Operating with an Incomplete Checklist

Michael P. Atkinson[*], Moshe Kress[†]

October 18, 2017

Abstract

We consider a time-critical operation that is contingent on completing a preliminary set of actions in a checklist. Aerial combat missions, emergency surgeries, launching a new product and rescuing hostages are a few examples of such situations. The operation may be executed before the full checklist is completed but then it may fail. The failure probability depends on the uncompleted actions. The question is when to abort the checklist and initiate the operation. In this paper we study this problem and prove that in certain realistic cases a simple myopic approach is optimal.

Supplementary materials are available for this article. Go to the publisher's online edition of IISE Transactions for the proofs of the propositions.

keywords: dynamic programming; optimal stopping; myopic policy

1 Introduction

Many time-critical operations must be preceded by a set of checks and actions specified in a *checklist*. For example, when launching an aerial combat mission, several technical, operational and logistical checks and actions must be performed before the aircraft can take off. If the time window for executing the mission is limited and the mission is critical, it is possible that the aircraft will take off before the checklist is completed. Another example is a critically wounded casualty who may enter the operating room and start surgery without completing all the necessary medical examinations and procedures that normally precede a surgery. A suspected location where terrorists

[*]mpatkins@nps.edu Operations Research Department, Naval Postgraduate School

[†]mkress@nps.edu Operations Research Department, Naval Postgraduate School

hold hostages may be raided before the complete intelligence picture becomes available, if it is feared that the hostages are at imminent risk. A final application would involve the decision to launch a new product. The company should complete quality control checks, stress tests, and focus group research before unveiling the product. However, the company may decide to launch early in fears that a competitor may introduce a similar product and gain control of the marketplace.

In this paper we model and analyze this situation and develop optimal stopping rules for when to abandon the checklist and immediately move forward to execute the operation. In particular, we show that there are situations where simple myopic policies are optimal.

The problem studied in this paper belongs to the general family of stopping rule problems, which has been extensively studied in the literature. From the Secretary Problem Freeman (1983), Ferguson (1989), to debugging and testing of new software Jelinski and Moranda (1972); Forman and Singpurwalla (1977), to test and evaluation Gaver and Jacobs (1997); Gaver et al. (2003), to estimating tolerated dosage in clinical trials O'Quigley and Reiner (1998) and information acquisition Browne and Pitts (2004), the question is when to stop a sequence of actions such that a certain objective attains its best expected reward. Stopping rules are also used in statistical analysis such as the sequential testing of two hypotheses Wald (1945). The classical book by Shiryaev (2007) presents a general theory of optimal stopping policies for the case of Markov processes. Other important references include Chow et al. (1971) and Ferguson (2004).

The most similar strand of literature to our work focuses on when to stop proofreading a manuscript for typos (see Chow and Schechner (1985); Ferguson and Hardwick (1989)) and when to stop debugging software (see Jelinski and Moranda (1972); Forman and Singpurwalla (1977); Dalal and Mallows (1988)). Early in the debugging/proofreading process, errors are found quickly, but later in the process the detection rate decreases as fewer errors remain. In our model if the actions on the checklist can be completed in parallel, the completion rate similarly decreases over time. Most prior work assumes homogeneity of the bugs in terms of detection rate and importance and a linear cost function that only depends upon the number of detected and undetected bugs. While a handful of articles relax some of these assumptions (see Ross (1985); Dalal and Mallows (1988); Ferguson and Hardwick (1989); Morali and Soyer (2003)), none take as general an approach to the problem as we do, especially with regard to the cost function. We allow the stopping cost to depend upon the identity of the completed checklist items, rather than just the number of completed items.

Fakhre-Zakeri and Slud (1996) and Dobson and Tezcan (2015) examine when to stop in a classic coupon collection problem, which is similar to our context and the debugging scenario. However, these coupon collection papers also take a more narrow approach to the problem and primarily focus on the asymptotic regime. One key aspect where our approach differs from the models in the debugging, proofreading, and coupon collection literature is that our model incorporates a window of opportunity that may close as we wait for more checks to complete.

A model that does incorporate a random window of opportunity is the search-and-interdiction scenario in Atkinson et al. (2016). In that setting, a searcher can potentially receive an unlimited stream of intelligence messages and she must decide when to act upon them. The differences between Atkinson et al. (2016) and the present one is in the operational setting, and more importantly, in the nature of the stopping rule that has in this paper more distinctive and elegant features, and is more easily implementable. Some articles that examine when a business should launch a product also include a limited window of opportunity before a competitor enters the market Armstrong and Lévesque (2002); Golany and Rothblum (2008). However, most of these articles include many factors, such as the type of financing, interest rates, and research burn rates, and formulate continuous-time optimal control models Roberts and Weitzman (1981); Armstrong and Lévesque (2002); Golany and Rothblum (2008); Lon and Zervos (2011). Our approach is less complex and more analytically tractable.

In Section 2 we describe the general setup of the problem. We then examine the special cases of performing the actions in parallel or in sequence in Sections 3 and 4, respectively. In Section 5 we allow for a mixture of both parallel and sequential actions. Section 6 presents some extensions to the basic model and Section 7 contains concluding remarks. Proofs appear in the Online Supplement.

2 Setting

In order to successfully execute an operation, the operator must complete n preliminary checks or procedures called henceforth simply *actions*. The durations of the actions are independent, exponentially distributed random variables, and the mean duration of action i is $1/\lambda_i$. The operation is time critical; it must be executed before a certain *window of opportunity* expires. The duration of the window of opportunity is exponentially distributed with rate parameter μ, and is independent

of the durations of the actions on the checklist. The decision maker can execute the operation before all actions are complete, but the operation will then fail with a certain probability. The actions on the checklist are not necessarily equally important; some actions may be more critical than others for the success of the operation. A function $f(\cdot)$ maps a list of incomplete actions to the probability of mission failure. Without loss of generality we assume that if the entire checklist is completed within the window of opportunity, then the probability of failure is 0. The cost of a failed operation is 1. If the window of opportunity expires before initiating the operation (e.g., the patient dies during a CT scan), then there is a cost d, which may be higher or lower than 1. We consider two types of checklists: a *parallel* list in which all actions start at the same time and are executed simultaneously, and a *sequential* list in which action i can start only upon the completion of action $i-1$, and the order of the actions is fixed and given. The operator must decide whether to execute the operation before all the actions complete, and if so when to initiate the operation. Due to the memoryless property of the exponential distribution, decisions need only be made at completion times of actions.

3 Parallel Checklist

All n actions start at the same time and are executed simultaneously. Obviously, because the durations of the actions are random, each action may end at a different time. We motivate the general model with the special case of homogeneous actions. That is $\lambda_i = \lambda$ for all i, and the failure probability only depends upon the number of incomplete actions. Specifically, if we denote j as the number of incomplete actions (henceforth called *state* j), and the dependence is linear, then $f(j) = aj + b$. According to our assumption $f(0) = 0$, hence $b = 0$, and without loss of generality we assume $a = \frac{1}{n}$. We next define the dynamic program formulation for the cost function $C(j)$:

$$C(j) = \min\left(\frac{j}{n}, \frac{\mu}{\lambda j + \mu} d + \frac{\lambda j}{\lambda j + \mu} C(j-1)\right) \qquad 1 \le j \le n \tag{1}$$

$$C(0) = 0.$$

The first term of the min function in equation (1) represents the expected cost of executing the operation in state j, and the second term represents the cost of waiting for the completion

of the next action. Because there are currently j ongoing actions, the time until the next action completion is an exponential random variable with rate parameter λj and thus we have a "race of exponentials": with probability $\mu/(\lambda j + \mu)$ the window of opportunity closes before the next action (out of the j actions yet to be completed) completes, which results in a cost of d, and with probability $\lambda j/(\lambda j + \mu)$, one of the j ongoing actions completes within the window of opportunity, which yields the recursive expected cost $C(j-1)$. The operator chooses one of the two options – execute the operation now or wait for the completion of the next action – that minimizes the expected cost. Because n is finite, solving (1) via backward induction will produce an optimal policy. We specify that if the operator is indifferent between executing the operation and waiting, she will execute. Using this tie-breaking rule, the optimal policy is well defined and unique. The homogeneous model defined in (1) makes similar assumptions to the debugging model of Jelinski and Moranda (1972), although the debugging model does not consider a window of opportunity.

While problems of the type described by equation (1) are usually solved via backward induction, this particular problem has a simple and elegant solution that boils down to a myopic policy (also called the *one-stage look-ahead policy* Ferguson (2004)). Using this policy, the operator compares the expected cost of executing the operation immediately to the cost of waiting for the completion of the next action and then executing the operation. If the expected cost of the latter (waiting for the next completed action in the checklist) is smaller than the expected cost of the former (executing the operation) then the operator waits, observes the next completed action (if the window of opportunity has not expired) and then repeats the myopic comparison. This myopic comparison is optimal and results in a threshold policy summarized in the following Proposition.

Proposition 1. *In the homogeneous case with $f(j) = j/n$, the optimal policy for the problem defined by equation (1) is to execute the operation if and only if*

$$\frac{j}{n} \leq \frac{\mu}{\lambda + \mu} d.$$

The operator should execute the operation as soon as at least a fraction $1 - \frac{\mu}{\mu+\lambda} d$ of the actions on the checklist have been completed. The proof of Proposition 1 appears in Section A of the Online Supplement. The optimality of the myopic policy follows from the monotone property introduced in Chapter 3.5 of Chow et al. (1971) (see also Chapter 5 of Ferguson (2004)).

Not surprisingly, the operator is more likely to execute sooner when the window of opportunity is short (large μ) and/or if failing to execute may result in severe consequences (large d). If the checklist can be completed very fast (very large λ), then the operator can afford to wait until a large portion of the checklist is completed.

3.1 General Model

We now present a more general model. Let x denote an n dimensional binary state vector, where $x(i) = 0$ if action i has been completed and $x(i) = 1$ otherwise. The mean completion time of action i is $\frac{1}{\lambda_i}$. In this case the cost function becomes:

$$C(x) = \min \left(f\left(x\right), \frac{\mu}{\sum_{k\,:\,x(k)=1} \lambda_k + \mu} d + \sum_{i\,:\,x(i)=1} \frac{\lambda_i C(x^{-i})}{\sum_{k\,:\,x(k)=1} \lambda_k + \mu} \right) \tag{2}$$

$$C(\vec{0}) = 0.$$

Where $\vec{0}$ is an n dimensional vector of all zeros (i.e., all actions have been completed) and x^{-i} is the state vector obtained from state x if $x(i) = 1$ and action i is the next completed action. Formally,

$$x^{-i}(j) = \begin{cases} 0 & \text{if } j = i \\ x(j) & \text{if } j \neq i \end{cases}.$$

The function $f(\cdot)$ is the failure probability, which depends on which actions remain incomplete. As with the homogeneous case, we can solve this problem via backward induction. However, the size of the state space is 2^n, which may lead to computational difficulties for even moderate values of n. The ease of solving the problem depends on the nature of the failure probability function $f(\cdot)$. For certain scenarios we can determine the optimal policy without using backward induction. In the following subsections we consider several forms of $f(\cdot)$.

3.1.1 Failure Probability with Marginally Decreasing Effect

It is reasonable to expect that the failure probability decreases if one additional action is completed. Formally, if $x > y$, that is, $x(i) \geq y(i),\ i = 1, ..., n$, with at least one sharp inequality, then $f(x) > f(y)$. It is also reasonable to assume, in some situations, a marginally decreasing effect of

completed actions. Formally, we say that $f(\cdot)$ exhibits *marginally decreasing effect* if $x > y$ and $x(i) = y(i) = 1$ for some $i = 1, ..., n$ imply that $f(x) - f(x^{-i}) \geq f(y) - f(y^{-i})$. If we view $f(\cdot)$ as a set function, then the marginally decreasing effect is equivalent to $f(\cdot)$ being a supermodular function. This property says that earlier completed actions are more effective than later ones in enhancing the probability of a successful operation. The next proposition states that if the failure probability satisfies these conditions, the myopic policy is optimal.

Proposition 2. *If $f(\cdot)$ satisfies the following two conditions,*

- *$x > y$ implies $f(x) > f(y)$,*
- *$x > y$ and $x(i) = y(i) = 1$ imply $f(x) - f(x^{-i}) \geq f(y) - f(y^{-i})$,*

then the optimal stopping rule is the myopic policy where the operator immediately executes the operation in state x if and only if

$$f(x) \leq \frac{\mu}{\sum_{k\,:\,x(k)=1} \lambda_k + \mu} d + \sum_{i\,:\,x(i)=1} \frac{\lambda_i f\left(x^{-i}\right)}{\sum_{k\,:\,x(k)=1} \lambda_k + \mu}.$$

The proofs of Proposition 2 and Corollary 1 appear in Section B of the Online Supplement.

In practice this simple policy is very valuable when n is large (e.g., $n > 30$) because the state space blows up quickly, which renders standard dynamic programming techniques impractical. With the aforementioned myopic policy, decisions could be made in real time. The following corollary connects the optimal action in state x with the optimal actions in future states or previous states.

Corollary 1. *If $f(\cdot)$ satisfies the conditions of Proposition 2, then the following relationships hold:*

- *If the optimal decision is to execute in state x, then that decision is optimal for any state y where $y \leq x$.*
- *If the optimal decision is to wait in state x, then that decision is optimal for any state y where $y \geq x$.*

Corollary 1 states that the optimal policy can be viewed as a generalized threshold policy; as soon as there is a state x in which it is optimal to immediately execute the operation, then it is also optimal to execute in any state in which the completed actions include all the completed actions in x.

3.1.2 Weighted Actions

In this section we assume that each action i has a weight $w_i > 0$, $\sum_{i=1}^{n} w_i = 1$, representing the relative contribution of that action to the success of the operation. The failure probability $f(\cdot)$ is monotone increasing in the sum of the w_i's corresponding to the incomplete actions. We can now rewrite (2) as:

$$C(x) = \min \left(f\left(\sum_{k\,:\,x(k)=1} w_k \right), \frac{\mu}{\sum_{k\,:\,x(k)=1} \lambda_k + \mu} d + \sum_{i\,:\,x(i)=1} \frac{\lambda_i C(x^{-i})}{\sum_{k\,:\,x(k)=1} \lambda_k + \mu} \right) \tag{3}$$

$$C(\vec{0}) = 0.$$

If the (now with unidimensional domain) failure probability $f(\cdot)$ is also convex, then the myopic policy is optimal.

Corollary 2. *If $f(\cdot)$ is increasing and convex, then the optimal stopping rule is the myopic policy where the operator immediately executes the operation in state x if and only if*

$$f\left(\sum_{k\,:\,x(k)=1} w_k \right) \leq \frac{\mu}{\sum_{k\,:\,x(k)=1} \lambda_k + \mu} d + \sum_{i\,:\,x(i)=1} \frac{\lambda_i f\left(\sum_{k\,:\,x(k)=1} w_k - w_i \right)}{\sum_{k\,:\,x(k)=1} \lambda_k + \mu}.$$

If we further assume that the failure probability is simply the total value of w_i for the incomplete actions, $f(z) = z$, then the condition in Corollary 2 simplifies significantly

Corollary 3. *If $f(z) = z$ then the optimal decision is to immediately execute the operation if and only if*

$$\sum_{k\,:\,x(k)=1} (\lambda_k + \mu) w_k \leq \mu d.$$

If we also assume that the nature of the actions is similar (e.g., technical checks) and therefore $\lambda_k = \lambda$ for all actions $k = 1, ..., n$, then the optimal decision is obtained as a simple threshold policy that is determined by the total relative contribution of the incomplete actions on the checklist. Specifically:

Corollary 4. *If $f(z) = z$ and $\lambda_k = \lambda$ for all $k = 1, ..., n$ then the optimal decision is to immediately*

execute the operation if and only if

$$\sum_{k\,:\,x(k)=1} w_k \leq \frac{\mu}{\lambda+\mu} d$$

If in addition we assume that all actions on the checklist are equally critical, that is, $w_k = \frac{1}{n},\ k = 1, ..., n$, we are back to the simple homogeneous model from Proposition 1.

3.1.3 Concave Failure Probability

Unfortunately, the myopic policy will not always be optimal for general failure probability functions. In this section we illustrate the suboptimality of the myopic policy when $f(\cdot)$ violates the conditions of Proposition 2. We maintain the assumption that each action has a weight w_i associated with it, and that $f(\cdot)$ is increasing in the sum of w_i corresponding to the incomplete actions. The convex failure probability $f(\cdot)$, studied in Sections 3.1.1–3.1.2, implies that the first few completed actions on the checklist are more significant for the success of the operations than later actions. While this may be the case in some situations, there may be other situations where the opposite is true and a concave failure probability $f(\cdot)$ will be more appropriate. This could occur in situations where significant confidence in the success of the operation hinges on the few final actions in the list.

The following example illustrates how the optimal stopping rule can deviate from the myopic one specified in Proposition 2–Corollary 4. Let $n = 6$, $\mu = 0.4$, $d = 1.8$, $\lambda_i = 1$ and $w_i = 1/6$ for all $i = 1, ..., 6$. We use the concave function $f(z) = z^{0.2}$ for the failure probability. In this case, the cost function is a slight modification of (1):

$$C(j) = \min\left(\left(\frac{j}{n}\right)^{0.2}, \frac{\mu}{\lambda j+\mu} d + \frac{\lambda j}{\lambda j+\mu} C(j-1)\right) \qquad 1 \leq j \leq n \tag{4}$$

$$C(0) = 0.$$

Table 1 presents the differences between the optimal and the myopic decisions. Each row in the table corresponds to a state j – the number of checklist actions yet to be completed. The second column is the cost to immediately execute in that state. This is the first term on the right-hand-side of (4). The third column is the cost of waiting and reassessing after the next action completion,

which corresponds to the second term on the right-hand-side of (4). The fourth column is the cost if the operator executes exactly after one additional action is completed. This is the right-hand-side of the condition in Corollary 2 and requires replacing $C(j-1)$ in (4) with $\left(\frac{j-1}{n}\right)^{0.2}$. The final two columns report the decision – execute (E) or wait (W) – when using the optimal or myopic policy, respectively. Evidently, the myopic policy is no longer optimal. With a concave function, the probability of mission failure drops quickly as j gets closer to 0, and the checklist is close to completion, but is relatively flat for larger j. The myopic policy, in this concave case, may miss the potential benefits of waiting for several more action completions, which may substantially reduce the failure probability down the road. Consequently, the myopic policy is more apt to execute the operation than the optimal decision, which may prefer to wait. Note that the optimal decision may oscillate between waiting (W) and executing (E) as j changes. For example, in Table 1 it is optimal to execute for larger j and $j = 0$ but wait for moderate values of j. This oscillating behavior occurs because of (a) potentially great benefits for waiting close to $j = 0$ due to the sharp decrease of $f(\cdot)$ in that region, and (b) the risk to waiting for very large j as the window of opportunity may close before the benefits of waiting can be realized. Starting from a void checklist, the optimal policy in this scenario is to execute immediately. It is still useful to know the optimal policy in future states (that theoretically should never be reached if the operator starts afresh) in case the operator takes over the decision-making duties after several actions have been completed, or perhaps some external force requires that some minimum number of actions need to be completed before execution.

State j	Cost to Execute	Cost to Wait	Myopic Waiting Cost	Optimal Decision	Myopic Decision
0	0	1.8	1.8	E	E
1	0.70	0.51	0.51	W	W
2	0.80	0.73	0.88	W	E
3	0.87	0.85	0.92	W	E
4	0.92	0.94	0.96	E	E
5	0.96	0.99	0.99	E	E
6	1	1.02	1.02	E	E

Table 1: Solution to the scenario with $f(z) = z^{0.2}$, $n = 6$, $\mu = 0.4$, $d = 1.8$, $\lambda_i = 1$ and $w_i = 1/6$.

4 Sequential Actions

If the actions must be completed sequentially in a certain pre-defined order of the checklist, then the state variable is simply the index of the next action to be completed. For consistency with Section 3, we use the reverse ordering of the indices, e.g., action 1 is the last action on the checklist and action n is the first item in the checklist. The state variable is j, the number of remaining incomplete actions. The cost function in this case is

$$C(j) = \min\left(f(j), \frac{\mu}{\lambda_j + \mu}d + \frac{\lambda_j}{\lambda_j + \mu}C(j-1)\right) \qquad 1 \leq j \leq n \tag{5}$$
$$C(0) = 0.$$

The general parallel model defined by equation (2) becomes computationally intractable for larger n. However, there are no such issues for the general sequential model, which can easily be solved numerically via backward induction. Unlike the parallel case, convexity of $f(\cdot)$ is generally insufficient for the myopic policy to be optimal. However, if we require the completion intensities λ_i to be monotone non-decreasing in i (i.e., earlier actions have higher values of λ_i), then the myopic policy becomes optimal. This result is summarized in the following Proposition.

Proposition 3. *If $f(\cdot)$ is increasing and convex and λ_i is non-decreasing in i, the operator should use the myopic policy. That is the operator should execute in state j if and only if*

$$f(j) \leq \frac{\mu}{\lambda_j + \mu}d + \frac{\lambda_j}{\lambda_j + \mu}f(j-1).$$

The proof of Proposition 3 and the related Corollaries appears in Section C of the Online Supplement.

The condition on λ_i implies that the earlier actions finish more quickly. The optimal policy can be rephrased as a threshold policy in j.

Corollary 5. *If $f(\cdot)$ is increasing and convex and λ_i is non-decreasing in i, then there is a threshold*

τ^* such that the operator should execute in state j if and only if $j \leq \tau^*$, where

$$\tau^* = \max\left\{ j \mid f(j) \leq \frac{\mu}{\lambda_j + \mu} d + \frac{\lambda_j}{\lambda_j + \mu} f(j-1) \right\}.$$

We now assume that the failure probability function takes the form $f(j) = f\left(\sum_{i=1}^{j} w_i\right)$, where w_i are the weights introduced in Section 3.1.2 that represent the relative contribution of action i. The conditions for Proposition 3 and Corollary 5 will be satisfied if $f(\cdot)$ is convex and the weights w_i are non-decreasing in i. This condition on the weights implies the earlier actions are more important than the later actions. As in the parallel case, when the failure probability is $f(z) = z$, the threshold in Corollary 5 simplifies greatly.

Corollary 6. *If $f(z) = z$ and the w_i and λ_i are both non-decreasing in i, the operator should execute in state $j \leq \tau^*$ if and only if*

$$\tau^* = \max\left\{ j \mid \lambda_j w_j + \mu \sum_{i=1}^{j} w_i \leq \mu d \right\}.$$

If all the actions are homogeneous, that is, $w_i = 1/n$ and $\lambda_i = \lambda$ for all $i = 1, ..., n$, then the threshold becomes even simpler.

Corollary 7. *If $f(z) = z$, $w_i = 1/n$, and $\lambda_i = \lambda$ for all $i = 1, ..., n$, then*

$$\tau^* = \max\left(0, dn - \lambda/\mu\right).$$

The final corollary compares the stopping rule (execute the operation) in the two – sequential and parallel – cases when the actions are homogeneous.

Corollary 8. *If $f(z) = z$, $w_i = 1/n$, and $\lambda_i = \lambda$ for all i, the operator should execute sooner (i.e., larger j) in the sequential case than in the parallel case.*

Corollary 8 is not surprising. The total completion rate of actions is slower in the sequential case compared to the parallel case, therefore it is more likely that the window of opportunity will close before many actions on the list are completed.

We conclude this section with an example where $f(z) = z$, $w_i = 1/n$, but the λ_i's are non-monotonic. Thus, the results from Proposition 3 – Corollary 8 do not apply. Namely, the myopic

policy is not necessarily optimal, nor is the optimal policy a threshold policy. The results appear in Table 2, which has a similar format to Table 1. The optimal policy oscillates between executing and waiting because the λ_i oscillate. It is risky to wait for an action with a small λ_i because the window of opportunity is more likely to close while waiting.

State j	λ_i	Cost to Execute	Cost to Wait	Myopic Waiting Cost	Optimal Decision	Myopic Decision
0	0	0	0.900	0.900	E	E
1	1	0.111	0.082	0.082	W	W
2	0.1	0.222	0.491	0.506	E	E
3	0.1	0.333	0.561	0.561	E	E
4	1	0.444	0.385	0.385	W	W
5	1	0.556	0.432	0.486	W	W
6	0.1	0.667	0.666	0.728	W	E
7	0.1	0.778	0.783	0.783	E	E
8	1	0.889	0.789	0.789	W	W
9	1	1	0.799	0.890	W	W

Table 2: Solution to the scenario with $f(z) = z$, $n = 9$, $\mu = 0.1$, $d = 0.9$, $w_i = 1/9$.

5 Mix of Parallel and Sequential Actions

What happens if the checklist contains a mixture of parallel and sequential actions? We assume that there are n_p parallel homogeneous actions and n_s sequential homogeneous actions. For simplicity we assume that both parallel and sequential actions share the same completion intensity λ. We also assume that the failure probability is a function of the sum of the weights of the incomplete actions and that all actions have the same weight $1/(n_p + n_s)$. A state in the decision process is a pair (j_p, j_s), where j_p and j_s are the remaining incomplete parallel and sequential actions, respectively. As in Section 4, the sequential actions must be completed in reverse order of the index: action n_s is the first action on the sequential checklist and action 1 is the last. Furthermore, we assume the parallel and sequential actions are completed on two independent tracks. That is completion of sequential action j_s only requires that sequential action j_s+1 be performed first; it does not depend on the completion of any parallel action. The cost function is:

$$
\begin{aligned}
C(j_p, j_s) = \min \Bigg(& f\left(\frac{j_p + j_s}{n_p + n_s}\right), \\
& \frac{\mu d}{\lambda(j_p+1)+\mu} + \frac{\lambda j_p C(j_p - 1, j_s)}{\lambda(j_p+1)+\mu} + \frac{\lambda C(j_p, j_s - 1)}{\lambda(j_p+1)+\mu} \Bigg) \quad 1 \le j_p \le n_p, 1 \le j_s \le n_s \\
C(j_p, 0) = \min & \left(f\left(\frac{j_p}{n_p + n_s}\right), \frac{\mu d}{\lambda j_p + \mu} + \frac{\lambda j_p C(j_p - 1, 0)}{\lambda j_p + \mu} \right) \quad 1 \le j_p \le n_p \\
C(0, j_s) = \min & \left(f\left(\frac{j_s}{n_p + n_s}\right), \frac{\mu d}{\lambda + \mu} + \frac{\lambda C(0, j_s - 1)}{\lambda + \mu} \right) \quad 1 \le j_s \le n_s \\
C(0, 0) = & \, 0
\end{aligned}
$$

In this setup, convexity is sufficient for the optimality of the myopic policy. This result appears in the following Proposition.

Proposition 4. *If $f(\cdot)$ is increasing and convex then the optimal policy is myopic. That is, the operator should execute in state (j_p, j_s) if and only if*

$$
f\left(\frac{j_p + j_s}{n_p + n_s}\right) \le \frac{\mu d}{\lambda(j_p + I(j_s > 0)) + \mu} + \frac{\lambda(j_p + I(j_s > 0)) f\left(\frac{j_p + j_s - 1}{n_p + n_s}\right)}{\lambda(j_p + I(j_s > 0)) + \mu}
$$

where $I(\cdot)$ is the indicator function.

The proof of Proposition 4 and Corollary 9 appears in Section D of the Online Supplement. As

with the purely parallel and sequential cases, when $f(z) = z$, the condition simplifies. We have a threshold policy for j_p, but the threshold depends upon the value of j_s.

Corollary 9. *If $f(z) = z$, the operator should execute if and only if $j_p \leq \tau^*(j_s)$, where*

$$\tau^*(j_s) = \frac{\mu d}{\mu + \lambda}(n_p + n_s) - \frac{\lambda I(j_s > 0) + \mu j_s}{\mu + \lambda}$$

For $j_s = 0$, the threshold in Corollary 9 is equivalent to the threshold in Corollary 4. As we increase j_s, the threshold decreases. However, the total number of checklist actions on the stopping boundary $(\tau^*(j_s) + j_s)$ does increase with j_s, which implies that the operator is more likely to execute if there are more sequential actions in the checklist.

6 Extensions

We analyze four extensions in this section. Section 6.1 incorporates a cost to complete each action. In Section 6.2 we assume that when the window of opportunity closes, the decision maker rushes to execute the operation. In this case the cost to rush the operation depends on the number of actions completed. Section 6.3 considers a form of dependency between the action completion times and the window of opportunity. Finally in Section 6.4 we allow for general distributions for the action completion time and the window of opportunity. We focus only on the parallel case from Section 3; the sequential model has very similar results.

6.1 Costs Associated with Action Completion

In our base analysis we consider two costs: the cost of a failed operation (normalized to 1) and the cost if the window of opportunity closes (d). In addition it may cost κ_i to complete action i. In this situation, (2) transforms to

$$C(x) = \min\left(f(x), \frac{\mu}{\sum_{k\,:\,x(k)=1}\lambda_k + \mu}d + \sum_{i\,:\,x(i)=1}\frac{\lambda_i(\kappa_i + C(x^{-i}))}{\sum_{k\,:\,x(k)=1}\lambda_k + \mu}\right) \tag{6}$$

$$C(\vec{0}) = 0.$$

The myopic policy discussed in Section 3 (Proposition 2) may not be optimal for an arbitrary action cost κ_i. If one of the incomplete actions is quite expensive, the myopic policy may recommend to stop immediately, whereas the optimal policy will wait as all other incomplete actions are much less costly. Below we state the modified version of Proposition 2 with the required restrictions on action costs κ_i.

Proposition 5. *If $f(\cdot)$ satisfies the conditions in Proposition 2 and satisfies*

$$f(x) - f(x^{-i}) \geq \kappa_i, \quad \textit{for all } x \textit{ such that } x(i) = 1$$

then the optimal stopping rule is the myopic policy.

The proof appears in Section E of the Online Supplement. The new condition in Proposition 5 limits the cost of action i. If action i completes, then the stopping cost drops from $f(x)$ to $f(x^{-i})$, and this differential captures the marginal operational benefit from completing action i. If that benefit exceeds the cost to complete action i, then the myopic policy remains optimal.

In the homogeneous action case (constant λ) with linear failure probability $f(j) = j/n$ and constant action cost κ, the new condition in Proposition 5 corresponds to $\kappa < \frac{1}{n}$. If this condition is met, then including the action cost modifies the stopping condition in Proposition 1 to

$$\frac{j}{n} \leq \frac{\mu}{\lambda(1 - \kappa n) + \mu} d.$$

6.2 Rushed Operation

The parameter d captures the expected cost when the window of opportunity closes before the operator executes the operation. In the base model, the operator cannot execute the operation after the window closes, and hence the cost d does not depend upon the completed actions. However, in some applications the window closing might actually represent the time when the operator is forced into executing the operation. For example, a doctor, observing that the condition of a patient has deteriorated to a critical point, may rush her into surgery, or a commander may scramble aircraft to provide immediate support to ambushed ground forces under heavy fire. In this situation, d is the expected cost to rush the operation due to the window closing. Since the operation is executed, albeit in a rushed fashion, the cost d may depend on the state x. We modify our cost function

slightly to account for this:

$$C(x) = \min\left(f(x), \frac{\mu}{\sum_{k\,:\,x(k)=1}\lambda_k + \mu} d(x) + \sum_{i\,:\,x(i)=1} \frac{\lambda_i C(x^{-i})}{\sum_{k\,:\,x(k)=1}\lambda_k + \mu} \right) \tag{7}$$

$$C(\vec{0}) = 0.$$

The myopic policy may no longer be optimal because $d(x)$ may change drastically as actions complete. The following proposition imposes restrictions on $d(x)$.

Proposition 6. *If $f(\cdot)$ satisfies the conditions in Proposition 2 and satisfies*

$$x > y \text{ implies } d(x) - f(x) \leq d(y) - f(y),$$

then the optimal stopping rule is the myopic policy.

The proof appears in Section F of the Online Supplement. We can view $f(x)$ as the expected cost to deliberately execute the operation in state x and $d(x)$ as the expected cost to rush to execute the operation when the window of opportunity closes. The condition $d(x) - f(x) \leq d(y) - f(y)$ when $x > y$ implies that as more actions complete it becomes more costly to execute a rushed operation relative to a deliberate execution. This is a reasonable assumption in many cases. Early in the process, there is often not much difference between a deliberate and rushed operation as both have little chance of succeeding. However, near the end after many actions have completed, there may be a significant difference between the success probability of a deliberate operation compared to a rushed one.

In the homogeneous action case (constant λ) with linear failure probability $f(j) = j/n$, the function $d(j) = (1-\alpha)\frac{j}{n} + \alpha$, for $\alpha \in [0,1]$ satisfies the new condition in Proposition 6. In this special case, the stopping condition in Proposition 1 changes to

$$\frac{j}{n} \leq \frac{\mu\alpha}{\lambda + \mu\alpha}.$$

When $\alpha = 1$, we revert back to the base model where d is a constant. For smaller α, the operator is less likely to execute the operation because waiting also reduces the cost associated with the window

of opportunity closing.

6.3 Actions-Window Dependence

In this section we present a possible manifestation of dependence between actions and the window of opportunity and define conditions when the myopic policy is optimal. We limit ourselves to the setup in Proposition 1: constant completion intensity λ, and linear failure probability ($f(j) = j/n$).

The underlying operation is one of two types: "routine" or "urgent." The action completion times have parameters λ_R and λ_U for routine and urgent operations, respectively. Similarly we define the parameters for the window closure time as μ_R and μ_U. We assume that $\frac{\mu_U}{\lambda_U} > \frac{\mu_R}{\lambda_R}$, which implies that the window of opportunity is more likely to close before the next action completion for urgent operations, that is, $\frac{\mu_U}{\lambda_U j+\mu_U} > \frac{\mu_R}{\lambda_R j+\mu_R}$. At the beginning of the process, the operator has a prior belief p_U about the likelihood that the operation is urgent. As the operator observes action completion times, the probability p_U updates in a Bayesian fashion. Conditioned on a routine (urgent) action completing before the window closes, the action completion time has an exponential distribution with rate $\lambda_R j+\mu_R$ ($\lambda_U j+\mu_U$). If we observe action interarrival time x, then we denote the updated probability that the operation is urgent by $\tilde{p}(x, j, p_U)$:

$$\tilde{p}(x, j, p_U) = \frac{p_U(\lambda_U j + \mu_U)e^{-(\lambda_U j+\mu_U)x}}{p_U(\lambda_U j + \mu_U)e^{-(\lambda_U j+\mu_U)x} + (1 - p_U)(\lambda_R j + \mu_R)e^{-(\lambda_R j+\mu_R)x}}$$

We augment our state space j with the probability that the operation is urgent p_U:

$$\begin{aligned} C(j, p_U) = \min \Bigg(& \frac{j}{n}, \left(p_U \frac{\mu_U}{\lambda_U j + \mu_U} + (1 - p_U)\frac{\mu_R}{\lambda_R j + \mu_R} \right) d \\ & + p_U \frac{\lambda_U j}{\lambda_U j + \mu_U} \int_0^\infty (\lambda_U j + \mu_U)e^{-(\lambda_U j+\mu_U)x} C(j - 1, \tilde{p}(x, j, p_U))dx \\ & + (1 - p_U)\frac{\lambda_R j}{\lambda_R j + \mu_R} \int_0^\infty (\lambda_R j + \mu_R)e^{-(\lambda_R j+\mu_R)x} C(j - 1, \tilde{p}(x, j, p_U))dx \Bigg) \quad 1 \le j \le n,\ p_U \in [0, 1] \\ C(0, p_U) = 0 \qquad & p_U \in [0, 1]. \end{aligned}$$

If the parameters for the routine and urgent operations differ substantially, we cannot guarantee the optimality of the myopic policy. The following Proposition formalizes how close the parameters need to be for the myopic policy to be optimal.

Proposition 7. *If*

$$\frac{\mu_R}{\lambda_R + \mu_R}\left(\frac{\lambda_R}{\mu_R} - \frac{\lambda_U}{\mu_U}\right) < \frac{1}{n},$$

then the optimal stopping rule is the myopic policy where the operator immediately executes the operation in state (j, p_U) *if and only if*

$$\frac{j}{n} \leq \left(p_U \frac{\mu_U}{\lambda_U j + \mu_U} + (1 - p_U)\frac{\mu_R}{\lambda_R j + \mu_R}\right) d + \left(p_U \frac{\lambda_U j}{\lambda_U j + \mu_U} + (1 - p_U)\frac{\lambda_R j}{\lambda_R j + \mu_R}\right) \frac{j-1}{n}.$$

The proof appears in Section G of the Online Supplement.

6.4 General Probability Distributions

Generalizing beyond the exponential distribution makes the analysis much more complicated. With the memoryless property, we only had to evaluate the operator's decision at action completion times. For general distributions, it might be optimal to execute the operation mid-action, and thus we must track the current time and residual lifetimes of the incomplete actions. As with the dependent case in Section 6.3, for simplicity we focus on the homogeneous actions case (constant λ) and linear failure probability ($f(j) = j/n$). To simplify the analysis, we discretize time and allow decisions every Δt time period. The bookkeeping required to analyze $\Delta t \to 0$ is unwieldy and, we believe, will not add any significant insight.

Let the window of opportunity have probability distribution G, and the completion times of the n actions be IID random variables with distribution H. Without loss of generality, we set time to 0 at the lower bound of the support of G, as the operator would never execute the operation before this time. We assume that the distributions have a finite support with upper bounds u_G and u_H, with $u = \min(u_G, u_H)$. The operator can only execute the operation at discrete time points $i\Delta t$ for $i = 0, 1, 2, \ldots, \frac{u}{\Delta t}$, for some fixed timestep Δt. The hazard functions play a significant role in our analysis as we consider the probability the window of opportunity will expire in the near future, given the window has not closed by a certain time. We next define the discretized version of the

hazard function for the window of opportunity:

$$r_G(t, \Delta t) = \frac{\frac{G(t+\Delta t) - G(t)}{\Delta t}}{1 - G(t)}.$$

The function $r_H(t, \Delta t)$ related to the action completion time is defined similarly. The expression $r_H(t, \Delta t)\Delta t$ is the probability an action completes in $(t, t + \Delta t]$, given that the action has not completed by time t.

We now account for both the number of remaining actions j and the current time t when making our decision. If the operator executes the operation in state (j, t), the expected cost is $\frac{j}{n}$. If the operator waits, with probability $r_G(T, \Delta t)\Delta t$ the window expires before the next decision epoch, which results in cost d. Otherwise, if the window does not close, some number of the remaining j actions will complete in the next Δt time. Combining these pieces yields the cost function:

$$C(j,t) = \min\Bigg(\frac{j}{n}, r_G(t,\Delta t)\Delta t \times d + (1 - r_G(t,\Delta t)\Delta t)\sum_{k=0}^{j}\binom{j}{k}(r_H(t,\Delta t)\Delta t)^k\,(1 - r_H(t,\Delta t)\Delta t)^{j-k}\,C(j-k, t+\Delta t)\Bigg) \quad 1 \le j \le n,\ \ t \le u - \Delta$$

$$C(j,u) = d \qquad \text{if } u_G \le u_H,\ \ 0 \le j \le n$$

$$C(j,u) = 0 \qquad \text{if } u_G > u_H,\ \ 0 \le j \le n$$

$$C(0,t) = 0 \qquad t \le u - \Delta t.$$

If the window does not close, a binomial number of actions will complete in $(t, t + \Delta t]$ because the remaining j actions are IID random variables. The base case value of $C(j, u)$ depends upon whether G or H has the smaller upper bound on the support. If $u_G \le u_H$, then the window will close with certainty in the last time period, whereas if $u_G > u_H$ all actions are guaranteed to complete by the last time period.

The myopic policy is optimal if the hazard functions meet certain conditions.

Proposition 8. *If*

- $r_H(t, \Delta t) + r_G(t, \Delta t) > 0$ *for all* $t \in [0, u]$
- $r_H(t, \Delta t)$ *is non-decreasing for all* $t \in [0, u]$
- $\frac{r_H(t,\Delta t)}{r_G(t,\Delta t)}$ *is non-increasing for all* $t \in [0, u]$

then the optimal stopping rule is the myopic policy where the operator immediately executes the

operation in state (j,t) *if and only if*

$$\frac{j}{n} < \frac{r_G(t,\Delta t)}{r_G(t,\Delta t) + r_H(t,\Delta t) - r_G(t,\Delta t) r_H(t,\Delta t)\Delta t} d.$$

The proof appears in Section H of the Online Supplement. The first condition of Proposition 8 ensures that before the next decision epoch, there is some possibility that either an action will complete or the window will close. The main condition for Proposition 8 is the last one, which effectively states that the hazard function for the window of opportunity must increase at a faster relative rate than the hazard function of the action completion times. For small Δt, the last term in the denominator of the stopping condition in Proposition 8 becomes negligible, and the stopping rule simplifies into a ratio of hazard functions. This produces a nice generalization of the result in Proposition 1.

If the action completion times are IID uniform random variables on $[0,a]$, and the time until the window of opportunity closes is a uniform random variable on $[0,b]$, then the conditions of Proposition 8 are satisfied for $b < a$. In this case the stopping condition is

$$\frac{j}{n} < \frac{a-t}{a+b-2t-\Delta t} d.$$

7 Conclusion

Initiating an operation may be contingent on completing a set of preliminary actions listed in a checklist. Sometimes, when the operation is time-critical, the operator may opt to stop the preliminary actions and start the operation before the checklist is complete. There are many relevant applications, e.g., in defense, industry, and healthcare, where such decisions are made. In this paper we study various manifestations of this situation and develop optimal stopping rules for when to abandon the checklist and execute the operation. While in the most general case the stopping rule must be solved computationally using backward induction, we identify several realistic cases where myopic, and even threshold, policies are optimal. These policies allow the operator to make optimal decisions in real-time. Future work could further develop the extensions introduced in Sections 6, especially the general distribution and dependence scenarios. The sequential model assumes the ordering of actions is exogenously given. Future work could analyze situations where

the operator chooses the ordering of actions ahead of time.

Acknowledgment

This work was funded by the Office of Naval Research (ONR).

References

Armstrong, M. J., Lévesque, M. (2002) Timing and quality decisions for entrepreneurial product development. *European Journal of Operational Research*, **141**(1) 88–106.

Atkinson, M. P., Kress, M., and Langer, R. J. (2016) When is Information Sufficient for Action? Search with Unreliable Yet Informative Intelligence. *Operations Research*, **64**(2) 315–328.

Browne, G. J. and Pitts, M. G. (2004). Stopping rule use during information search in design problems. *Organizational Behavior and Human Decision Processes*, **95**(2) 208–224.

Chow, C. W. and Schechner, Z. (1985) On stopping rules in proofreading. *Journal of Applied Probability*, **22**(4) 971–977.

Chow, Y. S., Robbins, H., and Siegmund, D. (1971) *Great expectations: The theory of optimal stopping*, Houghton Mifflin, Boston.

Dalal, S. R. and Mallows, C. L. (1988) When should one stop testing software? *Journal of the American Statistical Association*, **83**(403) 872–879.

Dobson, G. and Tezcan, T. (2015) Optimal sampling strategies in the coupon collector's problem with unknown population size. *Annals of Operations Research*, **233**(1) 77–99.

Fakhre-Zakeri, I. and Slud, E. (1996) Optimal stopping of sequential size-dependent search. *The Annals of Statistics*, **24**(5) 2215–2232.

Ferguson, T. S. (1989) Who solved the secretary problem? *Statistical Science*, **4**(3) 282–289.

Ferguson, T. S. (2004) *Optimal Stopping and Applications*. Mathematics Department, UCLA. `http://www.math.ucla.edu/~tom/Stopping/Contents.html`.

Ferguson, T. S. and Hardwick, J. P. (1989). Stopping rules for proofreading. *Journal of Applied Probability*, **26**(2) 304–313.

Forman, E. H. and Singpurwalla, N. D. (1977) An empirical stopping rule for debugging and testing computer software. *Journal of the American Statistical Association*, **72**(360) 750–757.

Freeman, P. R. (1983) The secretary problem and its extensions: A review. *International Statistical Review/Revue Internationale de Statistique*, **51**(2) 189–206.

Gaver, D. P, Jacobs, P. A., Glazebrook, K. D., and Seglie, E. A. (2003) Probability models for sequential-stage system reliability growth via failure mode removal. *International Journal of Reliability, Quality and Safety Engineering*, **10**(1) 15–40.

Gaver, D. P. and Jacobs, P. A. (1997) Testing or fault-finding for reliability growth: A missile destructive-test example. *Naval Research Logistics*, **44**(7) 623–637.

Golany, B. and Rothblum, U. G. (2008) Optimal investment in development projects. *Operations Research Letters*, **36**(6) 657–661.

Jelinski, Z. and Moranda, P. B. (1972) Software reliability research in *Statistical computer performance evaluation*, Freiberger, W. (ed) Academic Press, New York, 465–484.

Lon, P. C. and Zervos, M. (2011) A model for optimally advertising and launching a product. *Mathematics of Operations Research*, **36**(2) 363–376.

Morali, N. and Soyer, R. (2003) Optimal stopping in software testing. *Naval Research Logistics*, **50**(1) 88–104.

O'Quigley, J. and Reiner, E. (1998) A stopping rule for the continual reassessment method. *Biometrika*, **85**(3) 741–748.

Roberts, K. and Weitzman, M. L. (1981). Funding criteria for research, development, and exploration projects. *Econometrica*, **49**(5) 1261–1288.

Ross, S. M. (1985) Software reliability: the stopping rule problem. *IEEE Transactions on Software Engineering*, **11**(12) 1472–1476.

Shiryaev, A. N. (2007) *Optimal stopping rules*, vol. 8. Springer Science & Business Media.

Wald, A. (1945) Sequential tests of statistical hypotheses. *The Annals of Mathematical Statistics*, **16**(2) 117–186.