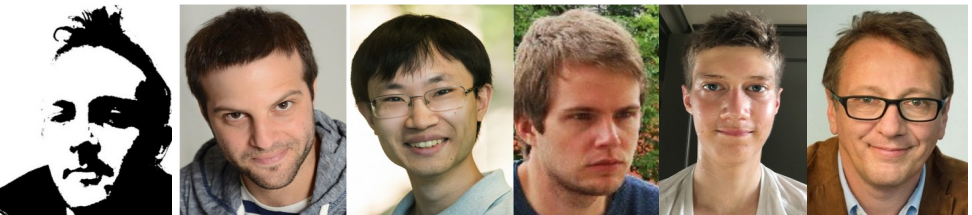
Contextual Dispatch for Function Specialization

Olivier Flückiger, Guido Chari, Ming-Ho Yee,
Jan Ječmen, Jakob Hain, Jan Vitek



Northeastern University
Czech Technical University

Problem

Ř: a fully compliant JIT for R

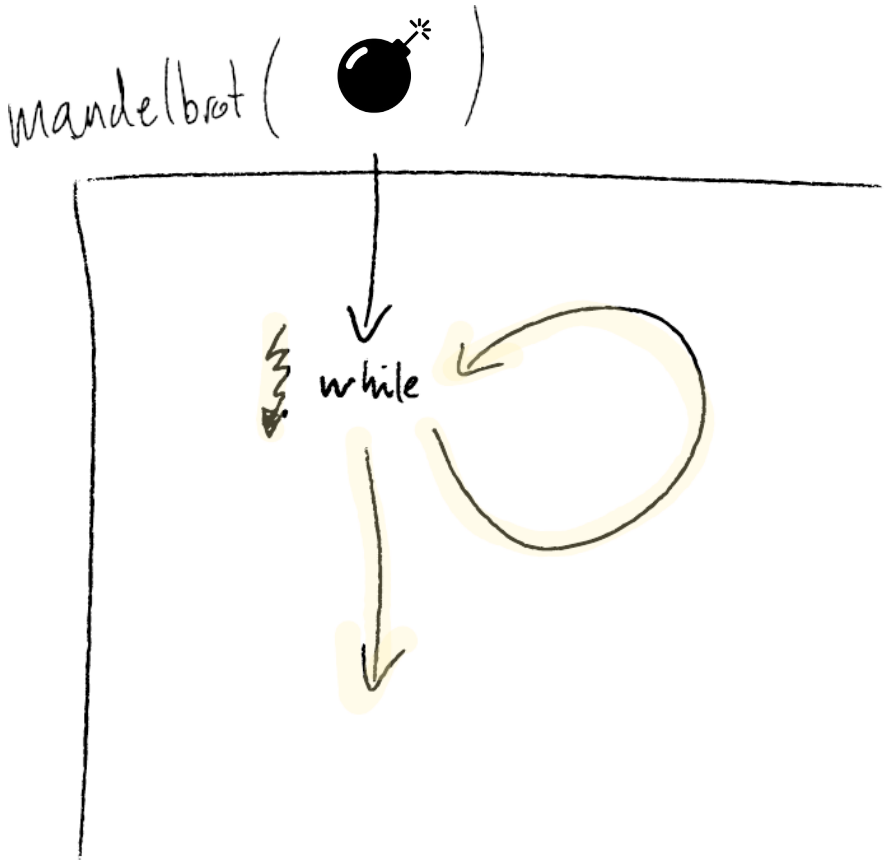
```
mandelbrot ← function (n) {
  i ← 0
  while (i < n) {
    ...
  }
}
```

Problem: Laziness + Reflection

```
mandelbrot(3)
mandelbrot({print("hi"); 3})
```

```
evil <- function() {
  rm(ls="n", envir = sys.frame(-1))
}
```

```
mandelbrot(evil())
```

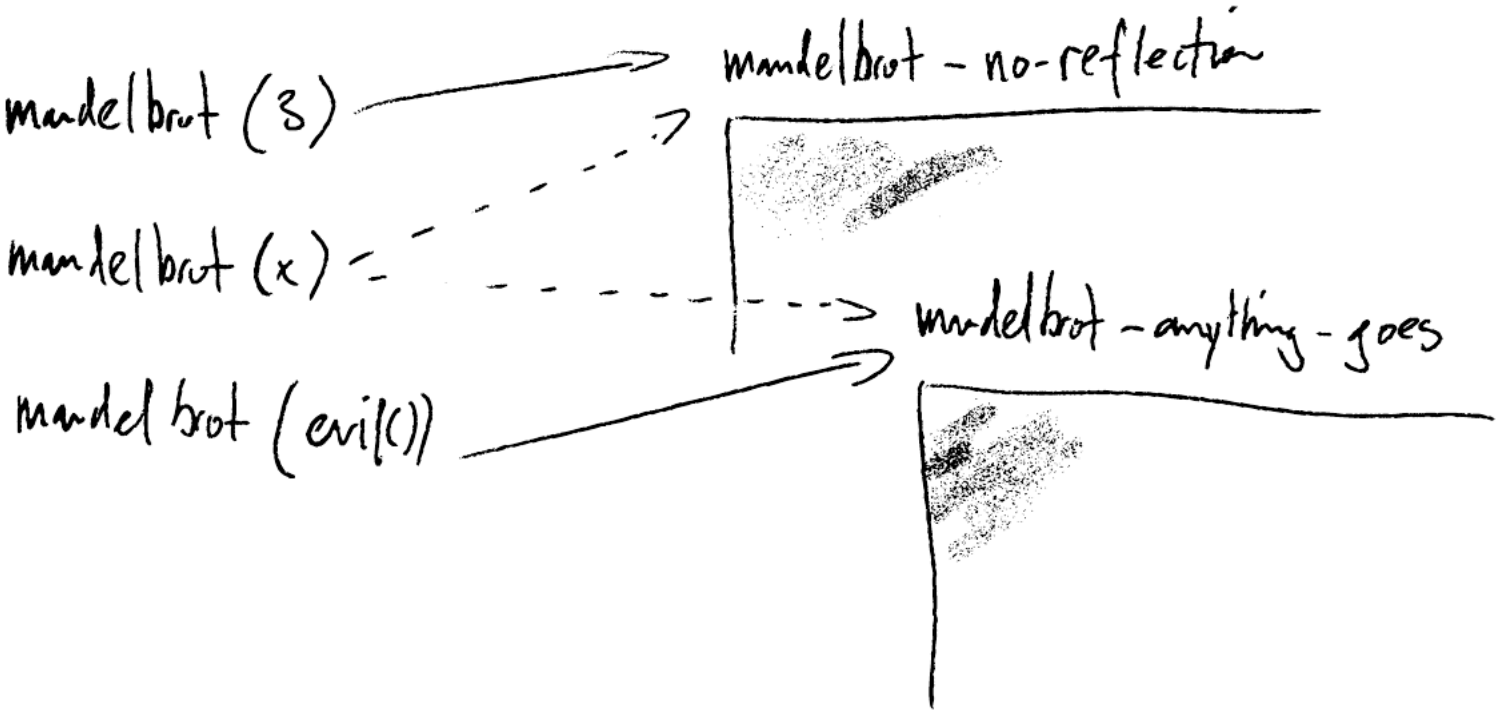

mandelbrot(
)
while

Most promises (thunks) are benign

... but who knows?

... the caller knows

Versioned Functions



Context

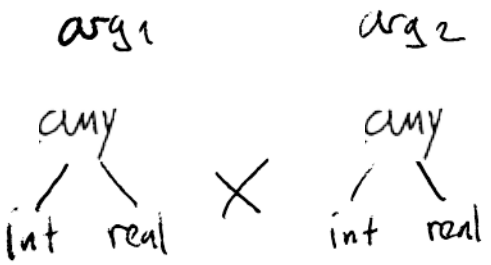
properties of program states

- Boolean Flags e.g. not-reflective
- Types e.g. arg_1 : int
- Shapes e.g. arg_n : scalar
- Quantities eg. n optional args missing

Context

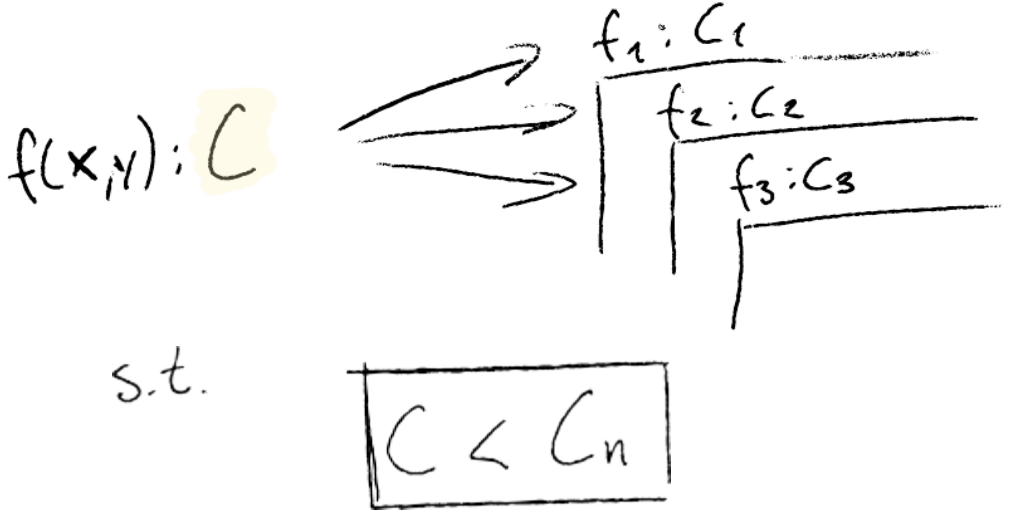
efficiently comparable
properties of program states

- Boolean Flags e.g. not-reflective
- Types e.g. arg_1 : int
- Shapes e.g. arg_n : scalar
- Quantities e.g. n optional args missing



Current Context

a context which holds
at the call site

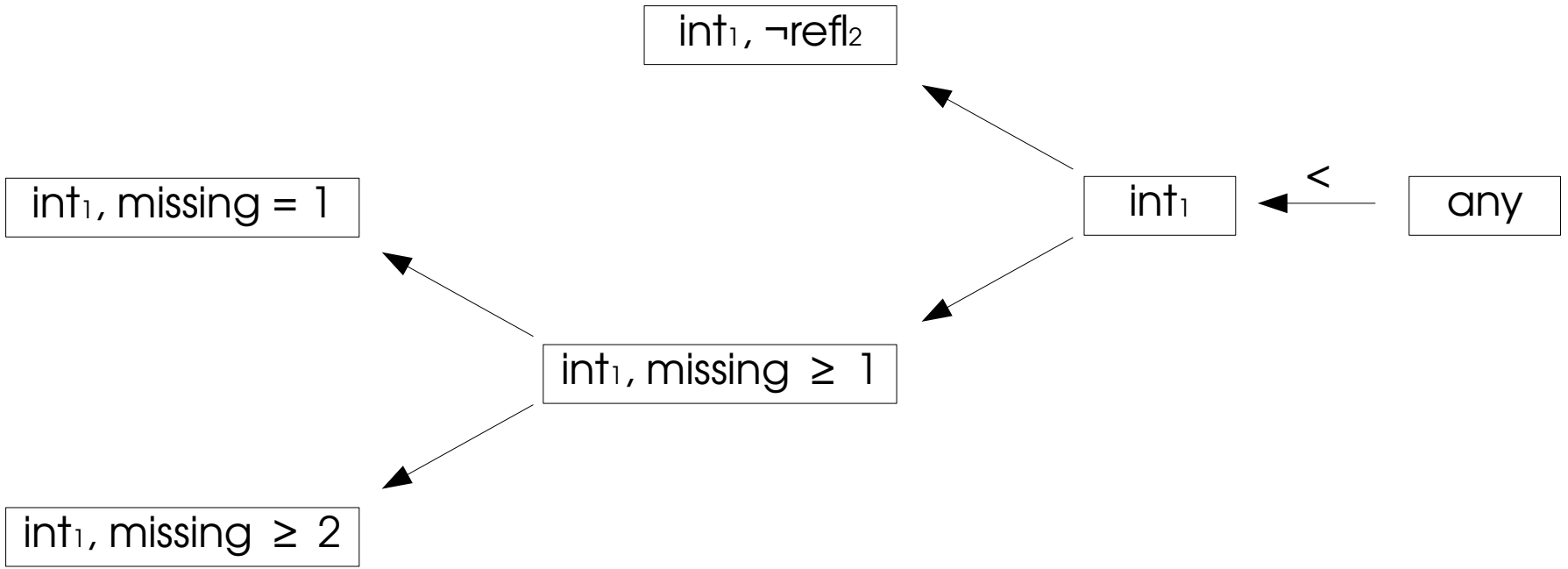


Incremental Computation

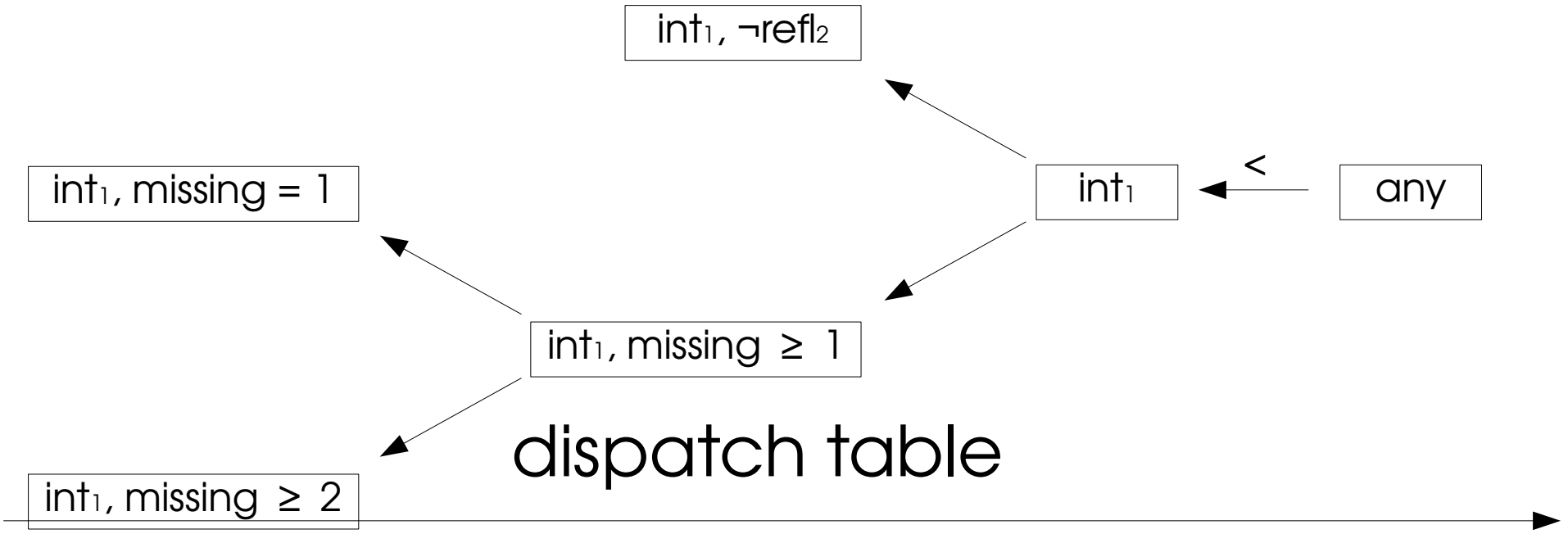
$f(x, 34)$: $C \equiv \underbrace{\langle \text{any}, \text{int} \rangle}_{\text{static}} \wedge \underbrace{\langle \text{real}, \text{any} \rangle}_{\text{dynamic}}$

closed under conjunction

Dispatch

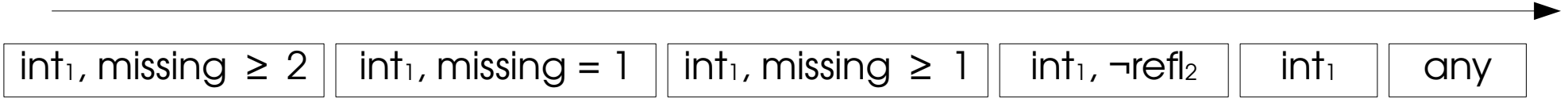

int1, ¬refl2
int1, missing = 1
int1
<
any
int1, missing ≥ 1
int1, missing ≥ 2

Dispatch

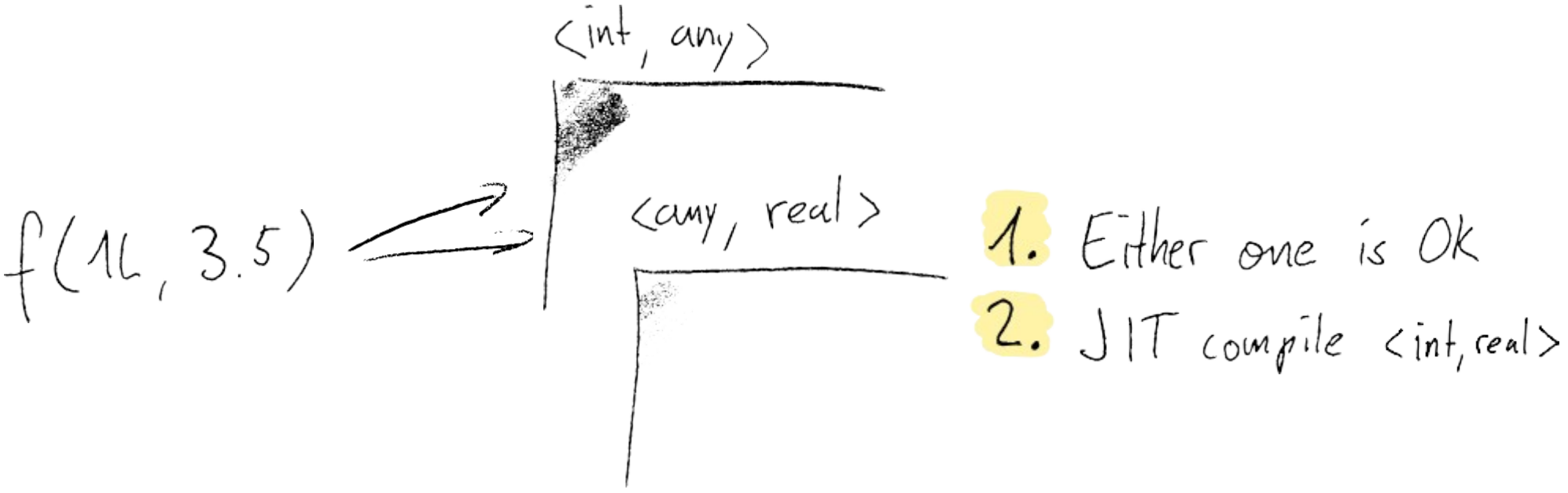

int1, ¬refl2
int1, missing = 1
int1
<
any
int1, missing ≥ 1
dispatch table
int1, missing ≥ 2

Dispatch

dispatch table



Best Target?


<int, any>
f(1L, 3.5)
<any, real>
1. Either one is OK
2. JIT compile <int, real>

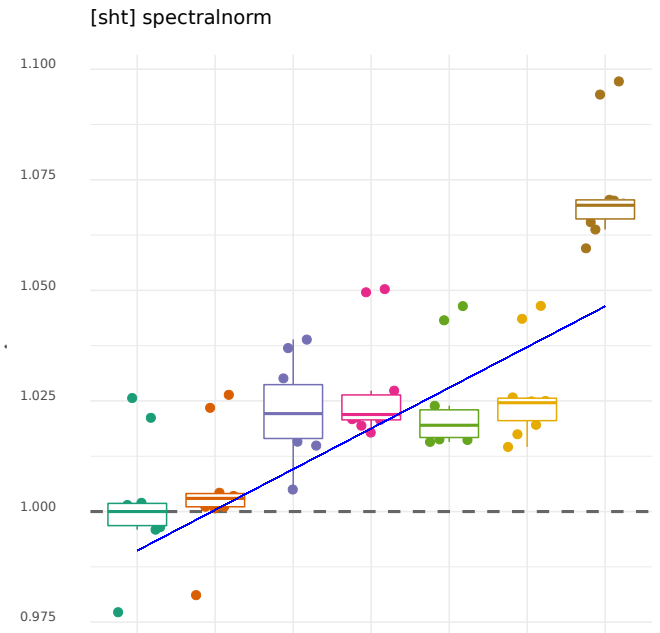
Results

Ř is 1.7x GNU R
(0.7x – 46x)

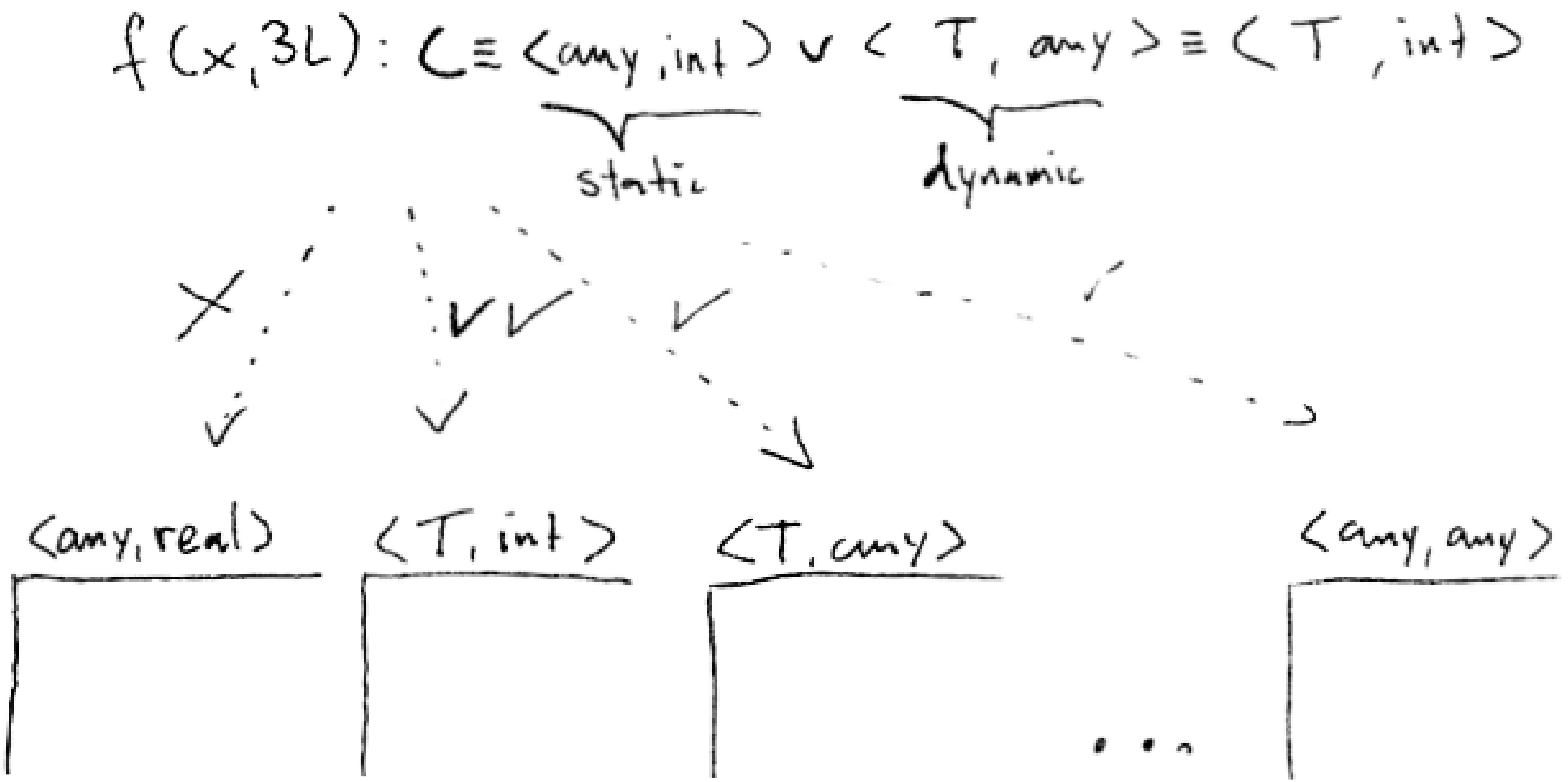
Ř is 0.6 x Fast R
(0.1 – 5.6)

Results

CD improves 18 out of 46



lower bound of $\rho = .05$ above the baseline



o1o.ch/about/cd

r-vm.net