Dragging debug into a new ERA

Using Data collection, Classification, and Analytics to Accelerate the Debug Cycle with AI/ML

Alan J. Pippin
Master Technologist
Fabric ASICs and Software Technologies Lab
Hewlett Packard Enterprise

Hewlett Packard Enterprise

Alan Pippin

HPE Master Technologist

- Degrees:
  - 2001 Bachelor of Science in Electrical Engineering from Brigham Young University
  - 2006 Master of Science in Electrical Engineering from Colorado State University
- Work Experience:
  - 23 years in FPGA and ASIC design and verification, Hardware design, Emulation, Systems IT support, Database engineering, and Embedded software development at HPE.
  - Led design and verification teams in delivering verified block IP for ASIC chips used in a variety of HPE products, spanning HPE Superdome class of supercomputers, HPE's Proliant servers, and HPE's optical business products.
- Current Responsibilities:
  - Leads the Slingshot Silicon Validation Team in delivering pre and post silicon validation solutions for the ASIC, SW, Driver, and FW Slingshot teams.

Aruba

Campus Edge Switching
Network switch ASICs

HPC and AI

Slingshot
Supercomputers, Data storage and analytics

Mission Critical x86 Systems
Scale Out and Scale Up systems

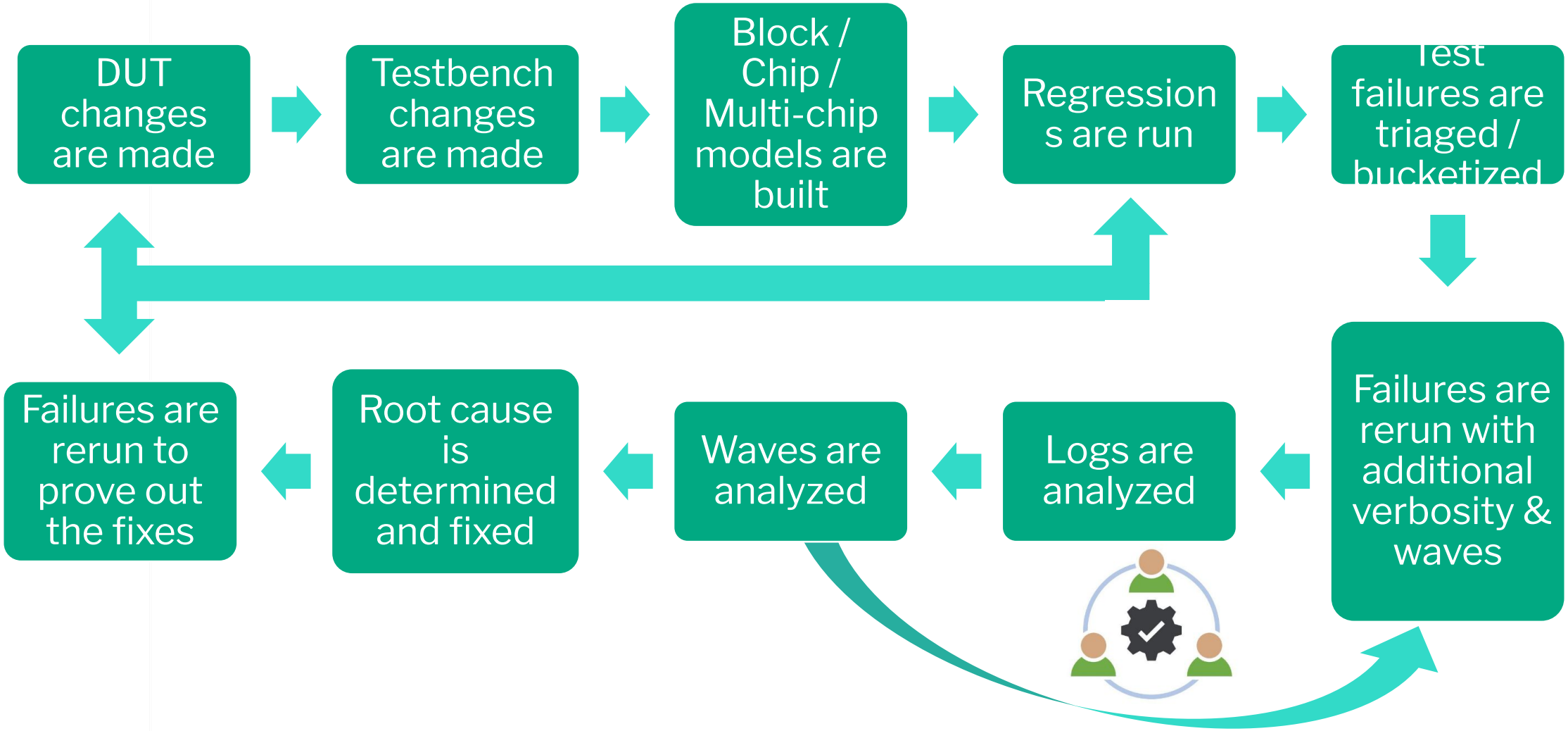
Hewlett Packard Labs

Research Silicon
Photonics and AI advanced research

Silicon Design Lab
Systems Management (iLO)

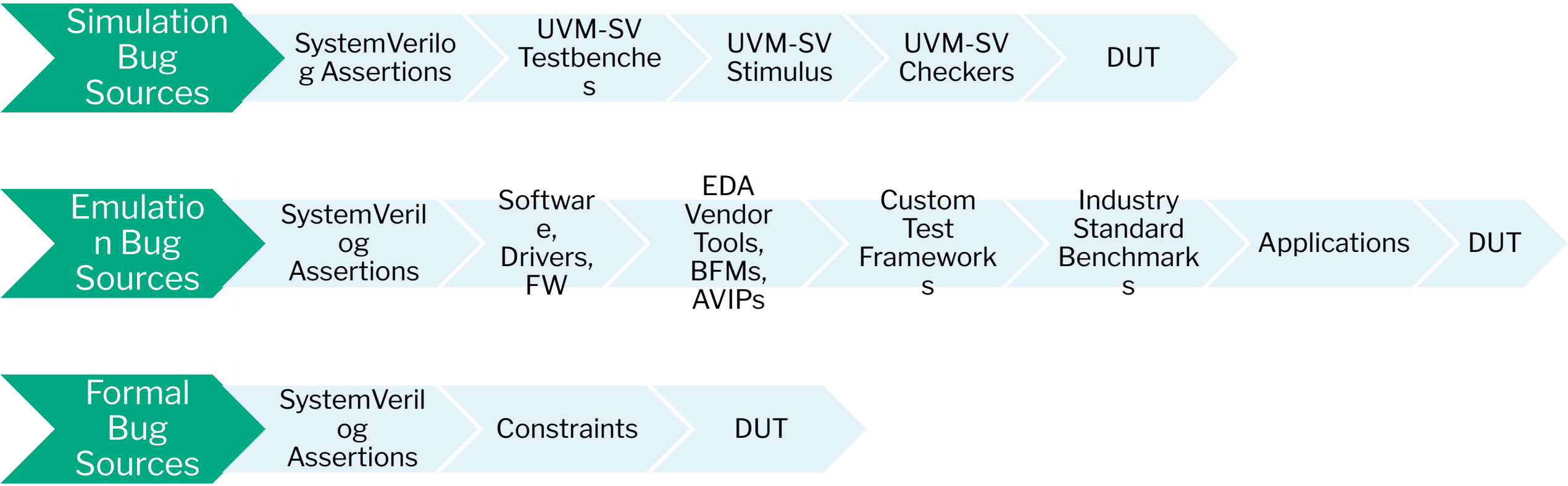
The Debug Cycle – A human driven task

What are we trying to get AI/ML to help us with in the debug cycle?



Sources of bugs are highly variant

How can AI/ML to help us isolate the root cause of different sources of bugs?



Finding bugs requires human insight

How can AI/ML replace human insight with intelligent insight?

The most critical kinds of bugs that need to be debugged

- Chip level
- Deep corner case bugs
- Queue overrun/underrun issues
- Performance bugs
- HW/SW interaction
- System level bugs: Architecture bugs and Chip bugs
- Software bugs: We are not just delivering chips, but a chips and software ecosystem

Current Verification Engines being used to catch bugs

- Simulation
- Formal
- Emulation
- Static tools (linting, CDC, etc)

Data Collection

We need to collect the right data and ask the right questions about it. Without data, there is no ML!

Compute Farm Data

- compute farm job data
- license usage data
- disk usage
- system load
- per site/business
- per user
- every few min

Performance Data

- test name
- traffic mixes
- bandwidth
- latency
- utilization
- swept variables and parameters
- per interface
- per topology

Test Results Data

- test metadata
- test definition
- tree revision
- pass/fail results
- error signatures
- durations
- memory usage
- cpu usage
- log files

Coverage Data

- per cover event
  - name
  - sim time
  - test id
  - value
- per test
  - name, user
  - cmdline, date
  - generator

Emulation Data

- Model build statistics
- Model health statistics
- Model run statistics

Issues that Data Collection helped us find

We were able to look for issues hiding in anomalies in the data we collected.

Compute Farm Data

- License Forecasting
- License Utilization
- Compute Farm Queuing Issues

Performance Data

- DUT performance bugs
- Architectural bugs
- Software bugs
- Testbench bugs
- Configuration bugs
- Stimulus bugs

Test Results Data

- Test throughput bugs
- DUT logic bugs
- Queue underrun/overrun bugs

Coverage Data

- Test stimulus bugs
- DUT assertion bugs
- Coverage reachability bugs

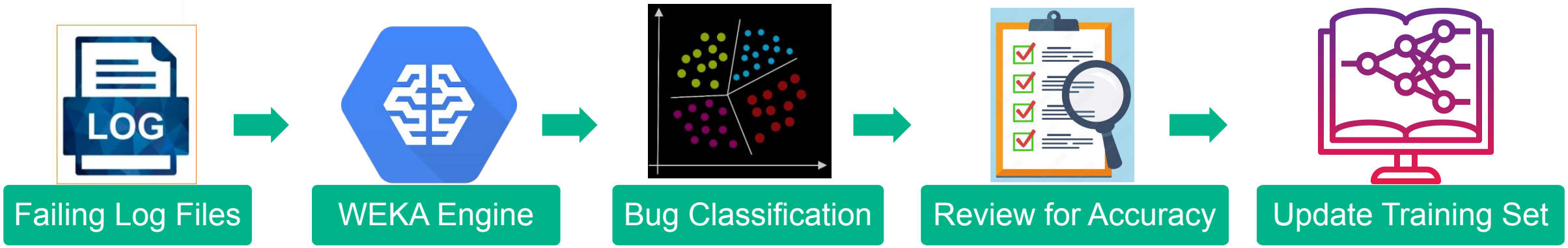
Emulation Data

- Vendor Build, tool, and BFM bugs
- HW/SW interaction bugs
- Software bugs

Pattern Recognition Engines

Take advantage of pattern recognition by using ML engines

- WEKA (Waikato Environment for Knowledge Analysis) is an open-source collection of machine learning algorithms for data mining tasks. It contains tools for data preparation, classification, regression, clustering, association rules mining, and visualization.
  - Wrote an in-house tool that trained WEKA with log data from our pre and post silicon test environments.
  - Failures were automatically bucketized daily to known failing signatures with more accuracy and less effort than a regexp based pattern recognition tool.

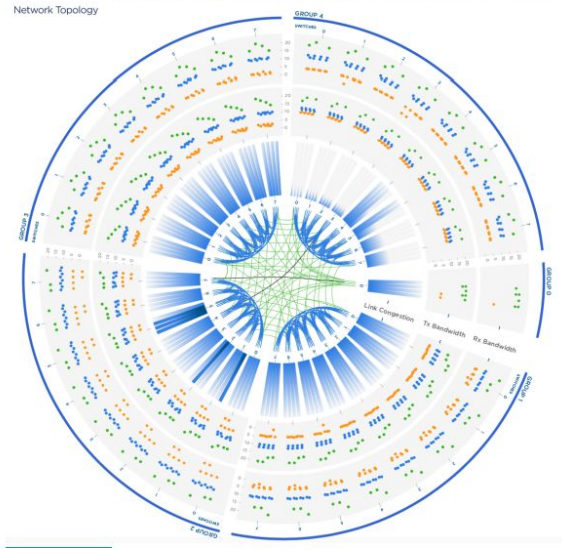


Analytics Frameworks

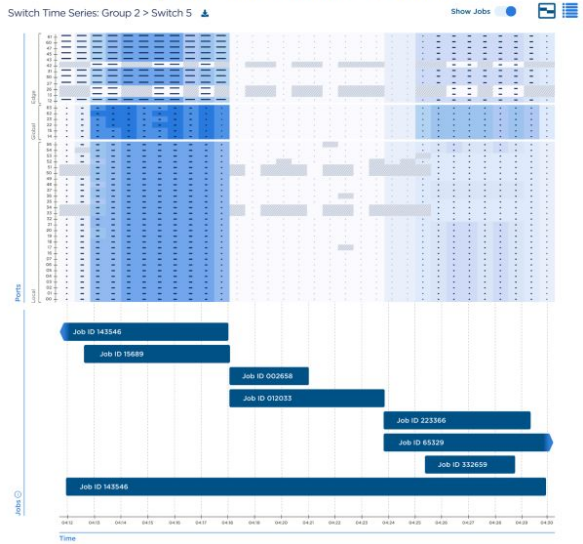
Take advantage of Analytics Frameworks to visualize patterns in the data

- Wide gap between Slingshot telemetry data and user's ability to assimilate and explore this data
- *trellis* is an analytical framework built on top of Slingshot system monitoring APIs.
- Currently investigating advanced machine learning models based on *trellis*.
- These models, once successful, will associate and predict job execution times with network performance.

Network Overview - Fabric Health and Link Failures

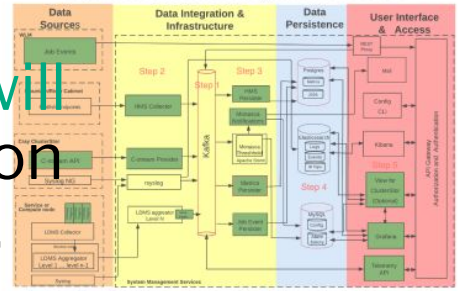


Overlay Jobs and Network Performance



Raw Telemetry

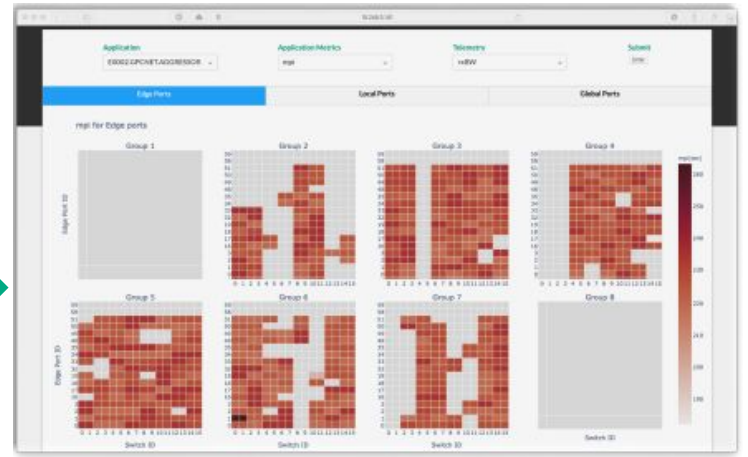
System Monitoring Framework



Exploring New Monitoring and Analysis Capabilities on Cray's Software Preview System. Jim Brandt et.al., 2019

~160,000 metrics @1Hz (1024-node Cray EX System)

trellis



AI/ML Needs & questions

We need AI/ML tools to make this easier! Here are some things to consider:

Apply ML to coverage closure

- Look at all test runs and all the constraint values
  - Identify constraints to close coverage in shortest amount of time possible
- Need an ML engine to train on constraint to coverage hit mapping
  - This is very controlled, very operational, very measurable, cause and effect.
- The Dream: Can AI engines consume our specs and generate tests to cover it?

ML engines need to be trained with what is right/expected

- With simulation, how do you tell it what is right? Signals, waves, logs, etc?
- What about randomization that changes conditions?
- How do you train these ML engines?

How do you know when things are running sub-optimally, performance bugs, etc?

- Can AI/ML be used to learn the normal behavior and flag inconsistent or unusual behavior?
- How to visualize data to spot problems and anomalies.
  - With larger systems, can't visualize all that data (it's too large and too complex)
  - How can we leverage AI/ML solutions to help us with that?