软件理论基础与实践

Types: Type Systems

熊英飞
北京大学

课程进展

- Coq √
- 数理逻辑基础 √
- 形式语义 √
- 类型系统

什么是类型系统?

- 类型系统是一种用来自动和高效证明程序没有某类错误的语法手段
  - 没有某类错误
    - C的类型系统：保证变量之间没有跨类型的错误赋值
    - Java的异常处理机制：保证程序执行中没有不被处理的异常（runtimeException除外）
  - 不能证明有某类错误
    - 通不过类型检查不代表运行时有可能出现错误
  - 语法手段
    - 通常需要程序员在程序中加上额外标记
  - 自动
    - 加上标记之后证明是自动的
  - 高效
    - 证明过程不能对编译速度有太大影响

类型系统的重要性

- 现代程序设计语言都有较为复杂的类型系统
- 类型系统设计和语言其他部分设计通常同时进行
- 类型理论是软件理论的核心组成部分
- 类型系统有很多作用
  - 编译时错误检测
  - 抽象：类型系统是对程序行为的抽象
  - 文档：类型系统帮助更好的撰写程序
  - 效率：加上类型系统的程序可能会更高效

类型系统的实现思想

- 对程序状态进行分类
- 程序分析角度——类型定义了系统状态的抽象域和抽象域上要检查的断言

带类型的算术表达式：语法

- 之前我们在语法上严格区分了aexp和bexp
  - 不可能写出if(a+1) then 1 else 2
- 为了体现类型的作用，我们在语法上混用自然数和布尔值
  - ```
    t ::= tru
        | fls
        | test t then t else t
        | zro
        | scc t
        | prd t
        | iszro t
    ```
- 但在更复杂的情况中（如有子类型），类型系统的作用很难被语法取代
    ```

语法元素的Coq定义


```
Inductive tm : Type :=
  | tru : tm
  | fls : tm
  | test : tm -> tm -> tm -> tm
  | zro : tm
  | scc : tm -> tm
  | prd : tm -> tm
  | iszro : tm -> tm.
```

定义值


```
Inductive bvalue : tm -> Prop :=
  | bv_tru : bvalue tru
  | bv_fls : bvalue fls.

Inductive nvalue : tm -> Prop :=
  | nv_zro : nvalue zro
  | nv_scc : forall t, nvalue t -> nvalue (scc t).

Definition value (t : tm) := bvalue t \/ nvalue t.
```

小步法语义

$$\frac{}{\texttt{test tru then } t_1 \texttt{ else } t_2 \rightarrow t_1} \quad \text{(ST\_TestTru)}$$

$$\frac{}{\texttt{test fls then } t_1 \texttt{ else } t_2 \rightarrow t_2} \quad \text{(ST\_TestFls)}$$

$$\frac{t_1 \rightarrow t_1'}{\texttt{test } t_1 \texttt{ then } t_2 \texttt{ else } t_3 \rightarrow \texttt{test } t_1' \texttt{ then } t_2 \texttt{ else } t_3} \quad \text{(ST\_Test)}$$

小步法语义

$$\frac{t_1 \rightarrow t_1'}{\text{scc } t_1 \rightarrow \text{scc } t_1'} \quad \text{(ST\_Scc)}$$

$$\frac{}{\text{prd zro} \rightarrow \text{zro}} \quad \text{(ST\_PrdZro)}$$

$$\frac{\text{numeric value } v}{\text{prd (scc } v) \rightarrow v} \quad \text{(ST\_PrdScc)}$$

$$\frac{t_1 \rightarrow t_1'}{\text{prd } t_1 \rightarrow \text{prd } t_1'} \quad \text{(ST\_Prd)}$$

小步法语义

$$\frac{}{\texttt{iszro zro} \rightarrow \texttt{tru}} \quad \text{(ST\_IszroZro)}$$

$$\frac{\texttt{numeric value v}}{\texttt{iszro (scc v)} \rightarrow \texttt{fls}} \quad \text{(ST\_IszroScc)}$$

$$\frac{t_1 \rightarrow t_1'}{\texttt{iszro } t_1 \rightarrow \texttt{iszro } t_1'} \quad \text{(ST\_Iszro)}$$

小步法语义的Coq定义


```
Reserved Notation "t '-->' t'" (at level 40).

Inductive step : tm -> tm -> Prop :=
  | ST_TestTru : forall t1 t2,
      (test tru t1 t2) --> t1
  | ST_TestFls : forall t1 t2,
      (test fls t1 t2) --> t2
  | ST_Test : forall t1 t1' t2 t3,
      t1 --> t1' ->
      (test t1 t2 t3) --> (test t1' t2 t3)
  | ST_Scc : forall t1 t1',
      t1 --> t1' ->
      (scc t1) --> (scc t1')
```

小步法语义的Coq定义


```
| ST_PrdZro :
    (prd zro) --> zro
| ST_PrdScc : forall v,
    nvalue v ->
    (prd (scc v)) --> v
| ST_Prd : forall t1 t1',
    t1 --> t1' ->
    (prd t1) --> (prd t1')
| ST_IszroZro :
    (iszro zro) --> tru
| ST_IszroScc : forall v,
     nvalue v ->
    (iszro (scc v)) --> fls
| ST_Iszro : forall t1 t1',
    t1 --> t1' ->
    (iszro t1) --> (iszro t1')

where "t '-->' t'" := (step t t').
```

标准型

- Strong Progress性质要求标准型一定是值
- 但现在的标准型却不一定是值，如
  - scc tru
- 称上面的项是卡住的项

```
Definition stuck (t : tm) : Prop :=
  normal_form step t /\ ~ value t.

Lemma some_term_is_stuck :
  exists t, stuck t.
```

类型

- 类型帮助我们避免卡住的项

```
Inductive ty : Type :=
  | Bool : ty
  | Nat : ty.
```

类型关系

- 将表达式关联到计算结果的类型上

$$\frac{}{\vdash \mathtt{tru} \in \mathtt{Bool}} \text{(T\_Tru)}$$

$$\frac{}{\vdash \mathtt{fls} \in \mathtt{Bool}} \text{(T\_Fls)}$$

$$\frac{\vdash t_1 \in \mathtt{Bool} \qquad \vdash t_2 \in T \qquad \vdash t_3 \in T}{\vdash \mathtt{test}\ t_1\ \mathtt{then}\ t_2\ \mathtt{else}\ t_3 \in T} \text{(T\_Test)}$$

$$\frac{}{\vdash \mathtt{zro} \in \mathtt{Nat}} \text{(T\_Zro)} \qquad \frac{\vdash t_1 \in \mathtt{Nat}}{\vdash \mathtt{prd}\ t_1 \in \mathtt{Nat}} \text{(T\_Prd)}$$

$$\frac{\vdash t_1 \in \mathtt{Nat}}{\vdash \mathtt{scc}\ t_1 \in \mathtt{Nat}} \text{(T\_Scc)} \qquad \frac{\vdash t_1 \in \mathtt{Nat}}{\vdash \mathtt{iszro}\ t_1 \in \mathtt{Bool}} \text{(T\_IsZro)}$$

类型关系的Coq定义


```
Reserved Notation "'|-
' t '\in' T" (at level 40).

Inductive has_type : tm -> ty -> Prop :=
  | T_Tru :
       |- tru \in Bool
  | T_Fls :
       |- fls \in Bool
  | T_Test : forall t1 t2 t3 T,
       |- t1 \in Bool ->
       |- t2 \in T ->
       |- t3 \in T ->
       |- test t1 t2 t3 \in T
  | T_Zro :
       |- zro \in Nat
```

类型关系的Coq定义


```
| T_Scc : forall t1,
      |- t1 \in Nat ->
      |- scc t1 \in Nat
| T_Prd : forall t1,
      |- t1 \in Nat ->
      |- prd t1 \in Nat
| T_Iszro : forall t1,
      |- t1 \in Nat ->
      |- iszro t1 \in Bool

where "'|-' t '\in' T" := (has_type t T).
```

进展性 & 保持性

- 进展性：类型正确的项如果不是值，就一定能往下约简
- 保持性：项往下约减之后还保持同样的类型

- 两个属性联合作用保证了项不会卡住

进展性Progress


```
Theorem progress : forall t T,
  |- t \in T ->
  value t \/ exists t', t --> t'.
```

作为对比，上一章strong progress属性是对所有term都成立

证明思路：根据推导出t \in T关系的类型规则做归纳证明。
证明需要用到如下lemma

```
Lemma bool_canonical : forall t,
  |- t \in Bool -> value t ->
bvalue t.

Lemma nat_canonical : forall t,
  |- t \in Nat -> value t ->
nvalue t.
```

保持性Preservation


```
Theorem preservation : forall t t' T,
  |- t \in T ->
  t --> t' ->
  |- t' \in T.
```

证明思路：还是根据推导出t \in T关系的类型规则做归纳证明

类型正确性


```
Definition multistep := (multi step).
Notation "t1 '-->*' t2" := (multistep t1 t2) (at level 40).

Corollary soundness : forall t t' T,
  |- t \in T ->
  t -->* t' ->
  ~(stuck t').
Proof.
  intros t t' T HT P. induction P; intros [R S].
  - apply progress in HT. destruct HT; auto.
  - apply IHP.
    + apply preservation with (t := x); auto.
    + unfold stuck. split; auto.
Qed.
```

作业

- 完成Types中standard非optional并不属于Additional Exercises的5道习题
  - 请使用最新英文版教材