OPTIMIZING DYNAMIC LANGUAGES

IIT Bombay

Introduction

- Dynamic languages like Python, JavaScript and R are everywhere.
- Despite their popularity, they suffer from performance bottlenecks and slow warmups.
- Static analysis tools help optimize static languages like C, C++ and Java, but these concepts seldom apply to dynamic languages.
- We are developing tools and frameworks to perform static analysis of JavaScript.

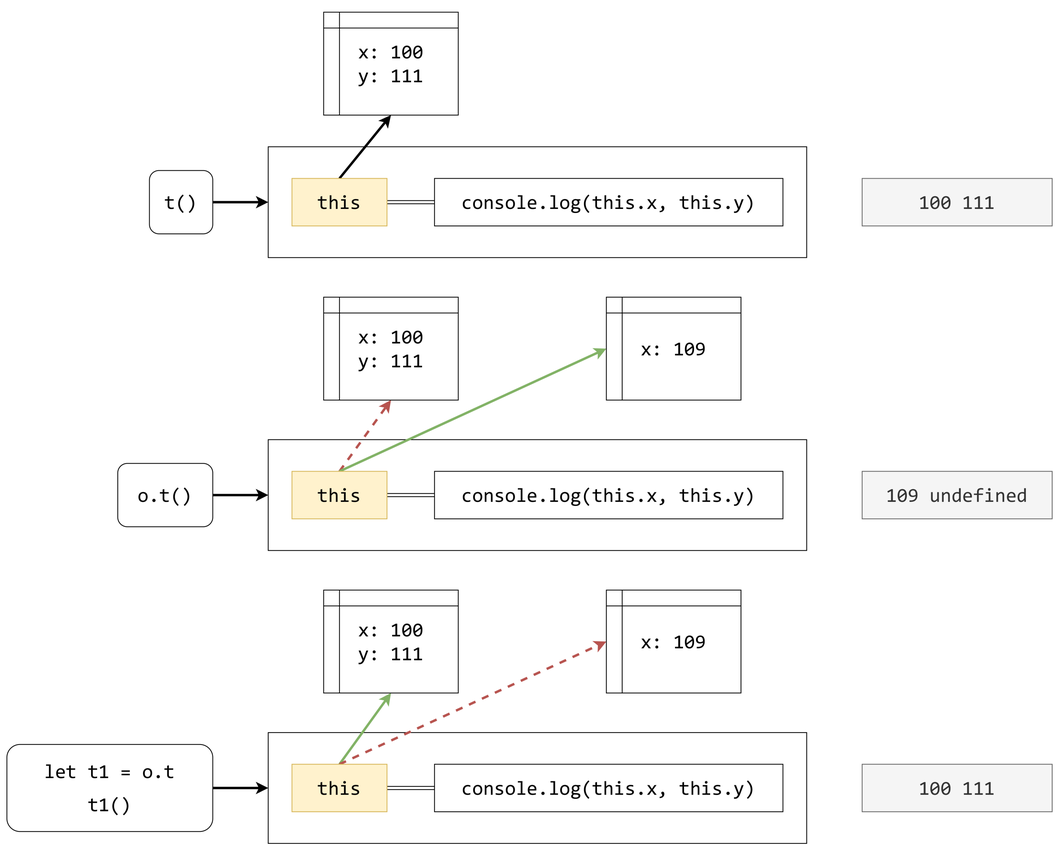
JavaScript is widely used to design User Interfaces, Servers and Mobile Applications.

Lack of standard representations and tools to study and optimize programs.

We are developing a framework called IRIDIUM that will be used to study and optimize JavaScript.

Challenges

```
1  const ff = function () {
2    this.x = 100
3    this.y = 111
4    // Returned Function foo
5    return function foo () {
6      console.log(this.x, this.y)
7    }
8  }
9  let t = ff()
10 t() // Callsite(1) ===> 100 111
11
12 const o = { x: 109, t: t }
13 o.t() // Callsite(2) ===> 109 undefined
14
15 let t1 = o.t
16 t1() // Callsite(3) ===> 100 111
```



- Implicit Contexts
- Chaining Operators
- Complex Lexical Scoping
- Temporal Dead Zones
- Dynamic pointers
- Complex Class Initialization

Our Work and Future

SOURCE CODE → 3JS → IRIDIUM → Iridium Passes → CODE GENERATION

Iridium Passes:
- NORMALIZING TDZ
- EXPLICIT CONTEXTS
- CLOSURE AWARE PTA
- SCOPE RESOLUTION

3JS is a fully compiant language subset of ECMAScript 2025

It is a Three-Address Like representation for JS

Optimization requires a precise pointer information, we are developing a closure aware pointer analysis for IRIDIUM to model the dynamic language featues of JavaScript.

Our next step is to integrate IRIDIUM to load optimized code into existing Virtual Machine systems like SpiderMonkey, V8 and JSC.

Meetesh Kalpesh Mehta (PhD IITB), Prof Manas Thakur (Advisor)

PLATO

Programming Languages Analysis Translation Optimization