L_nC_m Fault Model: Complexity and Validation

September 2016

Fatimah Adamu-Fika, PhD

f.adamu-fika@warwick.ac.uk　　*faateemahfika+work@gmail.com*

Host Institute Address:

Department of Computer Science
University of Warwick
CV4 7AL, Coventry.
UK.

Home Institute Address:

Federal Ministry of Agriculture and & Development
Federal Department of Agric Land Resources & Climate Change Management Services
Land Use, Planning Conversation & Climate Change Services Division
FCDA, 1 Capital Drive
Area 11, Garki
P. M. B 135
Abuja, Nigeria

Abstract:

Computer systems are ubiquitous in most aspects of our daily lives, as such the reliance of end users upon their correct and timely functioning is on the rise. With technology advancement, the functionality of these systems is increasingly being defined in software. On the other hand, feature sizes have drastically decreased, while feature density has increased. These hardware trends will keep happening as technology continues to advance. Consequently, power supply voltage is ever-decreasing and clock frequency and temperature hotspots are increasing. This steady reduction of integration scales is increasing the sensitivity of computer systems to different kinds of hardware faults. In particular, the likelihood of a single high-energy ion to cause double bit upsets (DBUs, due to its energy) or multiple bit upsets (MBUs, due to the incident angle) instead of single bits upsets (SBUs) is increasing. Furthermore, the likelihood of perturbations occurring in the logic circuits is also increasing. Owing to these hardware trends it has been projected that computer systems will expose such hardware faults to the software-level and accordingly the software is expected to tolerate such perturbations to maintain correct operations, i.e., the software needs to be dependable. Thus, defining and understanding the potential impact of such faults is required to propose the right mechanisms to tolerate their occurrence. To ascertain that software is dependable, it is important to validate the software system. This is achieved through the emulation of the type of faults that are likely to occur in the field during execution of the system, and through studying the effects of these faults on the system. Often, this validation process is achieved through a technique

called fault injection, that artificially perturbs the execution of the system through the emulation of hardware faults.

Traditionally, the single bit-flip (SBF) model is used for emulating single event upsets (SEUs) and single event transients (SETs) in dependability validation. The model assumes that only an SEU or SET occurs during a single execution of the system. However, with MBUs becoming more prominent, the accuracy of the SBF model is limited. Hence, the need for including MBUs in software system dependability validation. MBUs may occur as multiple bit errors (MBEs) is a single location (memory word or register) or as single bits errors (SBEs) in several locations. Likewise, they may occur as MBEs in several locations.

In the context of software-implemented fault injection (SWIFI), the injection of MBUs in all variables is infeasible due to the exponential size of the fault space, thereby making it necessary to carefully select those fault injection points that maximises the probability of causing a failure. A fault space, is the set of all possible fault under a given fault model. Consequently, research have started looking at a more tractable model, double bit upsets (DBU) in the form of double bit-flips within a single location, L_1C_2. However, with evidence of the possibility of corruption occurring chip wide, the applicability and accuracy of L_1C_2 is restricted. Following, this research focuses on MBUs occurring across multiple locations whilst seeking to address the exponential fault space problem associated with multiple fault injections.

In general, the thesis analyses the complexity of selecting efficient fault-injection locations (injection points that would uncover vulnerabilities and/or cause system failure) for injecting multiple MBUs. In particular, it formalises the problem of multiple bit-flip injections and found that the problem is NP-complete. There are various ways of addressing this complexity: (i) look for specific cases, (ii) look for heuristic and/or (iii) weaken the problem specification.

Next, the thesis presents one approach for each of the aforementioned means of addressing complexity:

- for the specific cases approach, the thesis presents a novel DBU fault model, that manifest as two single bit-flips across two locations. In particular, the research examines the relevance of the L_2C_1 fault model for system validation. It is found that the L_2C_1 fault model induces failure profile that is different from profiles induced by existing fault models.
- for the heuristic approach, the thesis uses an approach towards dependency aware fault injection strategies to extend the L_2C_1 fault model and the existing L_1C_2 fault model into L_nC_m (multiple location, multiple corruption) fault model, where n is the number of locations to target and m the maximum number of corruptions to inject in a

given location. It proposes two heuristics to achieve this: first, select the set of potential locations and then select the subset of variables within these locations, and it examines the applicability of the proposed framework.

- for the weakening of the problem specification approach, the thesis further refines the fault space and proposes a data mining approach to reduce the cost of multiple fault injections campaigns (in terms of number of multiple fault injections experiments performed). It presents an approach to refine the multiple fault injection points by identifying a subset of these points, whereby injection into this subset alone would be as efficient as injection into the entire set.

These contributions are instrumental to advance multiple fault injections and make it an effective and practical approach for software system validation.