مجله علوم آماری، بهار و تابستان ۱۳۹۵
جلد ۱۰، شماره ۱، ص ۱-۲۰
DOI: 10.7508/jss.2016.01.001

یک مدل تعمیم‌یافته برای ارزیابی قابلیت اعتماد نرم‌افزار براساس فرایند پواسون ناهمگن

شیوا اختریان، طاهره یعقوبی
گروه مهندسی کامپیوتر و فناوری اطلاعات، دانشگاه پیام نور

تاریخ دریافت: ۱۳۹۳/۸/۱۲ تاریخ آخرین بازنگری: ۱۳۹۴/۶/۲۴

چکیده : با توجه به کاربردهای گسترده سیستم‌های نرم‌افزاری، لزوم تولید نرم‌افزارهای تقریباً بدون خطا و با کیفیت بالا بیش از پیش اهمیت پیدا کرده است. قابلیت اعتماد نرم‌افزار یک رهیافت مهم برای ارزیابی کیفیت نرم‌افزار در نظر گرفته می‌شود. مدل‌سازی قابلیت اعتماد نرم‌افزار براساس فرایند پواسون ناهمگن یکی از روش‌های کاملاً موفق در مهندسی قابلیت اعتماد نرم‌افزار می‌باشد. در این مقاله ابتدا مدل عمومی رشد قابلیت اعتماد بررسی می‌شود. سپس با در نظر گرفتن دو نوع خطای ساده و پیچیده و لحاظ کردن وابستگی بین خطاهای پیچیده و نیز تاخیر زمانی بین کشف و حذف خطاهای پیچیده به ارائه یک مدل تعمیم‌یافته قابلیت اعتماد نرم‌افزار پرداخته می‌شود. برآورد پارامترهای مدل پیشنهادی با استفاده از مجموعه داده‌های شکست دو پروژه نرم‌افزاری واقعی و از طریق نرم‌افزار MATLAB انجام می‌شود. در پایان مدل پیشنهادی با دو مدل موجود با استفاده از معیارهای مختلف مقایسه می‌شود. نتایج نشان می‌دهد مدل ارائه شده در این تحقیق بر روی این مجموعه داده‌ها بهتر برازش شده، اطلاعات دقیق‌تری در مورد کیفیت

آدرس الکترونیک مسئول مقاله: طاهره یعقوبی، t.yaghoobi@pnu.ac.ir
کد موضوع‌بندی ریاضی (۲۰۱۰): ۶۲P۳۰

نرم‌افزار ارائه می‌کند.

واژه‌های کلیدی : قابلیت اعتماد نرم‌افزار، فرایند پواسون ناهمگن، خطاهای ساده و پیچیده، وابستگی خطاها، تاخیر زمان، اشکال‌زدایی کامل.

۱ مقدمه

امروزه نرم‌افزارها نقش بسیار مهمی در سیستم‌هایی با کاربردهای حساس همچون پزشکی، صنایع نظامی و راکتورهای هسته‌ای پیدا کرده‌اند، بنابراین خطرات و اثرات ناشی از شکست‌های نرم‌افزار می‌تواند جبران ناپذیر باشد. وجود چنین موقعیت‌هایی سازندگان سیستم‌های نرم‌افزاری را بر آن داشته است تا کیفیت نرم‌افزار را قبل از ارائه محصول به بازار و در مرحله تکمیل پروژه بررسی کنند. یک معیار مهم در بررسی کیفیت نرم‌افزار تعیین کمی میزان قابلیت اعتماد نرم‌افزار است. قابلیت اعتماد نرم‌افزار احتمال عملکرد بدون شکست یک برنامه کامپیوتری در محیطی مشخص برای زمانی معین تعریف می‌شود. باید توجه داشت یک سیستم کامپیوتری شامل دو مولفه سخت‌افزار و نرم‌افزار است. به گفته زاک (۱۹۹۲) قابلیت اعتماد نرم‌افزار از این جهت با قابلیت اعتماد سخت‌افزار مشابه است که هر دو به وسیله توابع توزیع احتمال توصیف می‌شوند. ولی سیستم‌های نرم‌افزاری برخلاف سیستم‌های سخت‌افزاری به معنای فیزیکی خراب نمی‌شوند. در سیستم‌های سخت‌افزاری بخش مهمی از خرابی‌ها به دلیل از کار افتادگی و استهلاک قطعات است، در حالی که خطاهای نرم‌افزاری بیشتر ناشی از اشتباهات ذهنی سازندگان آن است و سالخوردگی و استهلاک در نرم‌افزار وجود ندارد. همچنین خطاهای نرم‌افزاری سخت‌تر از خطاهای سخت‌افزاری قابل کشف و تصحیح هستند. بنابراین مدل‌های به‌کار رفته در اندازه‌گیری قابلیت اعتماد سیستم‌های سخت‌افزاری نمی‌توانند برای اندازه‌گیری قابلیت اعتماد نرم‌افزار مورداستفاده قرار گیرند (فام، ۲۰۰۶).

قابلیت اعتماد یک نرم‌افزار به فرایند کشف خطاها و حذف آن‌ها بستگی دارد. در مدل‌سازی قابلیت اعتماد نرم‌افزار فرض بر این است که وقوع شکست‌ها و

عیب‌یابی آن‌ها رویدادهای تصادفی هستند، بدین صورت که نرم‌افزار می‌تواند به عنوان تابعی در نظر گرفته شود که فضای ورودی را به فضای خروجی می‌نگارد. پس می‌توان مقادیر ورودی برای آزمون نرم‌افزار را به صورت تصادفی وارد کرد. وقتی مقادیر ورودی خطاها را فعال می‌کنند، آنگاه شکست‌ها نیز به صورت تصادفی نمایان می‌شوند. بنابراین قابل قبول است که از مدل‌های احتمالی برای مدل‌سازی فرایند شکست‌های نرم‌افزار استفاده شود. پژوهشگران فرایند کشف و برطرف کردن خطاها با هدف بهبود کیفیت نرم‌افزار را به‌وسیله روابط ریاضی مناسب توصیف می‌کنند. این روابط به نام مدل‌های قابلیت اعتماد نرم‌افزار[۱] (SRM) شناخته می‌شوند. هدف نهایی از ایجاد این مدل‌ها آن است که پیش‌بینی قابلیت اعتماد نرم‌افزارها را در مرحله اجرا، براساس داده‌های شکستی که به‌وسیله موارد نمونه در مرحله آزمون جمع‌آوری شده‌اند ممکن سازند. پیش‌بینی قابلیت اعتماد نرم‌افزار به کاربران نیز کمک می‌کند تا با اطمینان کافی برای خرید نرم‌افزار هزینه کنند.

اکثرمدل‌های رشد قابلیت اعتماد نرم‌افزار[۲] (SRGM) مبتنی بر فرایند پواسون ناهمگن[۳] (NHPP)، نمودار قابلیت اعتماد را به یکی از دو صورت مقعر یا S شکل نمایش می‌دهند (فام، ۲۰۰۶؛ کاپور و همکاران، ۲۰۱۱ و یامادا، ۲۰۱۴). اگر قابلیت اعتماد دارای رشد یکنواخت باشد از مدل‌های مقعر، در غیر این صورت از مدل‌های S شکل استفاده می‌شود. گوئل و آکوموتو (۱۹۸۵) مدل مقعر G-O (مدل نمایی) را ارائه کردند که در آن فرض می‌شود کلیه خطاهای نرم‌افزار از یکدیگر مستقل و احتمال کشف خطا در واحد زمان ثابت (مستقل از زمان) است. همچنین خطاها به محض شناسایی حذف و خطای جدیدی در این حین بروز نمی‌کند که به آن اشکال زدایی کامل گویند. مدل‌های اشکال‌زدایی کامل توسط گوئل (۱۹۸۵) و لای و گارج (۲۰۱۲) مورد بررسی و تحقیق قرار گرفتند. درمدل S شکل بررسی شده توسط یامادا (۲۰۰۰) و یامادا و همکاران (۱۹۸۴) نیزخطاها به محض شناسایی حذف می‌شوند و اشکال‌زدایی کامل است. با این تفاوت که نرخ کشف خطا ثابت نبوده و

---

۱ Software Reliability Model

۲ Software Reliability Growth Model

۳ Non Homogeneous Poisson Process

وابسته به زمان می‌باشد. در مدل S شکل تاخیری ارائه شده توسط یامادا و همکاران (۱۹۸۳) به این نکته مهم توجه می‌شود که با گذشت زمان تیم آزمون کننده نرم‌افزار با محیط آزمون، ابزارهای آزمون، نیازمندی‌های پروژه و ... آشنایی بیشتری پیدا می‌کند و بدین ترتیب فرایند آزمون نرم‌افزار پس از گذشت زمان بهبود می‌یابد. در این مدل فرض می‌شود که خطاها از نقطه نظر کشف شکست مستقل از یکدیگر هستند، اما در فرایند کشف خطا زمان بین $(i-1)$امین و iامین شکست بستگی به زمان وقوع $(i-1)$امین شکست دارد.

در برخی مدل‌هایی که توسط پژوهشگران ایجاد شده‌اند خطاهای شناسایی شده از نظر میزان پیچیدگی یا وابستگی بین آن‌ها تقسیم‌بندی می‌شوند. به‌عنوان مثال ایده وابستگی خطاها را اهبا (۱۹۸۴) مطرح کرد. با این فرض که خطاهای مستقل بلافاصله پس از شناسایی قابل حذف هستند، اما خطاهای وابسته پیش از حذف خطاهایی که به آن وابسته‌اند قابل حذف شدن نیستند. پس از آن کاپور و یونس (۱۹۹۵) نیز وابستگی خطاها را در مدل نمایی گوئل و آکوموتو (۱۹۸۱) تاثیر دادند، البته با توجه به این نکته که نمی‌توان از عامل تاخیر زمان بین کشف تا حذف خطاهای وابسته چشم‌پوشی کرد، موضوع تاثیر عامل تاخیر زمانی در حذف خطاهای وابسته را در نظر گرفتند. کاپور و همکاران (۱۹۹۹) مرحله آزمون نرم‌افزار را شامل سه فرایند مشاهده شکست، ایزوله کردن خطا و حذف خطا در نظر گرفتند و مدل SRGM را پیشنهاد دادند که در آن حذف و اصلاح خطاها براساس سطح پیچیدگی آن‌ها متمایز می‌باشد و مشتمل بر سه نوع خطا است. خطاهای نوع اول که ساده هستند و هیچ تاخیر زمانی بین مراحل مشاهده شکست تا حذف این دسته از خطاها تاثیر داده نمی‌شود و توسط مدل نمایی گوئل و آکوموتو (۱۹۸۱) مدل‌سازی می‌شوند. خطاهای نوع دوم خطاهای سخت هستند که بین مشاهده شکست و ایزوله کردن این دسته از خطاها تاخیر زمان تاثیر داده می‌شود و توسط مدل S شکل تاخیری مدل‌سازی می‌شوند و خطاهای نوع سوم خطاهای پیچیده‌اند و تاخیر زمان بین هر سه مرحله مشاهده شکست، ایزوله کردن خطا و حذف خطا تاثیر داده می‌شود. هانگ و لین (۲۰۰۴ و ۲۰۰۶) خطاها را به دو دسته مستقل و وابسته تقسیم نموده و برای مدل‌سازی وابستگی خطاها عامل‌های تاخیر متمایزی برای تاثیر در

حذف خطاهای وابسته بررسی کرده‌اند. سپس تاثیر وابستگی خطاها و عامل تاخیر بر تعدادی از مدل‌های پایه رشد قابلیت اعتماد نرم‌افزار براساس NHPP مورد تحلیل قرار گرفت.

در این مقاله مدلی ارائه می‌شود که در آن خطاهای نرم‌افزاری از دو منظر میزان پیچیدگی و وابستگی بین آن‌ها به سه نوع خطا تقسیم‌بندی می‌شوند. از نظر میزان پیچیدگی، خطاها ابتدا به دو دسته ساده و پیچیده تقسیم می‌شوند. خطای ساده به خطایی اطلاق می‌شود که بلافاصله و به‌سادگی بعد از شناسایی قابل حذف است. برای خطاهای پیچیده، امکان وابستگی بین آن‌ها در نظر گرفته می‌شود. سپس با توجه به اینکه خطاهای پیچیده بلافاصله بعد از کشف، قابل حذف نیستند و ممکن است ماه‌ها زمان و تلاش برای حذف آن‌ها نیاز باشد عامل تاخیر زمان بین کشف تا حذف این نوع خطاها در نظر گرفته می‌شود. بنابراین در مدل پیشنهادی خطاهای نرم‌افزاری به سه دسته ساده، پیچیده مستقل و پیچیده وابسته تقسیم می‌شوند. برای هر دسته، با توجه به نوع خطا مدل مناسب قابلیت اعتماد پیشنهاد می‌شود و مدل کلی قابلیت اعتماد نرم‌افزار برابر مجموع سه مدل فرض شده در نظر گرفته می‌شود.

بخش ۲ به توضیح پیرامون چارچوب نظری قابلیت اعتماد نرم‌افزار براساس NHPP اختصاص یافته است. در بخش ۳ مدل پیشنهادی همراه با پارامترها و پذیره‌های آن ارائه می‌شود. در بخش ۴ پارامترهای مدل پیشنهادی بر روی داده‌های شکست واقعی دو نرم‌افزار برآورد می‌شوند. در بخش ۵ مدل پیشنهادی با دو مدل پایه در زمینه قابلیت اعتماد نرم‌افزار مقایسه می‌شوند. سرانجام در بخش ۶ بحث و نتیجه‌گیری ارائه می‌شود.

۲ چارچوب نظری تحقیق

جهت تخمین قابلیت اعتماد نرم‌افزارها ابتدا مدل قابلیت اعتماد نرم‌افزار ایجاد می‌شود، سپس باید پارامترهای موجود در مدل برآورد شوند. در نهایت می‌توان توسط مدل ایجاد شده، قابلیت اعتماد نرم‌افزار را تخمین زد و زمان مناسب عرضه نرم افزار به بازار را پیش‌بینی نمود.

۱.۲ مدل‌سازی عمومی قابلیت اعتماد نرم‌افزار

مدل‌های قابلیت اعتماد براساس NHPP ابزارهای کاملا موفق در مهندسی قابلیت اعتماد نرم‌افزار هستند. برای مثال به کاپور و همکاران (۲۰۱۱)، لای و گارج (۲۰۱۲) و یامادا (۲۰۱۴) مراجعه شود. فرض اساسی در این مدل‌ها این است که فرایند شکست به‌وسیله NHPP قابل توصیف است. در اکثر مدل‌های رشد قابلیت اعتماد نرم‌افزار با توجه به فرایند کشف خطا پذیره‌های کلی به شرح ذیل در نظر گرفته می‌شوند:

۱- سیستم نرم‌افزاری محل وقوع شکست‌هایی با زمان تصادفی می‌باشد که توسط نقص‌های موجود در نرم‌افزار ایجاد می‌شوند.

۲- رخداد کشف یا حذف خطا به‌وسیله NHPP مدل‌سازی می‌شود.

۳- تمام خطاها از یکدیگر مستقل بوده و به یک میزان قابل شناسایی هستند.

۴- تعداد خطاهای کشف شده در هر زمان متناسب با تعداد خطاهای باقیمانده در نرم‌افزار است.

۵- هر زمان که شکستی رخ می‌دهد خطایی که منجر به آن شده بلافاصله و به‌طور کامل حذف می‌شود و خطای جدیدی دراین حین بروز نمی‌کند (اشکال‌زدایی کامل).

فرض کنید $N(t)$ تعداد خطاهای کشف شده تا زمان t و $m(t)$ میانگین تعداد خطاهای کشف شده تا زمان t، یا تابع مقدار میانگین[۴] (MVF) باشد. در حقیقت SRM تابع مقدار میانگین یک NHPP است. احتمال این که دقیقاً n شکست در بازه زمانی $(۰, t)$ رخ دهد به‌صورت

$$Pr\{N(t) = n\} = \frac{[m(t)]^n}{n!} e^{-m(t)}, \quad n = ۰, ۱, \ldots$$

است، که در آن $m(t) = \int_۰^t \lambda(s)ds$ و $\lambda(t)$ تابع شدت شکست نرم‌افزار (تعداد شکست‌ها در واحد زمان) است، به‌طوری که

$$\frac{dm(t)}{dt} = \lambda(t)$$

[۴] Mean Value Function

بنابراین مدل کلی قابلیت اعتماد مبتنی بر NHPP از حل معادله دیفرانسیل

$$\lambda(t) = \frac{dm(t)}{dt} = b(t)[a - m(t)] \qquad (۱)$$

با شرط اولیه $m(۰) = ۰$ به دست می‌آید، که در آن a تعداد کل خطاهای برآورد شده موجود در نرم‌افزار قبل از انجام آزمون، $[a - m(t)]$ متوسط تعداد خطاهای باقیمانده در نرم‌افزار تا زمان t و $b(t)$ نرخ کشف خطا در واحد زمان[۵] است.

جواب عمومی معادله دیفرانسیل (۱) به‌صورت

$$m(t) = e^{-\beta(t)}[C + \int ab(t)e^{\beta(t)}dt] \qquad (۲)$$

است، که در آن $\beta(t) = \int b(t)dt$ و C ثابت انتگرال‌گیری است. در نهایت قابلیت اعتماد نرم‌افزار از طریق فرمول

$$R(t) = e^{-\int_{۰}^{t} \lambda(s)}$$

به‌دست می‌آید. برای مثال کاپور و همکاران (۲۰۱۱) و گوئل (۱۹۸۵) مدل نمایی گوئل-آکوموتو و مدل S شکل تاخیری را به‌صورت جدول ۱ بیان نمودند.

جدول ۱: جزئیات مدل گوئل-آکوموتو و مدل S شکل تاخیری

مدل	$m(t)$	$b(t)$	$\lambda(t)$	$[a - m(t)]$
نمایی	$a(۱ - e^{-bt})$	b	abe^{-bt}	ae^{-bt}
S شکل تاخیری	$a[۱ - (۱ + bt)e^{-bt}]$	$\frac{b^{۲}t}{bt+۱}$	$ab^{۲}te^{-bt}$	$a(۱ + bt)e^{-bt}$

۲.۲ برآورد پارامترهای مدل‌های قابلیت اعتماد نرم‌افزار

مدل‌های قابلیت اعتماد نرم‌افزار دارای پارامترهای ناشناخته‌ای هستند که با استفاده از داده‌های شکست و معمولاً با روش ماکسیمم درستنمایی[۶] (MLE) برآورد می‌شوند. فرض کنید y_i، $i = ۱, \ldots, n$ که مجموع شکست‌های مشاهده شده واقعی

[۵] Fault Detection Rate

[۶] Maximum Likelihood Estimation

در بازه‌های زمانی $(۰, t_i)$، y_i باشد که $۰ < t_۱ < \cdots < t_n$. در این صورت لگاریتم تابع درستنمایی[۷] (LLF) به‌صورت

$$LLF = \sum_{i=۱}^{n} (y_i - y_{i-۱}) \log[m(t_i) - m(t_{i-۱})] - m(t_n)$$

خواهد بود. فام (۲۰۰۶) معادله

$$۰ = \sum_{i=۱}^{n} \frac{\frac{\partial}{\partial\theta} m(t_i) - \frac{\partial}{\partial\theta} m(t_{i-۱})}{m(t_i) - m(t_{i-۱})} (y_i - y_{i-۱}) - \frac{\partial}{\partial\theta} m(t_n)$$

را برای به‌دست آوردن ماکسیمم تابع درستنمایی به دست آورد، که در آن θ پارامتر مجهول مدل است.

به‌منظور محاسبه واریانس برآورد ماکسیمم درستنمایی پارامترها، ماتریس هسین با گرفتن مشتقات جزئی مرتبه دوم از لگاریتم تابع درستنمایی تشکیل می‌شود. ماتریس اطلاع فیشر، قرینه امید ریاضی این ماتریس است. از آن‌جایی که برخی از درایه‌های ماتریس مذکور فرم بسیار پیچیده‌ای دارند و محاسبه امید ریاضی آن‌ها امکان‌پذیر نیست، از قانون قوی اعداد بزرگ برای محاسبه امید ریاضی ماتریس هسین استفاده می‌شود. به این ترتیب که m مشاهده نمونه‌ای $N_i(t)$، $i = ۱, \ldots, m$ به‌طور مستقل به‌دست آورده می‌شود. سپس با جایگذاری در ماتریس هسین، برآورد ماتریس اطلاع فیشر به‌صورت

$$I(\boldsymbol{\theta}) \cong -\frac{۱}{m} \sum_{i=۱}^{m} H(\boldsymbol{\theta} | (N_i(t))_{۱ \le t \le n})$$

محاسبه می‌گردد، که در آن $(N_i(t))_{۱ \le t \le n}$ تحقق‌های مستقل از فرایند پواسون با پارامتر $m(t)$ و $\boldsymbol{\theta}$ بردار پارامترهای آن است. دراین صورت کران پایین کرامر–رائو برای ماتریس واریانس–کوواریانس بردار برآورد پارامترها برابر با $I^{-۱}(\boldsymbol{\theta})$ است.

---

[۷] Log Likelihood Function

۳.۲ پیش‌بینی زمان عرضه نرم‌افزار به بازار

یکی از کاربردهای پیش‌بینی قابلیت اعتماد نرم‌افزار، تعیین زمان مناسب برای خاتمه آزمون نرم‌افزار و عرضه محصول نرم‌افزاری به بازار است. با توجه به نظر گوئل و آکوموتو (۱۹۸۱) اگر نرم‌افزار به حد قابل قبولی از قابلیت اعتماد در بازه زمانی مشخصی طی فرایند آزمون نرم‌افزار رسیده باشد، آنگاه می‌توان برای انتشار آن به بازار تصمیم‌گیری کرد. برای رسیدن به این هدف از تابع قابلیت اعتماد شرطی $R(x|t)$ استفاده می‌شود. این تابع با فرض اینکه آخرین شکست در زمان t رخ داده باشد، احتمال عدم بروز شکست در بازه زمانی $(t, t+x)$ را به‌صورت

$$R(x|t) = Pr\{N(t+x) - N(t) = ۰\} = e^{[m(t+x)-m(t)]}$$

مشخص می‌کند.

۳ مدل پیشنهادی

در این بخش ابتدا پارامترهای مدل و سپس پذیره‌های آن ارائه می‌شود و در نهایت مدل جدید با توجه به پذیره‌های ارائه شده مدل‌سازی می‌شود. این مدل شامل دو نوع پارامتر مستقل از زمان و وابسته به زمان به‌صورت زیر است:

a: تعداد کل خطاهای برآورد شده اولیه موجود در نرم‌افزار

$a_۱, a_۲, a_۳$: تعداد کل خطاهای برآورد شده ساده، پیچیده مستقل، پیچیده وابسته

$b_۱, b_۲, b_۳$: نرخ کشف خطاهای ساده، پیچیده مستقل، پیچیده وابسته

p: نسبت خطاهای ساده به پیچیده در نرم‌افزار

q: نسبت خطاهای پیچیده مستقل به پیچیده وابسته

$\varphi_۱(t)$: فاکتور تاخیر زمانی خطاهای پیچیده مستقل

$\varphi_۲(t)$: فاکتور تاخیر زمانی خطاهای پیچیده وابسته

$m(t)$: تابع مقدار متوسط کل خطاهای نرم‌افزار

$m_۱(t)$: تابع مقدار متوسط خطاهای ساده

$m_۲(t - \varphi_۱(t))$: تابع مقدار متوسط خطای پیچیده مستقل

$m_۳(t-\varphi_۲(t))$: تابع مقدار متوسط خطای پیچیده وابسته

در عمل بعضی از پذیره‌های کلی ارائه شده در بخش ۲ ممکن است با واقعیت مطابقت نداشته باشند. مثلاً هر زمان که شکستی رخ دهد، ممکن است تاخیر در تلاش برای حذف علت شکست وجود داشته باشد. مدل پیشنهادی در پذیره‌های ۲،۱ و ۴ ارائه شده در بخش قبلی صدق می‌کند. با این حال مفروضات خاص ذیل نیز درمدل لحاظ شده‌اند:

الف- خطاهای نرم‌افزار به دو دسته ساده و پیچیده تقسیم می‌شوند. این تقسیم‌بندی با توجه به میزان تلاش مورد نیاز جهت رفع آن‌ها انجام می‌شود.

ب- خطاهای ساده بلافاصله بعد از شناسایی قابل اصلاح و حذف هستند.

ج- در سیستم نرم‌افزاری بعد از شناسایی خطاهای پیچیده، ممکن است مدت‌ها تلاش برای حذف و اصلاح نیاز باشد. این نوع خطاها خود در دو نوع مستقل و وابسته در نظر گرفته می‌شوند.

د- امکان حذف فوری خطاهای پیچیده مستقل وجود ندارد و برای حذف این دسته از خطاها تلاش و زمان بیشتری نسبت به خطاهای ساده نیاز است. بنابراین زمان تاخیر $\varphi_۱(t)$ برای حذف آن‌ها تاثیر داده می‌شود.

هـ- خطاهای پیچیده وابسته به علت وجود بعضی از خطاهای پیچیده مستقل به وجود می‌آیند و تا هنگامی که خطای پیچیده مستقل مربوط به آن حذف نشود قابل حذف نمی‌باشند. به علت تلاش مورد نیاز در حذف این دسته از خطاها عامل تاخیر زمان قابل چشم‌پوشی نبوده، بدین ترتیب زمان تاخیر $\varphi_۲(t)$ در اشکال زدایی آن‌ها تاثیر داده می‌شود.

و- احتمال ایجاد خطاهای جدید طی فرایند اشکال‌زدایی وجود ندارد (اشکال‌زدایی کامل).

فرض کنید مرحله آزمون نرم‌افزار از سه فرایند مشاهده شکست، کشف خطا و حذف خطا تشکیل شده باشد. براساس فرض‌های الف و ب خطاهای نرم‌افزاری به سه دسته ساده، پیچیده مستقل و پیچیده واسبته تقسیم می‌شوند. همچنین در حین فرایند اشکال‌زدایی، امکان بروز خطاهای جدید وجود ندارد (فرض و). بنابراین

تعداد کل خطاهای کشف شده در زمان $(۰,t)$ برابر است با

$$m(t) = m_۱(t) + m_۲(t - \varphi_۱(t)) + m_۳(t - \varphi_۲(t)) \qquad (۳)$$

الف- مدل‌سازی حذف خطاهای ساده $m_۱(t)$: براساس فرض ب، برای خطاهای ساده تاخیر زمانی بین مشاهده شکست، کشف و حذف خطا قابل چشم‌پوشی می‌باشد. با توجه به اینکه خطاهای ساده بلافاصله بعد از مشاهده شکست توسط فرد آزمون‌کننده سیستم قابل حذف می‌باشند، بنابراین فرایند کشف و حذف آن‌ها در مدل عمومی قابلیت اعتماد (۱) با نرخ کشف خطای ثابت $b_۱$ صدق می‌کند. با توجه به پذیره‌های این مدل خواهیم داشت.

$$\frac{dm_۱(t)}{dt} = b_۱[a_۱ - m_۱(t)] \qquad (۴)$$

چون $a_۱ = pa$ با حل معادله دیفرانسیل (۴) از طریق معادله (۲) تابع مقدار متوسط خطای کشف شده زیر ایجاد می‌شود.

$$m_۱(t) = a_۱(۱ - e^{b_۱ t}) = pa(۱ - e^{-b_۱ t})$$

ب- مدل‌سازی حذف خطاهای پیچیده مستقل $m_۲(t)$: در هنگام بروز خطاهای پیچیده، تیم آزمون‌کننده نرم‌افزار نیاز به زمان بیشتری جهت کشف علت خطا و سپس حذف آن دارد (فرض ج). در مورد خطاهای پیچیده مستقل تاخیر زمانی بین شناسایی یک خطا تا حذف آن قابل چشم‌پوشی نمی‌باشد (فرض د). برای این نوع خطاها نیز می‌توان از مدل عمومی قابلیت اعتماد استفاده کرد، با این حال لازم است عامل تاخیر زمان را در فرایند اشکال‌زدایی تاثیر داد. با فرض این که مقدار متوسط خطاهای مستقل شناسایی شده متناسب با تعداد خطاهای مستقل باقیمانده در سیستم باشد، داریم:

$$\frac{dm_۲(t)}{dt} = b_۲[a_۲ - m_۲(t)] \qquad (۵)$$

که در آن $a_۲ = (۱ - p)qa$ با حل معادله دیفرانسیل (۵) در شرایط $m_۲(۰) = ۰$ خواهیم داشت:

$$m_۲(t) = a_۲(۱ - e^{-b_۲ t}) = (۱ - p)qa(۱ - e^{-b_۲ t}) \qquad (۶)$$

با توجه به افزایش مهارت و یادگیری تیم آزمون‌کننده نرم‌افزار، عامل تاخیر زمان در اشکال‌زدایی خطاهای پیچیده مستقل را تابع صعودی $\varphi_1(t)$ درنظر گرفته، آن را در معادله (۶) تاثیر می‌دهیم.

$$\varphi_1(t) = (\frac{\ln(1 + b_2 t)}{b_2})$$

بنابراین

$$m_2(t - \varphi_1(t)) = (1 - p)qa[1 - (1 + b_2 t)e^{-b_2 t}]$$

که همان معادله مدل S شکل تاخیری است.

ج- مدل‌سازی حذف خطاهای پیچیده وابسته $m_3(t)$: براساس فرض ۵ امکان حذف این نوع خطاها قبل از آن که خطاهای پیچیده مستقل، که تولید کننده آن‌ها هستند، برطرف نشوند وجود ندارد. بنابراین مدل عمومی قابلیت اعتماد برای این نوع خطاها مستقیماً قابل به‌کارگیری نیست (فرض ۳ برقرار نمی‌باشد). دراین حالت مقدار متوسط خطاهای پیچیده وابسته کشف شده در سیستم متناسب با تعداد خطاهای پیچیده وابسته باقیمانده در سیستم و مقدار متوسط خطاهای پیچیده مستقل می‌باشد. بنابراین با قرار دادن

$$\frac{dm_3(t)}{dt} = b_3[a_3 - m_3(t)]\frac{m_2(t - \varphi_1(t))}{a_2 + a_3} \qquad (۷)$$

داریم $a_3 = (1 - p)(1 - q)a$ و از حل معادله (۷) خواهیم داشت:

$$\begin{aligned} m_3(t) &= a_3[1 - exp(\frac{-a_2 b_3(b_2 t + 2e^{-b_2 t} + b_2 t e^{-b_2 t} - 2)}{b_2(a_2 + a_3)})] \\ &= (1 - p)(1 - q)a[1 - e^{\frac{-(1-p)qab_3(b_2 t + 2e^{b_2 t} + b_2 t e^{-b_2 t} - 2)}{b_2((1-p)a)}}] \end{aligned} \qquad (۸)$$

با توجه به زمان مورد نیاز در حذف و اشکال‌زدایی خطاهای پیچیده وابسته، عامل تاخیر زمان $\varphi_2(t)$ در معادله (۸) تاثیر داده می‌شود و معادله نهایی خطاهای پیچیده وابسته به‌صورت زیر حاصل می‌شود.

$$\varphi_2(t) = (\frac{\ln(1 + b_3 t)}{b_3})$$

$$
\begin{aligned}
m_3(t-\varphi_2(t)) &= a_3[1-exp(\frac{-a_2b_3(b_2(t-\varphi_2(t))+2e^{-b_2(t-\varphi_2(t))}}{b_2(a_2+a_3)} \\
&+ \frac{b_2(t-\varphi_2(t))-2)}{b_2(a_2+a_3)})] \\
&= (1-p)(1-q)a[1-exp(-(1-p)qab_3(b_2(t-\varphi_2(t)) \\
&+ 2e^{-b_2(t-\varphi_2(t))+b_2(t-\varphi_2(t))e^{-b_2(t-\varphi_2(t))}-2}b_2((1-p)a))]
\end{aligned}
$$

بنابراین با درنظر گرفتن معادله (۳) تابع مقدار متوسط کل خطاهای نرم‌افزار عبارت خواهد شد از

$$
\begin{aligned}
m(t) &= pa(1-e^{-b_1t})+(1-p)qa[1-(1+b_2t)e^{-b_2t}] \\
&+ (1-p)(1-q)a[1-exp(\frac{-(1-p)qab_3(b_2(t-\varphi_2(t))}{b_2((1-p)a)} \\
&+ \frac{2e^{-b_2(t-\varphi_2(t))}+b_2(t-\varphi_2(t))e^{-b_2(t-\varphi_2(t))}-2}{b_2((1-p)a)})]
\end{aligned}
$$

۴ برازش مدل

مدل پیشنهادی با مجموعه داده‌های شکست نرم‌افزار فرماندهی و کنترل زمان واقعی که توسط موسی و همکاران (۱۹۸۷) گزارش شده است (جدول ۲)، تطبیق داده شد، اندازه تابع مقدار میانگین، تعداد خطاهای باقیمانده و قابلیت اعتماد سیستم برآورد شد. براساس داده‌های شکست ارائه شده در جدول ۲ ابتدا پارامترهای مدل پیشنهادی، مدل گوئل-آکوموتو و مدل S شکل تاخیری به روش MLE و با نرم‌افزار MATLAB 2012 برآورد شده و نتایج در جدول ۳ ارائه شده است.

اکنون با توجه به برآوردهای به‌دست آمده می‌توان تابع مقدار متوسط، تعداد خطاهای باقیمانده در نرم‌افزار و قابلیت اعتماد نرم‌افزار در بازه زمانی کوچکی همچون $(t, t+0/1)$ را برآورد کرد و زمان خاتمه آزمون نرم‌افزار را مشخص نمود. به‌عنوان مثال فرض کنید زمان عرضه نرم‌افزار به بازار زمانی است که تعداد خطاهای باقیمانده در آن کمتر از ۱۰ باشد، در این صورت می‌توان آزمون این نرم‌افزار را در ساعت ۲۴ام خاتمه داد. قابلیت اعتماد محصول نهایی برای ارائه به بازار در بازه

جدول ۲: مجموع داده شکست نرم‌افزار فزماندهی و کنترل زمان واقعی

زمان (ساعت)	تعداد خطای مشاهده شده	مجموع خطای مشاهده شده
۱	۲۷	۲۷
۲	۱۶	۴۳
۳	۱۱	۵۴
۴	۱۰	۶۴
۵	۱۱	۷۵
۶	۷	۸۳
۷	۲	۸۴
۸	۵	۸۹
۹	۳	۹۲
۱۰	۱	۹۳
۱۱	۴	۹۷
۱۲	۷	۱۰۴
۱۳	۲	۱۰۶
۱۴	۵	۱۱۱
۱۵	۵	۱۱۶
۱۶	۶	۱۲۲
۱۷	۰	۱۲۲
۱۸	۵	۱۲۷
۱۹	۱	۱۲۸
۲۰	۱	۱۲۹
۲۱	۲	۱۳۱
۲۲	۱	۱۳۲
۲۳	۲	۱۳۴
۲۴	۱	۱۳۵
۲۵	۱	۱۳۶

جدول ۳: برآورد پارامترهای مدل پیشنهادی، گوئل-آکوموتو و S شکل تاخیری

مدل	پارامتر	برآورد	واریانس	خطای استاندارد
پیشنهادی	a	۱۴۴	۱۵۲/۳۹	۱۲/۳۴
	b_1	۰/۳۷	۰/۰۰۳	۰/۰۶
	b_2	۰/۱۰	۰/۰۰۰۴	۰/۰۱
	b_3	۰/۹۹	۰/۰۶	۰/۰۲
	p	۰/۵۷	۰/۰۰۲	۰/۰۵
	q	۰/۴۴	۰/۰۲۲	۰/۱۴
گوئل-آکوموتو	a	۱۴۲	۱۴۸/۸۳	۱۱/۷۴
	b	۰/۱۲	$۲/۰۸۶۵e^{-۰۴}$	۰/۰۱
S شکل تاخیری	a	۱۳۶	۱۳۷/۸۳	۱۲/۱۹
	b	۰/۲۸	$۳/۴۱۲۵e^{-۰۴}$	۰/۰۱

زمانی کوتاه بعد از آن، (۲۵/۱ و ۲۵)، حدود ۰/۹ است. جدول ۴ اطلاعات فرایند آزمون نرم‌افزار سیستم فرماندهی و کنترل زمان واقعی، تطبیق داده شده با مدل پیشنهادی در زمان‌های ۱۷ تا ۲۵ آزمون، جهت تخمین زمان عرضه آن به بازار را ارائه می‌دهد. همچنین ماتریس اطلاع فیشر مدل به‌صورت

$$I(\theta)=\begin{bmatrix} ۷۴/۹۹ & ۰/۰۱۹ & ۰/۰۳۵ & -۰/۰۵۹ & -۰/۰۰۳ & ۰/۰۱۶ \\ ۰/۰۱۹ & ۰/۰۰۶ & ۰/۰۰۱ & -۰/۰۰۶ & -۰/۰۰۴ & ۰/۰۰۳ \\ ۰/۰۳۵ & ۰/۰۰۱ & ۰/۰۰۱ & -۰/۰۰۷ & -۰/۰۰۱ & ۰/۰۰۲ \\ -۰/۰۵۹ & -۰/۰۰۶ & -۰/۰۰۷ & ۰/۱۸۱ & ۰/۰۱۰ & -۰/۰۵۹ \\ -۰/۰۰۳ & -۰/۰۰۴ & -۰/۰۱۰ & ۰/۰۱۰ & ۰/۰۰۵ & -۰/۰۰۴ \\ ۰/۰۱۶ & ۰/۰۰۳ & ۰/۰۰۲ & -۰/۰۵۹ & -۰/۰۰۴ & ۰/۰۲۷ \end{bmatrix}$$

است، که در آن $\theta=(a,b_۱,b_۲,b_۳,p,q)$ است.

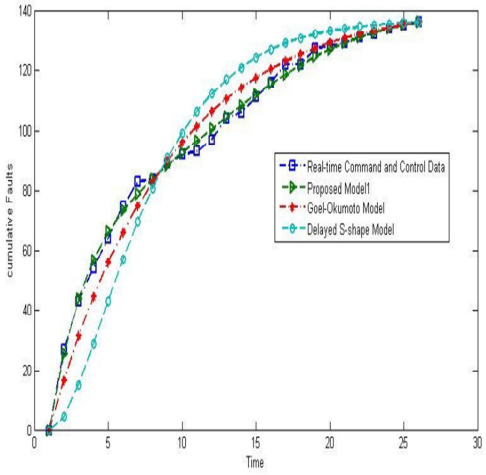
جدول ۴: اطلاعات کسب شده پس از هر بار آزمون نرم‌افزار فرماندهی و کنترل زمان واقعی برای تعیین زمان عرضه آن به بازار

آزمون تا زمان	$\hat{m}(t)$	$\hat{a}$	$\hat{a}-\hat{m}(t)$	$R(۰/۱\|t)$
۱۷	۱۲۲/۲۰۵	۱۹۸/۳۶۸	۷۶/۱۶۳	۰/۷۲۷
۱۸	۱۲۷/۰۰۰	۱۷۶/۶۸۰	۴۹/۶۸۰	۰/۶۸۰
۱۹	۱۲۷/۹۹۷	۱۵۲/۱۳۴	۲۴/۱۳۲	۰/۷۶۴
۲۰	۱۲۸/۹۹۷	۱۴۴/۴۰۹	۱۵/۴۱۲	۰/۸۱۹
۲۱	۱۳۱/۰۰۱	۱۴۴/۸۸۲	۱۳/۸۸۱	۰/۸۳۸
۲۲	۱۳۲/۰۰۰	۱۴۲/۴۵۹	۱۰/۴۵۹	۰/۸۶۹
۲۳	۱۳۴/۰۰۵	۱۴۵/۱۲۴	۱۱/۱۱۹	۰/۸۷۷
۲۴	۱۳۵/۰۰۲	۱۴۴/۰۷۸	۹/۰۷۵	۰/۸۹۶
۲۵	۱۳۶/۰۰۳	۱۴۳/۹۶۷	۷/۹۶۴	۰/۹۱۱

۵ مقایسه مدل پیشنهادی

در این بخش مدل پیشنهادی با مدل گوئل-آکوموتو و مدل S شکل تاخیری پیش‌بینی قابلیت اعتماد نرم‌افزار که جزئیات بیشتر آن‌ها در جدول ۱ ارائه شده است مقایسه می‌شود. مقایسه مدل‌ها می‌تواند توسط نمودارهای تطبیق مدل‌ها با مجموعه داده‌های شکست نرم‌افزارها نمایش داده شود. شکل ۱ نمودار تطبیق مدل پیشنهادی

و دو مدل مذکور را بر روی مجموعه داده شکست نرم‌افزار فرماندهی و کنترل زمان واقعی نشان می‌دهد. همان‌گونه که ملاحظه می‌شود، مدل پیشنهادی به خوبی بر



شکل ۱: نمودار مقایسه تطبیق مدل پیشنهادی، گوئل-آکوموتو و S شکل تاخیری روی مجموعه داده شکست نرم‌افزار فرماندهی و کنترل زمان واقعی

روی مجموعه داده شکست نرم‌افزار فرماندهی و کنترل زمان واقعی منطبق گردیده است.

برای مقایسه دقیق مدل‌ها می‌توان از سه معیار نرخ پیش‌بینی ریسک[۸] (PRR)، مجموع توان‌های دوم خطا[۹] (SSE) و متوسط توان‌های دوم خطا[۱۰] (MSE) که به‌صورت

$$PRR = \sum_{i=1}^{n} \left(\frac{m(t_i) - y_i}{m(t_i)}\right)^2$$

$$SSE = \sum_{i=1}^{n} (y_i - m(t_i))^2$$

$$MSE = \sum_{i=1}^{n} \frac{(m(t_i) - y_i)^2}{n}$$

[۸] Predictive-Ratio Risk

[۹] Sum of Squared Errors

[۱۰] Mean Squared Errors

تعریف می‌شوند، استفاده نمود، که در آن y_i تعداد کل خطاهای مشاهده شده در زمان t_i، $m(t_i)$ مجموع خطای تخمین زده شده در زمان t_i و n تعداد مشاهدات است.

جدول ۵ مقادیر سه معیار PPR، SSE و MSE در مدل‌های پیشنهادی، گوئل-آکوموتوو S شکل تاخیری را بر روی مجموعه داده‌ی شکست نرم‌افزار فرماندهی و کنترل زمان واقعی نشان می‌دهد.

جدول ۵: برآورد معیارها برای مقایسه روش پیشنهادی با مدل‌های گوئل-آکوموتو و S شکل تاخیری روی مجموعه داده‌ی شکست نرم‌افزار فرماندهی و کنترل زمان واقعی

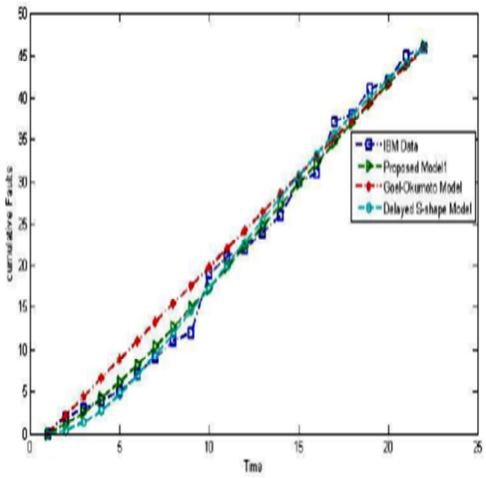
معیار	مدل پیشنهادی	مدل نمایی	مدل S شکل تاخیری
PRR	۰/۰۱۳	۰/۶۴	۲۸/۵۱
SSE	۸۴/۵۴	۹۰۳/۲۷	۴۲۱۷/۹
MSE	۴/۰۲۵	۳۶/۱۳	۱۶۸/۷۱

همان‌طور که ملاحظه می‌شود مدل پیشنهادی بر روی مجموعه داده‌های ذکر شده براز ش خوبی داشته، بهتر از دو مدل مذکور عمل می‌کند.

در ادامه مدل تعمیم‌یافته در این تحقیق با دو مدل مذکور با استفاده از مجموعه داده‌ای که از آزمون بسته نرم‌افزاری ورود اطلاعات برخط شرکت IBM، توسط اهبا (۱۹۸۴) به‌دست آمده است مقایسه و نتایج در جدول ۶ و شکل ۲ ارائه شده است.

جدول ۶: برآورد معیارها برای مقایسه روش پیشنهادی با مدل‌های گوئل-آکوموتو و S شکل تاخیری روی مجموعه داده IBM

معیار	مدل پیشنهادی	مدل نمایی	مدل S شکل تاخیری
PRR	۰/۷۱	۱/۰۴	۲۲/۲۲
SSE	۳۰/۵۰	۱۷۰/۸۲	۳۳/۸۸
MSE	۱/۶۹	۸/۱۳	۱/۶۸



شکل ۲: مقایسه تطبیق روش پیشنهادی با مدل گوئل-آکوموتو و مدل S شکل تاخیری بر روی مجموعه داده IBM

۶ بحث و نتیجه‌گیری

مدل‌های رشد قابلیت اعتماد نرم‌افزار برای اندازه‌گیری قابلیت اعتماد و کنترل فرایند آزمون نرم‌افزار مورد استفاده قرار گرفته‌اند. بعضی از مقیاس‌ها مانند تعداد خطاهای اولیه، شدت شکست، قابلیت اعتماد در یک دوره زمان خاص، تعداد خطاهای باقیمانده در نرم‌افزار، متوسط زمان بین شکست‌ها و متوسط زمان رسیدن به شکست توسط SRGMها تعیین می‌شوند. مدل جدید رشد قابلیت اعتماد ارائه شده در این مقاله خطاهای نرم‌افزاری را به سه دسته خطاهای ساده، خطاهای پیچیده مستقل و خطاهای پیچیده وابسته تقسیم کرده، تاثیر عامل تاخیر زمان در حذف و اصلاح خطاهای پیچیده را مورد توجه قرار داده است. نتایج حاصل از جداول و نمودارهای تطبیق مدل پیشنهادی با مجموعه داده شکست نرم‌افزار فرماندهی و کنترل زمان واقعی و مجموعه داده‌های شکست IBM نشان می‌دهد مفروضات مدل پیشنهادی به آنچه در جهان واقعی رخ می‌دهد نزدیکتر بوده، پاسخ منطقی‌تری جهت ارزیابی تعداد خطاهای باقیمانده در نرم‌افزار و تخمین وارزیابی قابلیت اعتماد نرم‌افزار ارائه می‌کند. بنابراین اطلاعات دقیق‌تری درباره کیفیت نرم‌افزار یا زمان عرض محصول

به بازار در اختیار می‌گذارد. در پایان پیشنهاد می‌شود مدل فوق با در نظر گرفتن احتمال بروز خطاهای جدید حین اشکال‌زدایی، به واقیت نزدیک‌تر شوند.

مراجع

Goel, A. L. (1985), Software Reliability Models: Assumptions, Limitations and Applicability, *IEEE Trans. On Software Engineering*, **11**, 1411-1423.

Goel, A. L. and Okumoto, K. (1981), Optimal Release Time for Software Systems Based on Reliability and Cost Criteria, *Journal of Systems and Software*, **1**, 315-318.

Huang, C. Y. and Lin, C. T. (2004), Software Reliability Growth Models Incorporating Fault Dependency with Various Debugging Time Lags, Proceedings of the 28th Annual International Computer Software and Application Conference, Hong Kong, China, 186-191.

Huang, C. Y. and Lin, C. T. (2006), Software Reliability Analysis by Considering Fault Dependency and Debugging Time Lag, *IEEE Trans on Reliability*, **55**, 436-450.

Kapour, P. K., Garg, R. B. and Kumar, S. (1999), *Contributions to Hardware and Software Reliability*, Word Scientific, Singapore.

Kapour, P. K., Pham, H., Gupta, A. and Jha, P. C. (2011), *Software Reliability Assessment with or Applications*, Springer-Verlag, London.

Kapour P. K. and Younes, S.(1995), Software Reliability Growth Model with Error Dependency, *Microelectron.Reliab.*, **35**, 273-278.

Lie, R. and Garg, M. (2012), A Detailed Study of NHPP Software Reliability Models, *Journal of Software*, **7**, 1296-1305.

Musa, J. D., Lannino, A. and Okumoto, K. (1987), *Software Reliability: Measurement, Prediction and Application*, MCGraw-Hill, New York.

Ohba, M. (1984), Software Reliability Analysis Models, *IBM Journal of Research and Development*, **28**, 428-443.

Pham, H. (2006), *System Software Reliability*, Springer Series in Reliability Engineering, London.

Yamada, S. (2000), Software Reliability Models and Their Applications: A Survey, In: *Proceedings of the International Seminar on Software Reliability of Man-Machine Systems*, Kyoto University, Japan, 56-80.

Yamada, S. (2014), *Software Reliability Modeling: Fundamentals and Applications*, Springer.

Yamada, S., Ohba, M. and Osaki, S. (1983), S-Shaped Reliability Growth Modeling for Software Error Detection, *IEEE Trans. On Reliability*, **32**, 475-484.

Yamada, S., Ohba, M. and Osaki, S. (1984), S-Shaped Software Reliability Growth Models and Their Applications, *IEEE Trans. On Reliability*, **33**, 289-292.

Yamada, S., Tokunou, K. and Osaki, S. (1992), S-Imperfect Debugging Models with Fault Introduction Rate for Software Reliability Assessment, *International Journal of System Science*, **23**, 2241-2252.

Zacks, S. (1992), *Introduction to Reliability Analysis*, Springer-Verlag, London.