Importance Sampling of Test Cases in Markovian Software Usage Models

Walter J. Gutjahr
Department of Statistics, Operations Research and Computer Science
University of Vienna

Abstract: Recently, some authors have suggested usage models of Markov type as a technique of specifying the estimated operational use distribution of a given program. A main purpose of such models is the derivation of random test cases allowing unbiased estimates on the (un)reliability of the program in its intended environment. In this article, we show that by a shift of the transition probabilities of the Markov chain corresponding to such a model, prior information on the error–proneness of single program operations can be taken into account. An unbiased unreliability estimator with reduced variance is obtained. Furthermore, it is shown that minimization of the variance leads to a special stochastic optimization problem which can be demonstrated to be convex, such that efficient solution techniques apply. Some related questions are also treated in a more general, non–Markovian framework.

Keywords: Software usage models, Markov chains, software reliability, software testing, importance sampling, stochastic optimization.

1 Introduction

In the last years, a growing interest in software usage modeling by means of Markov chain methods can be stated (see, e.g., [15], [16], [17], [13]). A software usage model characterizes the (estimated) distribution of the possible uses of a given program in its intended environment. Such a model is an indispensable prerequisite for "statistical testing", the only testing technique that allows predictions of the operational *reliability* of a given program. Statistical testing in its simplest variant consists in drawing test cases according to the operational use distribution and executing them, such that the frequency of failures during the test leads to unbiased estimates of the probability of failures in the application field. Of course, this approach presupposes that the operational use distribution can be described and generated in some way or other.

In its most primitive form, a (partial) usage model is already given by a table assigning to each program function the corresponding probability of a call in the intended usage environment. (For a completion of the model, it would have to be supplied with a component describing the selection of inputs for each program function; this problem is usually solved pragmatically.) However, the usage structure of most programs is more complicated such that more elaborate models are needed. Especially, the *time structure* of function calls can play an important role. In such cases, an appropriate description of the usage will be in terms of stochastic processes rather than in terms of single discrete random variables. Markov chains are very convenient model processes for a broad spectrum of applications.

A simple Markov usage model is depicted in Fig. 1. For generating a statistical test case based on this model, we have to step through the chain according to the transition probabilities. The sequence of visited states becomes the test case. This generation process can be repeated independently, producing an arbitrary number of test cases. Each test case is to be executed and evaluated, and the relative frequency of successful test executions can be used as an unbiased estimator for the operational reliability of the program.

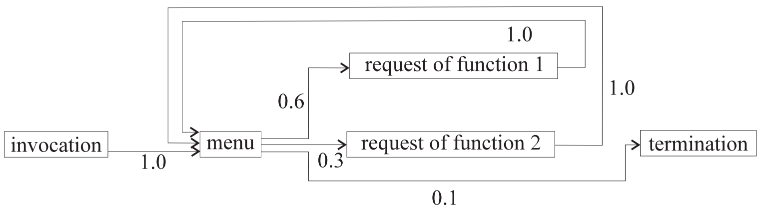

Fig. 1. A Markov chain usage model. The values assigned to the arcs are transition probabilities. The arcs themselves are operations of the program.

A serious disadvantage of this technique has been outlined by Walton, Poore and Trammell [13]: The most frequently used operations are always given the most testing, irrespectively of another important criterion, namely the relative error–proneness of the single operations. Intuitively, one would like to provide more test effort for operations that are expected to be especially error–prone, even if they are not very frequently used.

In [13], three solutions to this problem have been proposed: (1) separate usage models for infrequent, critical operations, (2) "anecdotal testing", (3) adjusting the usage distribution to ensure that critical operations are tested more frequently. However, the authors admit that these solutions do not completely satisfy, since their application implies that the relative failure frequency during test does not yield an unbiased unreliability estimate anymore.

The aim of the present paper is to show that the third suggestion by Walton, Poore and Trammell can be modified in such a way that

- we *do* get unbiased unreliability estimates, and
- taking account of prior knowledge on the error–proneness of each operation can be used for a variance reduction of the unreliability estimator.

The idea is to expose the (operational) transition probabilities in the Markov chain to a probability shift, as suggested in [13], but to compensate this shift afterwards by assigning weights to the test results. Essentially, this is the *importance sampling* technique of Rare Event Simulation. [1] Considering prior estimates of the failure probabilities connected with the operations (arcs) in the Markov chain as given, we shall see that the *optimal* probability shift results as the solution of a certain stochastic optimization problem. It will be shown that this problem has a unique solution and can be efficiently solved.

We start by developing, as a framework, a more general theory of optimal test case sampling from denumerable input domains (section 2). After that, the attention is restricted to the Markov chain situation (sections 3 – 5). Section 3 presents the formal details of reliability estimation by means of Markov chain usage models. Section 4 contains our main results, and in section 5, an extension of the method to a situation where the trajectories of the Markov chain have to be considered as "inhomogenous" with respect to their failure behavior, is outlined.

2 Minimum variance estimators for program unreliability

The following definitions have been taken from [4] and are explained there in more detail. Let D denote the *input domain* of a program π. By P, we denote the *operational distribution* of π, i.e., the probability distribution on D representing the use of π in real applications. Moreover, we consider a family $\mathcal{F}$ of possible *functionalities* of π. A functionality $F \in \mathcal{F}$ specifies a particular input/output behavior of a program. Note that the functionality F of a given program is in general unknown, at least before testing (otherwise testing would be unnecessary). So it is appropriate to model F by a *random variable*. The (hypothetical) distribution of F reflects the tester's prior information on the behavior of the given program π. We call it the *a–priori–distribution* on $\mathcal{F}$ and denote it by Q. To reassure the reader, we mention that a full description of this distribution will not be required; it suffices that certain parameters of Q can be estimated.

Let us emphasize the fact that we use two different "probability mechanisms": one of them, P, operates on D, the other, Q, on $\mathcal{F}$. (In our application of importance sampling, the probability shift will only be applied to the measure P.)

Next, we set

$$c(x,F) := \begin{cases} 0, & \text{if input } x \text{ is processed correctly by a program with functionality } F, \\ 1, & \text{otherwise.} \end{cases}$$

Therein, the decision on correctness is based on the program specification for π, which is considered as given and fixed.

[1] A survey on the application of importance sampling to reliability and queuing models can be found in [7]. The application of importance sampling to continuous–time failure/repair Markov chain models has been investigated in some recent articles, e.g., in [11].

By means of the quantities $c(x, F)$, the (un)reliability of a program can be defined:

Definition 2.1. The *unreliability* of a program with functionality F is given by

$$Z(F) := E_P(c(X, F)),$$

where X is a random input from D, and E_P (acting on X) is the expectation with respect to the operational distribution P. The *reliability* of the program is $1 - Z(F)$.

Since F is unknown before testing, also $Z(F)$ is unknown before testing. By means of the test results, it is possible to *estimate* $Z(F)$. A straightforward unbiased estimator for $Z(F)$ is $c(X, F)$, where X is a test input selected according to the operational distribution. The reader should notice that, although F itself is not known, the value of the (binary) random variable $c(X, F)$ can be derived from the result of the test. Of course, the variance of the estimator $c(X, F)$ can be reduced by repeated, independent sampling of test inputs, i.e., by using $(1/N)\sum_{\nu=1}^{N} c(X^{(\nu)}, F)$ with i.i.d. test inputs $X^{(\nu)} \in D$, $X^{(\nu)} \sim P$, instead of $c(X, F)$. This is the usual approach in reliability testing.

In [4] it has been shown that for finite D, the variance of the unreliability estimator can be further reduced by *importance sampling* based on the a–priori–distribution Q. In the sequel, we shall partially generalize this approach to a denumerable input domain $D = \{x^{(1)}, x^{(2)}, \ldots\}$. (In a certain aspect, our derivation will be more specific than in [4] insofar as we confine ourselves here to constant failure severities, while in [4] failure severities are allowed to vary.)

The basic idea is a *probability shift* for the test distribution: Instead of using the operational distribution P, we sample test inputs according to some appropriately chosen test distribution T on D satisfying $T(\{x\}) > 0$ for each $x \in D$ with $P(\{x\}) > 0$. For abbreviation, set $p(x) := P(\{x\})$ and $t(x) := T(\{x\})$. Now consider the estimator

$$S := \frac{p(X)}{t(X)} \cdot c(X, F), \tag{1}$$

where X is selected from D according to T. It is immediate that S is an unbiased estimator for $Z(F)$ (cf., e.g., [1], pp. 429 ff, [12], p. 337, or [2], p. 1367).

The function $L(X) := p(X)/t(X)$ is usually called the *likelihood ratio* of P with respect to T, or, in measure–theoretic terms, the *Radon–Nikodym derivative*. The property that $t(x) > 0$ for $p(x) > 0$ is the condition of *absolute continuity*. (Cf. [2], pp. 1379 – 1380.)

Again, the precision of the estimator can be boosted by repeated sampling, as indicated above.

We are interested in finding a test distribution T that yields an estimator S with minimum variance. The optimization problem $var_T(S) \to \min$, however, is not solvable in practice, since its solution would require the knowledge of F. So we consider the optimization problem

$$E_Q(var_T(S)) \to \min \tag{2}$$

instead, i.e., we ask for an unreliability estimator with minimum *expected* variance with respect to Q. The elements of D will be denoted by $x^{(1)}, x^{(2)}, \ldots$.

Proposition 2.1. Let $p(x^{(i)}) > 0$ and $E_Q(c(x^{(i)}, F)) > 0$ $(i = 1, 2, \ldots)$. Then the unique solution of (2) subject to $t(x^{(i)}) > 0$ $(i = 1, 2, \ldots)$ and $\sum_{i=1}^{\infty} t(x^{(i)}) = 1$ is given by

$$t(x^{(i)}) = \frac{p(x^{(i)}) \sqrt{E_Q\left([c(x^{(i)}, F)]^2\right)}}{\sum_{j=1}^{\infty} p(x^{(j)}) \sqrt{E_Q\left([c(x^{(j)}, F)]^2\right)}} \qquad (i = 1, 2, \ldots).$$

Proof. Since $var_T(S) = E_T(S^2) - (E_T(S))^2$, and since $E_T(S) = Z(F)$ does not depend on T, it suffices to minimize $E_Q(E_T(S^2))$. With the abbreviation $c(x) := E_Q([c(x, F)]^2)$, we obtain

$$\begin{aligned} E_Q(E_T(S^2)) &= E_Q(E_T([L(X)]^2 \, [c(X, F)]^2)) = E_Q(E_P(L(X) \, [c(X, F)]^2)) \\ &= E_P(L(X) \, c(X)) = E_P\left(\frac{p(X)}{t(X)} \, c(X)\right). \end{aligned}$$

So we have shown

$$\operatorname{argmin}_T E_Q(var_T(S)) = \operatorname{argmin}_T E_P\left(\frac{p(X)}{t(X)} \, c(X)\right). \tag{3}$$

For the solution of the problem

$$E_P\left(\frac{p(X)}{t(X)} \, c(X)\right) = \sum_i \frac{[p(x^{(i)})]^2}{t(x^{(i)})} \, c(x^{(i)}) \to \min$$

$$\text{such that } \sum_i t(x^{(i)}) = 1,$$

the Lagrange function

$$\mathcal{L} := \sum_i \frac{[p(x^{(i)})]^2}{t(x^{(i)})} \, c(x^{(i)}) + \lambda \left(\sum_i t(x^{(i)}) - 1\right)$$

can be used. Setting the partial derivatives of $\mathcal{L}$ (to the variables $t(x^{(i)})$ and λ) equal to zero yields a system of equations with solution

$$t(x^{(i)}) = \frac{p(x^{(i)}) \sqrt{c(x^{(i)})}}{\sum_j p(x^{(j)}) \sqrt{c(x^{(j)})}} \quad (i = 1, 2, \ldots).$$

□

For later reference, Proposition 2.1 has been formulated in a way where the fact that the $c(x, F)$ are 0–1–variables is not used. Within the context of the present chapter, the optimal test distribution indicated by Proposition 2.1 may also be written in the simpler form

$$t(x^{(i)}) = \frac{p(x^{(i)}) \sqrt{E_Q\left(c(x^{(i)}, F)\right)}}{\sum_{j=1}^{\infty} p(x^{(j)}) \sqrt{E_Q\left(c(x^{(j)}, F)\right)}} \qquad (i = 1, 2, \ldots). \tag{4}$$

Therein, $E_Q(c(x,F))$ is the (prior) probability of input x being error–causing. In applications with finite input domain D, these probabilities can be *estimated* by means of appropriate techniques (see [4]). Then (4) yields a practicable way of constructing optimal reliability tests. For an infinite input domain D with an intricate structure, the practical application of Proposition 2.1 is more difficult. Especially, in the cases of central interest in this article, namely those where the usage of the program is modeled by a Markov chain, Proposition 2.1 is hard to apply, even for relatively simple chains.

Before returning to this point in section 4, let us discuss another important aspect. One could suspect that, although test distributions yielding minimum variance reliability estimators are suitable tools for software reliability evaluation, they might be of limited value for other testing aims such as *error detection.* To refute this suspicion, we show that the optimal test distribution given by Proposition 2.1 has an equal or higher error–detection probability than the operational distribution:

Proposition 2.2. Let Φ_P and Φ_T denote the probability that a test input X selected according to the operational distribution resp. to the distribution (4) causes a program failure:

$$\Phi_P := E_Q(E_P(c(X,F))) \quad \text{and} \quad \Phi_T := E_Q(E_T(c(X,F))),$$

where T is defined by (4). Then $\Phi_T \geq \Phi_P$.

Proof. The proof relies on the stochastic inequation

$$E(C^{a+b}) \;\geq\; E(C^a)\, E(C^b) \qquad (C \geq 0;\; a,b \geq 0) \tag{5}$$

which can easily be derived from Lyapunov's inequation

$$[E(U^\alpha)]^{1/\alpha} \leq [E(U^\beta)]^{1/\beta} \qquad (U \geq 0;\; 0 < \alpha \leq \beta).$$

With $g_i := \sqrt{E_Q\left(c(x^{(i)},F\right))}$, one has

$$E_T(c(X,F)) = \sum_{i=1}^{\infty} t_i\, c(x^{(i)},F) = \frac{\sum_{i=1}^{\infty} p_i\, g_i\, c(x^{(i)},F)}{\sum_{j=1}^{\infty} p_j\, g_j}.$$

Thus, with $c(x) = E_Q(c(x,F)) = E_Q\left([c(x,F)]^2\right)$ as in the proof of Proposition 2.1,

$$\Phi_T = \frac{\sum_{i=1}^{\infty} p_i\, g_i^3}{\sum_{i=1}^{\infty} p_i\, g_i} = \frac{E_P\left([c(X)]^{3/2}\right)}{E_P\left([c(X)]^{1/2}\right)}.$$

On the other hand,

$$\Phi_P = E_P(E_Q(c(X,F))) = E_P(c(X)).$$

Setting $C := c(X)$, $a := 1$ and $b := 1/2$, we obtain from (5):

$$E_P\left([c(X)]^{3/2}\right) \geq E_P(c(X)) \cdot E_P\left([c(X)]^{1/2}\right).$$

This yields the assertion. $\square$

3 Markov usage models

Let us start this section with a formal description of a Markov usage model (for interpretations and examples, see [16], [13]). We suppose that each specific usage of the given program π is described by a trajectory $X = (X_0, X_1, \ldots)$ of a Markov chain G with state space $V = \{1, \ldots, n\}$, where $1, \ldots, n-1$ are transient states and n is an absorbing state (the state "program termination"). The Markov chain G can be represented by its transition graph, a connected directed graph with node set V and arcs (i, j) to which transition probabilities $p(i, j) > 0$ are assigned $(i, j \in V)$. Multiple arcs are not allowed. The transition probabilities satisfy the condition

$$\sum_{j \in Succ(i)} p(i, j) = 1 \qquad (i = 1, \ldots, n),$$

where $Succ(i)$ is the set of successors of node i, i.e., the set of nodes j such that (i, j) is an arc. In order to obtain a complete transition matrix, we define $p(i, j) := 0$ for $j \notin Succ(i)$. Since state n is absorbing, one has $p(n, n) = 1$. Fig. 2 shows an example of a Markov chain satisfying the above conditions.

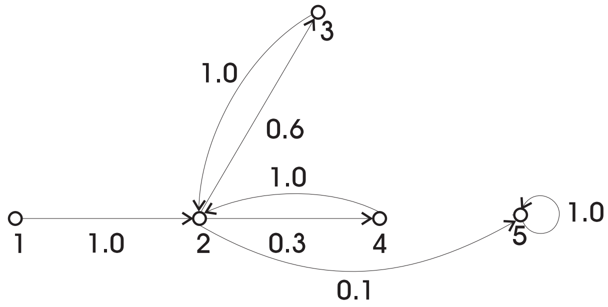

Fig. 2. Transition graph of the usage model of Fig. 1. A transition loop from node 5 to itself has been added in order to produce a complete Markov chain.

Furthermore, in Markov chain usage models it is usually assumed that the Markov chain always starts in state 1 (the state "program invocation"), i.e., $X_0 = 1$. In the ith transition, the arc (X_{i-1}, X_i) of the transition graph is activated.

By P, we denote the probability measure on the set of possible trajectories as defined by the transition probabilities $p(i, j)$. It is supposed that P represents the operational distribution of the trajectories.

In view of the conditions above, the time until absorption,

$$K := \min \{k \mid X_k = n\} = \min \{k \mid X_j = n \text{ for all } j \geq k\},$$

is P–a.s. finite. For the representation of the trajectory X, it is sufficient to indicate the prefix $(X_0, \ldots, X_{K-1})$ of the chain. Neglecting the set (with measure zero) of trajectories where $K = \infty$, we obtain a finite or denumerable set of possible trajectories.

Now let us return to the *reliability* of a program π with assigned Markov chain G. For the major part of our investigations, we confine ourselves to the following assumption: A fixed given trajectory $x = (x_0, x_1, \ldots)$ is either executed correctly by π in each case or executed incorrectly in each case. This is not the general situation, since two different inputs a_1, a_2 for π, where a_1 is correctly processed while a_2 is failure–causing, might nevertheless lead to the same trajectory x. In terms of the notion used by Hamlet and Taylor in [6], our assumption says that the subdomains D_x of the input domain D defined by

$$D_x := \text{ set of inputs leading to trajectory } x$$

are *homogenous*, i.e., they contain either only correctly processed or only failure–causing inputs. (Weyuker and Ostrand [14] call such subdomains *revealing*.) To a certain extent, our analysis also holds for the inhomogenous case (see section 5).

As in section 2, the input domain of a program is denoted by D. On the assumption of homogenity explained above, we may identify trajectories with inputs, such that D is the set of all possible trajectories $x = (x_0, x_1, \ldots)$. Each $x \in D$ may be processed correctly or incorrectly by program π.

The operational distribution P on D is immediately obtained:

$$P\{X = x\} = p(x) = p(x_0, x_1, \ldots) = \prod_{k=1}^{\infty} p(x_{k-1}, x_k). \tag{6}$$

(Since the transition probability $p(.,.)$ always depends on two operands, while the operational probability $p(x)$ depends on the infinite sequence $x = (x_0, x_1, \ldots)$, we may denote both functions without ambiguity by the same symbol p.)

Because of (6), the unreliability of a program with functionality F is given by

$$Z(F) = \sum_{x \in D} \prod_{k=1}^{\infty} p(x_{k-1}, x_k) \cdot c(x, F).$$

4 Probability shifts in Markov chains

In order to estimate $Z(F)$, one could use the straightforward Monte Carlo technique indicated at the beginning of section 2: Sample trajectories $X = (X_0, X_1, \ldots)$, using the operational transition probabilities $p(i, j)$, perform a test for each trajectory, and take the average of the outcomes $c(X, F)$ as an estimate of $Z(F)$. We are again interested in achieving a variance reduction of the estimator by exploiting prior information on the a-priori-distribution Q of F. As in the general framework of section 2, it is not necessary to know the distribution Q as a whole, but only to have estimates of the expected values $E_Q(c(x, F))$. As in the proof of Proposition 2.1, let us set $c(x) := E_Q([c(x, F)]^2) = E_Q(c(x, F))$. The function $c(x)$ represents the (prior) probability that trajectory x leads to a program failure.

Since there are (in general) infinitely many possible trajectories, the tester cannot estimate one $c(x)$ after the other. A general *model* for $c(x)$ is required which allows the reduction of the estimation process to the estimation of a limited set of parameters. In the sequel, we consider the following two natural models:

Model (A) (dynamic model): Each arc (i,j) in the Markov chain has, at each transition where it is activated, a prior probability $f(i,j)$ of failing. Failures of transitions occur independently from each other. The Markov chain fails, if any of its transitions (x_{k-1}, x_k) fails. Then

$$c(x,F) = 1 - \prod_{k=1}^{\infty}(1 - I_k(x)),$$

where $I_k(x)$ are independent Bernoulli variables with $E_Q(I_k(x)) = f(x_{k-1}, x_k)$, such that

$$c(x) = 1 - \prod_{k=1}^{\infty}(1 - f(x_{k-1}, x_k)). \tag{7}$$

We always assume $f(n,n) = 0$, which means that once the final state n is reached, no failure can occur anymore. For $x \neq y$, our assumptions imply that $c(x,F)$ and $c(y,F)$ are independent.

Model (B) (static model): Each arc (i,j) has a certain prior probability $f(i,j)$ of failing at *each* transition where it is activated. Failures of different arcs occur independently from each other. As in model (A), the Markov chain fails, if any of its transitions (x_{k-1}, x_k) fails. Then, denoting by $A(x) = \{(x_0,x_1),(x_1,x_2),(x_2,x_3),\ldots\}$ the set of arcs activated by trajectory x,

$$c(x,F) = 1 - \prod_{(i,j)\in A(x)} (1 - I(i,j)),$$

where $I(i,j)$ are independent Bernoulli variables with $E_Q(I(i,j)) = f(i,j)$, such that

$$c(x) = 1 - \prod_{(i,j)\in A(x)} (1 - f(i,j)). \tag{8}$$

Again, it is reasonable to assume that $f(n,n) = 0$. Here, of course, $c(x,F)$ and $c(y,F)$ are not independent anymore, except if x and y are edge–disjoint.

Let us emphasize that the two models above only decribe the *prior expectation* of the failure behavior of a given trajectory x. In the program test, program π will turn out to behave either correctly or incorrectly along trajectory x, but since we do not know the outcome of the test in advance, we need some model which allows the computation of a (prior) probability $c(x)$ of the failure event.

Now, as in section 2, a test distribution T different from the operational distribution P will be chosen in order to reduce the variance of the reliability estimator. However, we shall not consider the set of *all* possible test distributions, but restrict ourselves to a set of *easily realizable* distributions, and do the optimization within the restricted set. From a practical point of view, it is convenient to choose random trajectories that are realizations of another Markov chain themselves. So all we do is to change the operational transition probabilities $p(i,j)$ to *test transition probabilities* $t(i,j)$. A trajectory $x=(x_0,x_1,\ldots)$ has then the probability

$$P\{X=x\}=t(x):=\prod_{k=1}^{\infty}t(x_{k-1},x_k). \tag{9}$$

The numbers $t(i,j)$ have to satisfy the following constraints:

(i) $t(i,j)\geq 0 \qquad (1\leq i,j\leq n)$,

(ii) $\sum_{j=1}^{n}t(i,j)=1 \qquad (i=1,\ldots,n)$,

(iii) $t(i,j)=0$ iff $p(i,j)=0 \qquad (1\leq i,j\leq n)$,

(iv) $t(n,n)=1$.

By T, we denote the transition matrix $(t(i,j))$ $(1\leq i,j\leq n)$. Since the test distribution is uniquely determined by the transition probabilities $t(i,j)$, this notation is consistent with the use of the symbol T for the test distribution. Let $\mathcal{T}_P$ be the set of all $T=(t(i,j))$ satisfying the constraints (i) – (iv) above.

Inserting (6) and (9) into (1), we obtain (cf. also [7], eq. (11)): For a trajectory X selected according to T,

$$S:=\prod_{k=1}^{\infty}\frac{p(X_{k-1},X_k)}{t(X_{k-1},X_k)}\,c(X,F) \tag{10}$$

is an unbiased estimator for $Z(F)$.

(Observe that $t(X_{k-1},X_k)>0$ for all k holds T–a.s., so S is well–defined, and because of constraint (iii), the condition $T(\{x\})>0$ for $P(\{x\})>0$ is satisfied.)

Now we look for a minimizer of the (expected) variance of S within the set $\mathcal{T}_P$. We restrict ourselves to model (A); model (B) can be treated quite analogously.

From (3), we immediately conclude that under model (A), minimization of $E_Q(var_T(S))$ w.r.t. $T\in\mathcal{T}_P$ is equivalent to the solution of the following stochastic optimization problem:

$$G(T):=E_P\left\{\prod_{k=1}^{\infty}\frac{p(X_{k-1},X_k)}{t(X_{k-1},X_k)}\left[1-\prod_{l=1}^{\infty}(1-f(X_{l-1},X_l))\right]\right\}\to\min \tag{11}$$

such that $T\in\mathcal{T}_P$.

Obviously, the objective function $G(T)$ in (11) is just $E_P(E_Q(S))$.

The solution of (11) would be difficult in practical applications, if $G(T)$ should have many local optimizers. Fortunately, this is not the case: We show that the objective function $G(T)$ is *convex*. This allows a very efficient solution of (11) by means of the well–known Kiefer–Wolfowitz procedure ([8]; see also [10], p. 240, for further references).

Proposition 4.1. On $\mathcal{T}_P \cap \{T \mid G(T) < \infty\}$, the objective function $G(T)$ of (11) is a convex function of the variables $t(i,j)$ $(1 \le i,j \le n)$.

Proof. First, we observe that for arbitrary nonnegative integers $m_1, \ldots, m_n$, the function

$$h(x_1, \ldots, x_n) := x_1^{-m_1} \ldots x_n^{-m_n} \tag{12}$$

is a convex function on $(x_1, \ldots, x_n) \in]0, \infty[^n$. This follows from the fact that

$$\log h(x_1, \ldots, x_n) = -m_1 \log x_1 - \ldots - m_n \log x_n$$

is a convex function: each term $-m_i \log x_i$ is convex, and the sum of convex functions is convex again. So we obtain the log–convexity of h, which implies convexity.

Now let us return to the function $G(T)$. Since $p(X_{k-1}, X_k) > 0$ for all k holds P–a.s., we may restrict ourselves to trajectories X with $p(X_{k-1}, X_k) > 0$ and hence also $t(X_{k-1}, X_k) > 0$ $(k = 1, 2, \ldots)$. Set

$$\Psi(X) := \prod_{k=1}^{\infty} p(X_{k-1}, X_k) \left[1 - \prod_{l=1}^{\infty} (1 - f(X_{l-1}, X_l))\right].$$

The function Ψ does not depend on T, and

$$G(T) = E_P \left(\Psi(X) \cdot \prod_{k=1}^{\infty} [t(X_{k-1}, X_k)]^{-1} \right).$$

For fixed X, let $M(i,j)$ denote the number of transitions k with $(X_{k-1}, X_k) = (i,j)$. For $X \sim P$, $M(i,j)$ becomes a random variable which is P–a.s. finite unless $(i,j) = (n,n)$. So for each fixed X except such in a set with measure zero, the expression

$$\prod_{k=1}^{\infty} [t(X_{k-1}, X_k)]^{-1} = \prod [t(i,j)]^{-M(i,j)}, \tag{13}$$

where the second product is over all (i,j) with $1 \le i,j \le n$, $(i,j) \ne (n,n)$ and $t(i,j) > 0$, is a function of the type (12) in the variables $t_{ij} := t(i,j)$ with $1 \le i,j \le n$, $(i,j) \ne (n,n)$ and $t(i,j) > 0$. Using the observation at the beginning of this proof, we conclude that the r.h.s. of (13) is convex in the considered variables, which means that for $T = (t(i,j))$ and $T' = (t'(i,j))$ with $T, T' \in \mathcal{T}_M$ and $0 < \lambda < 1$,

$$\prod [\lambda t(i,j) + (1-\lambda) t'(i,j)]^{-M(i,j)} \le \lambda \prod [t(i,j)]^{-M(i,j)} + (1-\lambda) \prod [t'(i,j)]^{-M(i,j)}, \tag{14}$$

where the products are over all (i,j) with $1 \le i,j \le n$, $(i,j) \ne (n,n)$ and $p(i,j) > 0$. (Note that $t(i,j) > 0$ iff $p(i,j) > 0$.) Since $\Psi(X) \ge 0$, it follows that

$$\Psi(X) \prod_{k=1}^{\infty} [(\lambda t + (1-\lambda) t')(X_{k-1}, X_k)]^{-1}$$

$$\leq \lambda\, \Psi(X) \prod_{k=1}^{\infty} [t(X_{k-1}, X_k)]^{-1} + (1-\lambda)\, \Psi(X) \prod_{k=1}^{\infty} [t'(X_{k-1}, X_k)]^{-1} \tag{15}$$

P–a.s., which entails the assertion because of the monotonicity and linearity of E_P. □

Without proof we remark that the assertion corresponding to Proposition 4.1 for model (B) (equation (11) being replaced by its evident analogon) is also valid.

It is not difficult to see that the feasible set of the optimization problem (11) is convex itself:

Proposition 4.2. $\mathcal{T}_P \cap \{T \mid G(T) < \infty\}$ is a convex set.

Proof. Follows immediately from the fact that $\mathcal{T}_P$ can be represented as a cartesian product of one–point sets and simplices, and from the convexity of $\{T \,|\, G(T) < \infty\}$ which results from (15). □

Combining Proposition 4.1. and 4.2, we obtain by standard arguments:

Corollary. Each local minimizer of (11) is a global minimizer.

Note that for (11), it is easy to obtain a realization of the random variable whose expectation is to be minimized: All one has to do is to sample a trajectory X according to P and to compute the indicated expression from X. So, for the numerical solution of (11), the Kiefer–Wolfowitz procedure can be used (for literature, see above). In view of Proposition 4.1 and Theorems 1.5 and 1.11 in [9], this procedure yields, applied to (11), a sequence of approximate solutions converging almost surely to the global minimizer.

Remark 1. It could be conjectured that for equal failure probabilities $f(i,j) = \mathit{const}$ for all arcs (i,j), the solution T_{opt} of (11) is identical to the operational distribution P. This is not the case, as the following counter–example shows:

Let G be the Markov chain in Fig. 3, and $f(i,j) := 1/2$ for each of the three arcs.

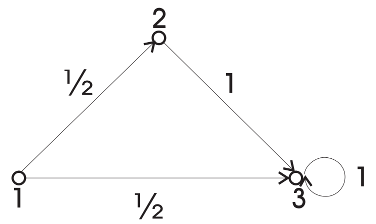

Fig. 3. Transition graph of the counter–example in Remark 1.

In this simple case, the solution of (11) can be computed explicitly. After a short calculation, one obtains

$$t(1,2) = 3 - \sqrt{6} = 0.5505...,$$
$$t(1,3) = -2 + \sqrt{6} = 0.4494...,$$
$$t(2,3) = 1$$

as the optimal test transition probabilities. This shows that even if no especially error–prone arcs can be identified in advance, testing with the operational distribution is nevertheless not optimal in general.

Remark 2. In the practical application, the computation of the optimal test distribution T_{opt} via (11) may fail if there is an arc (i, j) with very low operational probability $p(i, j)$, but high failure probability $f(i, j)$. In such a case, the Kiefer–Wolfowitz procedure does not necessarily improve the initial solution P by giving more weight to the transition (i, j), because, when $G(T)$ is estimated by means of operational sampling, it may happen that transition (i, j) never occurs. In order to avoid this situation, a more sophisticated approach can be used. Let $\bar{P} = (\bar{p}(i, j))$ be any distribution satisfying the constraints (i) – (iv) for test distributions. It can easily be shown that $G(T)$ may also be represented as

$$G(T) = E_{\bar{P}} \left\{ \prod_{k=1}^{\infty} \frac{[p(X_{k-1}, X_k)]^2}{\bar{p}(X_{k-1}, X_k)\, t(X_{k-1}, X_k)} \left[1 - \prod_{l=1}^{\infty} (1 - f(X_{l-1}, X_l)) \right] \right\}.$$

So, when minimizing $G(T)$, we can start the estimation of $G(T)$ with some distribution $\bar{P}$ which may be expected to be already close to the optimal test distribution T_{opt}. (In view of the results in section 2, this could be the distribution given by $\bar{p}(i, j)$ proportional to $p(i, j)\, [f(i, j)]^{1/2}$.) After each iteration step k in the Kiefer–Wolfowitz procedure, set $\bar{P} := T_k$, where T_k is the approximate solution obtained in this step k. Thus, already found good solutions for T are used for further improvements in a kind of boostrap procedure.

Remark 3. In the general (non–Markovian) context of section 2, we have seen that the most efficient test distribution from the viewpoint of reliability estimation is also superior to the operational distribution from the viewpoint of *error detection* (Proposition 2.2). It is an interesting question whether this also holds in the Markovian context, i.e., whether the solution T_{opt} of (11) has always a higher error–detection probability (in the sense formally defined in Proposition 2.2) than the operational distribution P. We conjecture that the question is to be answered in the affirmative.

Remark 4. It would be tempting to use the (unbiased, minimum–variance) *reliability* estimator

$$\tilde{S} := \frac{p(X)}{t(X)} \cdot [1 - c(X, F)] \tag{16}$$

instead of the (unbiased, minimum–variance) unreliability estimator (1). Indeed, it can be shown that for $\tilde{S}$, the optimization problem corresponding to (11) can be solved *explicitly*, using Perron–Frobenius theory (see, e.g., [2] and [3] for applications of this theory to importance sampling in Markov chains). Applying the reliability estimator instead of the unreliability estimator would have the additional advantage that, as can be demonstrated, the optimal test distribution on the set of trajectories is then automatically a Markov chain, i.e., the restriction to $\mathcal{T}_P$ would become unnecessary.

Unfortunately, however, on the usual assumption that the prior failure probabilities are rather small quantities [2], the estimator $\tilde{S}$ fails to achieve an essential variance reduction already in basic situations. To see this, consider the case of a Markov–chain graph G consisting of k *edge–disjoint* paths from the initial state to the absorbing state. Here, optimal test distributions and variances can be computed explicitly both for S and $\tilde{S}$. Let p_i and f_i denote the operational probability and the overall failure probability of path i, respectively ($i = 1, \ldots, k$). Then one findes that the minimum–variance testing probability of path i is given by

$$t_i^* = \frac{p_i \sqrt{f_i}}{\sum_{j=1}^k p_j \sqrt{f_j}} \tag{17}$$

for the unreliability estimator (1), and by

$$\tilde{t}_i^* = \frac{p_i \sqrt{1 - f_i}}{\sum_{j=1}^k p_j \sqrt{1 - f_j}} \tag{18}$$

for the reliabity estimator (16). From (17) and (18), we obtain with modest computational effort:

$$v(S) := E_Q(var_{T^*}(S)) = \sum_{i \neq j} p_i p_j \, \left(\sqrt{f_i f_j} - f_i f_j\right),$$

$$v(\tilde{S}) := E_Q(var_{\tilde{T}^*}(\tilde{S})) = \sum_{i \neq j} p_i p_j \, \left(\sqrt{(1 - f_i)(1 - f_j)} - (1 - f_i)(1 - f_j)\right),$$

and

$$v(S_0) := E_Q(var_P(S_0)) = \sum_{i \neq j} p_i p_j \, f_i (1 - f_j),$$

where T^* resp. $\tilde{T}^*$ are the optimal test distributions for S resp. $\tilde{S}$, and $S_0 = c(X, F)$ is the naive estimator based on the operational distribution. Now, as $f_i \to 0$ ($i = 1, \ldots, k$),

$$v(S) \sim \sum_{i \neq j} p_i p_j \cdot f_i^{1/2} f_j^{1/2},$$

$$v(\tilde{S}) \sim \sum_{i \neq j} p_i p_j \cdot (f_i + f_j)/2 = \sum_{i \neq j} p_i p_j \cdot f_i,$$

and

$$v(S_0) \sim \sum_{i \neq j} p_i p_j \cdot f_i,$$

which shows that for small failure probabilities, $v(\tilde{S})$ is always of the same order of magnitude as $v(S_0)$. On the other hand, $v(S) \leq v(S_0)$ by definition of T^*, and typically, $v(S)$ is even of a smaller order of magnitude than $v(\tilde{S})$ and $v(S_0)$: Let the values f_i be parametrized by a "rarity parameter" ϵ (see [11], p. 272, or [7], p. 46), f_i being of

[2]Note that only programs that are estimated relatively reliable are worthwile to be exposed to a reliability test!

order $\Theta(\epsilon^{\beta_i})$ in ϵ ($\beta_i \in R^+$). Generically, the values β_i are pairwise different. Then $v(S) = \Theta(\epsilon^{\gamma})$ and $v(\tilde{S}) = \Theta(\epsilon^{\tilde{\gamma}})$ with $\gamma = \min_{i,j}((\beta_i + \beta_j)/2)$ and $\tilde{\gamma} = \min_{i,j}(\min(\beta_i, \beta_j))$, such that $\gamma > \tilde{\gamma}$, i.e., $v(S)/v(\tilde{S}) \to 0$ as $\epsilon \to 0$.

The reason for the better variance reduction achieved by the minimum–variance unreliability estimator S is that it concentrates the probability measure around the rare event, i.e., the failure event, while the minimum–variance reliability estimator $\tilde{S}$ rather favors the un–rare event, namely the correct behavior (cf. (17) with (18)). Of course, the effect would turn to the opposite for very *unreliable* programs, which are, however, not a topic for quantitative reliability measurement.

The ineffectiveness of $\tilde{S}$ for "relatively reliable" programs can be verified on fairly general assumptions. E.g., it can be shown that whenever $E_P(c(X)) = \epsilon$ and $P\{c(X) > 1/2\} = o(\epsilon)$, the expected variance $E_Q(var_{\tilde{T}^*}(\tilde{S}))$ is of order $\Theta(\epsilon)$, which is not better than that of the naive estimator S_0. (For the sake of brevity, we omit the proof.)

5 The inhomogenous case.

In section 3 – 4, it has been assumed that the subdomains D_x corresponding to the trajectories x are *homogenous*. In this section, we turn to the more difficult inhomogenous case. If D_x is inhomogenous, then some program input leading to trajectory x is correctly processed, while another input leading to the same trajectory is failure–causing. We can deal with this situation by a slight extension of our formalism. Up to now, $c(x, F)$ was only allowed to take the values 0 or 1: Trajectory x is either correctly processed or failure–causing. From now on, let $c(x, F)$ denote the *probability* that a random input leading to trajectory x is failure–causing. As the distribution $P^{(x)}$ of the random input, the operational distribution P, restricted to subdomain D_x, is chosen.

Definition 2.1 of the unreliability remains formally unchanged:

$$Z(F) = E_P(c(X, F)),$$

where $c(X, F)$ can now assume any value between 0 and 1. The elements a of the input domain D of the given program can be classified according to the trajectories x they entail. So each input a can be uniquely represented by a pair (x, a'), where x is the trajectory forced by x and a' is an index distinguishing a from other inputs in D_x. We set

$$\bar{c}(x, a', F) := \begin{cases} 0, & \text{if } a = (x, a') \text{ is correctly processed,} \\ 1, & \text{otherwise,} \end{cases}$$

and

$$\bar{S} := \frac{p(X)}{t(X)} \cdot \bar{c}(X, A', F),$$

X resp. A' being randomly selected according to T resp. $P^{(X)}$. Let $x^{(1)}, x^{(2)}, \ldots$ be the possible trajectories. Then

$$E_T(\bar{S}) = \sum_{i=1}^{\infty} t(x^{(i)}) \cdot \frac{p(x^{(i)})}{t(x^{(i)})} E_{P^{(x^{(i)})}}(\bar{c}(x^{(i)}, A', F)),$$

where $E_{P(x^{(i)})}$ acts on A', such that with the extended definition of $c(x, F)$,

$$E_T(\bar{S}) = \sum_{i=1}^{\infty} t(x^{(i)}) \cdot \frac{p(x^{(i)})}{t(x^{(i)})} c(x^{(i)}, F) = Z(F).$$

Thus, $\bar{S}$ is an unbiased estimator for $Z(F)$.

The equivalence of the variance minimization problem to equation (11), however, does *not* generalize to the inhomogenous case: Throughout section 4, we have assumed that the equation $[c(x, F)]^2 = c(x, F)$ holds, which is only true for $c(x, F) = 0$ or 1. If $0 < c(x, F) < 1$, then $E_Q([c(x, F)]^2)$ is in general different from $c(x)$ as given by (7).

Nevertheless, well–designed Markov usage models can be expected to have "close–to–homogenous" subdomains D_x (cf. the discussion on the granularity of the usage model in [13]), which means that the prior distribution of the rate of failure–causing inputs in D_x is concentrated close to the values 0 and 1 (see [5]). In this case, $E_Q(c(x, F))$ may be a good approximation to $E_Q([c(x, F)]^2)$, and therefore the solution of (11) will give an unbiased unrelibility estimator with a variance that is possibly not minimal, but distinctly smaller than that induced by the operational distribution.

Acknowledgment. The author is indebted to I. Bomze, to M. Mandjes and especially to G. Pflug for valuable hints during the preparation of the paper. Furthermore, he wants to thank an anonymous referee for his or her detailed and helpful comments to the first version.

References

[1] Asmussen, S., Rubinstein, R., "Steady state rare event simulation in queuing models and its complexity properties", in: *Advances in Queuing: Theory, Methods and Open Problems*, J. Dshalalow (ed.), CRC Press, Boca Raton (1995).

[2] Glynn, P. W., Iglehart, D. L., "Importance sampling for stochastic simulations", *Management Science*, vol. 35 (11), 1367 – 1392 (1989).

[3] Glynn, P. W., "Importance sampling for Markov chains: asymptotics for the variance", *Commun. Statist. – Stochastic Models*, vol. 10 (4), 701 – 717 (1994).

[4] Gutjahr, W. J., "Optimal test distributions for software failure cost estimation", *IEEE Trans. Software Eng.*, vol. SE-21, 219 – 228 (1995).

[5] Gutjahr, W. J., "Partition testing vs. random testing: the influence of uncertainty", Technical Report, Univ. of Vienna (1995).

[6] Hamlet, R., Taylor, R., "Partition testing does not inspire confidence", in: *Proc. 2nd Workshop on Software Testing, Verification, and Analysis*, July 1988, 206 – 215 (1988).

[7] Heidelberger, P., "Fast simulation of rare events in queuing and reliability models", *ACM Trans. on Modeling and Computer Simulation*, vol. 5 (1), 43 – 85 (1995).

[8] Kiefer, J., Wolfowitz, J., "Stochastic estimation of the maximum of a regression function", *Ann. Math. Stat.*, vol. 23, 462 – 466 (1952).

[9] Ljung, P., Pflug, G., and Walk, H., *Stochastic Approximation and Optimization of Stochastic Systems*, Birkhaus, Basel (1992).

[10] Rubinstein, R. Y., *Simulation and the Monte Carlo Method*, Wiley (1981).

[11] Shahabuddin, P., "Fast transient simulation of Markovian models of highly dependable systems", *Performance Evaluation*, vol. 20, 266 – 286 (1994).

[12] Walrand, J., *An Introduction to Queuing Networks*, Prentice–Hall (1988).

[13] Walton, G. H., Poore, J. H., Trammell, J., "Statistical testing of software based on a usage model", *Software – Practice and Experience*, vol. 25 (1), 97 – 108 (1995).

[14] Weyuker, E. J., Ostrand, T. J., "Theories of program testing and the application of revealing subdomains", *IEEE Trans. Software Eng.*, vol. SE-6, 236 – 245 (1980).

[15] Whittaker, J. A., "Markov chain techniques for software testing and reliability analysis", Ph. D. dissertation, University of Tennessee (1992).

[16] Whittaker, J. A., Poore, J. H., "Markov analysis of software specifications", *ACM Trans. on Software Eng. and Method.*, vol. 2 (1), 93 – 106 (1993).

[17] Woit, D. M., "Operational profile specification, test case generation, and reliability estimation for modules", Thesis, Queen's University, Kingston, Ontario, Canada (1994).