

# Types and Programming Language

枫聆

2021 年 3 月 30 日

## 目录

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>Untyped Systems</b>	<b>3</b>
2.1	Syntax . . . . .	3

## Introduction

**Definition 1.1.** A **type system** is a tractable syntactic method for proving the absence of certain program behaviors by classifying phrases according to the kinds of value they compute.

type system 是一种用于证明某些确定的程序行为不会发生的方法，它怎么做呢？通过它们计算出值的类型来分类，有点抽象... 我想知道 the kinds of value they compute 是什么？如何分类？分类之后接下来该怎么做？

**Annotation 1.2.** Being static, type systems are necessarily also **conservative**: they can categorically prove the absence of some bad program behaviors, but they can't prove their presence.

**Example 1.3.**

```
1 if <complex test> then 5 else <type error>
```

上面这个 annotation 在说 type system 只能证明它看到的一些 bad program behavior 不会出现，但是它们可能会 reject 掉一些 runtime time 阶段运行良好的程序，例如在 runtime 阶段上面的 else 可能永远都不会进。即 type system 无法证明它是否真的存在。

# Untyped Systems

## Syntax

**Definition 2.1.** The set of terms is the smallest set  $\mathcal{T}$  such that

1.  $\{\text{true}, \text{false}, 0\} \subseteq \mathcal{T}$ ;
2. if  $t_1 \in \mathcal{T}$ , then  $\{\text{succ } t_1, \text{pred } t_1, \text{iszero } t_1\} \subseteq \mathcal{T}$ ;
3. if  $t_1 \in \mathcal{T}, t_2 \in \mathcal{T}, t_3 \in \mathcal{T}$ , then **if**  $t_1$  **then**  $t_2$  **else**  $t_3 \in \mathcal{T}$ .

**Definition 2.2.** The set of terms is defined by the following rules:

$$\begin{array}{c} \frac{\text{true} \in \mathcal{T}}{t_1 \mathcal{T}} \\ \hline \text{succ } t_1 \in \mathcal{T} \end{array} \quad \begin{array}{c} \frac{\text{false} \in \mathcal{T}}{t_1 \mathcal{T}} \\ \hline \text{succ } t_1 \in \mathcal{T} \end{array} \quad \begin{array}{c} \frac{0 \in \mathcal{T}}{t_1 \mathcal{T}} \\ \hline \text{succ } t_1 \in \mathcal{T} \end{array}$$
$$\frac{t_1 \in \mathcal{T} \quad t_2 \in \mathcal{T} \quad t_3 \in \mathcal{T}}{\text{if } t_1 \text{ then } t_2 \text{ else } t_3}$$