Sequential testing of k-out-of-n systems with imperfect tests

Wenchao Wei[1], Kris Coolen[2], Fabrice Talla Nobibon[3], and Roel Leus[4*]

Abstract. A k-out-of-n system configuration requires that, for the overall system to be functional, at least k out of the total of n components be working. We consider the problem of sequentially testing the components of a k-out-of-n system in order to learn the state of the system, when the tests are costly and when the individual component tests are imperfect, which means that a test can identify a component as working when in reality it is down, and vice versa. Each component is tested at most once. Since tests are imperfect, even when all components are tested the state of the system is not necessarily known with certainty, and so reaching a lower bound on the probability of correctness of the system state is used as a stopping criterion for the inspection.

We define different classes of inspection policies and we examine global optimality of each of the classes. We find that a globally optimal policy for diagnosing k-out-of-n systems with imperfect tests can be found in polynomial time when the predictive error probabilities are the same for all the components. Of the three policy classes studied, the dominant policies always contain a global optimum, while elementary policies are compact in representation. The newly introduced class of so-called 'interrupted block-walking' policies combines these merits of global optimality and of compactness.

Keywords: sequential testing, k-out-of-n systems, imperfect tests, sequencing and scheduling.

1 Introduction

System health monitoring for complex systems, such as a space shuttle, aircraft or integrated circuits, is crucial for reducing the likelihood of accidents due to sudden failures, and for improving system availability. It is also imperative that systems be tested before being put into operation, in order to ascertain their functionality. At manufacturing sites, for instance, products are typically inspected at the final stage of production before being shipped to the customer. Electronic equipment (smart mobile phones, laptops, etc.) in particular, which contains components from many different suppliers, has various tests executed throughout the different stages of manufacturing.

This text is a reworked and updated version of the earlier working paper KBI_1315.

[1]Faculty of Engineering, INESC, University of Porto, Portugal, wenchao.wei@inesctec.pt.
[2]HEC – Management School, QuantOM, Université de Liège, Belgium, kris.coolen@ulg.ac.be.
[3]Federal Express (FedEx), Brussels, Belgium, tallanob@gmail.com.
[4]ORSTAT, Faculty of Economics and Business, KU Leuven, Belgium, Roel.Leus@kuleuven.be.
*Corresponding author.

The k-out-of-n configuration is a special case of a complex system that requires that, for the overall system to be functional, at least k out of the total of n components must be working. This configuration has a wide range of applications in both industrial and engineering systems (Ünlüyurt, 2004), such as a multi-display system in a cockpit, the multi-radiator system in a heating system, a bridge with n cables where a minimum of k cables are necessary to support the bridge, and the power grid of a city with excess power generators. Consider, for example, an airplane with four engines. Furthermore, suppose that the design of the aircraft is such that at least two engines are required to function for the aircraft to remain airborne. This means that the engines are related in a k-out-of-n configuration, with $k = 2$ and $n = 4$. This is in literature sometimes also referred to as a "2-out-of-4:G" system, where G means the system works or is "good"; a k-out-of-n:G system is equivalent to an $(n - k + 1)$-out-of-n:F system, which fails ("F") if at least $(n - k + 1)$ components fail. The airplane is tolerant to failures in up to two engines. More generally, the k-out-of-n system configuration represents systems with built-in redundancy. A so-called *series* system is an n-out-of-n system and a *parallel* system is a 1-out-of-n system.

In sequential testing, the procedure of diagnosing a system consists in testing the components one by one in order to learn the state of the system (Ünlüyurt, 2004). The same diagnosis procedure may be repeated thousands of times, and so it is important to minimize the total expected costs in the long run. Additionally, besides this cost directly attributable to the test-set hardware and manpower, field return costs can be reduced by improving output quality through appropriate testing. In this article we search for an inspection *policy*, which is a set of decision rules that decide in which order to test the components, and respects specific stopping criteria. More specifically, we develop algorithms for finding optimal policies that minimize the expected testing expenses.

We focus on the case where individual component tests are *imperfect*, which means that a test can identify a component as working when in reality it is down, and vice versa; this can have severe implications. Obviously, different costs will be incurred in different ensuing situations, but these are neglected in this article: we only focus on the expected cost to assess the state of the system with a specific confidence level. Sequencing of imperfect component tests has already been studied in a number of isolated articles; we provide an overview in Section 2. The reference closest to our work is Nachlas et al. (1990), who focus only on series systems. To the best of our knowledge, however, the more general sequencing problem of imperfect tests

for k-out-of-n systems has not yet been treated in the existing literature. It is the goal of this paper to fill exactly this gap.

Throughout the text, we will say that a *positive* test outcome is associated with the discovery of a failure. Imperfect testing can involve two types of test errors. The first type of error is *false positive* (type-I error), which means the tester concludes the component fails (is 'down') when really it is not; for example, a product can fail a quality test before being shipped, while in reality it is in good condition (component is 'up'). The second type of test error is *false negative* (type-II error), which means the outcome is negative but in reality the component fails, for example, a system check-up failing to detect the fault it was designed to find, in a computer system that really has the fault. *Positive predictive value* is the proportion of positive results that are truly positive, whereas *negative predictive value* is the proportion of negative results that are truly negative. The following expressions show how these two values are related with type-I and type-II error:

$$
\begin{aligned}
&\text{positive predictive error } \epsilon_0 = \Pr\{ \text{ component up } | \text{ outcome positive } \},\\
&\text{negative predictive error } \epsilon_1 = \Pr\{ \text{ component down } | \text{ outcome negative } \},\\
&\text{positive predictive value } (1-\epsilon_0) = \Pr\{ \text{ component down } | \text{ outcome positive } \},\\
&\text{negative predictive value } (1-\epsilon_1) = \Pr\{ \text{ component up } | \text{ outcome negative } \},\\
&\text{type-I error} = \Pr\{ \text{ outcome positive } | \text{ component up } \},\\
&\text{type-II error} = \Pr\{ \text{ outcome negative } | \text{ component down } \},
\end{aligned}
$$

where $\Pr\{A|B\}$ is the conditional probability of event A knowing that event B has occurred. Note that predictive values are conditioned on the test results and represent the probability of presence or absence of a fault in a given component test. For definitions of similar terms in diagnostic tests, we refer the reader to Altman and Bland (1994a,b).

Consider a numerical example for testing one module of a specific product. Suppose the positive and negative predictive error are equal, $\epsilon_0 = \epsilon_1 \equiv \epsilon = 10\%$, so if a test outcome is positive, we know that the module is not functional with probability 0.9 and is working with probability 0.1; a similar interpretation applies for the negative outcome. If $\chi = 81.25\%$ of the test outcomes of the production output for a given day is negative, then we can infer that the a-priori probability that a produced module works, is $0.1(1-\chi)+0.9\chi = 0.75$. The type-I error is computed as $0.1875 \times 0.1/0.75 = 0.025$ and the type-II error is $0.8125 \times 0.1/0.25 = 0.325$.

In the foregoing example, we first fix the positive and negative predictive value and thereby

also the complement ϵ, which represents the probability that a test outcome is wrong. The type-I and type-II errors then follow implicitly from this choice for ϵ combined either with historical test data or with the a-priori probability that the module is functional. We will continue this approach throughout the article, and this contrasts with the few existing references (e.g., Ding et al., 1998; Nachlas et al., 1990), where the type-I and type-II errors are assumed to be known, which then implies a value for the predictive errors. Indeed, when a tester examines a component returning a positive result, he/she then wishes to evaluate what is the probability of the need to reject or repair this component. Type-I and type-II errors cannot be directly used to answer this question, because they are conditional on whether the component is actually functional or not. Predictive values are therefore very useful measures of diagnostic accuracy in routine inspection practice (Akobeng, 2006; Altman and Bland, 1994a,b). Additionally, and this is a rather stringent assumption, we will assume that the positive predictive error ϵ_0 is the same for all components, and likewise for the negative predictive error ϵ_1. One possible source of such a common error probability is the design of a single unreliable test machine. Benjamini and Hochberg (1995, 2000) also make a case for controlling what they call the "false discovery rate" in test design, which is equivalent to our positive predictive error ϵ_0. In the context of event detection by means of a network of sensors, Luo et al. (2006) also assume identical "false alarm probabilities," in a setup that is close to a k-out-of-n system. Sarma and Tufts (2001) describe that it is desirable in signal detection to maintain the "average probability of false alarm" at a fixed level despite time-varying noise, leading to "constant false alarm rate" (CFAR) detectors.

Two types of testing policies are defined by Butterworth (1972): *sequential* and *non-sequential*. Sequential then means that the testing order of the components is selected before the diagnosis begins, while non-sequential means the testing order is dynamic, in the sense that it can vary according to the results of the component tests. The terminology, however, varies between references (compare with Wald (1945), for example), and in the general setting of *sequential system diagnosis*, the adjective 'sequential' simply refers to the fact that component tests are conducted one after the other, and never simultaneously (in a scheduling context (Pinedo, 2012), one could speak of single-machine scheduling). In order to avoid misunderstanding, we therefore resort to the use of the name *elementary* policy to refer to what Butterworth (1972) defines as a sequential policy.

In the policy classes studied in this article, decisions are made dynamically, meaning that they are conditional on the observations (the outcomes) of the previous tests. As mentioned

above, this dynamic character of our policy classes constitutes a very natural motivation for conditioning on the test outcomes (fixing positive and negative predictive value) rather than on the actual (hence, unknown) state of the components (as would be done by specifying type-I and type-II errors). One can also imagine that under certain circumstances, testing the same component more than once would improve the quality of the output. We do not consider such retesting: each component is tested at most once. This allows us to focus only on the sequencing aspect.

The contributions of this article are threefold: (1) we describe a general setting for k-out-of-n system testing with imperfect tests; (2) we examine different classes of diagnosis policies and discuss global optimality of each of the classes; and (3) we present a polynomial-time algorithm to find a globally optimal policy. In the process, we also define and analyze other problem variants of k-out-of-n testing. The remainder of this text is structured as follows: Section 2 provides a review of the relevant literature. Some definitions and a formal problem statement are given in Section 3. A number of observations and results regarding the confidence level are presented in Section 4, and our main results are established in Section 5. We summarize and conclude this article in Section 6.

2 Literature review

An extensive literature review of different types of sequential testing problems can be found in Ünlüyurt (2004). Butterworth (1972) shows that the special cases of a parallel ($k = 1$) and a series ($k = n$) system without precedence constraints are polynomially solvable. A polynomial-time algorithm for arbitrary k was presented first by Salloum (1979), and independently by Ben-Dov (1981). Efficient implementations of this algorithm were proposed in Chang et al. (1990) (off-line algorithm requiring $O(n^2)$ space and $O(n^2)$ time) and in Salloum and Breuer (1997) (on-line algorithm requiring $O(n)$ space and $O(n \log(n))$ time). With general precedence constraints, the testing problem for series systems is NP-hard (Kelly, 1982; De Reyck and Leus, 2008). Computational results for sequencing tests of k-out-of-n systems with general precedence constraints can be found in Wei et al. (2013). One specific variant of classic k-out-of-n testing is the so-called *conservative* k-out-of-n testing, which is defined in the same way, but testing now continues until either k successful tests are observed or until all n tests have been performed (Hellerstein et al., 2011). In this text, we will assume component tests to be independent;

Cramer and Kamps (1996) present a generalization in which the failure rate of the untested components is parametrically adjusted based on the number of preceding failures.

Most research efforts in the system-testing literature have been directed at finding testing policies for systems with $k = 1$ and $k = n$ (parallel and series systems) and with special precedence constraints (e.g. a series-parallel or a tree precedence graph). Relatively less attention has been given to imperfect testing, where due to a defective test design or unforeseen errors, the outcome of a test only reflects the real condition of the component with a probability less than one, otherwise giving adverse information or even no information. The optimization of sequential search processes with imperfect tests has already been modeled and applied to various domains; we provide a survey below.

Imperfect testing has been introduced for so-called 'search problems.' In the *discrete search problem with a stationary target* (Ahlswede and Wegener, 1987), an item is assumed to be hidden in one of a set of boxes. Associated with each box i is a prior probability p_i ($\sum_i p_i = 1$) that the item is hidden in that box and the overlook probability (type-II error) a_i that the item will not be found in a particular search of that box even though the item is actually there. Value a_i remains the same for every search of box i, and the time (cost) consumed in examining box i is c_i. The search procedure does not stop before the item is found. Bellman (1957) was the first to describe an optimal policy to minimize the expected search cost by arranging the components in descending order of the ratio $p_i(1 - a_i)/c_i$. Using this result, Gluss (1959) develops an optimal procedure for detecting the breakdown in a complex multi-component system. The stop criterion is the same: the fault can be concealed due to test errors, but is ultimately discovered by repeating tests until it is properly isolated. Wagner and Davis (2001) extend the module concept proposed by Gluss (1959) for discrete sequential search and refer to it as a 'group activity.' Variations of stationary-object search are addressed in Song and Teneketzis (2004) and Stone et al. (1972).

The discrete search problem can also be seen as a representation of system testing. Besides the cost of testing, which depends on the set of components that are actually tested, Nachlas et al. (1990) also consider the consequences of a test error related to the disposition of the system after repair. If a false positive test result occurs, a functioning component is replaced and the failed component is left in place. If the system is then returned to service, the system fails immediately. If a false negative test result occurs, the overall test could indicate that no item fails; if the system is then returned to service, it fails immediately or it might be scrapped.

Nachlas et al. add these two events with corresponding cost coefficient into the objective function and perform efficient enumeration of permutations of test sequences to find an optimal one that minimizes the expected total costs for small systems (less than 10 components) with series structure.

Raghavan et al. (1999) study the *single-fault detection problem* with unreliable tests, where the fault is inherited from a finite failure source, and there is a finite set of available tests, each of which checks a subset of failure sources. The problem is to design a test policy with minimum expected total diagnostic cost to isolate the failure source with a specified confidence (typically within $[0.95, 0.99]$). This problem is treated as a partially observed Markov decision problem (POMDP), and is solved by a continuous-state dynamic programming recursion. Different fault detection problems are considered in Balakrishnan and Semmelbauer (1999), Shakeri et al. (2000), and Ruan et al. (2009).

In the testing department of a typical manufacturing company, for example a semiconductor manufacturer, the objective of the testing process is to achieve a high outgoing-product quality while minimizing the costs of testing, scrapping conforming products, and the opportunity cost of passing non-conforming items. Tzimerman and Herer (2009) consider a batch of products that was produced in a given order on a machine that is subject to random failures. They propose an exact dynamic programming algorithm and four heuristics to find an off-line inspection policy which finds the first non-conforming item (i.e., the point at which the machine fails) with a given confidence level while minimizing the expected number of inspected products in the batch. In the model of Tzimerman and Herer a conformity test may be wrong and it is assumed that an item is tested at most once. Since the tests are imperfect, however, applying the same test multiple times can be useful for obtaining higher outgoing quality and economic savings. Raouf et al. (1983) develop a model where accepted components are repetitively retested, and they determine the optimal number of repeat inspections for multi-characteristic components to minimize the total expected cost per accepted component due to type-I error, type-II error and cost of inspection. Greenberg and Stokes (1995) use the data acquired from retesting rejected items to estimate the probability of testing errors. Note that this stream of literature allows type-I and type-II errors of tests to vary after each test cycle. Ding et al. (1998) examine the question whether it is better to repetitively test rejected components or to repetitively test accepted components. Choosing between these two policies depends on the tradeoff between scrapping costs and outgoing quality; see also Ding and Gong (2008) and Quinino et al. (2010) for

variations of this problem. Sequencing issues and imperfect testing have both been considered in the inspection of multi-characteristic components; we refer to Raouf et al. (1983), Schmidt and Bennett (1972) and Tang and Tang (1994) for examples. The results in the foregoing references, however, do not apply for the case where retesting is not allowed or economically infeasible. Solutions have been published for the latter case also, albeit mainly in a different field; this is the subject of the next paragraph.

The reliability of a system is the probability that the system functions (ex ante, without diagnosis). The main concern of the reliability literature is the evaluation or the approximation of the reliability of a given system; see for instance Wu and Chen (1994) for weighted k-out-of-n systems, Ding et al. (2010), who develop approximation procedures for the reliability of multi-state weighted k-out-of-n systems, or Eryilmaz (2013), who analyzes the case when components have random weights. For a k-out-of-n configuration with imperfect information, most of the literature has focused on the design of the system such that it strikes a balance between reliability and cost. In general, the reliability of a k-out-of-n system for fixed k increases with the number n of components. As the required reliability of the system increases, the cost also goes up due to the increase in the number of redundant (idle) components in the system. Marseguerra et al. (2005) develop an approach to incorporate uncertainty (component failure probabilities are not known with certainty) into reliability calculations by using Monte-Carlo simulation and genetic algorithms. Amari et al. (2004) also study the design of systems with built-in redundancy, including k-out-of-n subsystems subjected to imperfect fault coverage (type-II error, see Arnold (1973) for the definition of fault coverage).

3 Definitions and problem statement

3.1 Definitions

We monitor a system consisting of n components; the component set is $N = \{1, 2, \ldots, n\}$. In order to discover the state of the system, we can test each component sequentially on a single test machine. Each component is in one of two *states*: either working (up) or not working (down). The system functions (succeeds) if at least $k \leq n$ of the n components are working and malfunctions (fails) if at least $(n-k+1)$ components are not working. Each component is tested at most once (Nachlas et al. (1990) refer to this setting as 'single-pass' testing). We also call the test of component i simply 'test i.' The *outcome* of test i is a binary value $x_i \in \mathbb{B} = \{0, 1\}$

with the interpretation that $x_i = 0$ if and only if the test detects a fault (a positive outcome). All outcomes can be gathered in an n-dimensional binary vector $\mathbf{x} = (x_1, x_2, \ldots, x_n) \in \mathbb{B}^n$.

We study the situation where the measurements (tests) are imperfect: the outcome can be positive while the component is actually working, and vice versa. As already stated in Section 1, each positive outcome has probability ϵ_0 of being wrong (in which case, the system is really functioning); we refer to ϵ_0 as the *positive predictive error*. Correspondingly, value ϵ_1 represents the *negative predictive error*, the probability that a negative outcome is incorrect. We define χ_i as the probability that $x_i = 1$ (negative outcome), and p_i is the prior probability that component i works. For ease of notation, we also define $q_i = 1 - p_i$ as the prior probability that component i is down, and $\lambda_i = 1 - \chi_i$ the probability that $x_i = 0$. Let X_i represent outcome i before testing, which is a Bernoulli random variable with parameter χ_i, and denote by $\mathbf{X} = (X_1, X_2, \ldots, X_n)$ the associated vector of random variables. The realization of each X_i is known only at the end of test i. We assume all variables X_i to be mutually independent.

A *schedule* $\mathbf{s} = (s_1, s_2, \ldots, s_{|\mathbf{s}|})$ is an ordered subset of N indicating the test sequence of a specific outcome, with s_t the index of the test in position t in the schedule $\mathbf{s}$. Let value c_i represent the cost of test i; then the cost of schedule $\mathbf{s}$ is $f(\mathbf{s}) = \sum_{t=1}^{|\mathbf{s}|} c_{s_t}$. Tests are conducted one at a time, so a sequence of (a subset of) tests indeed defines a schedule. Testing the components sequentially is a dominant decision for our objective function, but this does not necessarily apply for other objectives (such as makespan minimization; see, for instance, Pinedo, 2012).

3.2 Conditions on the predictive errors

From the foregoing definitions, it directly follows that

$$p_i = (1 - \epsilon_1)\chi_i + \epsilon_0(1 - \chi_i). \tag{1}$$

When $\epsilon_0 + \epsilon_1 \neq 1$, we can also express χ_i in function of the other parameters as follows:

$$\chi_i = \frac{p_i - \epsilon_0}{1 - \epsilon_0 - \epsilon_1} \tag{2}$$

From this expression, it is natural to distinguish the following three cases:

1. $\epsilon_0 + \epsilon_1 < 1$

From (1) it can be seen that the condition $\epsilon_0 + \epsilon_1 < 1$ implies that

$$\epsilon_0 < p_i < 1 - \epsilon_1 \tag{3}$$

$$\epsilon_1 < q_i < 1 - \epsilon_0 \tag{4}$$

In other words, the probability of having a working component increases after a negative test result and decreases after a positive test result. Similarly, the probability of a failing component increases after a positive test result and decreases after a negative test result. We conclude that the condition $\epsilon_0 + \epsilon_1 < 1$ is a characteristic of any reasonable component test in the sense that the outcome of the test does indeed provide more information on the true component state. Remark that in this case, it also holds that $0 < \chi_i < 1$.

2. $\epsilon_0 + \epsilon_1 = 1$

   In this case the strict inequalities in (3) and (4) become equalities, and therefore the test outcome does not change the likelihood of the component to be either working or failing. In other words, the component state and the component test outcome are statistically independent events; the test gives no information concerning the component state. Further note that in this case, χ_i is not even known from p_i, ϵ_0 and ϵ_1 (it can be any value between 0 and 1), and can only be determined when the type-I and type-II errors are also given.

3. $\epsilon_0 + \epsilon_1 > 1$

   This case corresponds to a test where the component is more likely to work after observing a positive test and more likely to fail after a negative test. Such a test would not be reasonable.

Based on the above analysis, we will assume that $\epsilon_0 + \epsilon_1 < 1$ in the remainder of this paper. We explicitly use this condition in Section 5.

3.3 Problem statement

A solution to the sequencing problem under study is a testing policy, which decides how to proceed at each stage based on diagnosis information from the preceding test outcomes. After each test, either a new component is selected or the diagnosis procedure is halted; this choice is based on the confidence level regarding the system's state. Let $\theta_1(r, \mathbf{x}, \mathbf{s})$ be the probability that there are k or more working components among the first r tested ones following schedule $\mathbf{s}$ and

with outcome vector $\mathbf{x}$ (which would mean that the system functions), with $r \leq |\mathbf{s}|$ and $r \leq |\mathbf{x}|$. Similarly, $\theta_0(r, \mathbf{x}, \mathbf{s})$ is the probability that there are at least $(n - k + 1)$ failing components among the first r tested components (which would mean that the system malfunctions). When there is no risk of confusion, we will use $\theta_1(r)$ and $\theta_0(r)$ for short. We define the overall system confidence level as $\theta(r) = \max\{\theta_0(r), \theta_1(r)\}$; in the context of isolating a single failure source, Raghavan et al. (1999) also use the term 'level of confidence' to refer to this value. We study the optimization problem of designing a test policy Π with minimum expected total diagnostic costs. Given a specified threshold value T on system confidence (e.g., $T = 95\%$), the inspection procedure stops when $\theta(r)$ reaches or exceeds T; if this stop criterion is never fulfilled then we perform all the n tests (which maximizes the confidence of the result). The value of T should follow from the requirements of the system or is specified by the user.

Consider an example 3-out-of-5 system for a given test outcome $(1, 1, 1, 1, 1)$ and schedule $(1, 2, 3, 4, 5)$. With negative predictive error rate $\epsilon_1 = 10\%$ (the value of ϵ_0 is irrelevant here because all test outcomes are negative), we have $\theta_1(3) = 72.9\%$, $\theta_1(4) = 94.77\%$ and $\theta_1(5) = 99.144\%$, and also $\theta_0(3) = 0.1\% = 0.1^3$, $\theta_0(4) = 0.37\%$ and $\theta_0(5) = 0.856\%$, while $\theta_0(r) = \theta_1(r) = 0$ for $r \leq 2$ (see Section 4 for details on the computation of these values). In case $T = 95\%$ then the diagnosis is not interrupted after the first four component tests because the obtained confidence level is not yet high enough, and so the fifth component is also tested; after this fifth test, $\theta_1(5) \geq T$ and so we conclude that the system is working. Had we worked with $T = 90\%$, however, then $\theta_1(4) \geq T$ and the fifth test would not have been necessary to achieve the (lower) desired confidence threshold. Value $T = 90\%$ would still have required four negative test outcomes although $k = 3$ because $\theta_1(3) = 0.729 < 0.9$. For $T = 99.5\%$, even after all five component tests the threshold is not attained, and so we halt without firm conclusion; we will call such a diagnosis *inconclusive*.

In line with the literature on stochastic scheduling (Igelmund and Radermacher, 1983), a policy Π can be modeled as a function $\Pi : \mathbb{B}^n \to \Sigma$ that maps outcomes $\mathbf{x}$ to schedules, where Σ is the set of all schedules; different outcomes may be mapped to the same schedule. Our objective is to find a policy Π^* within a specific class that minimizes the following expression, which represents the expected cost of policy Π:

$$\mathbb{E}[f(\Pi(\mathbf{X}))] = \sum_{\mathbf{x} \in \mathbb{B}^n} \left(\prod_{i:x_i=1} \chi_i \right) \left(\prod_{i:x_i=0} \lambda_i \right) f(\Pi(\mathbf{x})), \tag{5}$$

Table 1: Costs and probabilities of the example 3-out-of-5 instance

i	1	2	3	4	5
c_i	1	1	1	1	1
χ_i	$0.5+\delta$	0.5	0.5	0.5	$0.5-\delta$

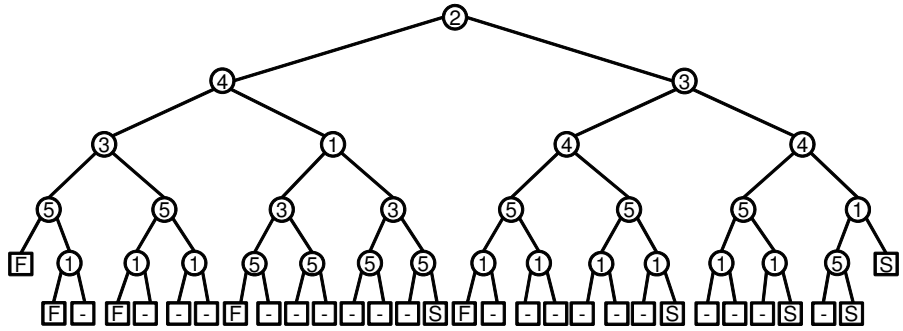

Figure 1: A globally optimal policy of the example instance

with $\mathbb{E}[\cdot]$ the expectation operator with respect to $\mathbf{X}$ and $f()$ the cost function as defined in Section 3.1. We say that a policy is *globally optimal* if it achieves the minimum expected cost over all possible policies.

The flexibility inherent in a testing policy allows to schedule different tests conditional on the outcomes of the previously conducted tests: the second component to be tested, for instance, might depend on the result (success or failure) of the first test. To capture this dynamic nature, a policy Π can also be represented by a binary decision tree (BDT). In such a BDT, each non-leaf node is labeled with the index of a component to be tested and has two child nodes. If the test outcome is positive then the left branch is entered, otherwise the right subtree is taken. Each leaf node either represents a conclusion of a specific system state (successful (S) or failed (F)) with the required confidence, or indicates that all components have been tested but the diagnosis is inconclusive (labeled by $-$). For a given outcome, the policy generates the schedule stepwise from the root node to a leaf node. Table 1 contains additional parameters for the example 3-out-of-5 system, with $0 \leq \delta < 0.5$; let the positive predictive error $\epsilon_0 = 5\%$. Figure 1 depicts a BDT representing a testing policy for the example instance of Table 1 with $T = 90\%$; its expected cost is $4.875 - 0.25\delta$. For outcome vector $(1, 0, 1, 0, 1)$, for example, the policy consecutively tests components 2, 4, 3, 5, and 1, and the tests are inconclusive.

3.4 Policy classes

In this section we briefly describe the classes of dominant and of elementary policies, which were both defined in Wei et al. (2013) for sequential testing with perfect tests.

We say that a policy is *dominant* if for any two vertices u_1 and u_2 with the same tested component set in the BDT representation of the policy (meaning that the same components were tested in the diagnosis process to reach u_1 and u_2) and with the same number of negative test outcomes in the preceding tests, it holds that the subtrees rooted at u_1 and u_2 are identical.

Lemma 1. *There exists a dominant policy that is globally optimal.*

All proofs appear in the appendix. Inspired by priority policies for standard scheduling problems (Pinedo, 2012), we also consider *elementary policies*, which are characterized by a total order of the component set N. Such an order can be represented by a list (or permutation) $L = (l_1, l_2, \ldots, l_n)$ of the elements of N. For a given outcome $\mathbf{x}$, the elementary policy Π characterized by a list L generates a unique schedule $\Pi(\mathbf{x}; L)$ by testing the components one by one in the order of the list (so first l_1, then l_2, etc.). The diagnosis stops when the confidence level reaches the threshold or all the components have been tested. The list $(1, 2, 3, 4, 5)$, for example, characterizes an elementary policy for the example instance presented above, with corresponding expected costs 4.875 for $T = 90\%$. Clearly, one direct advantage of elementary policies is their compact representation: one can also draw up a BDT for an elementary policy, but this is not necessary since the list contains all the information.

4 System confidence level

4.1 Computing the confidence level

We define $Q^{(\mathbf{x},\mathbf{s})}(w, r)$ as the probability that there are exactly w components working among the first r components tested (consequently, $r-w$ components are down) according to schedule $\mathbf{s}$ and outcome $\mathbf{x}$ (with $r \leq |\mathbf{s}|$ and $r \leq |\mathbf{x}|$). We use $Q(w, r)$ for short when no ambiguity is possible. We initialize $Q(0, 0) = 1$, and $Q(w, r) = 0$ when $w = -1$ or $w > r \geq 0$, and we propose the following recursive formula to generate each $Q(w, r)$ for $w = 0, 1, \ldots, n$; $r = 1, 2, \ldots, n$ with

Table 2: Values of $Q(w,r)$ for outcome $\mathbf{x}=(1,0,1,0)$ and schedule $\mathbf{s}=(1,2,3,4)$

$r=$	1	2	3	4
$x_{s_r}=$	1	0	1	0
$w=0$	ϵ_1	$\epsilon_1(1-\epsilon_0)$	$\epsilon_1^2(1-\epsilon_0)$	$\epsilon_1^2(1-\epsilon_0)^2$
$w=1$	$1-\epsilon_1$	$\epsilon_0\epsilon_1+(1-\epsilon_0)(1-\epsilon_1)$	$\epsilon_0\epsilon_1^2+2\epsilon_1(1-\epsilon_0)(1-\epsilon_1)$	$2\epsilon_0\epsilon_1^2(1-\epsilon_0)$ $+2\epsilon_1(1-\epsilon_0)^2(1-\epsilon_1)$
$w=2$	0	$\epsilon_0(1-\epsilon_1)$	$2\epsilon_0\epsilon_1(1-\epsilon_1)$ $+(1-\epsilon_0)(1-\epsilon_1)^2$	$\epsilon_0^2\epsilon_1^2+(1-\epsilon_0)^2(1-\epsilon_1)^2$ $+4\epsilon_0\epsilon_1(1-\epsilon_0)(1-\epsilon_1)$
$w=3$	0	0	$\epsilon_0(1-\epsilon_1)^2$	$2\epsilon_0^2\epsilon_1(1-\epsilon_1)$ $+2\epsilon_0(1-\epsilon_0)(1-\epsilon_1)^2$
$w=4$	0	0	0	$\epsilon_0^2(1-\epsilon_1)^2$
$\sum_{w=0}^{4} Q(w,r)=$	1	1	1	1

$w \leq r$:

$$\begin{aligned} Q(w,r) = {} & Q(w,r-1)(x_{s_r}\epsilon_1+(1-x_{s_r})(1-\epsilon_0)) \\ & + Q(w-1,r-1)(x_{s_r}(1-\epsilon_1)+(1-x_{s_r})\epsilon_0). \end{aligned} \tag{6}$$

The remaining values $Q(w,r)$ are set to zero. In words, in order to have w working components after r tests, either there were already w working components after $r-1$ tests (which corresponds with the first term in the right-hand side), and then the last test s_r pertains to a failing component, which is either correctly observed ($x_{s_r}=0$) with probability $1-\epsilon_0$, or wrongly observed ($x_{s_r}=1$) with probability ϵ_1. Alternatively, there were only $w-1$ working components after the first $r-1$ tests, and then a similar reasoning can be followed to obtain the second term in the right-hand side.

Table 2 gives an illustration of $Q(w,r)$ for a given outcome $\mathbf{x}=(1,0,1,0)$ and schedule $\mathbf{s}=(1,2,3,4)$ (we do not include the fifth component because this would make the table overly unwieldy, but line $w=5$ and column $r=5$ can be added, given knowledge of x_5, following exactly the same logic). Computing all values $Q(w,r)$ requires $O(n^2)$ operations in total for a given schedule and outcome vector. Given the values of $Q(w,r)$, we derive $\theta_1(r)$ and $\theta_0(r)$ as

follows:

$$\theta_1(r) = \begin{cases} 0 & \text{if } r < k \\ \sum_{w=k}^{r} Q(w,r) & \text{if } r \geq k \end{cases} \tag{7}$$

$$\theta_0(r) = \begin{cases} 0 & \text{if } r < n-k+1 \\ \sum_{w=0}^{r-n+k-1} Q(w,r) & \text{if } r \geq n-k+1 \end{cases} \tag{8}$$

The value of $\theta_1(4)$ computed in Section 3.3 for the 3-out-of-5 instance with outcome vector $(1,1,1,1,1)$, for instance, can be obtained as $Q(3,4) + Q(4,4) = 4 \cdot 0.9^3 \cdot 0.1 + 0.9^4 = 0.9477$.

We define $\dot{Q}(g;t,r)$ as the probability that there are exactly g working components in reality among the first r components already tested, when t negative results were obtained.

Lemma 2. *For $g,t,r = 0,1,2,\ldots,n$ with $g \leq r$ and $t \leq r$, we have:*

$$\dot{Q}(g;t,r) = \begin{cases} \sum_{l=0}^{\min\{r-g,t\}} \binom{r-t}{g-t+l}\binom{t}{l}(1-\epsilon_0)^{r-g-l}\epsilon_0^{g-t+l}(1-\epsilon_1)^{t-l}\epsilon_1^{l} & \text{if } g \geq t \\ \sum_{l=0}^{\min\{g,r-t\}} \binom{r-t}{l}\binom{t}{t-g+l}(1-\epsilon_0)^{r-t-l}\epsilon_0^{l}(1-\epsilon_1)^{g-l}\epsilon_1^{t-g+l} & \text{otherwise.} \end{cases} \tag{9}$$

Proposition 1. *$Q^{(\mathbf{x},\mathbf{s})}(w,r) = \dot{Q}(w;t,r)$, where t is the number of negative outcomes in $\mathbf{x}$.*

This proposition follows directly by observing that the computation of $\dot{Q}(g;t,r)$ in the proof of Lemma 2 can be carried through step by step for $Q^{(\mathbf{x},\mathbf{s})}(w,r)$. The underlying reason for the equality is the fact that we have assumed equal positive and equal negative predictive errors for all components. In Equations (7) and (8), we have implicitly (via Q) defined the confidence level as dependent on one specific pair of outcome and schedule. From Proposition 1, however, we see that as long as the number r of conducted tests is the same and the number of positive outcomes is the same (even if the set of components examined differs), the corresponding confidence will be identical. We therefore redefine the values $\theta_1(r;t)$ and $\theta_0(r;t)$ as conditional on the number r of tested components and the number t of observed negative outcomes; we will write parameter t explicitly only when necessary.

4.2 Comparing the confidence with the threshold T

The decisions to be made by a tester are: (1) when to halt the testing procedure, and (2) how to sequence the tests such that the total expected costs are minimized. In the remainder of

Section 4, we will answer the first question; we develop a solution to the second question in Section 5. We first state a few properties of $\theta_1(r)$ and $\theta_0(r)$.

Lemma 3. *$\theta_1(r, \mathbf{x}, \mathbf{s})$ and $\theta_0(r, \mathbf{x}, \mathbf{s})$ are non-decreasing functions of r for any given schedule $\mathbf{s}$ with outcome vector $\mathbf{x}$.*

Following the definitions of $\theta_1(r)$ and $\theta_0(r)$, this lemma is intuitive: increasing r means more tests, so more possibilities to reach the required number of successes and failures. This also implies that both positive and negative outcomes can contribute to both $\theta_1(r)$ as well as $\theta_0(r)$. In particular, for any fixed number $t \leq r$ of negative test outcomes the probabilities $\theta_1(r; t)$ and $\theta_0(r; t)$ are non-decreasing in the number of conducted tests r.

Lemma 4. *For $0 \leq r \leq n$, we have $\theta_0(r; t) + \theta_1(r; t) \leq 1$ for any value of t; equality holds when $r = n$.*

Informally, $1 - \theta_0(r) - \theta_1(r)$ is the probability that neither k successes nor $(n - k + 1)$ failures occur among the first r tested components. Since $k + (n - k + 1) = n + 1 > n$, these two events cannot occur simultaneously for any $r \leq n$, and when $r = n$ then exactly one of the two events will hold. From Lemma 4, we infer that:

Corollary 1. *If $T > 50\%$ then we cannot simultaneously have $\theta_1(r) \geq T$ and $\theta_0(r) \geq T$ for any number of tests r.*

In words, we will never conclude that a system works and fails at the same time. To illustrate the importance of the condition $T > 50\%$, consider the instance in Section 3.3 with $\epsilon_0 = \epsilon_1 = 40\%$ and $T = 45\%$. For outcome $(1, 1, 1, 0, 0)$ we have $\theta_1(5; 3) = 53.86\% > T$ and $\theta_0(5; 3) = 46.14\% > T$, whereas both $\theta_1(4; 3) < T$ and $\theta_0(4; 3) < T$. Clearly, this is a situation that renders the classification of the system highly confusing because the thresholds for concluding system success and system failure are reached at the same time. To avoid such situations, we thus assume $T > 50\%$ throughout this text. Intuitively, with only two system states possible, it is also clear that a tester would need to have more than 50% certainty about being in any specific state before drawing conclusions about what is the most plausible state.

For the 3-out-of-5 example instance of Section 3.3 with $T = 95\%$, we observed that the series of outcomes $(1, 1, 1, 1)$ from schedule $(1, 2, 3, 4)$ led to $\theta_1(4) = 94.77\% < T$, and we suggested that the final component 5 also needed to be tested in this case. From Lemma 3 we know that both $\theta_1(r)$ as well as $\theta_0(r)$ are non-decreasing in r, and we see that if $\epsilon_0 > 0$ then $\theta_1(5) > \theta_1(4)$

even if $x_5 = 0$. In particular, even for outcome $(1, 1, 1, 1, 0)$ and schedule $(1, 2, 3, 4, 5)$, and with $\epsilon_0 = 5\%$ as before, we have $\theta_1(5) = 95.01\% > T$. In conclusion, $\theta_1(5) > T$ whether x_5 takes value 0 or 1, and so whatever the outcome of the final test, the threshold for concluding a working system will be reached anyway, while (based on Corollary 1), the threshold for a failing system cannot be reached. This indicates that the diagnosis may be interrupted already after observing four negative results, and following similar arguments it can be seen that it is also optimal to halt after four positive results. In the remainder of this section we formalize this insight, which is based on the fact that the values $\theta_0(r)$ and $\theta_1(r)$ only pertain to the first r components while we learn from Lemma 3 that these values are non-decreasing with r.

With Theorem 1, we provide an alternative testing diagnosis based on simply counting the number of successful and unsuccessful outcomes such that more information is included in the stopping criterion. We first state the following intermediate results:

Lemma 5. *If* $\epsilon_0 + \epsilon_1 < 1$ *and* $0 \leq t_1 < t_2 \leq r$ *then* $\theta_1(r; t_1) \leq \theta_1(r; t_2)$.

Lemma 6. *If* $\epsilon_0 + \epsilon_1 < 1$ *and* $0 \leq l_1 < l_2 \leq r$ *then* $\theta_0(r; n - l_1) \leq \theta_0(r; n - l_2)$.

Under the stated condition, these lemmas imply that having more negative outcomes for the same number of tests will never decrease the probability of having a working system. Similarly, having more positive outcomes will never decrease the probability of having a failing system. We pointed out in Section 3.2 that the condition $\epsilon_0 + \epsilon_1 < 1$ is inherent to any reasonable component test, so it is definitely not a restrictive condition.

We define the following parameters:

$$K_1 = \min\{l \in \mathbb{N} : l \leq n \text{ and } T \leq \theta_1(n; l)\}; \tag{10}$$

$$K_0 = \min\{l \in \mathbb{N} : l \leq n \text{ and } T \leq \theta_0(n; n - l)\}. \tag{11}$$

In words, K_1 is the lowest integer not exceeding n such that $\theta_1(n)$ is at least equal to the threshold T when K_1 negative outcomes were observed. Similarly, K_0 is the lowest integer less than or equal to n such that when K_0 positive test results are obtained, the conclusion of a failing system will be drawn with the required confidence level after the n-th test or sooner. In case the min-operator applies to an empty set (there is no integer l that fulfills the conditions) then we will say that the parameter in question does not exist. Thus, if K_1 does not exist then it is known before the start of the system diagnosis that we will never reach the conclusion

that the system functions with the required confidence threshold T, whatever the outcomes. Similarly, when K_0 does not exist then the conclusion of system failure can never be drawn with the required confidence. If neither K_0 nor K_1 exist then the testing sequence becomes irrelevant because the diagnosis will conduct all the n tests under all possible outcomes (and will certainly be inconclusive).

Before the main theorem, we make a few observations. To start with, it can happen that K_1 equals 0, which is when $\theta_1(n;0) \geq T$. When $k = 1$ (only one working component needed for a successful system), for instance, then $\theta_1(n;0) = 1 - (1-\epsilon_0)^n$, and so $K_1 = 0$ when $n > \frac{\ln(1-T)}{\ln(1-\epsilon_0)}$. With $T = 75\%$ and $\epsilon_0 = 10\%$, $K_1 = 0$ for $n = 14$ and higher. This means that the system will work with at least 75% confidence whatever the outcomes of the tests (even of they are all positive), due to testing errors. The case where $K_0 = 0$ can be interpreted similarly. We need the following lemma (assuming $T > 50\%$):

Lemma 7. *If $K_1 = 0$ then K_0 does not exist, and vice versa.*

We can now state the main result of this section (assuming $\epsilon_0 + \epsilon_1 < 1$ and $T > 50\%$):

Theorem 1. *When the test procedure accumulates K_1 negative tests then it is a dominant decision to stop the procedure immediately, and the system will work with confidence level at least T. When the test procedure reaches K_0 positive tests then it is a dominant decision to halt the test procedure immediately, and the system will fail with confidence level at least T. Moreover, these stopping criteria are tight: one cannot stop testing before reaching either K_0 positive or K_1 negative outcomes and still guarantee the required confidence level.*

From Lemma 3 and Lemma 5 it follows that $\theta_1(r;t)$ is non-decreasing in both arguments r and t such that its maximum value is reached when $t = r = n$. Therefore, K_1 exists if and only if $\theta_1(n;n) \geq T$, and it can be computed by comparing T with the different θ-values. From Lemma 3 and Lemma 6, K_0 exists if and only if $\theta_0(n;0) \geq T$, and its value can be evaluated using a similar procedure. Overall, identifying K_0 and K_1 can be done in $O(n^3)$ time. This can be seen as follows: we need the values of $\theta_1(n;t)$ and $\theta_0(n;n-l)$ for $0 \leq t, l \leq n$. This can be achieved recursively by computing Q-values for appropriate outcome vectors; this takes $O(n^2)$ time per outcome. One such set of Q-values enables us to obtain $\theta_1(n;t)$ and $\theta_0(n;n-l)$ for a fixed value of t and l (pick an outcome vector with t ones and l zeros). This will cost another $O(n)$ additions. Therefore, we can evaluate $\theta_1(n;t)$ and $\theta_0(n;n-l)$ for fixed t and l in time $O(n^2) + O(n) = O(n^2)$. In the worst case, we need to do this for $n+1$ different

Algorithm 1 Procedure for computing K_1 and K_0

Let $\mathbf{s}$ be any schedule of n tests;
for $t = 0, \ldots, n$ **do**
    Choose a state vector $\mathbf{x}^t$ with t negative outcomes;
    Compute $Q^{(\mathbf{x}^t, \mathbf{s})}(w, n)$ for each $w \in \{0, \ldots, n\}$ recursively from Equation (6);
    Compute $\theta_1(n; \mathbf{x}^t, \mathbf{s}) = \theta_1(n; t)$ and $\theta_0(n; \mathbf{x}^t, \mathbf{s}) = \theta_0(n; t)$ from Equations (7) and (8);
end for
if $\theta_1(n; n) \geq T$ **then**
    Let K_1 be the smallest $t \in \{0, \ldots, n\}$ with $\theta_1(n; t) \geq T$;
else
    K_1 does not exist;
end if
if $\theta_0(n; 0) \geq T$ **then**
    Let K_0 be the smallest $l \in \{0, \ldots, n\}$ with $\theta_0(n; n - l) \geq T$;
else
    K_0 does not exist;
end if

outcome vectors, for instance $(0, 0, \ldots, 0)$, $(1, 0, \ldots, 0)$, $(1, 1, 0, \ldots, 0)$, $\ldots$, $(1, 1, \ldots, 1)$, leading to an overall time complexity of $O(n^3)$. Algorithm 1 shows the pseudo-code of the procedure to compute the values of K_1 and K_0.

Lemma 7 indicated that if one of K_0 and K_1 is zero then the system diagnosis is unambiguous but not very interesting. Ambiguity might arise, however, when both K_0 and K_1 exist (and hence are non-zero) but $K_0 + K_1 \leq n$. To see this, construct outcome $\mathbf{x}^*$ as follows: $x_1 = x_2 = \ldots = x_{K_1} = 1$ and $x_{K_1+1} = x_{K_1+2} = \ldots = x_n = 0$, and consider policy Π_1 that tests the components in increasing index. Clearly, Π_1 will observe K_1 successes under $\mathbf{x}^*$ while observing 0 failures, so if $K_0 > 0$ then Π_1 indeed concludes to system success. Similarly, if Π_2 conducts tests in decreasing index, it will reach the conclusion of system failure as long as $K_1 > 0$. Thus there exist outcomes $\mathbf{x}^*$ and policies Π_1 and Π_2 such that Π_1 under $\mathbf{x}^*$ will first observe K_1 working components while less than K_0 failing components are identified (and thus conclude that the system functions with the pre-specified confidence threshold T), whereas Π_2 under $\mathbf{x}^*$ will first observe K_0 failing components while less than K_1 working components are encountered (and thus halt the diagnosis with the conclusion of system failure). When $K_0 + K_1 \geq n + 1$, on the other hand, we cannot construct the vector $\mathbf{x}^*$ anymore. In conclusion, when $K_0 + K_1 \geq n + 1$ then the identification of the system state only depends on the test outcomes but not on the sequence in which the tests are executed, while the case $K_0 + K_1 \leq n$ can lead to ambiguous conclusions and would therefore require modifications in the problem statement (e.g., different stopping criteria or different statement of system identification). The following lemma reassures

us that ambiguity will not occur in the setting of Theorem 1, and justifies the restriction to $K_0 + K_1 \geq n + 1$ in Section 5.

Lemma 8. *If $T > 50\%$ and both K_0 and K_1 exist then $K_0 + K_1 \geq n + 1$.*

5 An optimal algorithm

In this section, we show that the testing problem with imperfect tests is polynomially solvable, and that a globally optimal solution for some special cases can be represented by a permutation of all the components. We investigate two different settings, namely the case where either K_0 or K_1 does not exist and the case where they both exist. In Section 4.2 we observed already that when neither K_0 nor K_1 exists, sequencing decisions are irrelevant.

5.1 Exactly one of K_0 and K_1 exists

Consider the *conservative k-out-of-n testing problem*, which is defined similarly as traditional k-out-of-n testing (with perfect tests), except that we perform tests either until we have observed k tests with negative outcome, or we have performed all n tests (Hellerstein et al., 2011). In particular, the diagnosis is not interrupted after observing $n-k+1$ tests with positive outcome. We have the following result:

Observation 1. *If exactly one of the parameters K_0 and K_1 exists then the k-out-of-n testing problem with imperfect tests as defined in Section 3 reduces to the conservative k-out-of-n testing problem.*

This is an immediate consequence of Theorem 1. More precisely, when only K_1 exists then an optimal policy for an instance of the k-out-of-n testing problem with imperfect tests corresponds exactly to an optimal policy for a corresponding instance of the conservative K_1-out-of-n testing problem with the χ_i-values as success probabilities. Such policy either concludes a working system with confidence level at least T after observing K_1 negative outcomes, or it tests all components and is inconclusive. When only K_0 exists then an optimal policy for an instance of the k-out-of-n testing problem with imperfect tests is constructed from an optimal policy for K_0-out-of-n conservative testing with values λ_i as success probabilities, by interchanging every pair of child nodes at every parent node in its BDT representation.

Hellerstein et al. (2011) mention the following result for conservative k-out-of-n testing, which follows from a more general result obtained in Boros and Ünlüyurt (1999).

Proposition 2. *For conservative k-out-of-n testing with perfect tests, an elementary policy represented by the permutation of all n components arranged in non-decreasing order of c_i/p_i is a globally optimal (elementary) policy.*

From Observation 1 and Proposition 2, we immediately obtain the following result.

Corollary 2. *When exactly one of the parameters K_0 and K_1 does not exist then the imperfect k-out-of-n testing problem is polynomially solvable, and an optimal policy can be stored in $O(n)$ space.*

5.2 Both K_0 and K_1 exist

Chang et al. (1990) propose a polynomial-time algorithm for the classic k-out-of-n (perfect) testing problem. We use their algorithm in Section 5.2.2 as a subroutine for solving our problem. For the sake of completeness, we first summarize this algorithm below in Section 5.2.1.

5.2.1 The algorithm of Chang et al.

Consider an instance of the k-out-of-n testing problem with perfect tests. We relabel the components such that:

$$\frac{c_1}{p_1} \leq \frac{c_2}{p_2} \leq \cdots \leq \frac{c_n}{p_n},$$

and we determine a permutation σ such that:

$$\frac{c_{\sigma(1)}}{q_{\sigma(1)}} \leq \frac{c_{\sigma(2)}}{q_{\sigma(2)}} \leq \cdots \leq \frac{c_{\sigma(n)}}{q_{\sigma(n)}}.$$

For any $U \subseteq N$, define $V_i(U)$ to be the subset of U containing the i components with smallest index, so $V_i(U) = \{j \in U | \, 1 \leq j \leq i\}$. Also, let $F_i(U)$ be the subset of components in U that occupy the first i positions in σ: $F_i(U) = \{\sigma(j) \in U | \, 1 \leq j \leq i\}$. Finally, we define $SS(U)$ as the component in U with the smallest index.

The following procedure, described by Chang et al. (1990), produces a globally optimal policy for this problem. We start with testing component $i = SS(V_k(N) \cap F_{n-k+1}(N))$. If component i is working then we choose component $SS(V_{k-1}(N \setminus \{i\}) \cap V_{n-k+1}(N \setminus \{i\}))$ to test; otherwise we select component $SS(V_k(N \setminus \{i\}) \cap F_{n-k}(N \setminus \{i\}))$ as the next component to be examined. Following this method recursively, we continue the test procedure until either k working components or $n - k + 1$ failing components are found.

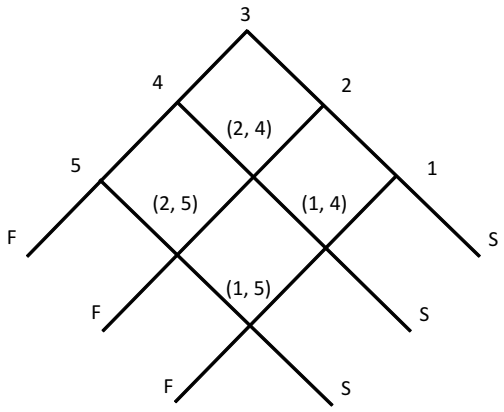

Figure 2: The BW representation of a globally optimal policy for the 3-out-of-5 example instance of Section 3.3 with $\epsilon_0 = \epsilon_1 = 0$

The BDT representation of an optimal policy generated by the above procedure can be simplified to a so-called *block-walking* (BW) representation. See Figure 2 for an illustration, which represents a globally optimal policy for the example instance with perfect tests. Testing proceeds from top to bottom along the grid. The line crossings on the outside of the grid are labeled by the index of the component to be tested, the nodes at the bottom represent the identification of the state of the system, and the inner crossings are marked by a pair of components, the first one of which will be tested if the previously tested component is working and the second one will be tested if the previously tested component is not working. Similarly as for the BDT representation, if a component is found to be working then the right downward line segment is taken, otherwise the left segment is selected. A testing policy that has this structure is referred to as a BW policy.

5.2.2 Generalized testing

We define a new generalized testing problem (with perfect tests) called (k_0, k_1)-out-of-n testing problem, defined similarly as the classic k-out-of-n testing problem in that an instance is defined by a component set N with parameters c_i and p_i, but instead of parameter k, a generalized testing instance takes two parameters k_1 and k_0 and the system diagnosis continues until either k_1 working components or k_0 failing components are found, or until all tests are performed without reaching any of the two previous conditions. The system is therefore in one of three states; in order of the foregoing three possible diagnosis outcomes: *working*, *failing* or *inconclusive*, respectively. Clearly, the classic k-out-of-n testing problem is a special case of (k_0, k_1)-out-of-n testing where $k_1 = k$ and $k_0 = n - k + 1$, and the inconclusive system state never occurs. Sim-

ilarly, the conservative k-out-of-n testing problem is a subproblem of the generalized problem with $k_1 = k$ and $k_0 = n$. We now turn back to k-out-of-n testing with imperfect tests. The following result holds.

Proposition 3. *If both K_0 and K_1 exist then the k-out-of-n testing problem with imperfect tests as defined in Section 3 is equivalent to the (k_0, k_1)-out-of-n testing problem with perfect tests.*

The result in Proposition 3 follows immediately from Theorem 1 after setting $k_1 = K_1$ and $k_0 = K_0$. As a consequence, any policy for the (K_0, K_1)-out-of-n testing problem with perfect tests can also be interpreted as a policy for the k-out-of-n testing problem with imperfect tests (assuming that K_0 and K_1 exist). The generalized testing problem with $k_0 + k_1 = n + 1$ is equivalent with classic (perfect) k-out-of-n testing, and so is polynomially solvable by the algorithm of Chang et al. described in Section 5.2.1. Notice that in this case the system is never in the inconclusive state. The following theorem contains a generalization of this observation. We will not consider $k_0 + k_1 \leq n$ in this text because this situation exhibits similar ambiguities as those mentioned for the case $K_0 + K_1 \leq n$ in Section 4.2.

Theorem 2. *The (k_0, k_1)-out-of-n testing problem with perfect tests is polynomially solvable when $k_0 + k_1 \geq n + 1$. An optimal policy can be stored in $O(n^2)$ space.*

The idea of the proof of Theorem 2 is to show that the (k_0, k_1)-out-of-n testing problem reduces to the classic k-out-of-n testing problem. In the reduction (see the proof for details) one adds $k_0 + k_1 - n - 1$ dummy components to the original n components. By construction, in any optimal policy, the dummy components are only tested after all original components. By removing the dummy components from such policies, a policy for the original (k_0, k_1)-out-of-n testing problem is obtained. Such a policy is also a feasible solution to the equivalent k-out-of-n testing problem with imperfect tests. We refer to a policy for the generalized testing problem that is obtained by removing the dummy components from a BW policy as a *generalized block-walking* (GBW) policy. We also use the name GBW policy for the corresponding policy for the equivalent imperfect testing problem.

The following result ensues directly from Proposition 3, Lemma 8, Theorem 2 and the previous discussion.

Corollary 3. *Consider the imperfect k-out-of-n testing problem as defined in Section 3 and assume that K_0 and K_1 both exist. If $\epsilon_0 + \epsilon_1 < 1$ and $T > 50\%$, then the problem is polynomially solvable. An optimal GBW policy can be stored in $O(n^2)$ space.*

Algorithm 2 Optimal algorithm for the k-out-of-n testing problem with imperfect tests

For each component i, compute χ_i via Equation (2);
Compute K_1 and K_0 by Algorithm 1;
if K_1 and K_0 do not exist **then**
    Return (every policy is always inconclusive);
else if K_0 does not exist **then**
    Test the components in non-decreasing order of c_i/χ_i until either K_1 negative test outcomes are observed (in which case the system works with confidence T), or all components are tested (in which case testing is inconclusive);
else if K_1 does not exist **then**
    Test the components in non-decreasing order of $c_i/(1-\chi_i)$ until either K_0 positive test outcomes are observed (in which case the system fails with confidence T), or all components are tested (in which case testing is inconclusive);
else
    Solve the (K_0, K_1)-out-of-n testing problem for perfect tests with value χ_i as the probability that component i works and c_i as its testing cost;
end if

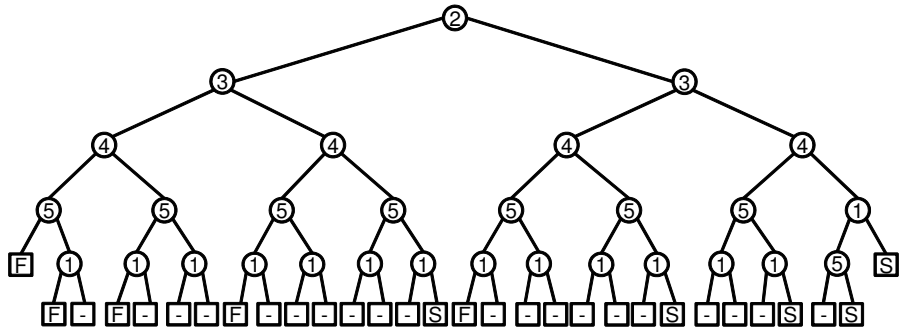

Figure 3: Another globally optimal policy for the example instance in Section 3.3

An overview of the optimal algorithm is presented in the pseudo-code of Algorithm 2, assuming $\epsilon_0 + \epsilon_1 < 1$ and $T > 50\%$.

The class of GBW policies combines the advantages of dominant and of elementary policies, namely global optimality (Lemma 1) and compact representation (polynomial in n). We note that the class of GBW policies is a subset of the class of dominant policies, but the reverse is not true. For the example instance, the dominant policy depicted in Figure 1 is not a GBW policy. Figure 3 gives another globally optimal dominant policy that does belong to the GBW class, and its compact representation is shown in Figure 4. As before, the bottom end points of the edges in the grid are labeled by one of the three symbols S, F and $-$, which indicate the system is working, not working or the diagnosis is inconclusive, respectively (for the specified confidence threshold).

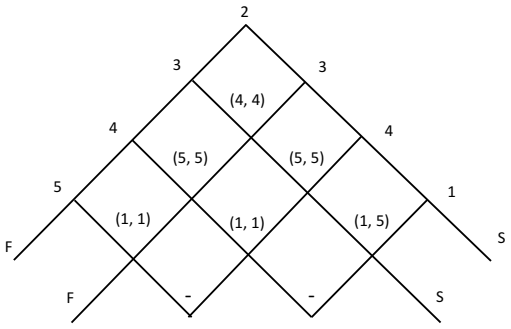

Figure 4: The GBW representation of the globally optimal dominant policy in Figure 3

5.2.3 Counterexample for global optimality of elementary policies

Elementary policies have the benefit of compact representation, but they may 'miss' the global optimum. For the example instance of Section 3.3 with $T = 95\%$, $\epsilon_1 = 10\%$ and $\epsilon_0 = 5\%$, we have $K_0 = K_1 = 4$ by Equations (10) and (11). The globally optimal GBW policy depicted in Figure 4 has expected total costs $4.875 - 0.25\delta$, while the optimal elementary policy characterized by list $(1, 2, 3, 4, 5)$ has expected costs 4.875, which is strictly higher than the global optimum.

6 Summary and outlook on future work

In this article, we have studied the problem of diagnosing k-out-of-n systems when the tests are imperfect. When positive and negative predictive values are the same for all components and with reasonable restrictions on the values of the parameters, this problem can be transformed into a generalized testing problem with perfect tests. We describe a polynomial-time algorithm for the generalized testing problem with perfect tests, which implies that the k-out-of-n test sequencing problem with imperfect tests is also polynomial (again given specific assumptions on the parameters). We examine different policy classes for the imperfect testing problem, namely the class of dominant policies, of elementary policies and of GBW policies. The class of dominant policies always contains a global optimum, while elementary policies are compact in representation (polynomial in the number of components). GBW policies have the merits of both global optimality and of compactness.

For further research, it may be interesting to take retesting into consideration: finished products that do not pass the quality inspection (they have a positive test outcome), for instance, may be retested to reduce the scrapping cost. Accepted components may also be retested to improve outgoing quality. More generally, performing a component test more than once will

allow the tester to obtain more information about the system under study. This option was not explored in this article because it requires additional modeling assumptions. A second option for further work to pursue is a different interpretation of imperfect testing, in which the outcome of a particular test might also be unknown or indeterminate instead of simply wrong. This setting has already received some attention in recent literature, see for instance Balakrishnan and Semmelbauer (1999).

Appendix

Proof of Lemma 1: Consider the BDT of a globally optimal policy Π^* and suppose that there are two vertices u_1 and u_2 in the tree that have the same tested component set (as in the definition of dominant policies) but with different subtrees rooted from u_1 and u_2, so that Π^* is not dominant. By replacing u_1 together with its subtree by u_2 together with its subtree we obtain a different policy Π_1, and similarly we can also replace u_2 and subtree by u_1 and subtree to obtain Π_2. The objective function value of either Π_1 or Π_2 will be at most as high as that of Π^*. By retaining the best policy of either Π_1 or Π_2 and applying similar changes until a dominant policy is obtained, we arrive at a dominant globally optimal policy. The confidence threshold will also be reached in exactly the same leaf nodes because the confidence depends only on the number of conducted tests and the number of positive outcomes (see the discussion in Section 4.2). □

Proof of Lemma 2: There are t components whose outcomes are negative and $(r-t)$ components with positive outcomes. We first focus on the case $g \geq t$: there are at least as many components (g) working as there were negative observations, so at least $g-t$ tests gave a wrong outcome. Therefore,

$$\begin{aligned}
\dot{Q}(g;t,r) &= \Pr\{g \text{ components working} \mid t \text{ negative outcomes, } r \text{ tested }\} \\
&= \sum_{l=0}^{\min\{r-g;t\}} \Pr\{g-t+l \text{ false positive, } l \text{ false negative} \mid t \text{ negative, } r-t \text{ positive }\} \\
&= \sum_{l=0}^{\min\{r-g;t\}} \Pr\{g-t+l \text{ false positive} \mid r-t \text{ positive }\} \cdot \Pr\{l \text{ false negative} \mid t \text{ negative }\}.
\end{aligned}$$

The summation index l represents the number of incorrect negative outcomes, which cannot be more than the total number t of negative outcomes, and it can also not exceed the number $r-g$ of

failing components. Based on our definitions, each positive outcome is actually an independent Bernoulli trial with probability ϵ_0 of being a false positive, and so the total number of false positive observations out of $(r-t)$ follows a Binomial distribution. Similarly, the number of false negative observations is Binomial with t experiments and probability ϵ_1. Consequently,

$$\Pr\{g-t+l \text{ false positive} \mid r-t \text{ positive}\} = \binom{r-t}{g-t+l}\epsilon_0^{g-t+l}(1-\epsilon_0)^{r-g-l}$$

and

$$\Pr\{l \text{ false negative} \mid t \text{ negative}\} = \binom{t}{l}\epsilon_1^{l}(1-\epsilon_1)^{t-l}.$$

This leads to the first term in Equation (9).

We now turn to the case $g < t$, where there are at least $t-g$ false negative outcomes. In line with the derivation above, we obtain

$$\dot{Q}(g;t,r) = \sum_{l=0}^{\min\{g;r-t\}} \Pr\{l \text{ false positive} \mid r-t \text{ positive}\}\cdot\Pr\{t-g+l \text{ false negative} \mid t \text{ negative}\},$$

where the counter l in the summation now represents the number of false positive outcomes. □

Proof of Lemma 3: We wish to establish that $\theta_1(r+1) \geq \theta_1(r)$ for $0 \leq r \leq n-1$. If $0 \leq r < k$ then $\theta_1(r) = 0$, so $\theta_1(r+1) \geq \theta_1(r)$ is always true. If $k \leq r \leq n-1$, then using Equation (7) we have

$$\theta_1(r+1) = \sum_{w=k}^{r+1} Q(w, r+1).$$

Substituting $Q(w, r+1)$ by $Q(w,r)$ and $Q(w-1,r)$ according to Equation (6), we obtain

$$\begin{aligned}\theta_1(r+1) = &\sum_{w=k}^{r+1} Q(w,r)(x_{s_{r+1}}\epsilon_1 + (1-x_{s_{r+1}})(1-\epsilon_0)) \\ &+ \sum_{w=k}^{r+1} Q(w-1,r)(x_{s_{r+1}}(1-\epsilon_1) + (1-x_{s_{r+1}})\epsilon_0).\end{aligned}$$

Taking $Q(r+1,r)$ and $Q(k-1,r)$ out of the first and second summation, respectively, we have

$$\begin{aligned}\theta_1(r+1) = &Q(r+1,r)(x_{s_{r+1}}\epsilon_1 + (1-x_{s_{r+1}})(1-\epsilon_0)) + \sum_{w=k}^{r} Q(w,r)(x_{s_{r+1}}\epsilon_1 + (1-x_{s_{r+1}})(1-\epsilon_0)) \\ &+ Q(k-1,r)(x_{s_{r+1}}(1-\epsilon_1) + (1-x_{s_{r+1}})\epsilon_0) + \sum_{w=k}^{r} Q(w,r)(x_{s_{r+1}}(1-\epsilon_1) + (1-x_{s_{r+1}})\epsilon_0).\end{aligned}$$

With $Q(r+1,r)=0$, this reduces to

$$\begin{aligned}\theta_1(r+1) &= Q(k-1,r)(x_{s_{r+1}}(1-\epsilon_1)+(1-x_{s_{r+1}})\epsilon_0)+\sum_{w=k}^{r} Q(w,r)\\ &= Q(k-1,r)(x_{s_{r+1}}(1-\epsilon_1)+(1-x_{s_{r+1}})\epsilon_0)+\theta_1(r) \qquad (12)\\ &\geq \theta_1(r).\end{aligned}$$

We conclude that $\theta_1(r)$ increases with r.

Via analogous reasoning, we infer that

$$\theta_0(r+1)=\theta_0(r)+Q(n-k,r)(x_{s_{r+1}}\epsilon_1+(1-x_{s_{r+1}})(1-\epsilon_0)). \qquad (13)$$

From this, the result for θ_0 follows. □

Proof of Lemma 4: Using Equations (7) and (8) and Proposition 1, we have

$$\theta_1(r;t)+\theta_0(r;t)=\sum_{g=k}^{r}\dot{Q}(g;t,r)+\sum_{g=0}^{(k-1)+(r-n)}\dot{Q}(g;t,r).$$

It then follows that

$$\theta_1(r;t)+\theta_0(r;t)\leq\sum_{g=0}^{n}\dot{Q}(g;t,r)\leq\sum_{g=0}^{n}\dot{Q}(g;t,n)=1.$$

The first inequality holds because $r\leq n$ and so the summation index g does not necessarily take all values in $\{0,1,2,\ldots,n\}$. The second inequality follows from Proposition 1 and Lemma 3. It is easy to verify that when $r=n$ (all tests are applied), $\theta_1(r;t)+\theta_0(r;t)=1$ holds regardless of the value of t. □

Proof of Lemma 5: It suffices to prove that $\theta_1(r;t)\leq\theta_1(r;t+1)$ holds when $\epsilon_1+\epsilon_0<1$ and $0\leq t<r$. Let $\mathbf{x}$ be any $(r-1)$-dimensional outcome vector with $\sum_{i=1}^{r-1}x_i=t$. We construct two new vectors $\mathbf{x}'=(\mathbf{x},0)$ and $\mathbf{x}''=(\mathbf{x},1)$ and a schedule $\mathbf{s}=(1,2,\ldots,r-1)$ and $\mathbf{s}'=(1,2,\ldots,r)$.

From Eq. (12) and with explicit mention of the arguments of θ_1 and Q as in Sections 3.3 and 4.1, we have

$$\theta_1(r,\mathbf{x}',\mathbf{s}')=Q^{(\mathbf{x},\mathbf{s})}(k-1,r-1)\epsilon_0+\theta_1(r-1,\mathbf{x},\mathbf{s})$$

and

$$\theta_1(r, \mathbf{x}'', \mathbf{s}'') = Q^{(\mathbf{x},\mathbf{s})}(k-1, r-1)(1-\epsilon_1) + \theta_1(r-1, \mathbf{x}, \mathbf{s}).$$

Since $\epsilon_0 < 1-\epsilon_1$, we have $\theta_1(r, \mathbf{x}', \mathbf{s}') \leq \theta_1(r, \mathbf{x}'', \mathbf{s}'')$. Using the redefinition of θ_1 in the discussion of Proposition 1, this implies that $\theta_1(r;t) \leq \theta_1(r;t+1)$. □

Proof of Lemma 6: Analogous to the proof of Lemma 5, based on Eq. (13) instead of (12). □

Proof of Lemma 7: If $K_1 = 0$ then $\theta_1(n;0) \geq T$, and $\theta_0(n;0) = 1 - \theta_1(n;0) \leq 1 - T$. Since $T > 0.5$, it holds that $1 - T < T$; therefore $\theta_0(n;0) < T$, and so K_0 does not exist.

The same reasoning can be set up for the case where $K_0 = 0$. □

Proof of Theorem 1: We provide the proof for K_1, the result for K_0 can be verified analogously. In the first paragraph we show the dominance result for K_1 and in the second paragraph we show that it is tight.

Consider a sequence of tests of all n components, and for each $r \leq n$ let $K(r)$ denote the number of negative test outcomes obtained after the first r tests. Clearly, $K(r)$ is non-decreasing with r. Let r^* be the lowest number of conducted tests such that $K(r^*) = K_1$ (we assume that r^* exists). If $\theta_1(r^*; K_1) \geq T$ then the system works with confidence level at least T, in which case it is a dominant decision to stop testing after r^* tests with K_1 negative outcomes. Now assume that $\theta_1(r^*; K_1) < T$. Then, by the definition of K_1 and by Lemma 5, it follows that the threshold is still reached after performing all n tests, whatever the outcome of the tests after test r^*, because $\theta_1(n; K(n)) \geq \theta_1(n; K_1) \geq T$. Therefore, it is a dominant decision to stop the testing procedure after r^* tests since the threshold for a working system will eventually be reached anyway. It remains to show that the threshold will not be reached earlier for θ_0 (from Corollary 1 we already know that the threshold cannot be reached simultaneously). To see this, let $\bar{r}$ be the smallest number of tests with $\theta_1(\bar{r}; K(\bar{r})) \geq T$ (with $r^* < \bar{r} \leq n$). Then for any $r < \bar{r}$, it holds that $\theta_0(r; K(r)) \leq \theta_0(\bar{r}; K(\bar{r})) \leq 1 - \theta_1(\bar{r}; K(\bar{r})) \leq 1 - T < T$, where the first two inequalities follow from Lemma 3 and Lemma 4, respectively.

To show that K_1 is tight, assume by contradiction that the threshold is reached before observing K_1 negative outcomes. Then, with the same notation as in the previous paragraph, $\theta_1(r'; K(r')) \geq T$ for some test r' with $r' < r^* \leq n$ and $K(r') < K_1$. But by Lemma 3 and the definition of K_1, we find that $\theta_1(r'; K(r')) \leq \theta_1(n; K(r')) < T$, which is not possible. □

Proof of Lemma 8: If K_1 and K_0 both exist then both values are greater than zero (Lemma 7), and by definition

$$\theta_1(n; K_1) \geq T \qquad \text{and} \qquad \theta_0(n; n - K_0) \geq T.$$

Suppose (reductio ad absurdum) that $K_1 + K_0 \leq n$; then $K_1 \leq n - K_0$, implying $\theta_1(n; n - K_0) \geq \theta_1(n; K_1) \geq T$. Then $\theta_0(n; n - K_0) + \theta_1(n; n - K_0) \geq 2T > 1$ since $T > 0.5$, which contradicts Lemma 4. □

Proof of Theorem 2: For an arbitrary instance of generalized testing with $k_0 + k_1 \geq n + 1$, we construct an instance of classic k-out-of-n' testing, as follows. The component set includes all components of the generalized instance but we add $(k_0 + k_1 - n - 1) \geq 0$ identical dummy components to the system, resulting in a system with $n' = k_0 + k_1 - 1$ components in total, and with parameter $k = k_1$. The constructed k-out-of-n' system works if at least $k = k_1$ components work and fails if at least $n' - k + 1 = k_0$ components fail. We wish to make sure that the costs of the dummy components are very high such that the dummy components have the highest ratios of both c/p and c/q; each cost can for instance be chosen as $\max\{c_n/p_n; c_{\sigma(n)}/q_{\sigma(n)}\}$ and the probability of success as 0.5. In this way, the dummy components will not be tested until all the ordinary components (inherited from the generalized testing instance) have been inspected, and they will always be sequenced last in the diagnosis procedure. Therefore, the probability that dummy components will be tested does not depend on the sequence of the ordinary components, such that the contribution of testing all dummy components to the objective function can be calculated independently and remains the same in any optimal policy.

We apply the algorithm of Chang et al. (1990) to thus constructed k_1-out-of-$(k_0 + k_1 - 1)$ system to obtain an optimal BW policy. Removing the dummy components from this optimal policy then leads to an optimal policy for the original generalized testing instance, where the (k_0, k_1)-out-of-n system is classified as 'inconclusive' in exactly those scenarios that need one or more dummy components to be inspected after all n original components were inspected in the optimal BW policy of the constructed k_1-out-of-$(k_0 + k_1 - 1)$ system. □

References

Ahlswede, R. and Wegener, I. (1987). *Search Problems.* Wiley-Interscience, New York.

Akobeng, A. (2006). Understanding diagnostic tests 2: Likelihood ratios, pre- and post-test probabilities and their use in clinical practice. *Acta Paediatrica*, 96:487–491.

Altman, D. and Bland, J. (1994a). Statistics notes: Diagnostic tests 1: Sensitivity and specificity. *British Medical Journal*, 308:1552.

Altman, D. and Bland, J. (1994b). Statistics notes: Diagnostic tests 2: Predictive values. *British Medical Journal*, 309:102.

Amari, S. V., Pham, H., and Dill, G. (2004). Optimal design of k-out-of-n:G subsystems subjected to imperfect fault-coverage. *IEEE Transactions on Reliability*, 53(4):567–575.

Arnold, T. (1973). The concept of coverage and its effect on the reliability model of a repairable system. *IEEE Transactions on Computers*, 22:251–254.

Balakrishnan, A. and Semmelbauer, T. (1999). Circuit diagnosis support system for electronics asssembly operations. *Decision Support Systems*, 25(4):251–269.

Bellman, R. (1957). *Dynamic Programming*. Princeton University Press, Princeton.

Ben-Dov, Y. (1981). Optimal testing procedures for special structures of coherent systems. *Management Science*, 27(12):1410–1420.

Benjamini, Y. and Hochberg, Y. (1995). Controlling the false discovery rate – a new and powerful approach to multiple testing. *Journal of the Royal Statistical Society B*, 57:289–300.

Benjamini, Y. and Hochberg, Y. (2000). On the adaptive control of the false discovery rate in multiple testing with independent statistics. *Journal of Educational and Behavioral Statistics*, 25(1):60–83.

Boros, E. and Ünlüyurt, T. (1999). Diagnosing double regular systems. *Annals of Mathematics and Artificial Intelligence*, 26:171–191.

Butterworth, R. (1972). Some reliability fault-testing models. *Operations Research*, 20(2):335–343.

Chang, M.-F., Shi, W., and Fuchs, W. (1990). Optimal diagnosis procedures for k-out-of-n structures. *IEEE Transactions on Computers*, 39(4):559–564.

Cramer, E. and Kamps, U. (1996). Sequential order statistics and k-out-of-n systems with sequentially adjusted failure rates. *Annals of the Institute of Statistical Mathematics*, 48(3):535–549.

De Reyck, B. and Leus, R. (2008). R&D-project scheduling when activities may fail. *IIE Transactions*, 40(4):367–384.

Ding, J. and Gong, L. (2008). The effect of testing equipment shift on optimal decisions in a repetitive testing process. *European Journal of Operational Research*, 186(1):330–350.

Ding, J., Greenberg, B., and Matsuo, H. (1998). Repetitive testing strategies when the testing process is imperfect. *Management Science*, 44(10):1367–1378.

Ding, Y., Zuo, M. J., Lisnianski, A., and Li, W. (2010). A framework for reliability approximation of multi-state weighted k-out-of-n systems. *IEEE Transactions on Reliability*, 59(2):297–308.

Eryilmaz, S. (2013). On reliability analysis of a k-out-of-n system with components having random weights. *Reliability Engineering and System Safety*, 109:41–44.

Gluss, B. (1959). An optimum policy for detecting a fault in a complex system. *Operations Research*, 7(4):468–477.

Greenberg, B. and Stokes, S. (1995). Repetitive testing in the presence of inspection errors. *Technometrics*, 37:102–111.

Hellerstein, L., Özkan, O., and Sellie, L. (2011). Max-throughput for (conservative) k-of-n testing. In *Algorithms and Computation*, volume 7074, pages 703–713. Springer Berlin Heidelberg.

Igelmund, G. and Radermacher, F. (1983). Preselective strategies for the optimization of stochastic project networks under resource constraints. *Networks*, 13:1–28.

Kelly, F. P. (1982). A remark on search and sequencing problems. *Mathematics of Operations Research*, 7(1):154–157.

Luo, X., Dong, M., and Huang, Y. (2006). On distributed fault-tolerant detection in wireless sensor networks. *IEEE Transactions on Computers*, 55(1):58–70.

Marseguerra, M., Zio, E., Podofillini, L., and Coit, D. (2005). Optimal design of reliable network systems in presence of uncertainty. *IEEE Transactions on Reliability*, 54(2):243–253.

Nachlas, J., Loney, S., and Binney, B. (1990). Diagnostic-strategy selection for series systems. *IEEE Transactions on Reliability*, 39(3):273–280.

Pinedo, M. (2012). *Scheduling. Theory, Algorithms, and Systems.* Springer.

Quinino, R., Colin, E., and Ho, L. (2010). Diagnostic errors and repetitive sequential classifications in on-line process control by attributes. *European Journal of Operational Research*, 201(1):231–238.

Raghavan, V., Shakeri, M., and Pattipati, K. (1999). Test sequencing algorithms with unreliable tests. *IEEE Transactions on Systems, Man and Cybernetics - Part A: Systems and Humans*, 29(4):347–357.

Raouf, A., Jain, J., and Sathe, P. (1983). A cost-minimization model for multicharacteristic component inspection. *IIE Transactions*, 15:187–194.

Ruan, S., Y. Zhou, F. Y., Pattipati, K. R., Willett, P., and Patterson-Hine, A. (2009). Dynamic multiple-fault diagnosis with imperfect tests. *IEEE Transactions on Systems, Man and Cybernetics - Part A: Systems and Humans*, 39(6):1224–1236.

Salloum, S. (1979). *Optimal testing algorithms for symmetric coherent systems.* PhD thesis, University of Southern California.

Salloum, S. and Breuer, M. (1997). Fast optimal diagnosis procedures for k-out-of-n:G systems. *IEEE Transactions on Reliability*, 46(2):283–290.

Sarma, A. and Tufts, D. (2001). Robust adaptive threshold for control of false alarms. *IEEE Signal Processing Letters*, 8(9):261–263.

Schmidt, J. W. and Bennett, G. K. (1972). Economic multiattribute acceptance sample. *AIIE Transactions*, 4(3):194–199.

Shakeri, M., Raghavan, V., Pattipati, K. R., and Patterson-Hine, A. (2000). Sequential testing algorithms for multiple fault diagnosis. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 30(1):1–14.

Song, N. and Teneketzis, D. (2004). Discrete search with multiple sensors. *Mathematical Methods of Operations Research*, 60(1):1–13.

Stone, L., Stanshine, J., and Persinger, C. (1972). Optimal search in the presence of Poisson-distributed false targets. *SIAM Journal on Applied Mathematics*, 23(1):6–27.

Tang, K. and Tang, J. (1994). Design of screening procedures: A review. *Journal of Quality Technology*, 26(3):209–226.

Tzimerman, A. and Herer, Y. T. (2009). Off-line inspections under inspection errors. *IIE Transactions*, 41:626–641.

Ünlüyurt, T. (2004). Sequential testing of complex systems: A review. *Discrete Applied Mathematics*, 142:189–205.

Wagner, B. and Davis, D. (2001). Discrete sequential search with group activities. *Decision Sciences*, 32(4):557–573.

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

Wei, W., Coolen, K., and Leus, R. (2013). Sequential testing policies for complex systems under precedence constraints. *Expert Systems with Applications*, 40:611–620.

Wu, J. and Chen, R. (1994). An algorithm for computing the reliability of weighted-k-out-of-n systems. *IEEE Transactions on Reliability*, 43(2):3276–328.