Módulo I – Fundamentos Computacionais
Sessão 5 - Aula 1 – Teste de Software – Considerações iniciais

Dr. Luiz Alberto Vieira Dias
Prof. Dr. Lineu Mialaret

ITAú Analytics – Fundamento de Computação – CEDS 111
2º SEMESTRE 2018

Teste de Software
Considerações iniciais

CEDS -111 - Teste de Software - Sessão 5 - Aula 1

Prof. Dr. Luiz Alberto **Vieira Dias** (ITA)
Prof. Dr. Adilson Marques da **Cunha** (ITA)
Prof. Dr. **Lineu** Mialaret (ITA/IFSP)

Por que testar software se é caro e demorado?

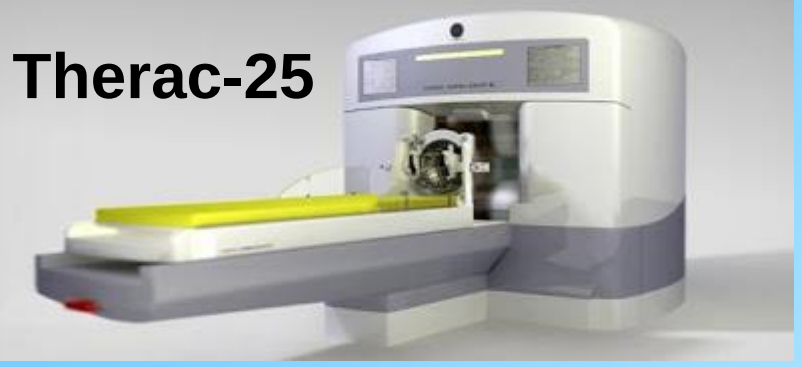
- Porque não testar sai ainda mais caro

- Não testar pode causar desastres e perda de $$$ e até vidas humanas

EXEMPLOS

- Therac-25
- Vôo inaugural do Ariane 5
- Vôo de teste do Airbus 320
- Finanças

EXEMPLOS (2)



https://youtu.be/I9gELPxPG8Q

E em Finanças? Crash na Wall Street (1987)?

- **Perda:** $500 bilhões em um dia!
- **Desastre:** Em 19 de outubro de 1987, o índice Dow Jones caiu 508 pontos, perdendo 22,6% de seu valor total. Esta foi a **maior** perda que Wall Street já sofreu em um único dia
- **Causa:** Um mercado em grande alta foi interrompido por uma série de investigações conduzidas pela SEC e por outras forças do mercado. Como os investidores fugiram de ações investigadas, um número muito grande de ordens de venda foram gerados pelos **software de compra de computadores**, quebrando sistemas e deixando os investidores efetivamente cegos

Teste de Software

- Caixa Preta
- Caixa Branca

Dificuldades do Teste

- Custo
- Tempo para Teste
- Número ”alto” de Casos de Teste

Caixa Preta

- Classes de Equivalência
- Valor de Fronteira
- Outros

Caixa Branca

- Control Flow

- Data flow

Como reduzir o número de casos de teste, sem perder a cobertura?

Testar somente os casos estatisticamente mais críticos (perde cobertura)

Usar MATEMÁTICA para reduzir o número de casos de teste

Análise Estatística

- Aplicar Análise Estatística para mostrar que realmente a técnica em questão (pairwise testing, no caso) produz o mesmo resultado que o Teste Exaustivo com (muito?) menor número de casos de teste

Considerações iniciais
Teoria – Artigo Microsoft

Pairwise Testing in the Real World: Practical Extensions to Test-Case Scenarios

Jacek Czerwonka
Microsoft Corporation
February 2008

Jacek Czerwonka works for Microsoft Corporation in one of its test organizations. For the last few years, he has been involved in the design and implementation of pairwise-related tools, and the evangelization of pairwise testing. You can reach Jacek at **jacekcz@microsoft.com**

Summary: **Pairwise testing** has become an indispensable tool in a software tester's toolbox. This article pays special attention to usability of the pairwise-testing technique. In particular, it focuses on ways in which the pure pairwise-testing approach must be modified to become practically applicable, and on the features that tools must offer to support the tester who is trying to use pairwise testing in practice. (15 printed pages)

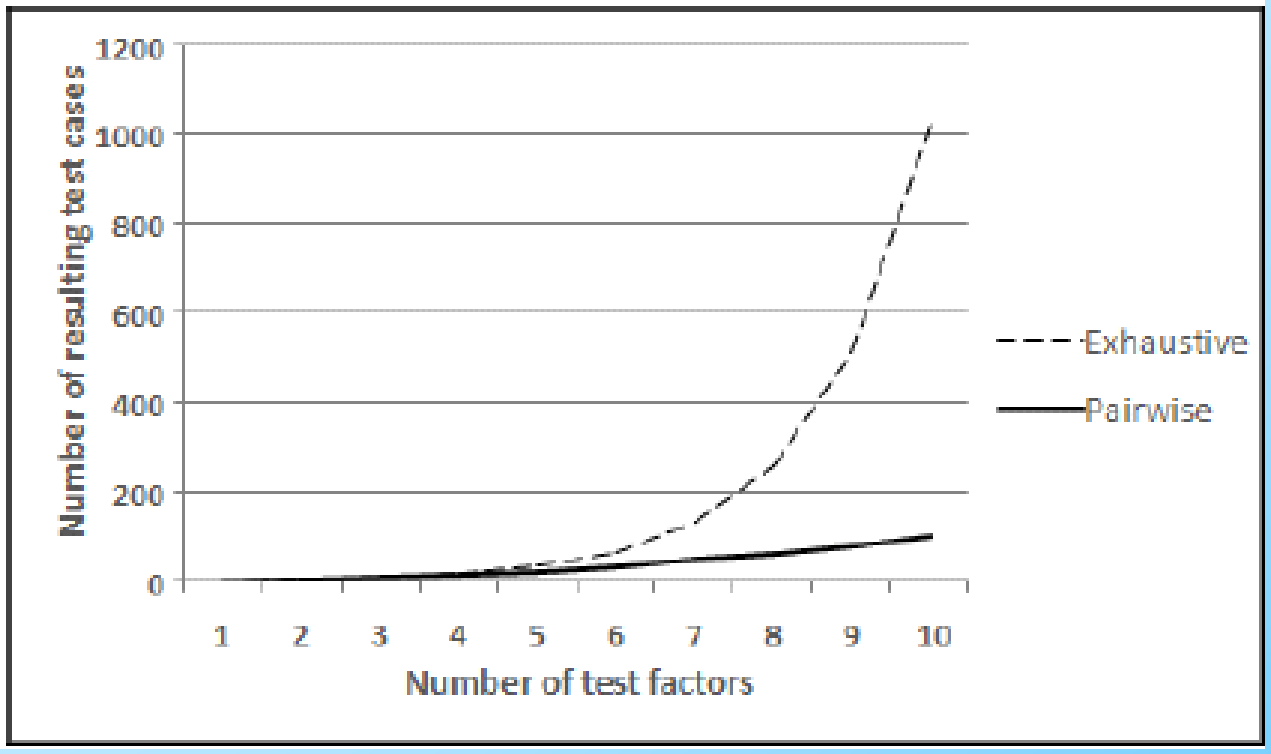
Contents

Overview

Pairwise testing has become an **indispensable** tool in a software tester's toolbox. The technique has been known for almost 20 years [22], but it is only in the last five years that we have seen a tremendous increase in its popularity

So far, information on at least 20 tools that can generate pairwise test cases has been published [1]. Most tools, however, lack practical features that are necessary for them to be used in the industry

A set of possible inputs for any nontrivial piece of software is too large to be tested exhaustively. Techniques such as equivalence partitioning and boundary-value analysis [17] help convert even a large number of test levels into a much smaller set with comparable defect-detection power. Still, if software under test (SUT) can be influenced by a number of such factors, exhaustive testing again becomes impractical.



References

[1] Czerwonka, J. “Pairwise Testing: Available Tools.” Available at:

[2] Ammann P. E., and A. J. Offutt. “Using Formal Methods to Derive Test Frames in Category-Partition Testing.” In Ninth Annual Conference on Computer Assurance (COMPASS’94), Gaithersburg, MD, pages 69–80, 1994.

[3] Bach, J., and P. Shroeder. “Pairwise Testing: A Best Practice that Isn’t.” In Proceedings of the 22nd Pacific Northwest Software Quality Conference, pages 180–196, 2004.

[4] Burr, K., and W. Young. “Combinatorial Test Techniques: Table-Based Automation, Test Generation, and Test Coverage.” In Proceedings of the International Conference on Software Testing, Analysis, and Review (STAR), San Diego, CA, 1998.

[5] Burroughs, K., A. Jain, and R. L. Erickson. “Improved Quality of Protocol Testing Through Techniques of Experimental Design.” In Proceedings of the IEEE International Conference on Communications (Supercomm/ICC’94), May 1–5, New Orleans, LA, pages 745–752, 1994.

[6] Cohen, D. M., S. R. Dalal, M. L. Fredman, and G. C. Patton. “The AETG System: An Approach to Testing Based on Combinatorial Design.” IEEE Transactions on Software Engineering, 23(7), 1997.
AETG is a trademark of Telecordia Technologies.
[7] Cohen, D. M., S. R. Dalal, J. Parelius, and G. C. Patton. “The Combinatorial Design Approach to Automatic Test Generation.” IEEE Software, 13(5):83–87, 1996.
[8] Colbourn, C. J., M. B. Cohen, and R. C. Turban. “A Deterministic Density Algorithm for Pairwise Interaction Coverage.” In Proceedings of the IASTED International Conference on Software Engineering, 2004.
[9] Dalal, S. R., A. J. N. Karunanithi, J. M. L. Leaton, G. C. P. Patton, and B. M. Horowitz. “Model-Based Testing in Practice.” In Proceedings of the International Conference on Software Engineering (ICSE 99), New York, pages 285–294, 1999.

[10] Dunietz, I. S., W. K. Ehrlich, B. D. Szablak, C. L. Mallows, and A. Iannino. “Applying Design of Experiments to Software Testing.” In Proceedings of the International Conference on Software Engineering (ICSE 97), New York, pages 205–215, 1997.
[11] Grindal, M., J. Offutt, and S. F. Andler. “Combination Testing Strategies: A survey.” GMU Technical Report, 2004.
[12] Hartman, A., and L. Raskin. “Problems and Algorithms for Covering Arrays.” Discrete Mathematics, 284(1–3):149–56, 2004.
[13] Kuhn, R., and M. J. Reilly. “An Investigation of the Applicability of Design of Experiments to Software Testing.” In Proceedings of the 27th NASA/IEEE Software Engineering Workshop, NASA Goddard Space Flight Center, 2002.
[14] Lei, Y., and K. C. Tai. “In-Parameter-Order: A Test-Generation Strategy for Pairwise Testing.” In Proceedings of the Third IEEE International High-Assurance Systems Engineering Symposium, pages 254–261, 1998.

[15] Malaiya, Y. K. “Antirandom Testing: Getting the Most Out of Black-Box Testing.” In Sixth International Symposium on Software Reliability Engineering, Oct. 24–27, 1995, pages 86–95, 1996.
[16] Mandl, R. “Orthogonal Latin Squares: An Application of Experiment Design to Compiler Testing.” Communications of the ACM, 28(10):1054–1058, 1985.
[17] Myers, G. J. The Art of Software Testing. New York: John Wiley and Sons, 1979.
[18] Sherwood, G. “Effective Testing of Factor Combinations.” In Proceedings of the Third International Conference on Software Testing, Analysis and Review, Washington, DC, pages 133–166, 1994.
[19] Shimokawa, H., and S. Satoh. “Method of Setting Software Test Cases Using the Experimental Design Approach.” In Proceedings of the Fourth Symposium on Quality Control in Software Production, Federation of Japanese Science and Technology, pages 1–8, 1984.

[20] Smith, B., M. S. Feather, and N. Muscettola. “Challenges and Methods in Testing the Remote Agent Planner.” In Proceedings of AIPS, 2000.
[21] Tai, K. C., and Y. Lei. “A Test-Generation Strategy for Pairwise Testing.” IEEE Transactions of Software Engineering, 28(1), 2002.
[22] Tatsumi, K. “Test-Case–Design Support System.” In Proceedings of the International Conference on Quality Control (ICQC), Tokyo, 1987, pages 615–620, 1987.
[23] Wallace, D. R., and D. R. Kuhn. “Failure Modes in Medical Device Software: An Analysis of 15 Years of Recall Data. International Journal of Reliability, Quality and Safety Engineering, 8(4), 2001.
[24] Williams, A. W. “Determination of Test Configurations for Pairwise Interaction Coverage.” In Proceedings of the 13th International Conference on Testing Communicating Systems (Test-Com 2000), pages 59–74, 2000.
[25] Williams, A. W., and R. L. Probert. “A Practical Strategy for Testing Pairwise Coverage of Network Interfaces.” In Proceedings of the Seventh International Symposium on Software Reliability Engineering (ISSRE ’96), page 246, 1996.