# Type Checking - 1

Dynamics - Version 1

SERC/IIITH

March, 2015

The main reference for the present material is [1]. Students must note that test questions will depend not only on these skeletal notes but also the material covered in class.

## 1 The Language $L(num, plus)$

We will consider set the $Exp$ of expressions $e$ as described below.

| | Abstract Syntax | Concrete Syntax | Description |
|---|---|---|---|
| $e ::=$ | | | |
| | $num[n]$ | $n$ | numeral |
| | $plus(e_1; e_2)$ | $e_1 + e_2$ | addition |

### 1.1 The size of an expression

We define the size function $|.| : Exp \to \mathbb{N}$ by induction on $Exp$.

1. $|num[n]| = 1$.
2. $|plus(e_1; e_2)| = 1 + |e_1| + |e_2|$.

For example, $|plus(num[2]; num[3])| = 3$.

### 1.2 Problem

Q: What is the meaning of these expressions?

A: The meaning is described in terms of the transitions of an abstract *transition system* as described next.

### 1.3 Transition Systems

A transition system $TS$ consists of

1. A set of states $S$.
2. A subset of initial states $S_{init} \subseteq S$.
3. A subset of final states $S_{fin} \subseteq S$.
4. A binary relation $\mapsto$ on states.

$TS$ is deterministic if each state transits to at most one state.
We express the properties of transition using judgements. Let $\overset{*}{\mapsto}$ represent the iteration of the binary relation $\mapsto$. This extended relation $\overset{*}{\mapsto}$ is defined by the three following judgements.

1. $\frac{}{s \overset{*}{\mapsto} s}$
2. $\frac{s \mapsto s' \quad s' \overset{*}{\mapsto} s"}{s \overset{*}{\mapsto} s"}$
3. $\frac{s \overset{*}{\mapsto} s' \quad s' \mapsto s"}{s \overset{*}{\mapsto} s"}$

**Question:** Are both the properties 2 and 3 above necessary to be stated? Would either one of them be enough?

## 1.4 Structural Dynamics

The structural dynamics for the language $L(num, plus)$ is given by the transition systems whose states are expressions, any expression can be an initial one, and the final states are *values*, which represent completed computations. The transitions are given by the judgements listed below.

### 1.4.1 Value Judgements

$$\frac{}{num[n] \quad val}$$

### 1.4.2 Transition Judgements

The first one gives the primitive application $PLUS_N$.

$$\frac{n_1 + n_2 = n_3 \quad nat}{plus(num[n_1]; num[n_2]) \mapsto num[n_3]} \quad PLUS_N$$

A different way of stating this judgement is as below.

$$\frac{}{plus(num[n_1]; num[n_2]) \mapsto num[n_1 + n_2]} \quad PLUS_N$$

The next two judgements are concerned with the order of evaluation. $L$ stands for 'left' and $R$ stands for 'right'.

$$\frac{e_1 \mapsto e_1'}{plus(e_1; e_2) \mapsto plus(e_1'; e_2)} \quad PLUS_L$$

$$\frac{e_1 \quad val \qquad e_2 \mapsto e_2'}{plus(e_1; e_2) \mapsto plus(e_1'; e_2')} \quad PLUS_R$$

**Question:** Why is it necessary to add the condition that $e_1$ is a value in the judgement rule $PLUS_R$ when such a condition was not stated for $PLUS_L$?

### 1.5 Semantics

The semantics of an expression $e_0$ is defined by a transition sequence

$$e_0 \mapsto e_1 \mapsto e_2 \mapsto \cdots .$$

Each step $e_i \mapsto e_{i+1}$ is derived using one of the transition judgements listed above.

**Example:**

| $\underline{Expression}$ | $\underline{Remarks}$ |
|---|---|
| $plus(plus(num[2]; num[3]); plus(num[4]; num[5]))$ | $initial$ |
| $\mapsto plus(num[5]; plus(num[4]; num[5]))$ | $PLUS_N \; \& PLUS_L$ |
| $\mapsto plus(num[5]; num[9])$ | $PLUS_N \& PLUS_R$ |
| $\mapsto num[14]$ | $PLUS_N$ |

## References

[1] Harper, R.: *Practical Foundations for Programming Languages*, Cambridge University Press, December 2012.