Ein Stichprobenverfahren zur Abschätzung der Fehlerzahl

Karl-Heinz Wichert, Melanie Späth

Capgemini sd&m AG
Carl-Wery-Str.42
81739 München
karl-heinz.wichert@capgemini-sdm.com
melanie.spaeth@capgemini-sdm.com

Abstract: Eine häufige Ursache für ineffiziente Tests und mangelnde Testendequalität ist die ungenügende Eingangsqualität der Testobjekte. Es fehlen praktisch anwendbare Verfahren, mit denen die Qualität der Testobjekte frühzeitig im Test bewertet werden kann. Dieses Dokument beschreibt ein Stichprobenverfahren, mit dem die Gesamt-Fehlerzahl eines Softwaresystems nach Durchführung weniger Testfälle abgeschätzt werden kann. Das Verfahren kombiniert ein einfaches mathematisches Modell für Fehlerabdeckung mit der Methodik des Pairwise-Testens. Mit dieser Information kann frühzeitig entschieden werden, ob ein weiterer Test sinnvoll ist oder konstruktive Maßnahmen zu diesem Zeitpunkt besser geeignet sind. Dadurch kann ineffizientes Testen vermieden, der Testaufwand genauer abgeschätzt und die Qualitätsverantwortung weg vom Testen und hin zum Entwicklungsprozess delegiert werden.

1 Einleitung

Testverfahren haben meistens das Ziel, möglichst viele Fehler eines Softwaresystems zu finden. Häufig erfolgt der Test in mehreren Stufen: Zunächst testet jeder Entwickler seinen Code, üblicherweise mit Whitebox-Techniken. Dann beginnen weitere Testschritte, die von Testteams mittels Blackbox-Verfahren durchgeführt werden (z.B. der Systemtest). Sie sollen die verbleibenden Fehler finden. Dabei treten oft folgende Probleme auf:

- **Die Qualitätsverantwortung wird dem Systemtest untergeschoben.**
  Entwickler verlassen sich darauf, dass der Systemtest alle verbleibenden Fehler findet. Damit sinkt die Motivation, bereits bei der Programmierung hochwertige Qualität zu produzieren und durch den Entwicklertest abzusichern.

- **Blackbox-Test ist ineffizient bei hoher Fehlerdichte.**
  Enthält die Software noch viele Fehler, sind Whitebox-Tests, Code-Reviews etc. erwiesenermaßen effizienter, um die Fehler zu beheben [Jo97].

- **Der Testaufwand ist nicht abschätzbar.**
  Beschreibung, Kategorisierung, Nachstellung und Verwaltung der Fehler sind wesentliche Aufwandstreiber beim Systemtest. Eine vernünftige Planung für den Systemtest ist somit nur möglich, wenn die Fehlerzahl ungefähr bekannt ist.

Dies sind alltägliche Erfahrungen in vielen Projekten. Häufig wird jedoch, statt die Ursache zu ergründen, entweder Testen an sich für ineffizient gehalten oder das Testvorgehen im konkreten Projekt angezweifelt. Die Probleme können adressiert werden, wenn möglichst früh und mit wenig Aufwand die Fehlerzahl abgeschätzt und auf dieser Basis entschieden wird, ob die Software bereits die nötige Reife für den Systemtest besitzt.

Ein hierfür geeignetes Verfahren sollte folgende Anforderungen erfüllen:

A1. Es liefert eine Abschätzung der Fehlerzahl, die beim Test auftreten wird. Eine hohe Genauigkeit ist wünschenswert, aber nicht notwendig.

A2. Der Aufwand zur Abschätzung der Fehlerzahl ist sehr viel kleiner als der Aufwand für den gesamten Systemtest.

A3. Das Verfahren ist anwendbar ohne Kenntnis über die interne Struktur und Vorgeschichte der Software. Nur fachliches Wissen wird vorausgesetzt.

A4. Das Ergebnis ist nachvollziehbar.

Unser Vorschlag für ein solches Verfahren besteht aus zwei Schritten: Zunächst wird ermittelt, wie viele Fehler die Software ungefähr enthält. Dann wird entschieden, ob diese Fehlerzahl für den Eingang in den Systemtest akzeptabel ist. Da der zweite Schritt stark abhängig ist von den Qualitätszielen des jeweiligen Projektes, werden wir diesen nur am Rande betrachten.

2 Related Work

Es gibt bereits eine Reihe von Ansätzen zur Fehlerprognose, wobei das Ziel meistens in einer Vorhersage der Zuverlässigkeit der Software für die Produktion ist [FN99]. Solche Prognosen sind schwierig, weil im Test gefundene Fehler sich nur mit erheblicher Unsicherheit auf die Zuverlässigkeit im Betrieb abbilden lassen. Unser Ziel ist bescheidener; wir wollen lediglich die Reife einer Software für den Test nachweisen.

Eine in der Industrie gängige Eingangsprüfung für Systemtests sind sog. Smoke-Tests [SL05], [Bi99]. Diese prüfen, ob das System grundsätzlich lauffähig und testbar ist. Wir halten dieses Kriterium für notwendig, aber nicht hinreichend. Wir wollen darüber hinaus auch prüfen, ob die Software effizient getestet werden kann.

Häufig untersucht wurden z.B. sog. Reliability Growth Models [Ly95], [FP98]. Diese prognostizieren die Zuverlässigkeit eines Systems aus den Ausfalldaten operationeller Tests. Für unsere Zwecke sind die Modelle nicht geeignet, weil wir die Fehlerabschätzung zu einem frühen Zeitpunkt im Entwicklungszyklus benötigen.

Zudem gibt es Veröffentlichungen zur Aufwandschätzung für Tests an Hand gemittelter Erfahrungswerte von Fehlerdichten [KV03]. Erfahrungsgemäß schwanken Fehlerdichten beträchtlich [FO00]. Unser Verfahren soll feststellen, ob die Fehlerdichte in der untersuchten Software im üblichen Bereich liegt oder stark davon abweicht.

Viele Arbeiten, darunter [Bi99], [Wo95], haben sich mit dem Zusammenhang von Codeabdeckung mit Fehlerabdeckung beschäftigt. Durch Messung der Codeabdeckung beim Ausführen von Testfällen lässt sich damit die Gesamt-Fehlerzahl extrapolieren. Diese Methoden betrachten wir als hilfreich zur Überprüfung der Qualität von Entwicklertests. Für unsere Zwecke sind sie nicht anwendbar, da es sich um Whitebox-Verfahren handelt, für deren Durchführung Wissen über die Codestruktur notwendig ist.

3 Eigener Ansatz

Um eine Aussage über die ungefähre Anzahl der Fehler in einem Software-System zu erhalten, verwenden wir ein Stichprobenverfahren. Die Stichprobe ziehen wir aus der Menge aller theoretisch möglichen Testfälle. Die Fehlerzahl, die bei Ausführung aller Testfälle gefunden werden würde, ist genau die unbekannte Fehlerzahl, die wir ermitteln wollen. Klären müssen wir, welchen Anteil A_F der gesamten Fehler wir finden, wenn eine Stichprobe T der gesamten Testfallmenge gezogen wird. Dies hängt von der Wahl der Testfälle ab. Dass die Fehler zudem nicht über die Module hinweg gleichverteilt sind, führt bei Verwendung eines Stichprobenverfahrens naturgemäß zu einer statistischen Ungenauigkeit. Diese nehmen wir in Kauf; sie begrenzt die Prognosegenauigkeit.

Wir diskutieren nun ein einfaches Modell für die Abhängigkeit des Fehleranteils von der Anzahl durchgeführter Testfälle, $A_F(T)$. Die grundlegende Annahme dabei ist, dass die Wahrscheinlichkeit, mit einem Testfall einen neuen Fehler zu finden, linear abnimmt mit der Anzahl der bereits durchgeführten Testfälle. Es gilt also

$$\Delta A_F \sim (1- A_F) \Delta T,$$

wobei T: Anzahl bereits durchgeführter Testfälle; A_F: Fehlerabdeckung = Anteil der bereits entdeckten Fehler an allen Fehlern; ΔA_F: Änderung der Fehlerabdeckung; ΔT: Anzahl neu durchgeführter Testfälle.

Bei einer hohen Anzahl durchgeführter Testfälle kann Δ als klein betrachtet werden. Aus obiger Gleichung ergibt sich eine einfache Differentialgleichung mit folgender Lösung:

$$A_F (T) = 1-e^{-\lambda * T}$$

Die Funktion ist eine aus vielen natürlichen Prozessen bekannte Sättigungskurve mit nur einem freien, zunächst unbekannten, Parameter, λ. Bei geschickter Wahl von Testfällen steigt die Fehlerabdeckung schneller an als bei ungeschickter Wahl. Diese Wahl repräsentiert λ. Abbildung 1 zeigt fiktive Kurven für verschiedene Werte von λ.

Für die Fehlerprognose müssen wir das λ finden, das für ein gegebenes System und eine gegebene Abfolge von Testfällen die reale Kurve näherungsweise am besten beschreibt.

Im Folgenden zeigen wir, wie dieser Parameter bestimmt werden kann für Testobjekte, für die sich Pairwise-Testen [Ma85], [Wi00] anwenden lässt. Pairwise-Testen ist eine Blackbox-Testmethode (Anforderung A3), die Testfälle durch gezielte Kombination von Parametern bildet. Zur Erzeugung von Pairwise-Testfällen gibt es verschiedene Algorithmen und Werkzeuge, darunter [Co97], [HR04].

Für den Test mit einem vollständigen Pairwise-Testfallsatz geben [WK01], [KR02] und [KWG04] Fehlerabdeckungen zwischen 50% und 100% an. Dies legen wir zugrunde für die Annahme, dass durch die Pairwise-Methode eine optimale Fehlerabdeckung erreicht wird und diese bei 85% ± 15% liegt. Damit bestimmen wir für ein System bei bekannter Anzahl Pairwise-Testfälle das λ für die Durchführung einer optimalen Testfallmenge:

$$\lambda = -\ln(1- A_p)/ T_p ,$$

wobei T_p: Anzahl aller Pairwise-Testfälle; A_p: Fehlerabdeckung bei deren Ausführung.

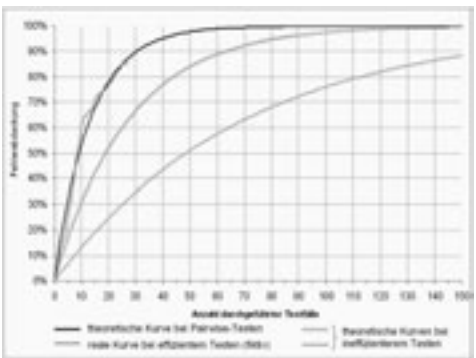

Abbildung 1 - Fehlerabdeckung abhängig von der Effizienz des Testvorgehens

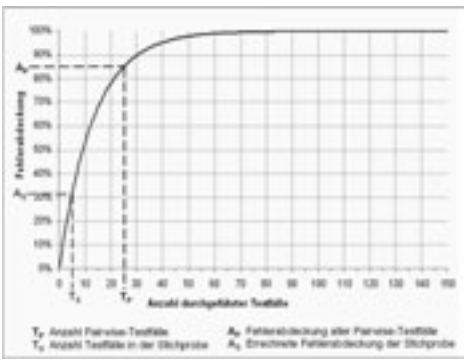

Abbildung 2 - Verlauf der Fehlerabdeckung bei optimaler Testfallwahl

Durch die Kenntnis der Kurve können wir nun für A_p=0,85 bei Wahl einer Testfall-Teilmenge T den zugehörigen Fehlerabdeckungsgrad A_F bestimmen als

$$A_F (T) = 1- e^{-0,15(T/ Tp)}.$$

Die geschätzte Gesamtfehlerzahl im System ist dann:

$$F_{Gesamt} = F_{Test}/ A_T = F_{Test} / (1 - (1 - A_p)^{(T_{Test} / T_p)}),$$

wobei F_{Test}: gefundene Fehleranzahl; T_{Test}: durchgeführte Testfälle; T_p: Anzahl aller Pairwise-Testfälle; A_p: Fehlerabdeckung; A_T: erreichte Fehlerabdeckung.

3.1 Anwendung des Verfahrens

Basierend auf dem mathematischen Modell werden folgende Schritte durchgeführt:

1. Ermittle für das zu prüfende System die Eingabe- und sonstigen Parameter und deren mögliche Werte durch Äquivalenzklassen- und Grenzwertanalyse.

2. Berechne mit Hilfe von Werkzeugen die Anzahl aller Pairwise-Testfälle.

3. Führe eine Teilmenge der Pairwise-Testfälle durch. Faustregel hierbei: Wähle die Testfallanzahl so, dass mindestens 20% Fehlerabdeckung erreicht werden.

4. Zähle die Anzahl der Fehler, die bei der Durchführung auftreten.

5. Wende das entwickelte mathematische Modell an. Die Gesamtfehlerzahl kann mit einem einfachen Tabellenkalkulationsprogramm berechnet werden.

Das Ergebnis liefert die geschätzte Gesamtzahl der Fehler in dem getesteten System (Anforderungen A1, A4). Diese wird gegen das Abbruchkriterium geprüft. Die maximal zu erwartende Anzahl von Fehlern wird in der Testplanung festgelegt: Deren Protokollierung und resultierende Retests sind wesentliche Größen der Testaufwandsschätzung.

Da die Schritte 1 bis 3 ohnehin im Systemtest ausgeführt werden, ist der Zusatzaufwand durch unser Verfahrens gering (Anforderung A2). Das Verfahren anzuwenden ist dann sinnvoll, wenn der Aufwand für die noch verbleibende Testdurchführung deutlich höher ist als die Hochrechnung der Fehlerzahl. Dies ist in der Praxis nur dann nicht der Fall, wenn die Durchführung aller verbleibenden Testfälle vollständig automatisiert wurde.

3.2 Genauigkeit des Verfahrens

Bezüglich der Fehlerabdeckung bei Ausführung aller Pairwise-Testfälle nehmen wir an, dass sie bei 85 ± 15% liegt. Die Anwendung bekannter Regeln zur Fehlerfortpflanzung ergibt eine Genauigkeit der Hochrechnung von ± 50%.

Bei der Hochrechnung der gefundenen Fehler zur Ermittlung der Gesamt-Fehlerzahl tritt zusätzlich ein statistischer Fehler auf. Die Ungenauigkeit ist stark abhängig von der Zahl gefundener Fehler (Gesetz der großen Zahlen). Dies führt dazu, dass Software mit wenigen Fehlern eher zu gut bewertet wird. Bei einer nicht zu kleinen Stichprobe ist der Fehler fast immer geringer als der oben ermittelte Fehler durch die Ungenauigkeit von T_p. Deshalb kann eine Genauigkeit des Verfahrens von ± 50% angenommen werden.

4 Zusammenfassung und Ausblick

Mit unserem Verfahren können wir nach Ausführen weniger Testfälle auf die Gesamtfehlerzahl schließen. Das Modell ist anwendbar auf Systeme, die mittels Pairwise-Testen testbar sind. Die Genauigkeit liegt hierbei bei ± 50%. Dies ist für viele Zwecke ausreichend. Höhere Genauigkeit kann durch Ausführen von mehr Testfällen erreicht werden.

Das Verfahren kann angewendet werden, um die Qualität eines Systems zu prüfen und abhängig von dieser weitere Maßnahmen zu ergreifen.

Insbesondere verhindert es, dass Software mit zu schlechter Qualität in den Systemtest gelangt und damit zu hohen Testaufwänden führt. Bei unkritischen Systemen und gemessener guter Qualität kann der Test unter Umständen schnell beendet werden.

Wir erproben das Verfahren derzeit in konkreten Projekten bei Capgemini sd&m.

Interessant wäre zudem, den Anwendungsbereich auf Systeme zu erweitern, für die Pairwise-Testen keine geeignete Testfallauswahl bietet. Dies ist beispielsweise der Fall für Systeme mit verschiedenen Prozessdurchläufen mit jeweils unterschiedlichen Parametern. Das Prinzip unseres Verfahrens bleibt auch für diese Art von Systemen anwendbar. Nur die Justierung der Fehlerkurve muss dann auf andere Weise erfolgen.

Literaturverzeichnis

[Bi99] Binder, R.: Testing Object-Oriented Systems. Addison-Wesley, 1999.

[Co97] Cohen, D. M. et. al.: The AETG System: An Approach to Testing Based on Combinatorial Design. In (IEEE Computer Society Hrsg.): IEEE Transactions on Software Engineering (TSE), 23(7), 1997; S. 437-444.

[HR04] Hartman, A.; Raskin, L.: Problems and Algorithms for Covering Arrays. In: Discrete Mathematics, 284(1-3), 2004; S. 149-156.

[FN99] Fenton, N.E.; Neil, M.: A Critique of Software Defect Prediction Models. In (IEEE Computer Society Hrsg.): IEEE TSE, 25(5), 1999; S. 675-689.

[FO00] Fenton, N.E.; Ohlsson, N.: Quantitative Analysis of Faults and Failures in a Complex Software System. In (IEEE Computer Society Hrsg.): IEEE TSE, 26(8), 2000; S. 797-814.

[FP98] Fenton, N.E.; Pfleeger, S.L.: Software Metrics: A Rigorous and Practical Approach (2nd ed.). PWS Publishing Co., 1998.

[Jo97] Jones, C.: Software Quality - Analysis and Guidelines for Success. Int. Thomson Computer Press, 1997.

[KR02] Kuhn, D.R.; Reilly, M.J.: An Investigation of the Applicability of Design of Experiments to Software Testing. In (IEEE Computer Society Hrsg.): Proc. 27th Annual NASA Goddard Software Engineering Workshop, Washington, 2002; S. 91-95.

[KV03] Kropfitsch, D.; Vigenschow, U.: Das Fehlermodell: Aufwandsschätzung und Planung von Tests. Objekt-Spektrum, (6), 2003; S. 30-35.

[KWG04] Kuhn, D.R.; Wallace, D.R.; Gallo, A.M.: Software Fault Interactions and Implications for Software Testing. In (IEEE Computer Society Hrsg.): IEEE TSE, 30(6), 2004; S. 418-421.

[Ly95] Lyu, M.R.: Handbook of Software Reliability Engineering. McGraw-Hill, 1995.

[Ma85] Mandl, R.: Orthogonal Latin Squares: An Application of Experiment Design to Compiler Testing. In: Communications of the ACM, 28(10), 1985; S. 1054-1058.

[SL05] Spillner, A., Linz, T.: Basiswissen Softwaretest. dpunkt.verlag, Heidelberg, 2005.

[Wi00] Williams, A.W.: Determination of Test Configurations for Pair-Wise Interaction Coverage. In (Kluwer, B.V. Hrsg.): Proc. 13th Int. Conf. on Testing Communicating Systems, 2000; S. 59-74.

[WK01] Wallace, D.R.; Kuhn, D.R.: Failure Modes in Medical Device Software: An Analysis of 15 Years of Recall Data, In (World Scientific Publishing Company Hrsg.): International Journal of Reliability, Quality and Safety Engineering, 8(4), 2001; S. 351-371.

[Wo95] Wong, W.E. et. al.: Effect of Test Set Minimization on Fault Detection Effectiveness. In (ACM Hrsg.): Proc. 17th Int. Conf. on Software Engineering, Seattle, 1995; S. 41-50.