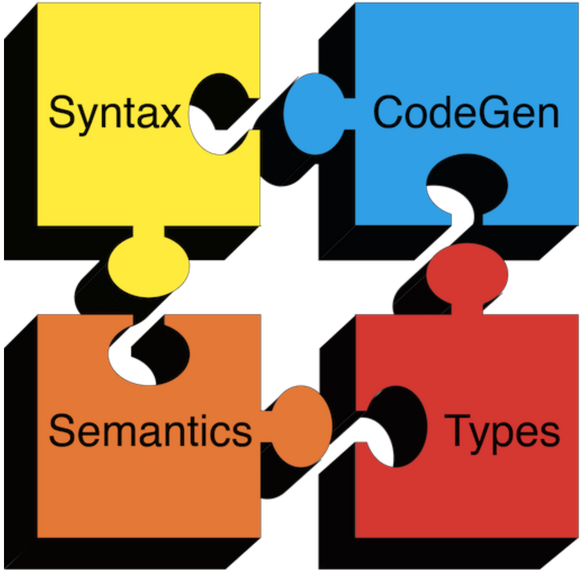


Compilers

Temporaries

Alex Aiken

Temporaries

- Idea: Keep temporaries in the AR

- The code generator must assign a location in the AR for each temporary

Temporaries

```
def fib(x) = if x = 1 then 0 else
    if x = 2 then 1 else
        fib(x - 1) + fib(x – 2)
```

Temporaries

- Let $NT(e)$ = # of temps needed to evaluate e
- $NT(e_1 + e_2)$
  - Needs at least as many temporaries as $NT(e_1)$
  - Needs at least as many temporaries as $NT(e_2) + 1$
- Space used for temporaries in e_1 can be reused for temporaries in e_2

Temporaries

$$NT(e_1 + e_2) = \max(NT(e_1), 1 + NT(e_2))$$

$$NT(e_1 - e_2) = \max(NT(e_1), 1 + NT(e_2))$$

$$NT(\text{if } e_1 = e_2 \text{ then } e_3 \text{ else } e_4) = \max(NT(e_1), 1 + NT(e_2), NT(e_3), NT(e_4))$$

$$NT(id(e_1,\ldots,e_n) = \max(NT(e_1),\ldots,NT(e_n))$$

$$NT(int) = 0$$

$$NT(id) = 0$$

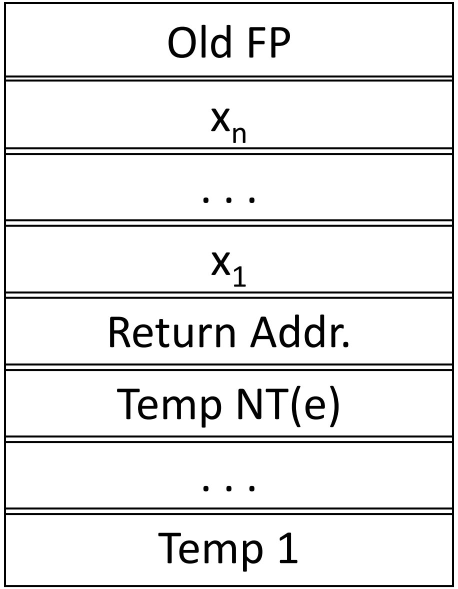
Temporaries

```
def fib(x) = if x = 1 then 0 else
                 if x = 2 then 1 else
                    fib(x - 1) + fib(x – 2)
```

Temporaries

- For a function definition $f(x_1,\ldots,x_n) = e$ the AR has $2 + n + NT(e)$ elements
  - Return address
  - Frame pointer
  - n arguments
  - $NT(e)$ locations for intermediate results

Temporaries



Old FP

x_n

. . .

x_1

Return Addr.

Temp NT(e)

. . .

Temp 1

Temporaries

For the powerOfTwo() function at right, what are the numbers of temporaries required to evaluate each sub-expression, and the total number of temporaries required for powerOfTwo()?

```
def powerOfTwo(x) =
  if x % 2 == 0
  then powerOfTwo(x / 2)
  else x == 1
```

	x % 2 == 0	powerOfTwo(x / 2)	x == 1	Total
○	1	2	2	3
○	1	1	1	1
○	2	1	0	2
○	2	1	0	3

Temporaries

- Code generation must know how many temporaries are in use at each point

- Add a new argument to code generation
  - the position of the next available temporary

- The temporary area is used like a small, fixed-size stack

Temporaries

```
cgen($e_1$ + $e_2$) =
                cgen($e_1$)
                sw $a0 0($sp)
                addiu $sp $sp -4
                cgen($e_2$)
                lw $t1 4($sp)
                add $a0 $t1 $a0
                addiu $sp $sp 4
```

Temporaries

```
cgen($e_1$ + $e_2$, nt) =
        cgen($e_1$, nt)
        sw $a0 nt($fp)
        cgen($e_2$, nt + 4)
        lw $t1 nt($fp)
        add $a0 $t1 $a0
```