Data Assurance in Opaque Computations

High Assurance Systems Engineering

Guy Haworth

guy.haworth@bnc.oxon.org
Topics

- Motivation
- Systems Engineering Cycle
  - Definition: the Problem Domain and the Systems Response
  - Computation
  - Management and use of the data created
- 'Matters Arising' in computations of Endgame Tables
- The Declarative Approach
  - The generic approach and benefits
  - HOL, Chess and BDDs
- The Future: Opportunities and Challenges for Assurance
  - Parallelism
  - Community Computing, e.g. The Chess Studies Community
- Summary
Motivation

- My interest in the endgame and in the use of EGTs
  - A concern for the future integrity of EGTs
  - The 'single thread' today is the Bourzutschky/Konoval partnership

- Mathematical Background:
  - 'Unto thyself be true, as the night followeth the day' (Laertes, Hamlet)
  - Theorems have integrity
  - A search for 'The Grail': Programs with the integrity of theorems
  - Research on Proving Programs Correct ... Turing, 1949
  - 'Defensive' if not infallible programming' style
  - Rigorous approach in the '70s and '80s to
    - The Four Colour Conjecture, Mersenne Number testing

- Lifestyle globally and increasingly dependant on Systems
- Need for 'vehicles' to help teach Systems Engineering principles
The Systems Engineering Cycle

- The Scenario and the 'System Response'

- Phase 1: Definition - the author
  - models the scenario of the computation
  - analyses the requirements and designs a systems response
  - implements and tests the System Response

- Phase 2: Computation
  - the author runs the computation and generates output

- Phase 3: Use
  - the author manages the output: publishes, promulgates, comments
  - the reader uses and interprets the results of the computation
SEC Phase 1: Definition

- Translating 'real world' into a 'computer model' of same
- This task is eased by:
  - the simplicity of the scenario
  - complete knowledge about the scenario
  - the maturity of the translator: training, skill, experience
  - the method and tools used, esp. the target language
- Modelling failures arise:
  - 1.1: in setting up the initial 'static aspects' of the scenario
  - 1.2: in emulating the 'dynamic aspects' of the process
- 1.3: Inadequate testing:
  - Boundary problems, 'One out' problems
  - Testing only proves that bugs 'of certain types' do not exist
EDSAC I: First software bug

- Maurice Wilkes:
  "... the realisation came over me that a good part of the remainder of my life was going to be spent in finding the errors in my own programs."

Memoirs, p145
Implementation Error: Ariane 5 1996-6-04

Data conversion from 64-bit floating point to a 16-bit signed integer failed.

The ADA code software handler had been disabled. Cost $1Bn of your money.

A Chinook crash may have been caused by engine control sw bugs (1994)
System error: Therac 25 misuse

1985-7: 6 dead, others injured
Root cause: the 'guard' on the high-power beam was inadequate
SEC Phase 2: The Computation

- Thompson's Turing lecture 'Reflections on Trusting Trust' (1984)
  - "Nuances can be inserted at any level of the infrastructure"
  - … deliberately or accidentally

- Levels
  2.1: Hardware:
    - systematic, contingent and transient errors … chips, discs
  2.2: Microcode, kernel, operating system
  2.3: Compiler, collector, library routine
  2.4: Wrong input data … 'garbage in …'

- Consequent errors may be:
  - Systematic, contingent or transient
Systematic error: chip design

- Do we take chip integrity for granted?
- Pentium FDIV processor
- 1 in 9,000,000,000 operations wrong
- Some missing entries in a table
- Estimated cost $800m
- Intel now using HOL
Contingent error: Harvard Mark II

- The first computer bug ... but not the first bug (Edison, 1878)
Transient Error: Radar Interference

- Field computer kept falling over quickly
- When we looked out of the window for inspiration, we saw …
SEC Phase 3: Use of the Output Data

3.1 Labelling or accessing the data incorrectly
3.2 Building on inadequate foundations
3.3 Shortcomings in the user's understanding
3.4 Physical data decay – file coatings are 'plastic' in nature
3.5 Constructing poor arguments based on probabilities
EGT-specific issues in SEC Phase 1

- Ambitious modelling of subgames using chessic logic:
  - 1.1a 1986: Komissarchik's KQPKQ EGT
  - 1.1b 1987: Van Den Herik's KRP(a2)KbBP(a3) EGT
- 1.1c Hiatus in DTM EGTs: mates in $m$ but not in $m-1$
- 1.1d Forced capture by the loser: RETROENGINE, Wirth (1999)
- 1.1e FEG:
  - The 'KNNK' bug: missing 'losses in 0'
  - The 'Transparent Pawn' bug
EGT-specific issues in SEC Phase 2

- 2.1: Hardware errors, CPU, RAM, Disc [Schaeffer]
- 2.3a: Compiler errors: using 32-bit working in a 64-bit context [Schaeffer]
- 2.4a: Wrong input files:
  - 2-byte instead of 1-byte Nalimov format
  - the subgame's DTZ rather than DTZ50 EGT for a DTZ50 calculation
- 2.4b: Physical file decay
  - prevented only by using and checking signatures
EGT-specific issues in SEC Phase 3

- 3.1a: Mislabelling the output: Nalimov's mystery KBPKN stats file
- 3.1b: Using the wrong access routine: KINGSROW
- 3.1c: Using the wrong files:
  - DTC rather than DTM: watch the engine balk at actual capture!
  - Using DTZ rather than DTZ50 EGTs
  - 'Non peers' promulgated pornography under Nalimov filenames
- Thompson's EGTs
  - 3.2a Forgetting that KT's early KQPKQ EGTs ignored underpromotion
  - 3.2b Forgetting that they are White wins / does_not_win EGTs
    - Type 2 (010) zugs invisible; type 1 (121) and type 3 (020) indistinguishable
  - 3.3a Misinterpreting Thompson's depth-data
- 3.3c: Forgetting that EGTs do not include castling rights
The Declarative Approach
## The Generic Approach and Benefits

<table>
<thead>
<tr>
<th>Activity</th>
<th>Benefits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set up the 'model world', i.e. the 'givens', within the logic</td>
<td>More powerful language</td>
</tr>
<tr>
<td>Prove 'theorems' in the logic; logic engine verifies the proof</td>
<td>English-like statements</td>
</tr>
<tr>
<td>Outputs provably follow from inputs</td>
<td>Combines human induction with silicon deduction</td>
</tr>
<tr>
<td></td>
<td>Much lower risk that the outputs are not correct</td>
</tr>
</tbody>
</table>

HOL is the (Higher Order) Logic language referred to in this paper. However, the above is generic and applies to all logic languages.
HOL, Chess and EGTs

- Note: SEC phases 1 and 2 conflate to a degree …
  - HOL is an Interactive Theorem Prover
- Phase 1
  - Model 'chess': FIDE Articles
    - Simplifications though: no Pawns, no castling rights
  - Model the Endgame Table
    - Using BDDs, first used by Gordon to provide solutions to Solitaire
    - Define 'the set of wins (losses) of depth \(d\)'
  - These are 'static aspects' of the model
- Phase1/2:
  - prove that the contents of the BDD follow from the definition of chess as modelled from the FIDE Articles in HOL
HOL definition of Chess and EGTs

- Take a subset of the FIDE Articles of Chess, singly (or not):
  - those defining the Game but not those defining Rules of Play
  - not those defining pawn moves or castling
- Translate the text of the FIDE Articles into HOL
  - A task eased by the power and 'naturalness' of HOL
  - 'Higher Order' ⊆ ∀Sets S = {m} and ∀Functions f:S₁→S₂ as well as ∀m
  - this formalisation process might even reveal infelicities in the text
- Define EGTs in terms of Binary Decision Diagrams (BDDs)
  - Gordon first combined HOL and BDD re Peg Solitaire (2002)
  - Work back from checkmates, but 'symbolically' using BDDs
  - JH's work is the first demonstration of HOL/BDDs on 2-person games
- Result: not just text, but 'givens' (axioms) of the 'world' created
  - A starting-point for proving subsequent theorems (providing results)
The definition of the Rook Move

- square $\equiv N \times N$
  position $\equiv$ side $\times$ (square $\rightarrow$ (side $\times$ piece) option)
- rook_attacks $p \ a \ b$
- $a \neq b \ ^\wedge (file \ a = file \ b \ \vee \ rank \ a = rank \ b)$
- $^\wedge \forall c. \ square\_between \ a \ c \ b \ \Rightarrow \ empty \ p \ c$
- The other rules of chess are similarly easy

Articles 3.3 and 3.5 translated in combination …
3.3: line-piece
3.5: non-hopping piece
HOL Results

- 4-man pawnless Chess EGTs which have been proved ...
  - to follow from the Laws of Chess

- Caveat at the logic level:
  - The 'environmental axioms' of this proof are that ...
  - Everything the proof depends on is working properly
  - Hardware, the logic-engine and its runtime realisation
  - [ ... and this is where the JH-GH discussion started ]

- Caveat at the physical level:
  - The price of this approach is more space and more time
  - we look to Moore's Law to ramp up memory and processor power
The Future
Emerging Opportunities and Challenges

- **Parallel Computing**
  - Has been 'in play' since 'Set Level Requests' were conceived
  - SQL is perhaps the most notable interface in this category
  - 'CPU' route is power-constrained: 'more' rather than 'faster'
  - Symmetric Multiprocessing is now 'on chip' on 'in-box' networks
  - This has created problems for both customers and suppliers
    - Customers have still not moved fully to a 'parallelised approach'
    - Customers are having to manage change in CPU/Memory balance
    - Suppliers are concerned that customers will not be able to do this
  - Supercomputing is an opportunity for the 'Declarative Approach'

- **Community Computing**
  - Using shared systems on the Web to energise various Diaspora
  - Enrich relationships within the Diaspora, mobilise activity, …
The Studies Community

- A (Win) Chess Study requires White to find the 'unique' winning line
- 'Unique' means 'essentially unique', not 'absolutely unique'
- But what alternative moves may be discarded?
- The FIDE PCCC has declared that 'cycling moves' may be ignored
  - these allow Black, defending, to force White to repeat a position
- The Study Community has long sought a tool to detect cycling moves
  - "the detection of blind alleys in general is notoriously difficult"
  - "detecting cycling moves can be ... essentially impossible to do by hand"

- GH has now defined an algorithm, SEA, to detect cycling moves
  - Identifies the area of 'no return' to which White should not move
  - An implementation is in prospect ... but what about Assurance?
Studies Community: Future Scenario

- There are some 70,000 studies in the corpus so far
- Members of the Studies Community apply SEA to a study
  - and report their findings on cyclic moves to the community
    - "given that positions $p_1$ to $p_n$ have been visited, move $m$ cycles"
    - these are non-trivial statements, easily mis-stated
  - The Mandler KNPKPP study of the Zugzwang paper would be 'target'
- Assurance issues, given the above framework:
  - Will the implementation of SEA be correct? Perhaps the least risk.
  - Will the users use the SEA tool correctly? Users are a big 'unknown'.
  - Will their results be correctly transmitted and understood?
  - Will their results be easier to verify than to find in the first place?
    - Does this 'desirable' increase the information that should be tabled?
- All these considerations have an effect on 'SEA' implementation
Summary

- The creation of EGTs is a complex and little understood task
- The EGTs now 'front' the domain of sub-7-man Chess
- They must therefore be correct but this is not certain in the future

Themes from this review:
- Collect data on errors as the foundation for Assurance Discussions
- No magic solutions but a framework of generic remedies
- At root, the precise meaning of the objects of the computation … and the context in which they are used … must be defined

The future: Community, and Parallel, Computing
- Provides opportunities for enriching the social fabric
- Provides opportunities for greater use of the declarative approach