The Wheel of Fault Injection

Grenoble, October the 11th, 2016

jean-louis.lanet@inria.fr

Inria
INVENTEURS DU MONDE NUMÉRIQUE

50

An old research field

Design and Use of Fault Simulation for Saturn Computer Design

FRED H. HARDIE AND ROBERT J. SUHOCKI

4000 Saturn instructions in either
s (up to 33 faults per IBM 7090 run)

BIBLIOGRAPHY

[1] P. W. Case, H. H. Graff, L. E. Griffith, A. R. Leclercq, W. B. Murley, and T. M. Spence, "Solid logic design automation," *IBM J. R & D*, vol. 8, pp. 127–140, April 1964.
[2] R. E. Ide and R. J. Suhocki, "User's manual, logic system simulator," IBM, Owego, N. Y., Rept. 65-578-01, January 8, 1965.
[3] IBM Dept. 578, "Final report work package 3860," IBM 65-578-03, January 12, 1965.
[4] "AES-EPO study program—final study report," IBM, 65-562-012.

During the subject study, several hundred selected and randomly chosen failures were simulated by a

The *failure injection program* is run only when a group of faults is to be compiled and simulated.

Full central processing unit simulation while containing in one 32K memory the complete compiled logic

mation of one logical function into another. A fault could be considered to be a terminal node of a logic block stuck at logical ONE or logical ZERO. If the user

Parallel Error Simulation

In parallel error simulation, each bit position in the 36-bit 7090 computer word can be used to contain the binary value of a specified fault or a specified group of multiple faults. In actual practice, the Saturn design

The path to failure

- Fault impact the hardware (bit-flip, stuck at,…)
- Error is an activated fault,
  - Could remain dormant,
  - Can be intermittent or permanent,
  - Impact the software.

The path to failure

- Fault impact the hardware (bit-flip, stuck at,…)
- Error is an activated fault,
- Failure is the propagation or an error in the system such that the service is no more nominal.
  - No effect,
  - Detected,
  - Time out (freeze)
  - Silent Data Corruption (SDC)
- Failure FI *vs.* Security FI
  - Fault distribution
  - Reduced attack surface if no side effect

FI techniques

- Hardware based
  - Lack of repeatability, sampling, scalability
  - Fist, Messaline, Rifle,…
- Simulation based
  - High level models: VHDL model, SystemC,…, slow, sampling, small size
  - Mefisto-C, VERIFY,…
- Emulation Based
  - Hardware prototyping on FPGA emulation system, sampling, not the “real HW”
  - Antoni, Civerra…

SWIFI

- Software Implemented FI
  - Either pre run time (*a.k.a* mutant, static, compile) or run time (*a.k.a.* saboteur, dynamic)
  - System Under Test
    - Pure software: processor and/or system virtualization (à la SmartCM, Celtic)
    - Use of external simulator (à la Warsaw)
    - Use of a external hardware (à la EFS)
  - State space, time granularity…
  - `If` interface `then`

21

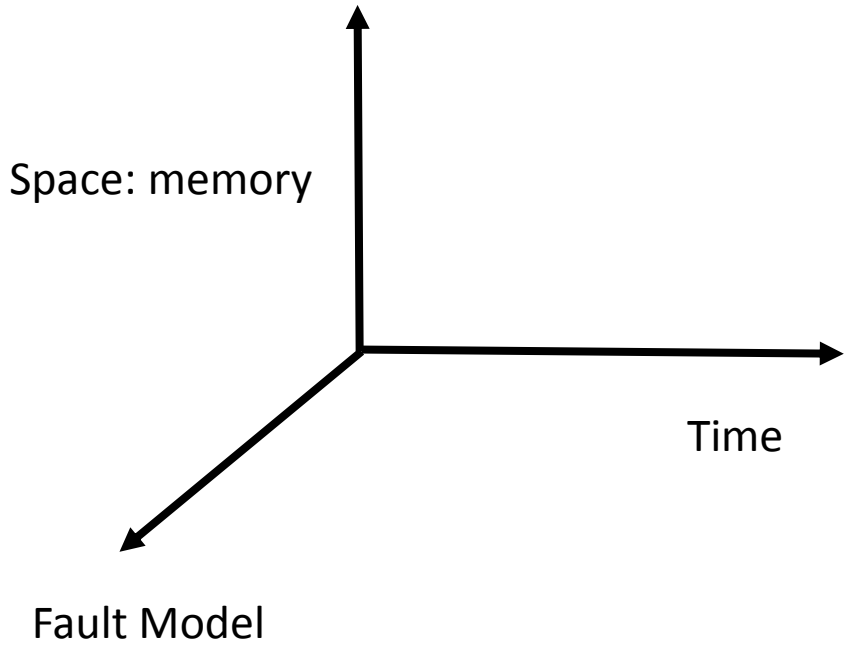
Combining Software-Implemented and Simulation-Based Fault Injection into a Single Fault Injection Method*

Jens Güthoff and Volkmar Sieh

Department of Computer Science III
University of Erlangen-Nürnberg
Martensstr. 3, 91058 Erlangen, Germany
email: {jsguetho,vrsieh}@immd3.informatik.uni-erlangen.de

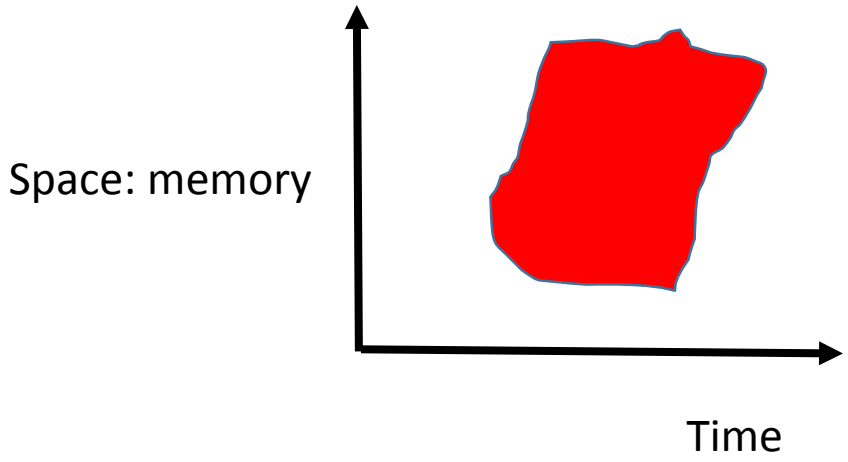
An old problem: 1995 (J. Güthoff &V. Sieh)

- From a volume to a plane



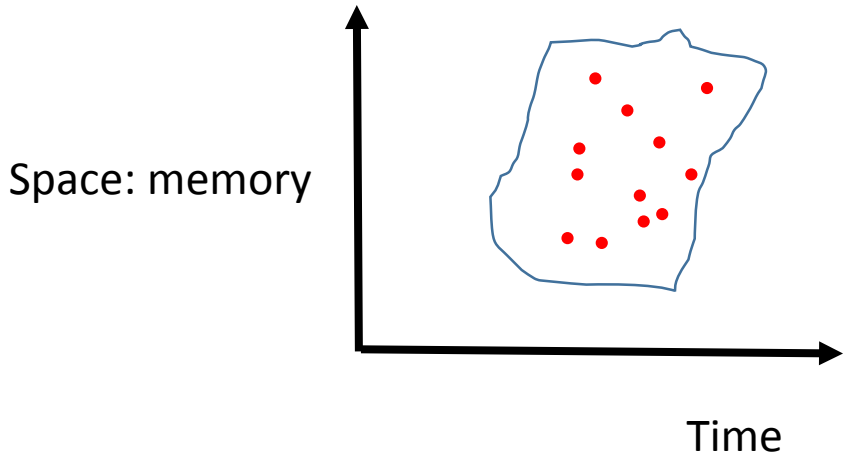
An old problem: 1995 (J. Güthoff &V. Sieh)

- From a volume to a plane
- Fix a fault model
- Prune the space & time



An old problem: 1995 (J. Güthoff &V. Sieh)

- From a volume to a plane
- Fix a fault model
- Prune the space & time
- Most of the tools use Sampling
- Except:
  - Relyzer, Fail*, SmartInjector & SmartCM



Optimization

- Data input
  - Chillarege (1985) Error latency depends of the workload
- Time
  - Benyo (1994) shows that 99% are active in the 200 next instructions
- Space
  - Güthoff & Sieh (1995) : Operational *profile based fault injection*, wwr & rww
  - Tsai (1999), stress based & path-based FI & live memory
  - Barbosa (2005) 90% of randomly injected fault are never activated.

Optimization

- Barbosa (2005) optimization *Inject on read* reduces between 2 and 5 order of magnitudes the state space.
- SmartInjector (2013) claims 99,81% of initial fault removed and 36% reduced simulation time *vs.* Relyzer.
- With its pruning heuristic, Fail* (2014) reduces 2 order of magnitude the state space

Pruning

- Conservative def/use analysis (Smith 1995)
- Fault outcome Prediction Benso (1998), Berrojo (2002),
  - State comparisons to reduce the simulation time *vs.* GR
- Equivalence Class fault pruning (Hari 2011), (Hari 2014)
  - Control equivalence & data equivalence
- Known outcome fault pruning (Li 2013)
  - Illegal instruction, unused bits, logic masking bit, memory access instructions
- Evolutionary pruning (Schmirner 2014)
  - Genetic algorithm for state equivalence

8

Open source tools

- Grinder, https://github.com/DEEDS-TUD/GRINDER
- LLFI, https://github.com/DependableSystemsLab/LLFI
- FlipIt, https://github.com/aperson40/FlipIt
- KFI, https://github.com/ut-osa/fault-injection
- VulFI, http://utahfmr.github.io/VULFI/
- Fail, https://github.com/danceos/fail
- GemFI, https://github.com/koparasy/gemfi
- Cfi-c, http://cfi-c.gforge.inria.fr/

8

Smart Card based Simulators

- 2004, No Name, Rothbart *et alii*,
  - Simulation based: SystemC
  - Single stuck at fault model
  - Memory and peripherals

Smart Card based Simulators

- 2004, No Name, Rothbart *et alii*,
- 2009, SmartCM, A. Sere PhD,
  - SWIFI at Java level, static,
  - No sampling,
  - ISA-Fault model.

Smart Card based Simulators

- 2004, No Name, Rothbart *et alii*,
- 2009, SmartCM, A. Sere PhD,
- 2012, cfi-c, X. Kaufmann PhD,
  - SWIFI at C source level (but also binary), static & dynamic (gdb)
  - ISA-fault model restricted to `nop` and `jump`

Smart Card based Simulators

- 2004, No Name, Rothbart *et alii*,
- 2009, SmartCM, A. Sere PhD,
- 2012, cfi-c, X. Kaufmann PhD,
- 2014, No Name, Lackner PhD
  - Emulation with FPGA at Java level
  - Static & dynamic, sampling.
  - ISA-Fault model, NVM and RAM bit precise Fault Model

Smart Card based Simulators

- 2004, No Name, Rothbart *et alii*,
- 2009, SmartCM, A. Sere PhD,
- 2012, cfi-c, X. Kaufmann PhD,
- 2014, No Name, Lackner PhD
- 2014, No Name, Chorko *et alii*.
  - SWIFI, at Java level,
  - No sampling (?), static, cref simulator based,
  - ISA-Fault Model.

Smart Card based Simulators

- 2004, No Name, Rothbart *et alii*,
- 2009, SmartCM, A. Sere PhD,
- 2012, cfi-c, X. Kaufmann PhD,
- 2014, No Name, Lackner PhD
- 2014, No Name, Chorko *et alii*.
- 2015, EFS, L. Riviere PhD,
  - Dynamic *à la FERRARI*, sampling,
  - ISA-Fault model restricted to `jump` and `nop` and Memory

Smart Card based Simulators

- 2004, No Name, Rothbart *et alii*,
- 2009, SmartCM, A. Sere PhD,
- 2012, cfi-c, X. Kaufmann PhD,
- 2014, No Name, Lackner PhD
- 2014, No Name, Chorko *et alii*.
- 2015, EFS, L. Riviere PhD,
- 2016, Celtic, L. Dureuil PhD
  - See tomorrow !

Smart Card based Simulators

- 2004, No Name, Rothbart *et alii*,
- 2009, SmartCM, A. Sere PhD,
- 2012, cfi-c, X. Kaufmann PhD,
- 2014, No Name, Lackner PhD
- 2014, No Name, Chorko *et alii*.
- 2015, EFS, L. Riviere PhD,
- 2016, Celtic, L. Dureuil PhD
- 2016, Classifier, C. Yayaoui PhD

1

Only one important research point

- A clever approach to generate a mutant.
  - Most of the mutants are useless,
  - Model checking state space representation and comparison
  - Test techniques & compilation optimizations.
- Re-use existing frameworks,
  - Open source, maintenance,
  - Avoid processor simulation use prototype hardware.

Innovations or improvements ?