What Went Wrong?

Automatic Triage of Precision Loss During Static Analysis of JavaScript

Safe Harbor Statement

The following is intended to provide some insight into a line of research in Oracle Labs. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. Oracle reserves the right to alter its development plans and practices at any time, and the development, release, and timing of any features or functionality described in connection with any Oracle product or service remains at the sole discretion of Oracle. Any views expressed in this presentation are my own and do not necessarily reflect the views of Oracle.

Oracle Labs Australia

http://labs.oracle.com/locations/australia

- Focus on Program Analysis
- Current projects
  - Java vulnerability detection
  - Frappé: impact analysis
  - Soufflé: Datalog engine for program analysis
  - **Web-based vulnerability detection**
- Past project
  - Oracle Parfait: bug-checker for C/C++

Precision Loss During Static Analysis of JavaScript

- ✓ Introduction
- Idea and motivation
- Our current implementation

Precision in JavaScript static analysis

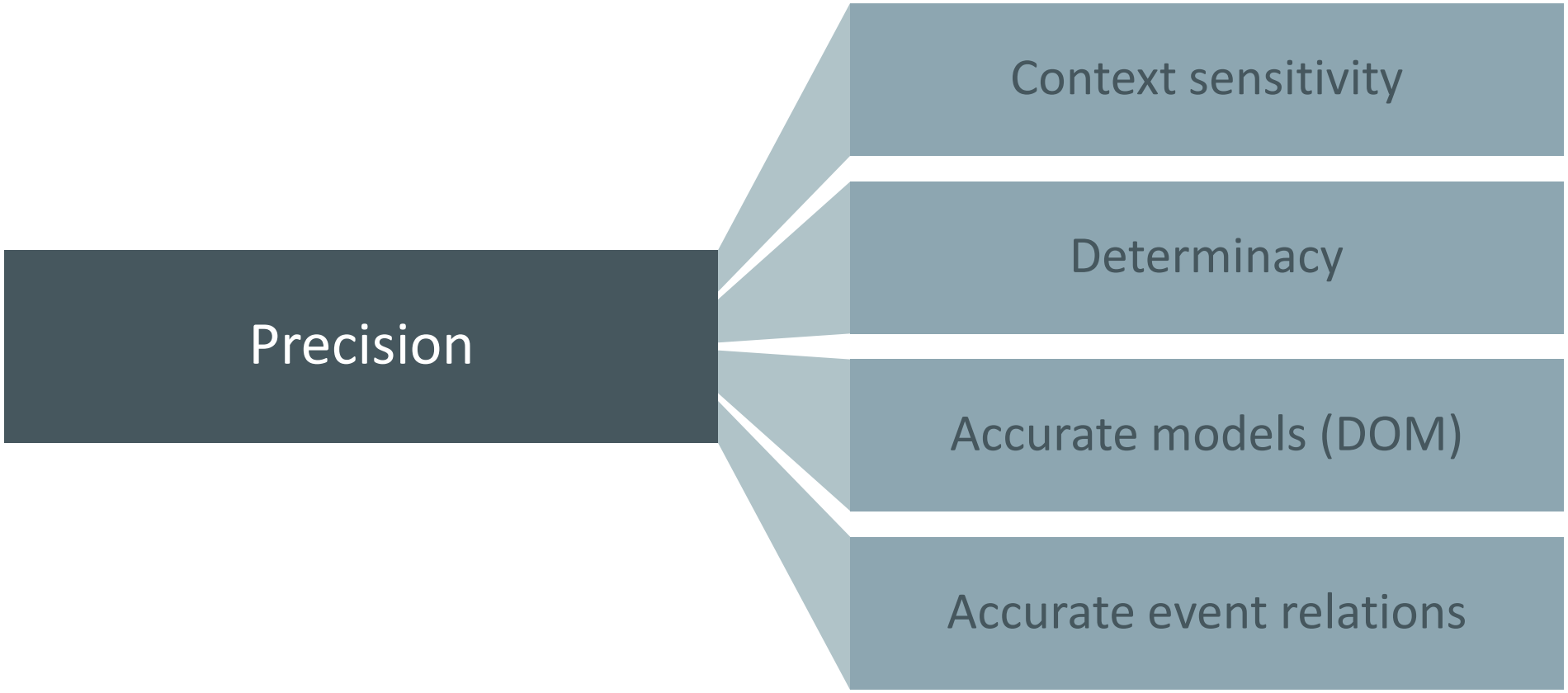
Why keep track?

JavaScript v. Static Analysis

A non-exhaustive list of grievances:

- Few static guarantees (un-typed & dynamic)
- Ubiquitous use of reflection
- Intricate semantics and side-effects

JavaScript v. Static Analysis



When analysis fails

At least in our experience

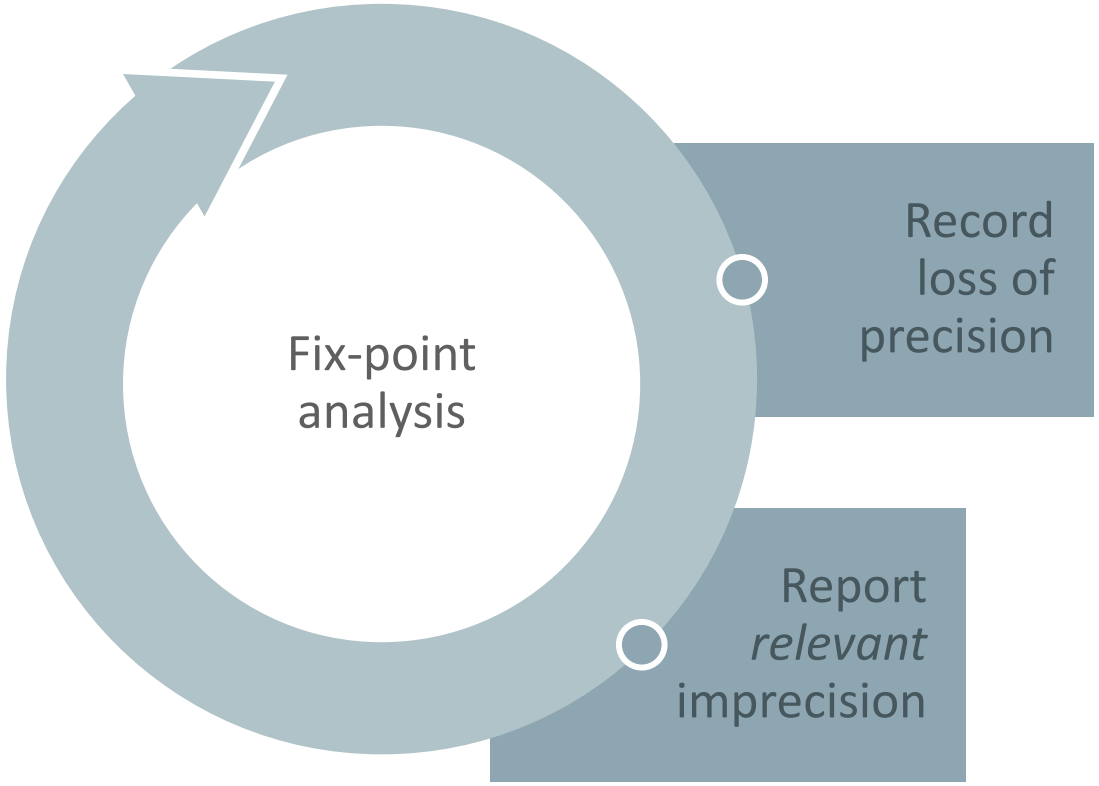
1. ¯_(ツ)_/¯
2. Try again with different analysis settings
3. Tweak the program under analysis
4. Last resort: inspect abstract state, add logging

When analysis fails

A better way

- Fail fast (early during analysis)
- Explain why

Precision tracking



User notified in two ways:

1. Warnings about potentially detrimental effects of precision loss
2. When imprecision causes analysis to become infeasible: stop and emit all precision loss information relevant to current state

Precision tracking

- Better means of debugging analysis problems related to precision
  - Ensuring precision is hard
  - It is easier to validate soundness of our analysis
- Pinpoint static analysis “hotspots”
  - *That DOM function that is only a stub*
  - *That essential part of the program that needs more context sensitivity*

"There is always a larger volume of work that is worth doing than can be done currently"

– Mervin Kelly, Bell Labs

Precision tracking in SAFE

A work in progress

Precision tracking in SAFE

1. Imprecision Sources
   - Abstract domain operations
   - Partially modelled DOM and built-in functions
2. Attach imprecision hints to abstract values
   - $\widehat{Value} = \widehat{PValue} \times p(\widehat{Loc})$
3. Sinks
   - Call, apply
   - Property access

In addition: keep track of the size of the work list

Results

Example: jQuery analyzed with SAFE's highest sensitivity setting

Precision problems found:

- Math.random
- jQuery.extend
- SAFE's String and Number domains
- jQuery's list of event handlers
- SAFE's Array.prototype.sort model

Summary

- Precision tracking now
  - Means of debugging analysis problems
  - Guiding efforts to model built-in and library code
- Future applications?
  - Automatic analysis refinement (e.g. context-sensitivity, abstract domains)

Integrated Cloud

Applications & Platform Services


ORACLE®