Imperial College London

Google Summit

Automatic JavaScript Program Verification Using Bi-Abduction

Gabriela Sampaio

Petar Maksimović

José Fragoso Santos

Thomas Wood

Philippa Gardner

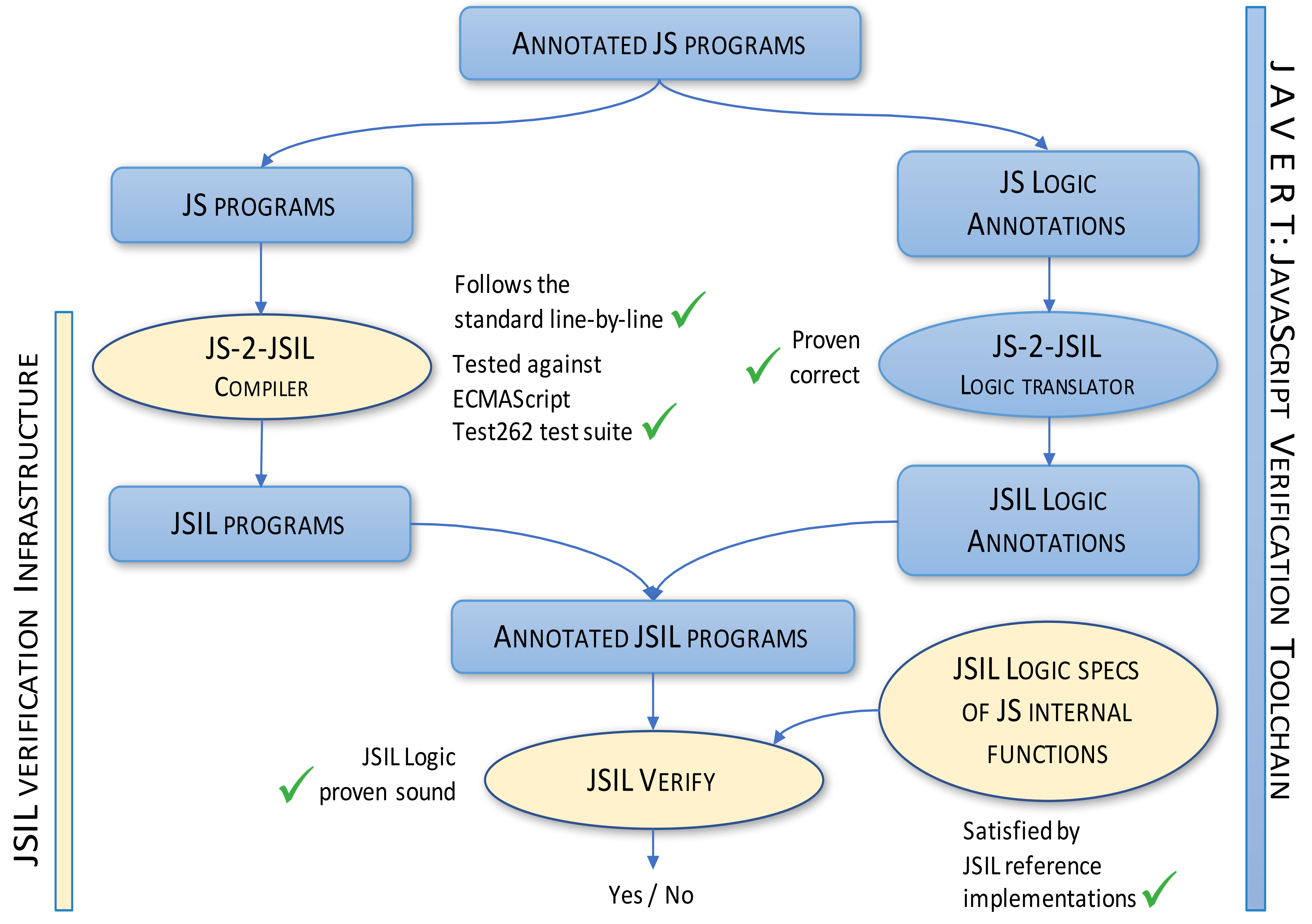
MOTIVATION

Problem: due to the highly dynamic nature of the JavaScript language and its complex semantics, applications might have bugs which are hard to find without tool support.

Our vision: correctness of JavaScript programs verified by inferring pre- and post-conditions automatically, in a scalable way.

JAVERT[1]

- Semi-automatic tool for reasoning about JavaScript programs using separation logic
- Programs annotated with pre- and post-conditions, loop invariants and directions for folding and unfolding predicates
- JaVerT specifications are written in JS Logic, our logic assertion language for JavaScript



ANNOTATED JS PROGRAMS
JS PROGRAMS
JS LOGIC ANNOTATIONS
JS-2-JSIL COMPILER
Follows the standard line-by-line ✓
Tested against ECMAScript Test262 test suite ✓
✓ Proven correct
JS-2-JSIL LOGIC TRANSLATOR
JSIL PROGRAMS
JSIL LOGIC ANNOTATIONS
ANNOTATED JSIL PROGRAMS
JSIL LOGIC SPECS OF JS INTERNAL FUNCTIONS
✓ JSIL Logic proven sound
JSIL VERIFY
Yes / No
Satisfied by JSIL reference implementations ✓
JSIL VERIFICATION INFRASTRUCTURE
JAVERT: JAVASCRIPT VERIFICATION TOOLCHAIN

BI-ABDUCTION[2,3]

- Form of logical inference for separation logic
- Automates the key ideas about local reasoning
- Allows one to infer pre- and post-conditions
- Infer tool uses it for static languages

EXAMPLE

```
function swap(x, y){
```

State: { emp }
Missing: { emp }

```
    var aux = x;
```

State: { aux = #x }
Missing: { x = #x }

```
    x = y;
```

State: { aux = #x * x = #y }
Missing: { x = #x * y = #y }

```
    y = aux;
```

State: { x = #y * y = #x}
Missing: { x = #x * y = #y}

```
}
```

Bi-abduction

```
Pre:  {x = #x * y = #y}

Post: {x = #y * y = #x}
```

References:

1. J. Fragoso Santos, P. Gardner, P. Maksimović, D. Naudžiūnienė. JaVerT: JavaScript Verification using Separation Logic. *POPL*, Accepted for publication, 2018.
2. P. O'Hearn, J. Reynolds, H. Yang. Local Reasoning About Programs that Alter Data Structures, *Computer Science Logic*, 2001.
3. C. Calcagno, D. Distefano, P. W. O'Hearn, H. Yang. Compositional Shape Analysis by Means of Bi-Abduction. *Journal of the ACM*, 2011.