Optimizing Regression with VSO.ai

Andy Li, Jimmy Chuang, Realtek
Dung Bui, Alvin Chen, Synopsys

Outline

- Introduction of VSO.ai
- RTK evaluation

Introduction

The Problem

- Coverage is the key metric for verification sign-off.
- Running regression for coverage:
  - Is time consuming.
  - Has a lot of built-in redundancy.
  - Poorly targeted regression leads to a lack of diversity.

Introduction

VSO.ai Regression Optimizer

1. Reduces redundancy in your regressions comprising directed + random tests
   - –Automatically identifies an optimized set of test runs and test options for the regression
   - –Helps reach the same coverage goals faster
   - –Orchestrates tests to minimize a user-selected objective function

     (Ex: regression CPU time, number of test runs, simulation cycles, or cycles-per-second)

Introduction

VSO.ai Regression Optimizer

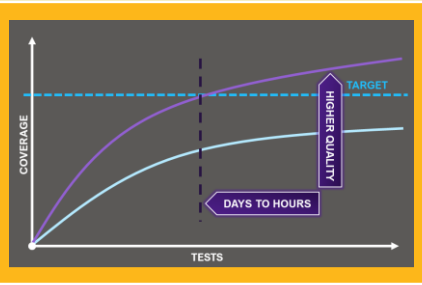
2. Increases coverage diversity in your regressions
   - Helps in increasing the probability of covering rare and complex coverage targets
   - Potentially improves QoR

3. Provides analytics to better visualize coverage results
   - Information on correlated tests, coverage targets and their probabilities
   - Details on unhit, extra-hit, and rare bins

VSO.ai Focus Areas

Coverage Closure

Test Efficiency
High Diversity

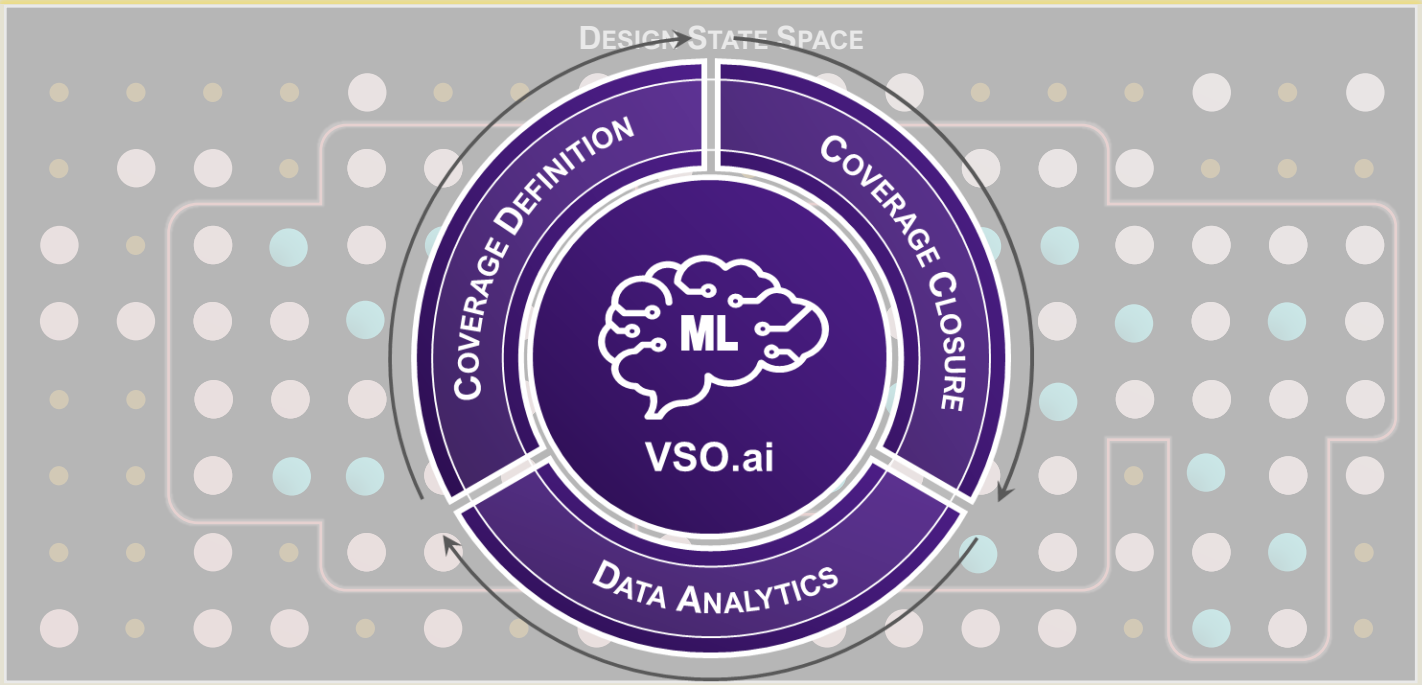


Targeted Stimulus
Higher QoR

Coverage Definition

Inferred Coverage
Higher Fidelity

Coverage Guidance
Identify Gaps



Analytics & Debug

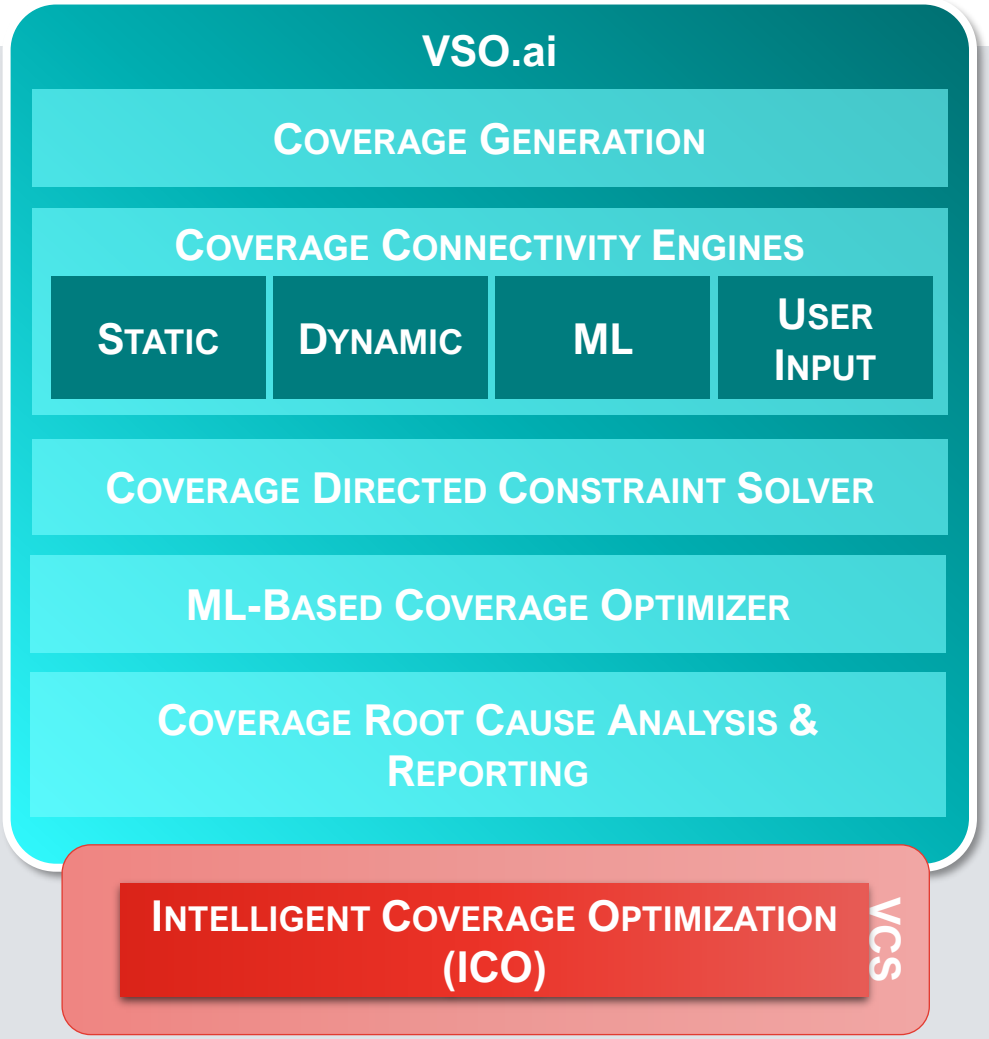
Automated Analysis
Better Insights

Coverage
Reachability

Configured Design

VSO.ai Architecture & Vision

Native Integration of VCS Core Engines & AI/ML Algorithms



Better Coverage: *Coverage Inference Engine Helps to Extract Coverage from Stimulus and RTL*

Productivity Boost: *Connectivity Engines & Directed Solver Targets Hard to Hit Coverage*

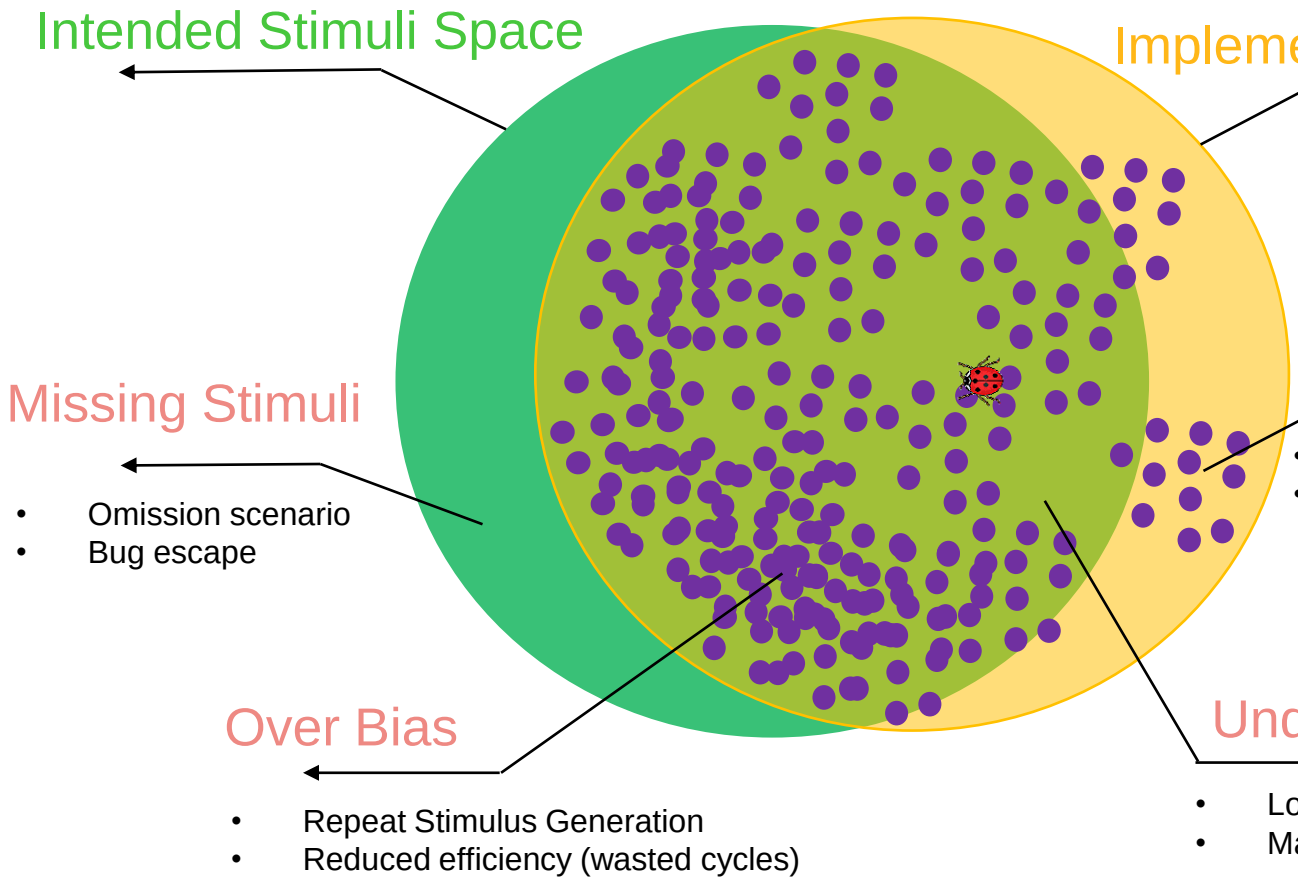
Improved HW Utilization: *Regression Optimizer Ensures Highest ROI Tests Run First*

Higher User Productivity: *Advanced Root Cause Analysis to Identify Unreachable Coverage*

Better Stimulus Quality: *High stimulus diversity improves bug hunting, TB visibility, coverage*

VSO.ai/ICO benefit: Testbench is the key for verification

VSO.ai/ICO Provide visibility/Root Cause Analysis/AI-based autonomous for TB quality/Improvement



TB Developer

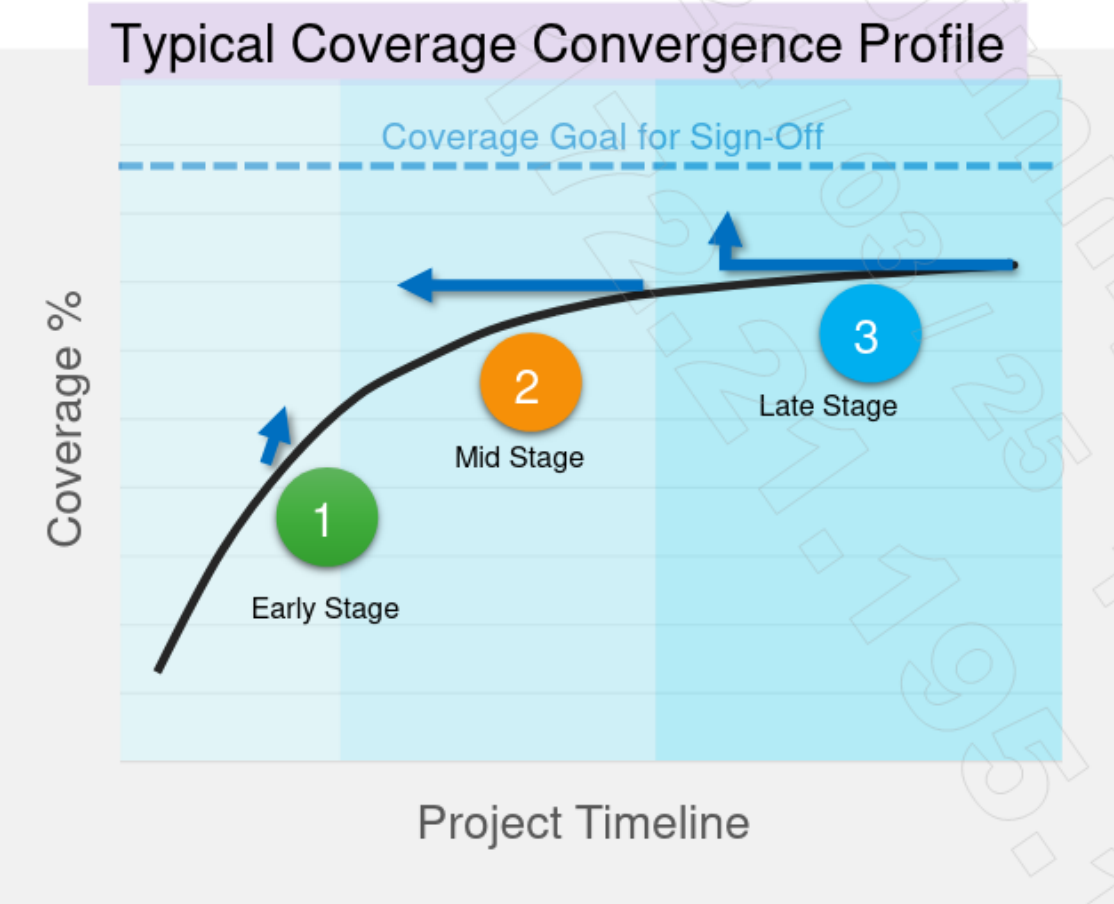
Do you know my pains?

- No Visibility to Distribution
- Complex constraints
- Under/Over constraints
- Distribution Bias
- Evolving coverage spec
- Holes in coverage specs
- Non-trivial TB failures
- Randomization issue
- Overflow issue
- Test non-terminating

…

“ICO can help from Day 1”

VSO.ai/ICO benefit: Shift-left Functional Coverage



1 TB Development

Ask – Stabilize and Mature TB faster

2 Optimize Regression, Achieve Coverage Goals

Ask – Accelerate Coverage and Bug rate

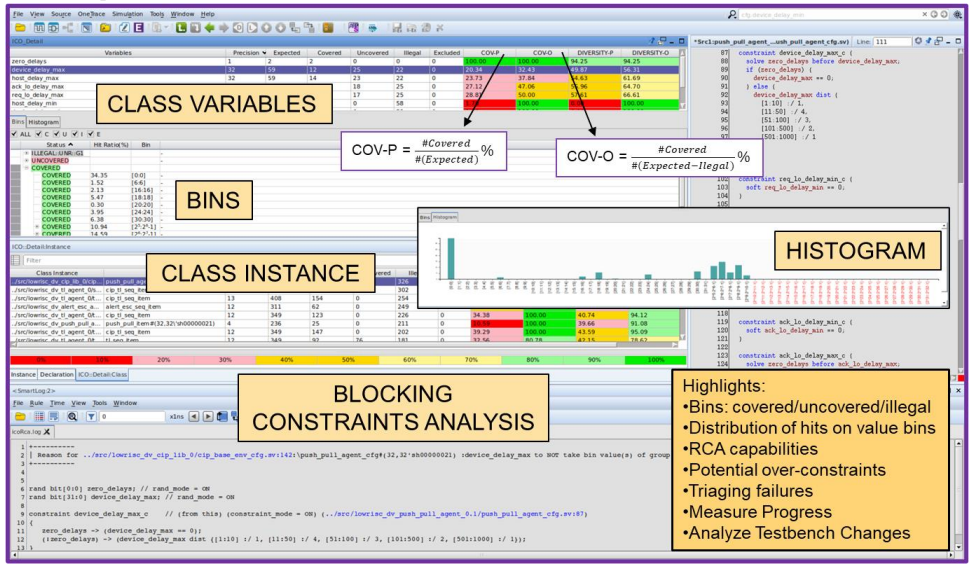
3 Coverage Convergence & Closure

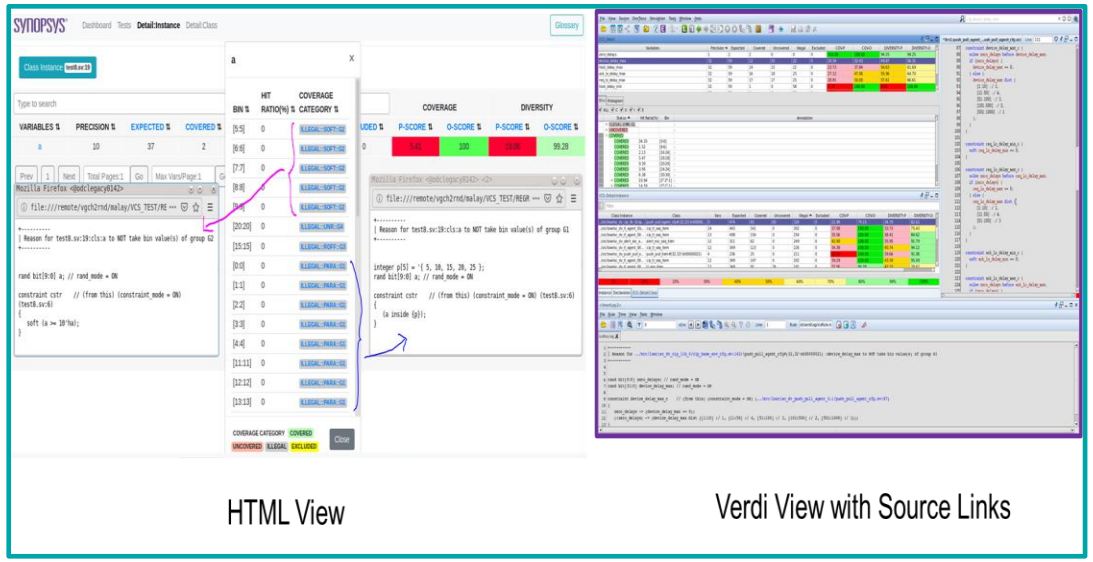
Ask – Faster closure, Save writing directed tests

"ICO can help from Day 1"

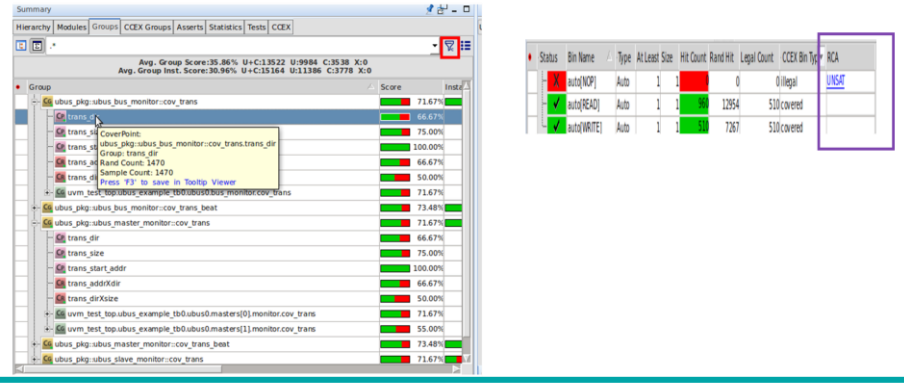
VSO.ai/ICO Verdi Integration/Simple integration with VCS

DV hard problems (ex: over constraint) now can have more fine-grain visibility to improve





- Option to enable Verdi coverage report with explorer : `verdi -cov -ccex -covdir simv.vdb`
- Explorer targeted cover objects are marked in Group page.
- Explorer specific information like rand hit, RCA etc are available in bin table.



1 CCEX Connections — Generate Static Inference Connectivity report:

```
% vcs -vso ccex -ccex_opts connect_report+nocycl -lca <other_switches>
```

2 CCEX Compile — Compile with Coverage Explorer (Static Inference)

```
% vcs -sverilog -vso ccex -lca <other_compile_switches>
```

3 CCEX Runs — Simulation with Coverage Explorer (Static Inference)

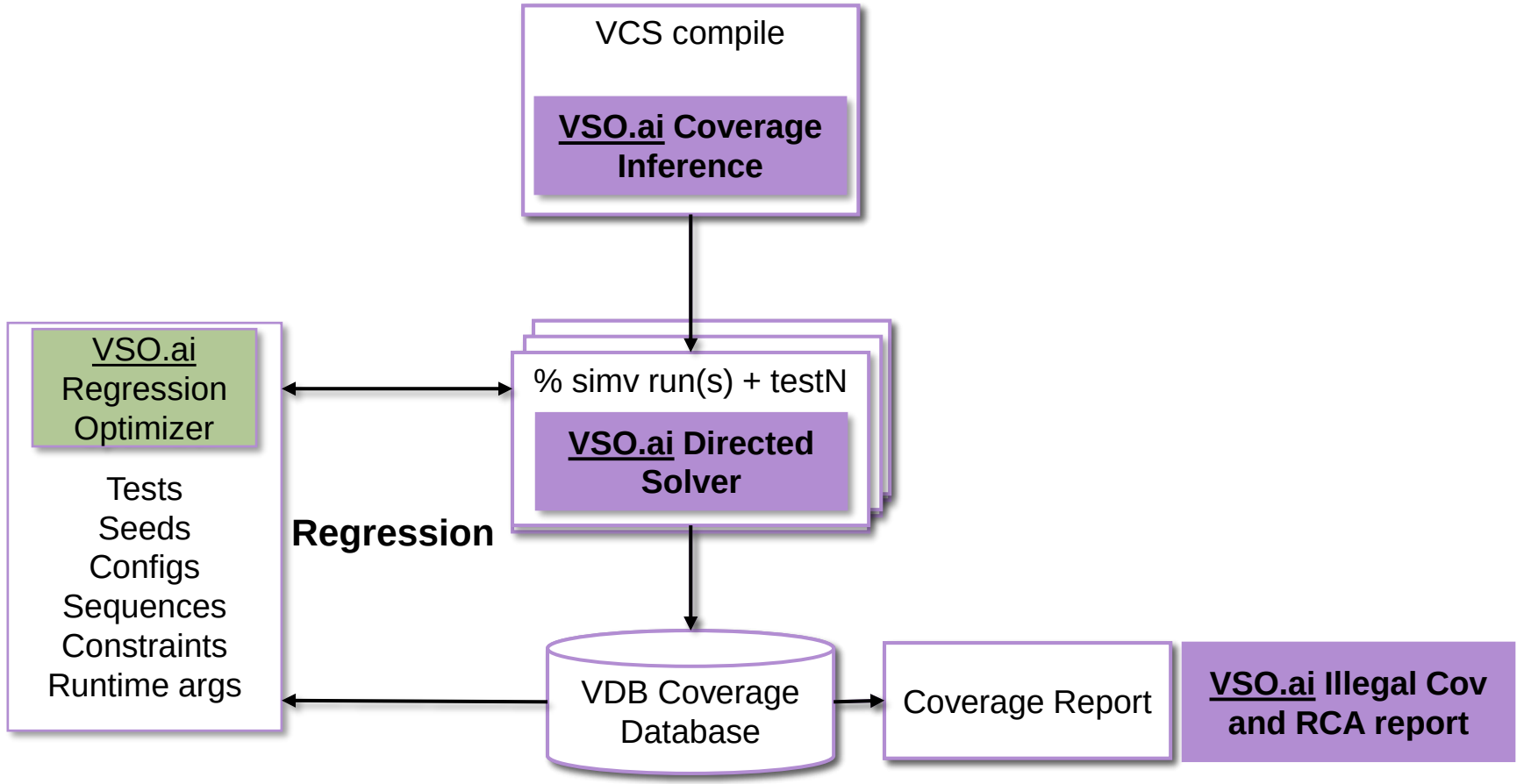
```
% simv -vso ccex <other_compile_switches> -ccex_opts
merged_vdb_dir=<dir>
```

4 CCEX Report — Report generation with URG or Verdi

```
% urg -ccex -dir <simv.vdb> [-dump full_exclusions ccex_illegal]
% verdi -cov -ccex -covdir <simv.vdb>
```

VSO.ai Regression Optimization with VCS

- The RO works at build, test, and test option level



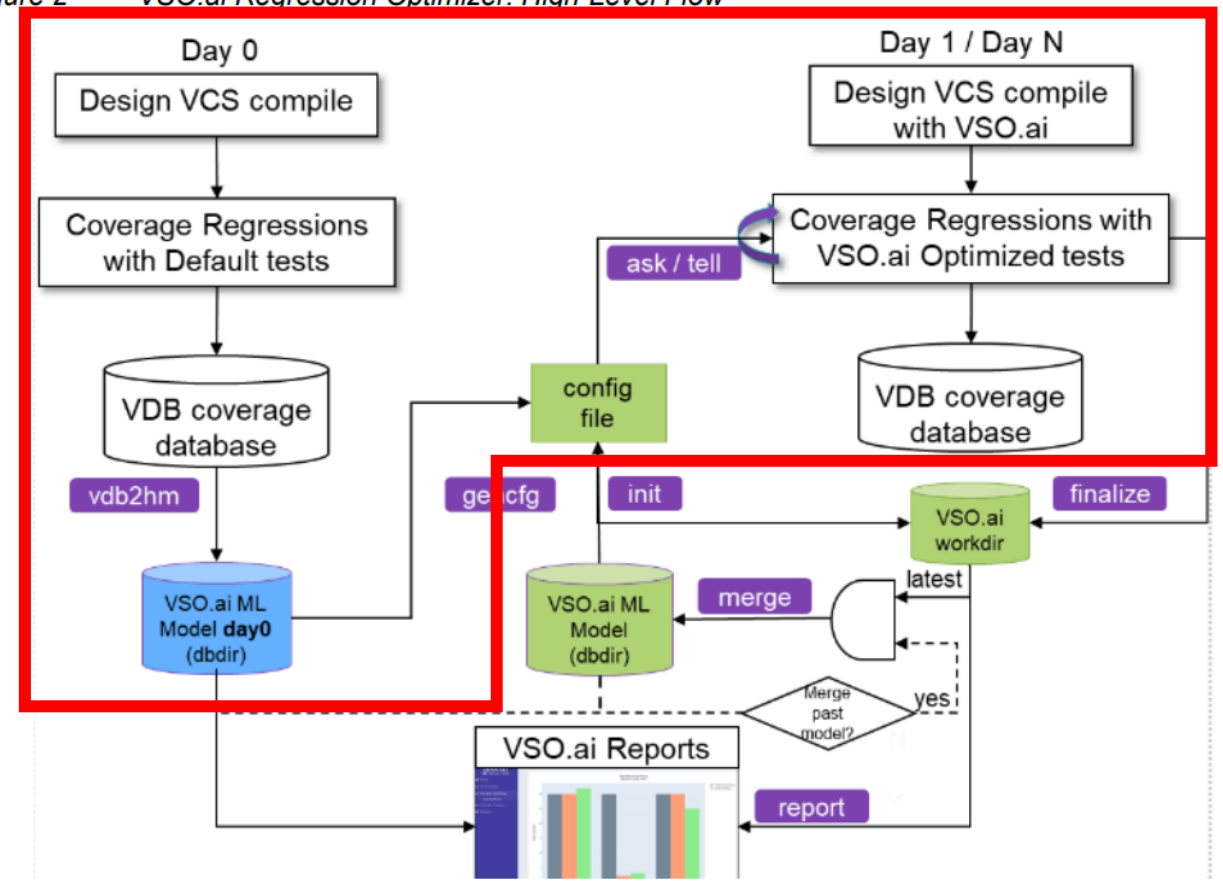
VSO.ai Regression Optimizer Flow

RTK Experiment

- RTK evaluates TTR feature
  - Day 0 vs Day 1
- Test case
  - 1100 test runs: 4 regression with ~279 test runs each.
- EDA Tool
  - VCS 2023.03-SP2-2

The following figure illustrates the high-level flow of VSO.ai Regression Optimizer:

Figure 2 VSO.ai Regression Optimizer: High-Level Flow



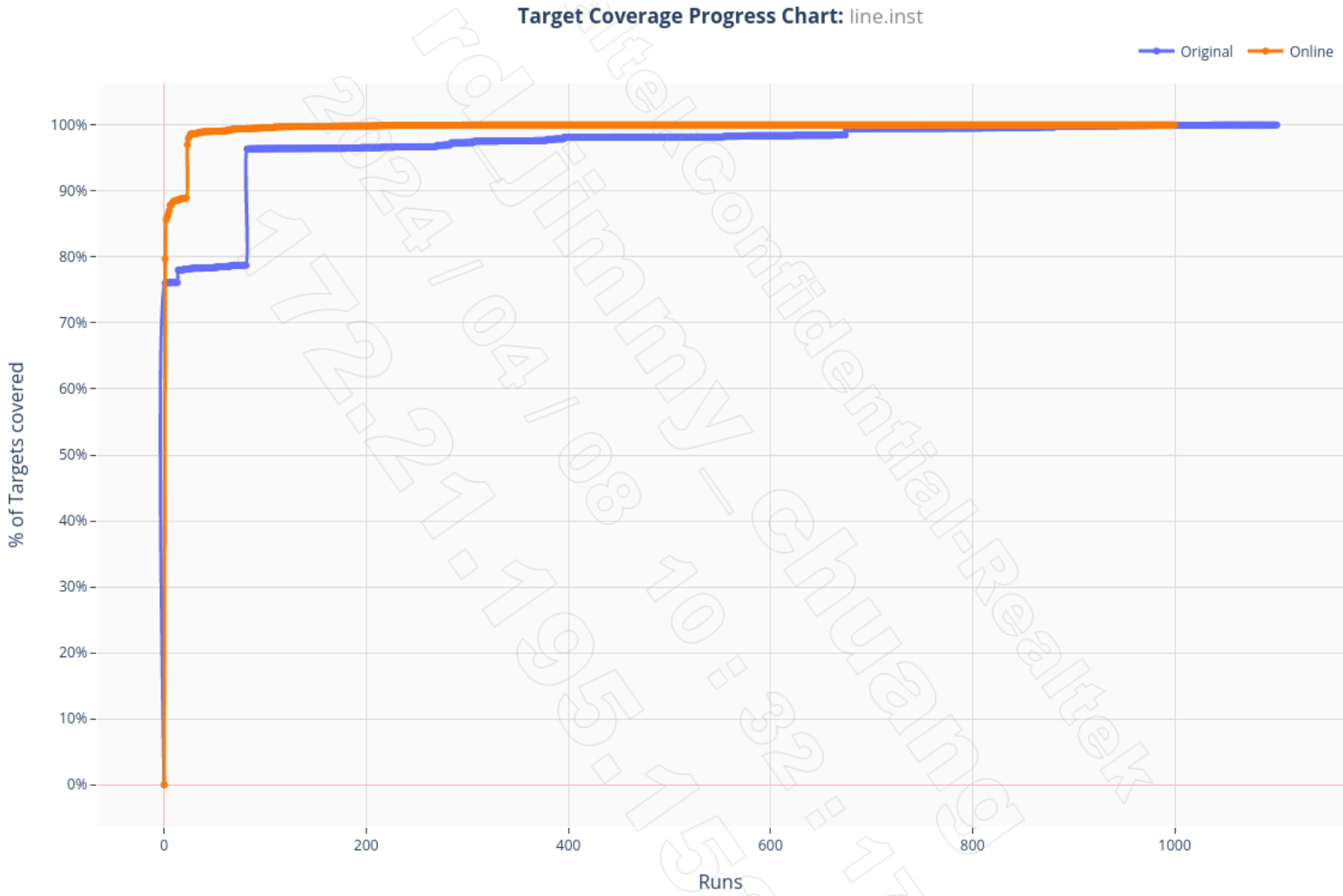
Experiment Result

- With VSO.ai, the test runs is reduced to 326 runs.
- Gain is 3.38X
  - 1101/326 = 3.38

Online

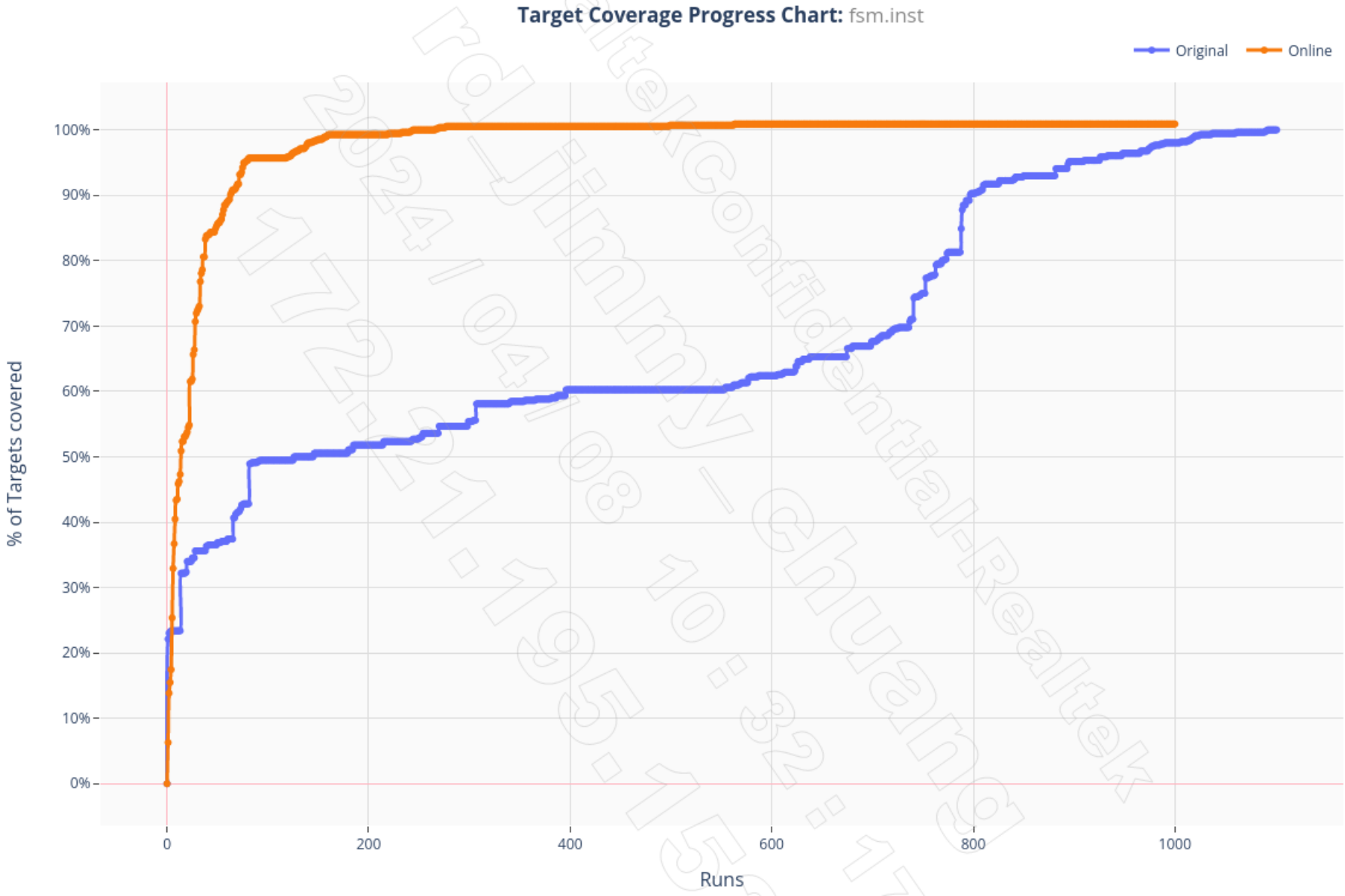
1101	326	134	3.38	6/6
Ref Runs	Runs	Tests	Gain	Metrics_met

Results – Line.inst



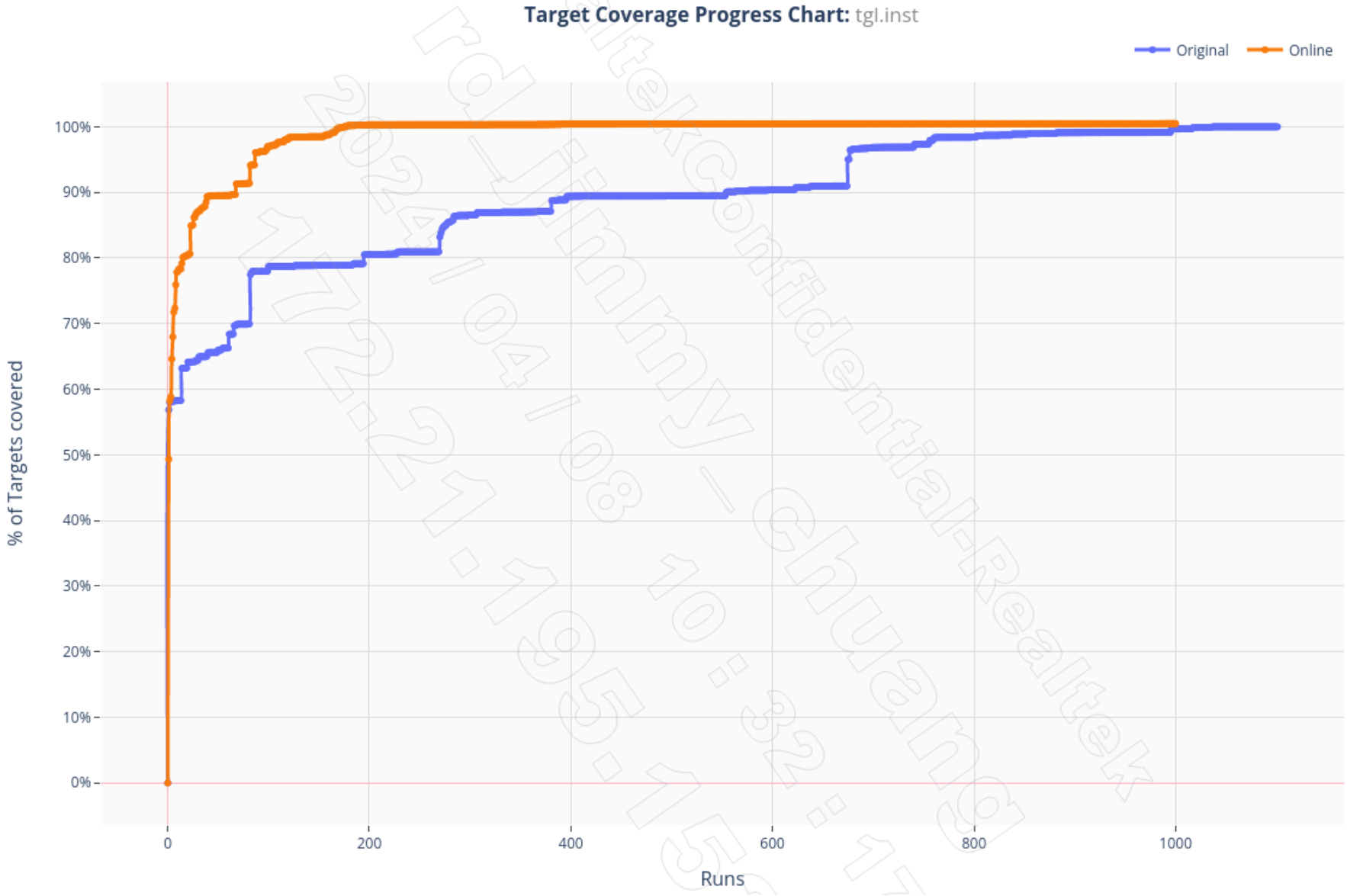
- Original: 1100 runs
- Online: 310 runs
- Gain: 3.55 X

Results – FSM.inst



- Original: 1100 runs
- Online: 260 runs
- Gain: 4.23 X

Results – tgl.inst



- Original: 1100 runs
- Online: 175 runs
- Gain: 6.29 X

Conclusions

- **VSO.ai - AI based Regression optimizer ensure high ROT tests**
- Overall gain is 3.38X
- VSO.ai helps to reduce turnaround time(TAT) for regression
- The feature could help to shorten project schedule in mid-late phase


THANK YOU
Our Technology, Your Innovation™