SCHEME: An Interpreter for Extended Lambda Calculus

Gerald J. Sussman and Guy L. Steele Jr.

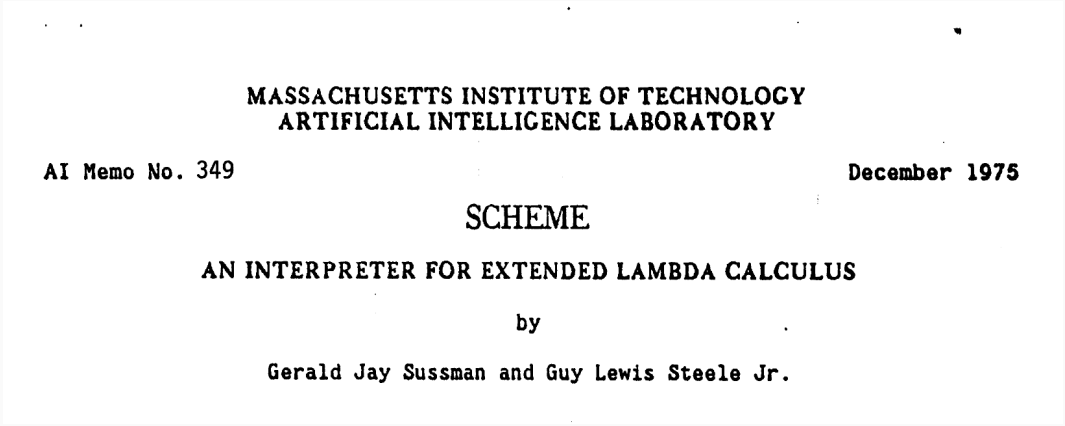
Veit Heller
Papers We Love Berlin

February 16, 2020

Agenda

- Introduction and historical context
- Scheme primer
- A “hairy control structure”
- Let’s see some code!
- Implementation notes

In 1975, a 21-year-old grad student named Guy Steele and his thesis advisor Gerald Sussman had something to show to the world: a little programming language called Scheme.



MASSACHUSETTS INSTITUTE OF TECHNOLOGY
ARTIFICIAL INTELLIGENCE LABORATORY

AI Memo No. 349 December 1975

SCHEME

AN INTERPRETER FOR EXTENDED LAMBDA CALCULUS

by

Gerald Jay Sussman and Guy Lewis Steele Jr.

Figure: A wild paper appears.

The Paper

The paper has all the goods a hacker could wish for: a reference, cool code examples, and an implementation of Lisp in Lisp.

The Name

The language was originally intended to be called SCHEMER, in reference to its ancestors PLANNER and CONNIVER.

In Scheme, we define functions using *define*—you might know it as *defn* or *defun* in other Lisps:

```
(define add
  (lambda (x y)
    (+ x y)))
```

Listing 1: Defining addition

NB: I eschewed the all-caps notation, and I hope your eyes will thank me for it.

We can *quote* things using either the function or the abbreviation '.

```
; this will always return the symbol x
(define gimme-x (lambda () 'x))
```

Listing 2: Using symbols as values

There is also the somewhat idiosyncratic *labels*, which allows you to define local functions that can be called inside a context, and can call themselves and other local functions in that context. You might know it as *letrec** from later Schemes, and as simply *let* in Common Lisp.

```
; lets define cells!
(define cons-cell (lambda (contents)
    (labels ((the-cell
                (lambda (msg)
                  (if (eq msg 'contents) contents
                    (if (eq msg 'cell?) 'yes
                      (if (eq (car msg) '<-)
                        (block (aset 'contents (cadr msg))
                               the-cell)
                        (error '|Unrecognized Message - Cell|
                               msg
                               'wrng-type-arg)))))))
      the-cell)))
```

Listing 3: Let's define something!

And now?

There is more, though!
Let's get to the good stuff.

TODO: Talk about new control structures.

TODO: Talk about code examples: samefringe, pattern matchin, multiprocessing:

TODO: Talk about the implementation.