeling and analysis of reliability of multi-release open source software incorporating both fault detection and correction processes. *Journal of Systems and Software* 115: 102–110.
35. Zeephongsekul, P. & Chiera, C. (1995). Optimal software release policy based on a two-person game of timing. *Journal of Applied Probability* 32: 470–481.
36. Zheng, S. (2002). Dynamic release policies for software systems with a reliability constraint. *IIE Transactions* 34: 253–262.
37. Zhu, M. & Pham, H. (2018). A multi-release software reliability modeling for open source software incorporating dependent fault detection process. *Annals of Operations Research* 269: 773–790.