LiP

An IMP compiler

IMP: an imperative language

Features:

- Declarations: **const** n=51; **var** x; **array** A[1000]
- Assignment: x := (A[i]+1) * (n-1)
- Sequence: f := f*i; i := i+1
- Conditional: **if** (i=5) **then** a := A[i] **else** a := B[i]
- Iteration: **while** (i<n) **do** f := f*i; i := i+1

Expressions (1)

```
e ::= v             constant
   |  x             identifier
   |  x[e]          identifier + index
   |  e + e'        addition
   |  e – e'        subtraction
   |  e * e'        multiplication
   |  ...
```

Expressions (2)

```
e ::= ...
    |   e < e'        less than
    |   e = e'        equals to
    |   e and e'      logical conjunction
    |   e or e'       logical disjunction
    |   not e         logical negation
```

We represent boolean values as integers: 0 (false) and ≠0 (true)

Declarations

d ::= **const** x = v	constant
| **var** x	variable
| **array** x[v]	array
| d; d'	sequence

Example: **const** n=2; **var** x; **array** A[5]

Commands

c ::=	**skip**	no operation
|	x := e	assign to variable
|	x[e'] := e	assign to array
|	c; c'	sequence
|	**if** e **then** c **else** c'	conditional
|	**while** e **do** c	iteration

Compilation

Compilation functions:

- T_{exp} : IMP expressions → ASM
- T_{com} : IMP commands → ASM
- T_{prog} : IMP programs → ASM

The compilation functions are **partial**! Some programs cannot be compiled.

Compiling expressions

$T_{exp}(e, r, \rho, F)$

- e is the expression to be compiled
- r is the register that will contain the result of the evaluation of e
- ρ is the environment, i.e. a function from identifiers to pairs (type,value/address)
- F is the set of available registers (used to choose tmp registers)

Example

Compile:

$$T_{exp}((A[i]+1) * (n-1), r, \rho, F)$$

in the environment:

$$\rho = \{(var,\ell i)/i, (const,7)/n, (array,\ell a)/A\}$$

and with available registers:

$$F = \{bi, i, ba, a, t1, t2, t3\}$$

Example

$T_{exp}((A[i]+1) * (n-1), r, \rho, \{bi,i,ba,a,n,t1,t2,t3\})$

```
Addi $bi ℓi $0          // ρ(i) = (var,ℓi)
Load $i $bi[$0]         // i
Addi $ba ℓa $0          // ρ(A) = (array,ℓa)
Load $a $ba[$i]         // A[i]
Addi $t1 1 $a           // t1 = A[i] + 1
Addi $n 7 $0            // ρ(n) = (const,7)
Addi $t2 1 $0           // t2 = 1
Sub $t3 $n $t2          // t3 = n – t2
Mul $r $t1 $t3          // r = t1 * t3
```

Values

$T_{exp}(v, r, \rho, F)$ = `Addi $r v $0`

Constants

$T_{exp}(x, r, \rho, F) =$ Addi $r v $0

if $\rho(x) = (\text{const}, v)$

Variables

$T_{exp}(x, r, \rho, F)$ =

```
Addi $bx ℓx $0
Load $r $bx[$0]
```

if $\rho(x) = (\text{var}, \ell x)$, $bx \in F$

Access to array

$T_{exp}(x[e_1], r, \rho, F)$ =

C1
Addi \$bx ℓx \$0
Load \$r \$bx[\$t1]

if $\rho(x) = (\text{array}, \ell x)$, t1, bx $\in$ F

C1 = $T_{exp}(e_1, t1, \rho, F)$

Addition

$T_{exp}(e_1+e_2, r, \rho, F)$ =

```
C1
C2
Add $r $t1 $t2
```

where t1, t2 $\in$ F

C1 = $T_{exp}(e_1, t1, \rho, F)$

C2 = $T_{exp}(e_2, t2, \rho, F-\{t1\})$

Subtraction

$T_{exp}(e_1 - e_2, r, \rho, F)$ =

C1
C2
Sub $r $t1 $t2

where t1, t2 $\in$ F

C1 = $T_{exp}(e_1, t1, \rho, F)$

C2 = $T_{exp}(e_2, t2, \rho, F-\{t1\})$

Multiplication

$T_{exp}(e_1 * e_2, r, \rho, F) =$

C1

C2

Mul $r $t1 $t2

where t1, t2 $\in$ F

C1 $= T_{exp}(e_1, t1, \rho, F)$

C2 $= T_{exp}(e_2, t2, \rho, F-\{t1\})$

Comparison =

$T_{exp}(e_1=e_2, r, \rho, F)$ =

```
      C1
      C2
      Beq $t1 $t2 eq
      Addi $r 0 $0
      Jmp cont
eq:   Addi $r 1 $0
cont: ...
```

where t1, t2 $\in$ F

C1 = $T_{exp}(e_1, t1, \rho, F)$

C2 = $T_{exp}(e_2, t2, \rho, F\text{-}\{t1\})$

Warning!

Labels must be **unique** in ASM programs

Comparison <

$T_{exp}(e_1<e_2, r, \rho, F)$ =

C1
C2
Slt $r $t1 $t2

where t1, t2 $\in$ F

C1 = $T_{exp}(e_1, t1, \rho, F)$

C2 = $T_{exp}(e_2, t2, \rho, F\text{-}\{t1\})$

Not

$T_{exp}(\text{not } e_1, r, \rho, F)$ =

```
        C1
        Addi $r 1 $0
        Beq $t1 $0 cont
        Addi $r 0 $0
cont: ...
```

where $t1 \in F$

C1 = $T_{exp}(e_1, t1, \rho, F)$

And

$T_{exp}(e_1 \text{ and } e_2, r, \rho, F) =$

```
        C1
        C2
        Addi $r 0 $0
        Beq $t1 $0 cont
        Beq $t2 $0 cont
        Addi $r 1 $0
cont: ...
```

where t1, t2 $\in$ F

C1 $= T_{exp}(e_1, t1, \rho, F)$

C2 $= T_{exp}(e_2, t2, \rho, F\text{-}\{t1\})$

Or

$T_{exp}(e_1 \text{ or } e_2, r, \rho, F) =$

```
        C1
        C2
        Addi $r 1 $0
        Bne $t1 $0 cont
        Bne $t2 $0 cont
        Addi $r 0 $0
cont: ...
```

where t1, t2 $\in$ F

C1 = $T_{exp}(e_1, t1, \rho, F)$

C2 = $T_{exp}(e_2, t2, \rho, F-\{t1\})$

Compiling commands

$T_{com}(c, \rho, F)$

- ρ is an environment, i.e. a function from identifiers to pairs (type, value/address)
- F è the set of available registers (used to choose tmp registers)

Skip

$$T_{com}(\textbf{skip}, \rho, F) = \text{Nop}$$

Assignment to variables

$T_{com}(x := e, \rho, F) =$

```
C1
Addi $bx ℓx $0
Store $bx[$0] $t1
```

where $\rho(x) = (\text{var}, \ell x)$, $t1, bx \in F$

C1 = $T_{exp}(e, t1, \rho, F)$

Assignment to arrays

$T_{com}(x[e_1] := e_2, \rho, F) =$

```
C1
C2
Addi $bx ℓx $0
Store $bx[$t1] $t2
```

where $\rho(x) = (\text{array}, \ell x)$, $t1, t2, bx \in F$

C1 $= T_{exp}(e_1, t1, \rho, F)$

C2 $= T_{exp}(e_2, t2, \rho, F-\{t1\})$

Sequence

$T_{com}(c_1; c_2, \rho, F)$ =

C1
C2

where:

C1 = $T_{com}(c_1, \rho, F)$

C2 = $T_{com}(c_2, \rho, F)$

Conditional

$T_{com}(\textbf{if}\ e\ \textbf{then}\ c_1\ \textbf{else}\ c_2, \rho, F) =$

```
        Ce
        Beq $t $0 FF
        C1
        Jmp cont
FF:     C2
cont: ...
```

where $t \in F$

$\textbf{Ce} = T_{exp}(e, t, \rho, F)$

$\textbf{C1} = T_{com}(c_1, \rho, F)$

$\textbf{C2} = T_{com}(c_2, \rho, F)$

While

$T_{com}(\textbf{while}\ e\ \textbf{do}\ c, \rho, F) =$

```
loop: C
      Beq $t $0 cont
      C'
      Jmp loop
cont: ...
```

where $t \in F$ $\quad \mathbf{C} = T_{exp}(e, t, \rho, F)$

$\mathbf{C'} = T_{com}(c, \rho, F)$

Compilation of programs

ASM code

$$T_{prog}(\textbf{program}\ d\ \textbf{begin}\ c\ \textbf{end}) = \left(\begin{array}{l} \mathbf{C} \\ \text{Halt} \end{array}, \ell \right)$$

Address of first instruction

Environment (symbol table)

$$\text{dove}\ (d,\{\},0) \rightarrow_{dec} (\rho, \ell)$$

Set F of available registers

$$\mathbf{C} = T_{com}(c, \rho, [1..63])$$