

Note

These slides were mangled by a Keynote “upgrade” after the talk was given. There may be display issues.

Profile-Guided Meta-Programming

William J. Bowman, Swaha Miller,
Vincent St-Amour, R. Kent Dybvig



What is ...

Meta-Programming

- Extended syntax
- Generate source code
- Embedded DSLs

What is ...

Profile-Guided (Optimization)

- Collect profile at runtime
- Use profile as oracle
- Write optimizations

What is ...

Profile-Guided

- Collect profile at runtime *for*
- Use profile as oracle *to*
- Write optimizations *from*

Meta-Programming

- Extended syntax
- Generate source code
- Embedded DSLs

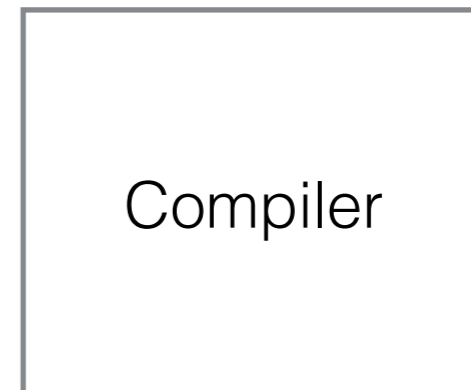
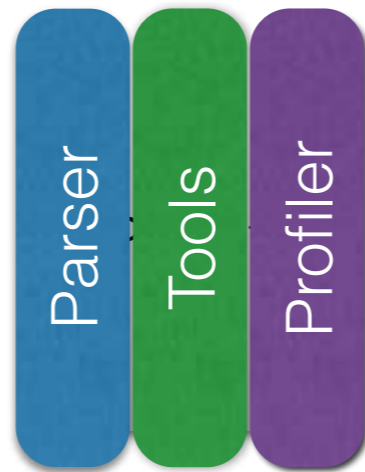
Profile-Guided Meta-Program

An example

```
define-syntax if-r(test t-branch f-branch):  
  if profile(t-branch) < profile(f-branch):  
    generate(#'(if not(test) f-branch t-branch))  
  else:  
    generate(#'(if test t-branch f-branch))
```

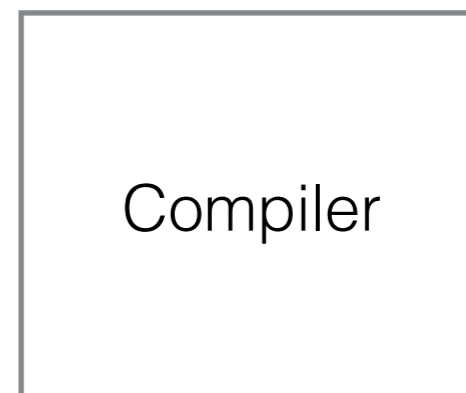
Problems

For Optimization **Writers**



Problems

For Optimization **Writers**



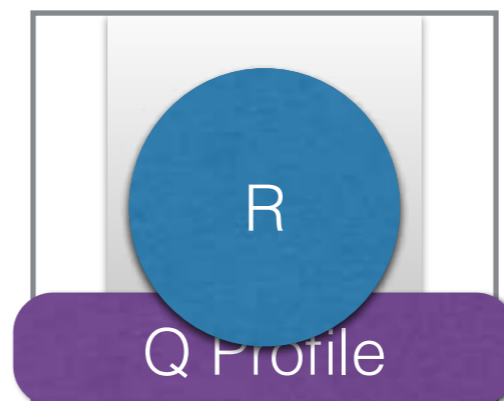
Problems

For Optimization **Users**



Problems

For Optimization **Users**

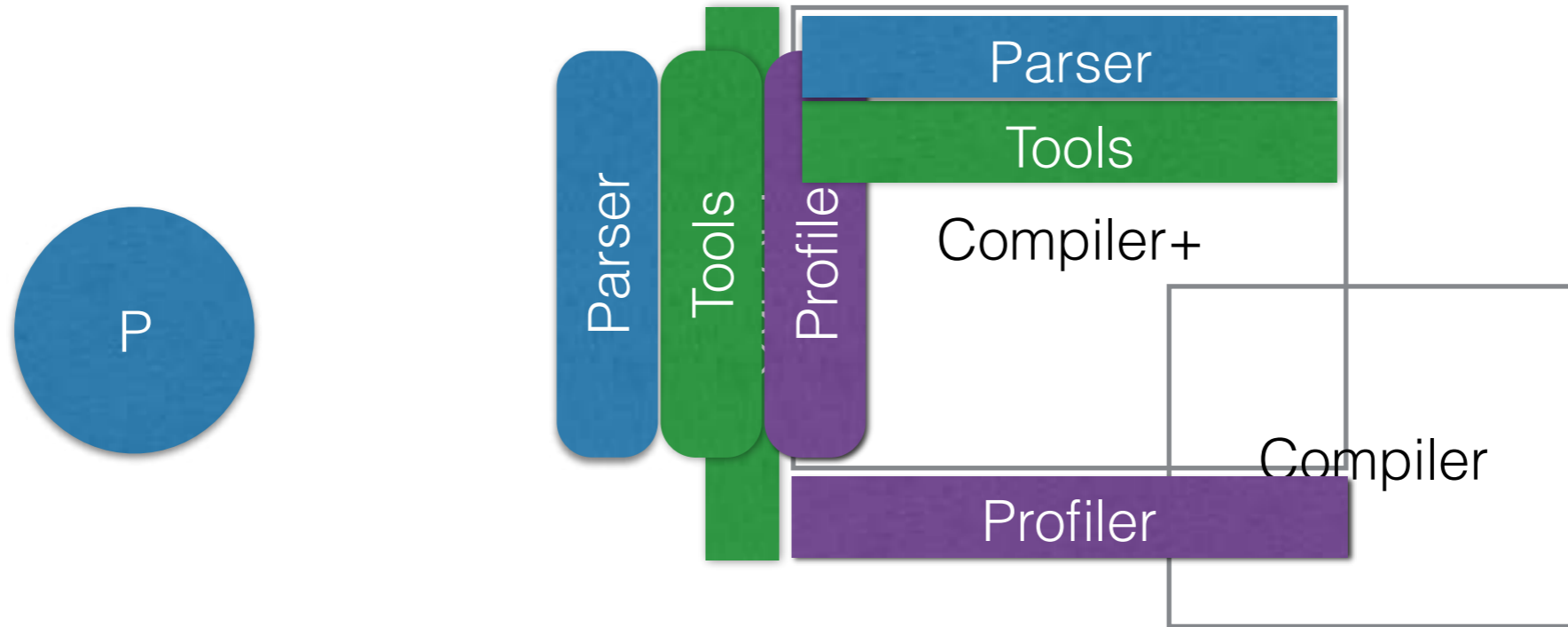


Our Design...

- Saves PGO writers effort
- Saves PGO users effort
- Simple to implement
- Expresses standard optimizations
- Enables new optimizations

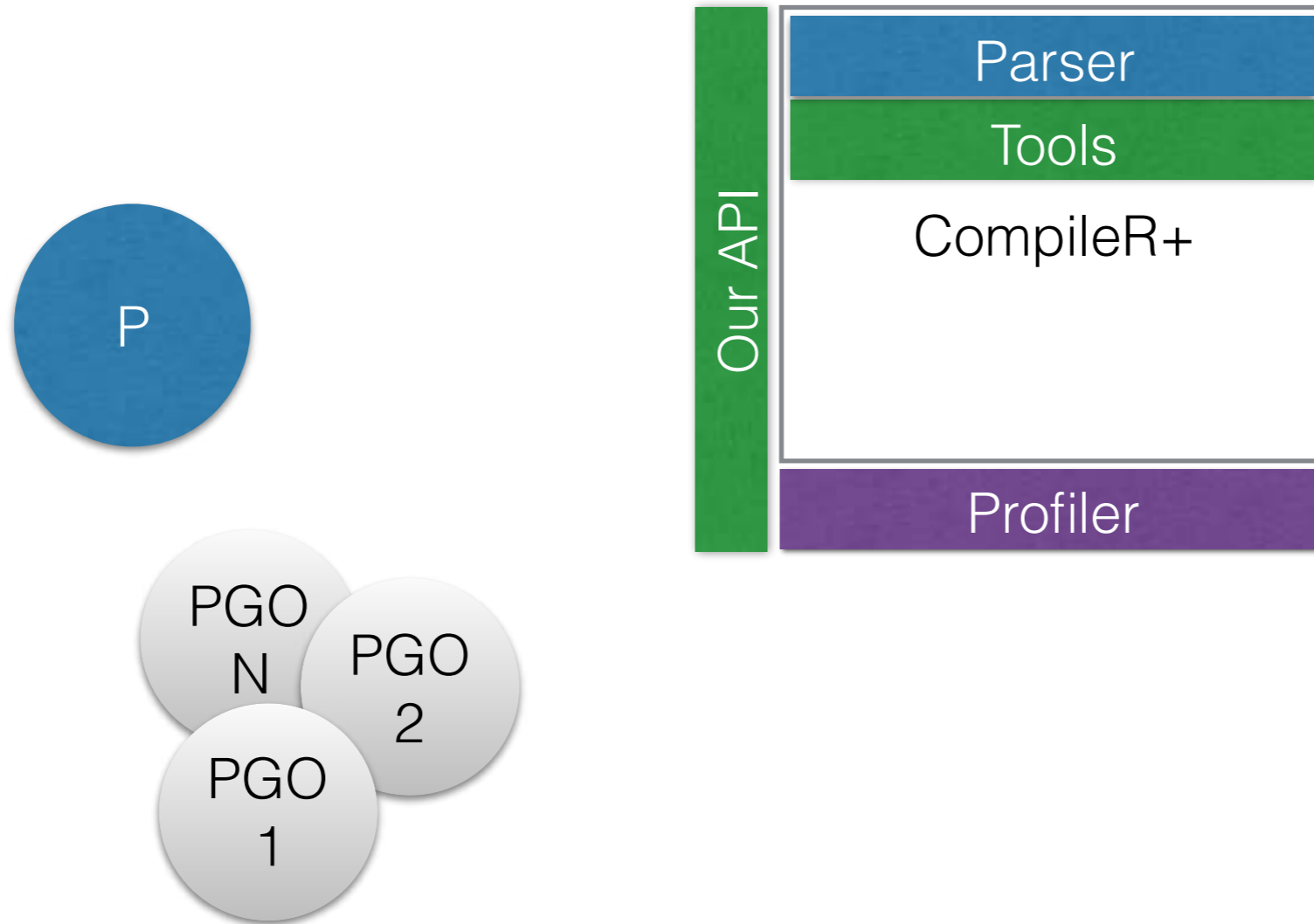
Solutions

For everyone!



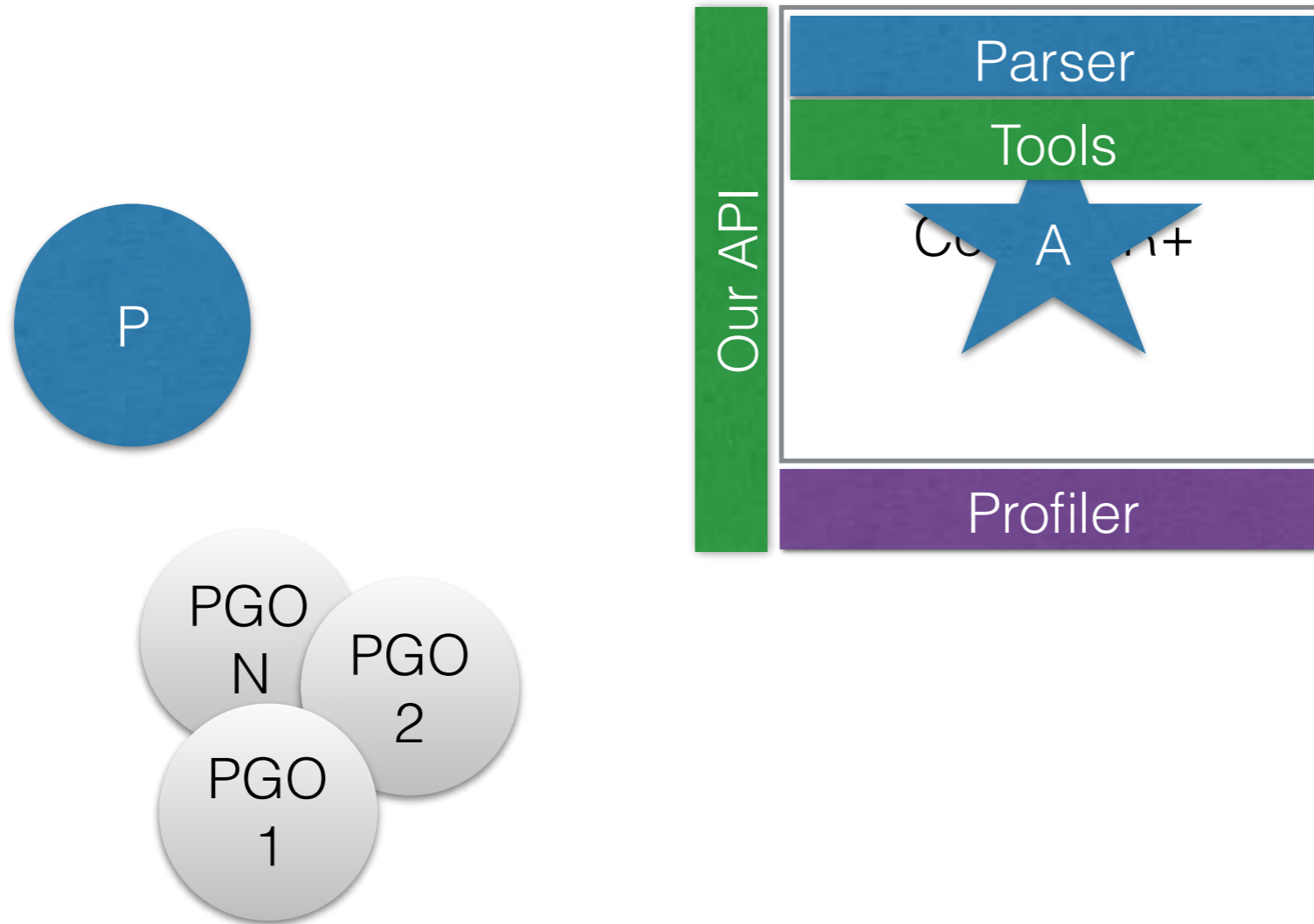
Solutions

For everyone!



Solutions

For everyone!



Design highlights

- Profile Points
- Profile Weights
- API

Profile Points

What the profiler Cares about

- Abstraction of source expressions for profiler
- Profile Point \leftrightarrow Source Expression
- Create new profile points
- Attach profile points to source expressions

Profile Points

An Example

**Source
Loc.**

**Profile
Point**

- Profile Points via source location

File a
Line 2 Char 3 1

- Generate new points
(deterministically)

File A
Line 4
Char 3 2

<GEN3> 3

Profile Weight

What the Meta-Program Cares about

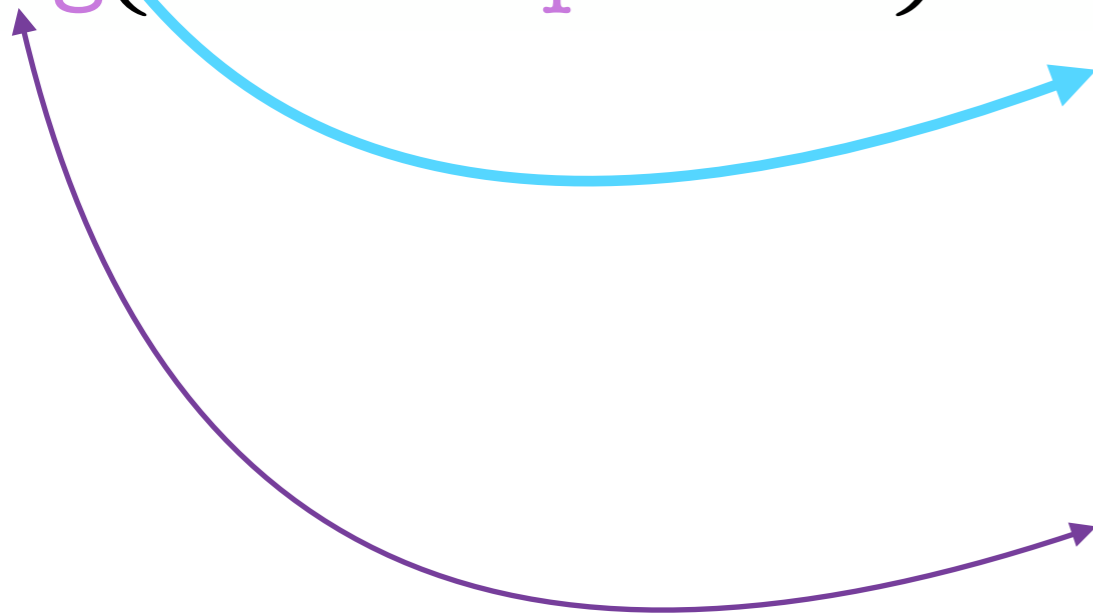
- Abstraction of profile information for PGMPs
- Profile Point \leftrightarrow Profile Weight
- Single value for relative importance
- Easy to combine multiple data sets

Profile Weights

An Example

```
if-r (subject-contains? "PLDI")  
  flag(email spam)  
else:  
  flag(email important)
```

Source Loc.	DS1	DS2	Merged
File a Line 2 Char 3	10/10 = 1	10/100 = .1	$(1 + .1)/2$ = 0.55
File A Line 4 Char 3	5/10 = .5	100/100 = 1	$(.5 + 1)/2$ = 0.75



The API

A brief look at

Evaluation

- We implemented it.
- Profiler: unchanged
- Meta-programming:
unchanged
- Racket API: 134 lines

Evaluation

- We implemented it.
- Profiler: unchanged
- Meta-programming:
unchanged
- Racket API: 134 lines

Evaluation

Profile-Guided ...

- Conditional Branch Optimization:
81 lines
- Receiver Class Prediction:
44* lines
- Data Structure Specialization:
111 lines

Conditional Branch Reordering

```
(define (parse stream)
  (case (peek-char stream)
    [(#\space #\tab) (white-space stream)]
    [(0 1 2 3 4 5 6 7 8 9) (digit stream)]
    [(#\() (start-paren stream)]
    [(#\)) (end-paren stream)]
    ....))
```


Conditional Branch Reordering

```
(define (parse stream)
  (let ([t (peek-char stream)])
    (cond
      [(key-in? t '(#\space #\tab))
       (white-space stream)]
      [(key-in? t '(0 1 2 3 4 5 6 7 8 9))
       (digit stream)]
      [(key-in? t '(#\() (start-paren stream))]
      [(key-in? t '(#\)) (end-paren stream)]
      ....)))
```

Conditional Branch Reordering

```
(define (parse stream)
  (let ([t (peek-char stream)])
    (exclusive-cond
      [(key-in? t '(\space \tab))
       (white-space stream)]
      [(key-in? t '(0 1 2 3 4 5 6 7 8 9))
       (digit stream)]
      [(key-in? t '(\() ) (start-paren stream)]
      [(key-in? t '(\) ) (end-paren stream)]
      ....)))
```

Exclusive-Cond

```
(define-syntax (exclusive-cond syn)
; Internal definitions
(define (clause-weight clause)
  (syntax-case clause ()
    [(test e1 e2 ...) (profile-query #'e1)]))
(define (sort-clauses clause*)
; Sort clauses greatest-to-least by weight
  (sort clause* > #:key clause-weight))
; Start of code transformation
(syntax-case x ()
  [(_ clause ...)
; Splice sorted clauses into a cond expression
  #`(cond #,@(sort-clause #'(clause ...)))]))
```

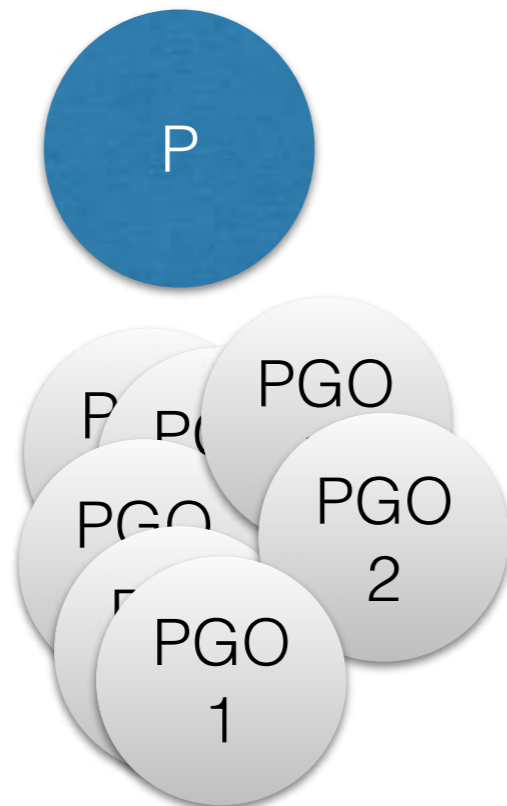
Take a Step bAck

Our Design...

- Saves PGO writers effort
- Saves PGO users effort
- Simple to implement
- Expresses standard optimizations
- Enables new optimizations

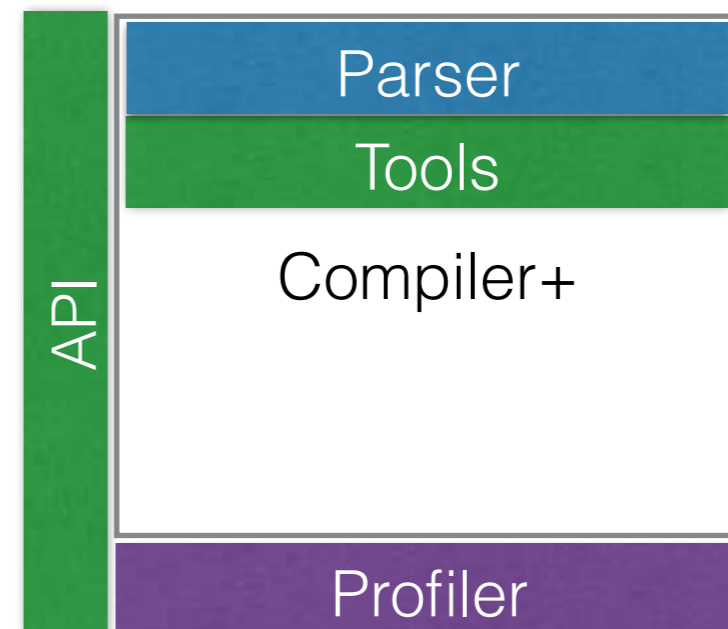
Profile-Guided

- Collect profile at runtime *for*
- Use profile as oracle *to*
- Write optimizations *from*



Meta-Programming

- Extended syntax
- Generate source code
- Embedded DSLs



Why No Benchmarks?

- I do not claim to make anything faster.
As I do not make that claim, I do not support it.
- I do claim a design that reduces programmer effort.
As I make that claim, I support it.
Effort was measured with lines of code.

Non-Scheme languages?

- Yes. See the paper.

optimizations with dependency

- If one PGO depends on another?
Then the user doesn't get composition for free.

C/C++?

- Maybe, with extensions to templates or macros.

Path Profiling?

- Our design may not work with path profiling.
- It is unclear how to apply profile points to paths.

Staged Meta- PRogramming?

- Should work with staged meta-programming.
See the paper.

