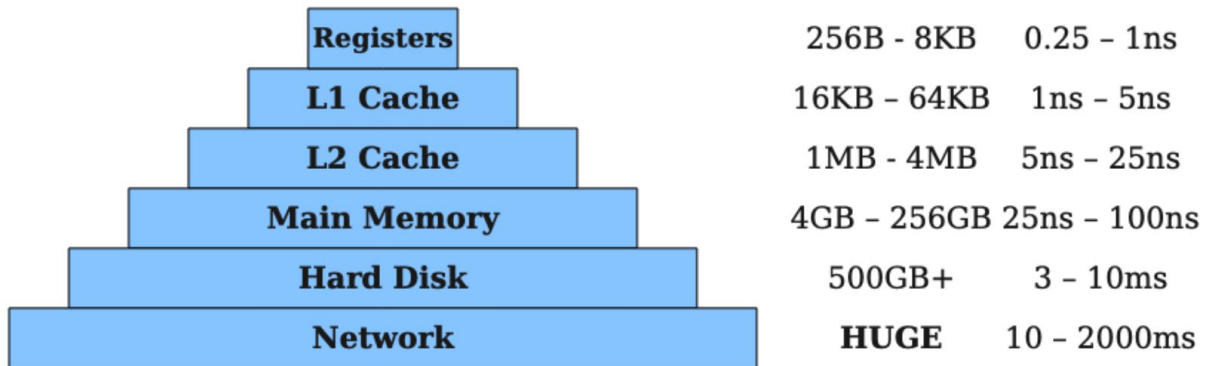


Let's add **register allocation** to our compiler

faster, smaller



slower, bigger

(via Max New)

So far: all variables/values stored on stack (or heap) (**easy, but slow**)

Next: Use the REGISTERS 3-10x **performance gains**, variable access are ubiquitous!

Let's add **register allocation** to our compiler

Example 1

```
(let ((a1 (+ 10 10))
      (a2 (* 2 a1))
      (a3 (* 3 a2)))
  (* 10 a3))
```

Example 2

```
(let ((n (* 5 5))
      (m (* 6 6))
      (x (+ n 1))
      (y (+ m 1)))
  (+ x y)
)
```

Example 3

```
(defn (f a)
  (let ((x (* a 2))
        (y (+ x 7)))
    y))
```

Example 4

```
(defn (f a)
  (let ((x (* a 2))
        (y (+ x 7)))
    (g x y)))
```

But ... what if the programmer *instead* wrote

Example A

```
(* 10 (* 3 (* 2 (+ 10 10))))
```

Example B

```
(+ (+ (* 5 5) 1) (+ (* 6 6) 1))
```

Example C

```
(defn (f a)  
  (+ (* 2 a) 7))
```

1. Administrative Normal Form (ANF)

Immediate expressions: whose values don't require any computation!

- **Constant**, e.g. 1, true, false,
- **Variable**, e.g. x, y, z (whose value is on the stack/reg)

An expression is in ANF when all **primitive operations** have **immediate** arguments

QUIZ: ANF? Yes or No : Example 1, 2, 3, 4, A, B, C

Expr

ANF Expr

(+ (+ 1 2) 3)

(+ (+ (+ 1 2) 3) 4)

(+ (+ (+ 1 2) 3)
(+ (+ 4 5) 6))

2. Compiling with **Allocations**

3. Computing **Allocations** by Graph Coloring