TypesA4M36TPJ, 2013/2014

Motivation

"Well-typed programs never go wrong"

What does "go wrong" mean?

- Program crash
- Execution error
- Divergence
- Unknown method call.....

The Purpose of Typing

 Well-typed programs are more likely to actually do what they are supposed to do.

Two Kinds of Typing

- Static typing.
- Dynamic checking.

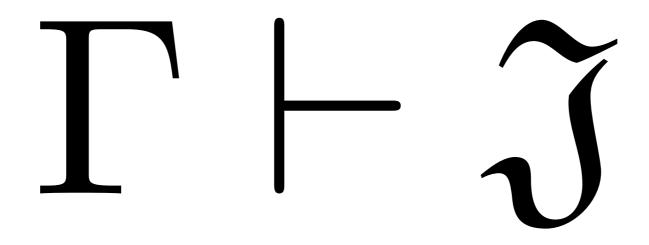
Static Typing

Formally specified by a Type System.

Simple Type System Specification

- Judgements.
- Type rules.
- Environment.

Judgements General Form



Judgements Examples

$$\Gamma \vdash M : A$$
 $\emptyset \vdash true : Boolean$
 $\emptyset \vdash 1 : Nat$

Type Rules

- Type rules assert the validity of certain judgments on the basis of other judgments that are already known to be valid.
- The process gets off the ground by some intrinsically valid judgment (usually: empty environment is well formed).

Type Rules (II)

$$\frac{\Gamma_1 \vdash \mathfrak{J}_1 \cdots \Gamma_n \vdash \mathfrak{J}_n}{\Gamma \vdash \mathfrak{J}}$$

Example of EXPR Language

```
Expr ::= Num \mid
Bool \mid
\triangle Expr \mid
Expr \odot Expr \mid
Expr \leq Expr \mid
Expr nand Expr \mid
if Expr then Expr else Expr,
```

where Num is a predefined set of integer numbers (a.k.a. Z) and Bool is a predefined set of boolean values.

Rules for EXPR

Convention: $e, e', e'', \ldots \in Expr, b, b' \in Bool \text{ and } n, n' \in Num.$

 $\frac{}{n:Number}$ (Val N)

(Val B)

 $\overline{b:Boolean}$

Rules (II) for EXPR

```
\frac{e:Number}{\triangle e:Number} \text{ (Val triangle)}
```

```
\frac{e:Number}{e\odot e':Number} \frac{e':Number}{e:Number} \text{ (Val circle)} \\ \frac{e:Number}{e':Number} \frac{e':Number}{e':Boolean} \text{ (Val leq)}
```

Rules(III) for EXPR

```
\frac{e:Boolean}{e \text{ nand } e':Boolean} \text{ (Val nand)}
```

```
\frac{e:Boolean}{\text{if }e \text{ then }e' \text{ else }e'':Number}{e:Boolean} \text{ (Val ifNum)} \\ \frac{e:Boolean}{\text{if }e \text{ then }e' \text{ else }e'':Boolean}{\text{if }e \text{ then }e' \text{ else }e'':Boolean} \\ \text{(Val ifBool)}
```

Judgement for EXPR

- Do we need an Environment in Type System of EXPR?
- Do we have any expressions that contain variables in EXPR?

Derivation in EXPR

```
\frac{(\text{Val N}) - \frac{(\text
```