Soutenance de Compilation

Paul Charles Arsène Volte

Enseirb-Matmeca

Décembre 2017

Table des matières

Variables

Déclaration

```
int a,b;

%a = alloca i32
%b = alloca i32
```

Enjeux

- Reconnaître le type : `int` ou `float`
- Erreur si le type déclaré est `void`
- Déclarer plusieurs variables simultanément

Affectation

```
a = 1;

%r0 = add i32 1, 0
store i32 %r0, i32* %a
```

Enjeux

- Stocker la variable/constante à affecter dans un registre
- Convertir éventuellement ce registre avec `sitofp` ou `fptosi`
- Afficher les flottants en hexadécimal

Fonctions

Déclaration et retour

```
int foo(int a, float b) {
    return a;
}

define i32 @foo(i32 %a.val, float %b.val) {
    %a = alloca i32
    %b = alloca float
    %r0 = load i32, i32* %a
    ret i32 %r1
}
```

Enjeux

- Allouer les arguments
- Vérifier la cohérence du type du registre retourné
- Retourner `void` si rien n'est retourné

Tables de correspondance

Associent des identifiants à des données, afin de savoir :

- si une variable a déjà été déclarée/initialisée
- le type d'une variable, son nombre de référencements, et sa portée
- le type de retour d'une fonction
- le nombre d'arguments d'une fonction et leurs types

Expressions

Les tables servent lors de l'évaluation d'une variable/l'appel d'une fonction :

```
int f(int x) {
    return x;
}
```

```
int a;
a = 1;
a = f(2) + a * 2.0;
```

Booléens

Évalués avec paresse

Booléens et labels

`bool` :

- label si vrai
- label si faux

$1 == 2$

```
%r0 = add i32 1, 0
%r1 = add i32 2, 0
%r2 = icmp seq i32 %r0, %r1
br i1 %r2, label %L0, label %L1
```

Évaluation paresseuse

$1 == 2 \;\&\&\; 3 == 3$

```
%r0 = add i32 1, 0
%r1 = add i32 2, 0
%r2 = icmp seq i32 %r0, %r1
br i1 %r2, label %L0, label %L1
L0:
%r3 = add i32 3, 0
%r4 = add i32 3, 0
%r5 = icmp seq i32 %r3, %r4
br i1 %r5, label %L2, label %L3
L3:
br label %L1
```

Label si vrai : L2, label si faux : L1

Structures de contrôle

if..else, while..do, do..while

if..else

```
if (1 == 1) {
    ..
} else {
    ...
}
```

```
%r0 = add i32 1, 0
%r1 = add i32 1, 0
%r2 = icmp seq i32 %r0, %r1
br i1 %r2,label %L0,label %L1
L0:
..
br label %L2
L1:
...
L2:
```

do..while

```
do {
    ...
} while (1 == 1);
```

```
L0:
...
%r0 = add i32 1, 0
%r1 = add i32 1, 0
br %r2 = icmp seq i32 %r0, %r1
br i1 %r2,label %L1,label %L2
L1:
br label %L0
L2:
```