KJS: A Complete Formal Semantics of JavaScript

Link: https://pdfs.semanticscholar.org/031e/1eb62214945aae260802275c0338dece87d3.pdf

This paper presents KJS, the most complete and throughly tested formal semantics of JavaScript to date. Being executable, KJS has been tested against the ECMAScript 5.1 conformance test suite, and passes all 2,782 core language tests. Among the existing implementations of JavaScript, only Chrome V8's passes all the tests, and no other semantics passes more than 90%. In addition to a reference implementation for JavaScript, KJS also yields a simple coverage metric for a test suite: the set of semantic rules it exercises. Our semantics revealed that the ECMAScript 5.1 conformance test suite fails to cover several semantic rules. Guided by the semantics, the author wrote tests to exercise those rules. KJS is symbolically executable, thus it can be used for formal analysis and verification of JavaScript programs. they verified non-trivial programs and found a known security vulnerability.

Related Work:

Previoulsly it was defined a small-step semantics of EC- MAScript 3 and proved some basic properties. Their semantics is based on the older ECMAScript 3, and does not cover the modern JavaScript features such as the strict mode. Also, it is not executable, and cannot be validated against conformance test suites. They defined a core language, , and a translation from JavaScript to together with a (runtime) environment containing internal semantic functions written in itself. They also implemented an interpreter for , which, combined with the translator and the runtime environment, allows to execute and test their semantics. Although the reduced semantics is helpful to understand the essentials of JavaScript, there is a gap between it and the actual language specification. Since their semantics does not directly follow the structure of the language specification, it is difficult to manually/visually inspect it and, indeed, it contains a number of bugs. they found that the JavaScript language specification, unlike for other languages, is quite well written, They decided to follow it faithfully

Future work:

Although KJS passes all the tests in the ECMAScript 5.1 conformance test suite for the core language, which is the reason why they call it a 'complete semantics', there is no guarantee that our semantics is necessarily correct. In the absence of a reference semantics, they believe that the best they can do to validate our semantics at this stage is to test it heavily against as many tests as possible, which they did, and to reason with it and prove certain expected properties of it, which they have not done yet but they plan to do as soon as a Coq backend becomes available for K .One of the most promising directions of future work is to use KJS to formally verify JavaScript programs against security properties of popular JavaScript applications