Разработка и реализация модульной системы проверки и вычисления типов

М. А. Буряков

СПбГПУ, каф. РВКС

Научный руководитель:
ст. преп. Д. А. Тимофеев

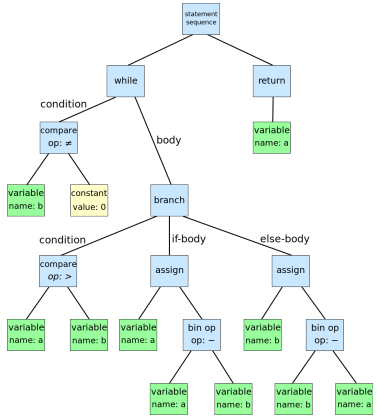
СПб, 2014

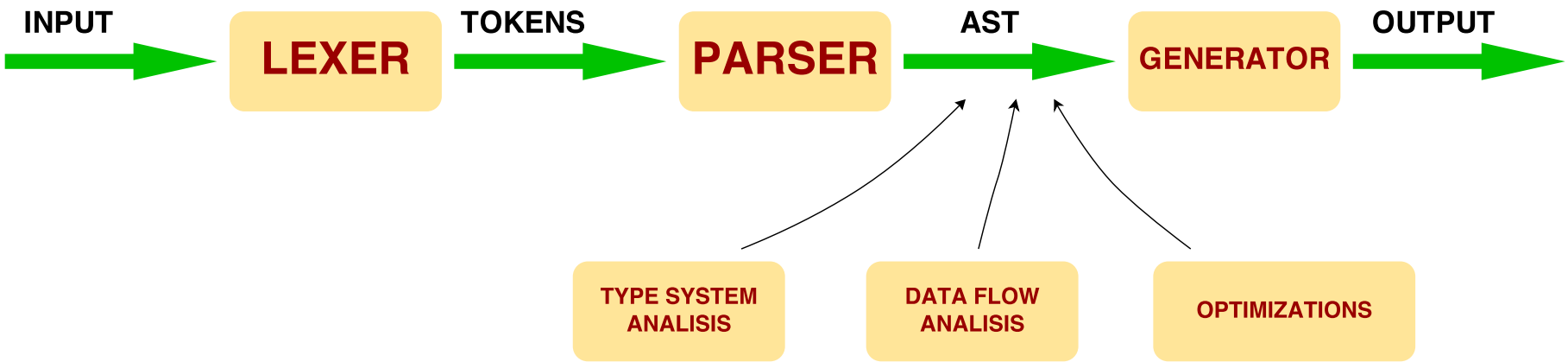
Порядок обработки программного кода

```
/**
 * Simple HelloButton() method.
 * @version 1.0
 * @author john doe <doe.j@example.com>
 */
HelloButton()
{
  JButton hello = new JButton( "Hello, wor
  hello.addActionListener( new HelloBtnList

  // use the JFrame type until support for t
  // new component is finished
  JFrame frame = new JFrame( "Hello Button"
  Container pane = frame.getContentPane();
  pane.add( hello );
  frame.pack();
  frame.show();           // display the fra
}
```

```
123
if
2.03
*=
;
(
++
"abc"
==
class
.
while
}
```





Система типов

Система типов:

- приписывает типы выражениям

`2 + 3`	`int`	типы выражений Java
`2.0 + 3.0`	`double`	
`2.0 + 3`	`double`	

- проверяет корректность типов

`int i = 1000` `byte b = 2*64 - 1`	корректные выражения Java
`int i = 1.0` `byte b = 200`	некорректные выражения Java

Система типов

Спецификация:

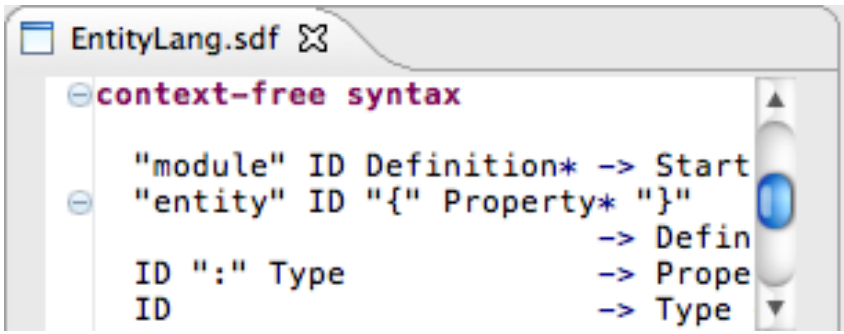
- Разрабатывается создателем языка
- Описывается словами в документации
- Использует придуманную специально для неё терминологию

Реализация:

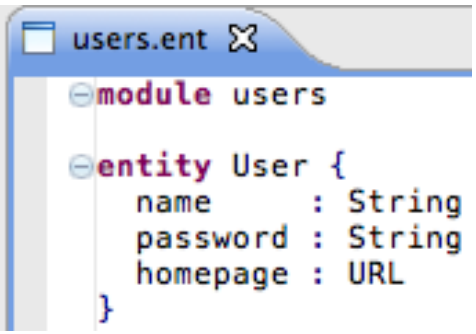
- Входит в состав компилятора или среды разработки (IDE)
- Реализуется независимо для каждого языка
- Реализуется в виде алгоритма на языке общего назначения

Платформы метапрограммирования

- Предначначены для разработки предметно-ориентированных языков (DSL)
- Содержат средства для описания языков (метаязыки)
- Генерируют трансляторы и IDE для разработанных языков



грамматика языка



IDE для языка

Популярные платформы метапрограммирования:

- XText
- MPS
- Spoofax

Системы типов в платформах метапрограммирования

Необходим единый интерфейс взаимодействия IDE с системой типов

Система типов описывается вместе с создаваемым языком

Система типов описывается на специальном предметно-ориентированном языке

Системы типов в платформах метапрограммирования

Необходим единый интерфейс взаимодействия IDE с системой типов

Система типов описывается вместе с создаваемым языком

Система типов описывается на специальном предметно-ориентированном языке

Известные среды метапрограммирования включают в себя поддержку систем типов:

- XTypes
- XText Typesystem Framework
- MPS typesystem aspect
- Spoofax typesystem integration

Недостатки поддержки систем типов в среде MPS

Язык описания систем типов имеет модель вычисления, несовместимую с алгоритмами анализа типов многих языков программирования

Несовместимые алгоритмы описать сложно или невозможно

Недостатки поддержки систем типов в среде MPS

Язык описания систем типов имеет модель вычисления, несовместимую с алгоритмами анализа типов многих языков программирования

Несовместимые алгоритмы описать сложно или невозможно

Эта модель вычисления выбрана в качестве компромиссной модели и не соответствует ни одному существующему языку

Поддержка систем типов в MPS предназначалась для языка Java, однако модель вычисления языка ближе к системе типов Хиндли-Милнера

Модель вычисления системы типов MPS

При обходе дерева создаются уравнения, неравенства и другие ограничения на типы вершин

```
typeof(integerLiteral) :==: <int>;
infer typeof(expression) :<=: <SModel>;
typeof(castExpression.expression) :~: castType ;
```

уравнение
неравенство
сравнимость

Специальный решатель пытается найти решение, удовлетворяющее всем ограничениям

Если решений несколько, выбирается одно из них

Постановка задачи

Разработать интерфейс взаимодействия IDE и системы типов

Постановка задачи

Разработать интерфейс взаимодействия IDE и системы типов

Разработать язык описания систем типов

- Модель вычисления этого языка должна быть в достаточной степени универсальна, чтобы выразить системы типов разных языков

Постановка задачи

Разработать интерфейс взаимодействия IDE и системы типов

Разработать язык описания систем типов

На разработанном языке описать основную часть системы типов языка Haskell.

- Система типов должна описываться модульно в виде набора правил, объявленных для конструкций языка

Постановка задачи

Разработать интерфейс взаимодействия IDE и системы типов

Разработать язык описания систем типов

На разработанном языке описать основную часть системы типов языка Haskell.

На разработанном языке описать элементы системы типов Java (отношения над типами и операция присваивания).

- Отношения над типами должны описываться расширяемо, чтобы при добавлении в язык новых типов отношения можно было расширить на них

Репозиторий типов

Все типы хранятся в репозитории

Типы привязываются к вершинам AST

Операции с репозиторием:

- прочитать тип вершины
- записать тип вершины

Множественные репозитории

Могут существовать несколько независимых репозиториев

- Если работают одновременно несколько разных систем типов, они будут использовать разные репозитории

Правила типизации

Алгоритм вывода и проверки типов реализуется в виде правил

Правило — фрагмент кода на языке, расширяющем Java

Правило может делать запросы на:

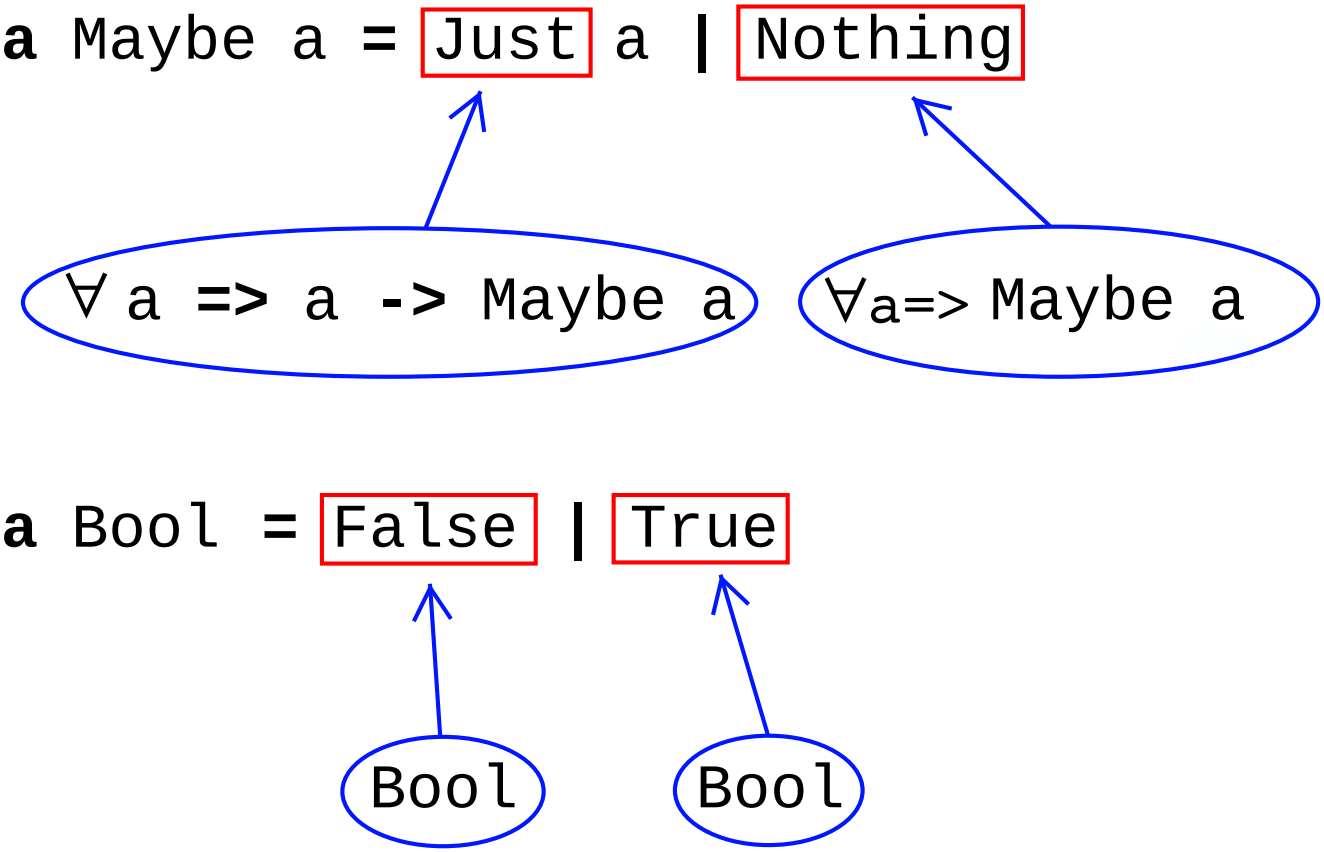
- чтение небольшого фрагмента AST
- чтения из репозитория типов
- запись в репозиторий типов

Правило пишется для определённой конструкции языка

- Каждый раз, когда эта конструкция используется в программе, создаётся экземпляр правила

Пример: конструкторы в языке Haskell

Конструкторы возникают в объявлении алгебраического типа:



Правило для объявления алгебраичекого типа обходит все конструкторы и присваивает им тип.

Правила типизации

Тип, вычисленный одним экземпляром, может использоваться другим экземпляром

- важен порядок применения экземпляров правил

Правила типизации

Тип, вычисленный одним экземпляром, может использоваться другим экземпляром

- важен порядок применения экземпляров правил

Правила снабжаются аннотациями

- вершины AST, читаемые экземпляром этого правила
- типы, читаемые экземпляром этого правила
- типы, записываемые экземпляром этого правила

Правила типизации

Тип, вычисленный одним экземпляром, может использоваться другим экземпляром

- важен порядок применения экземпляров правил

Правила снабжаются аннотациями

- вершины AST, читаемые экземпляром этого правила
- типы, читаемые экземпляром этого правила
- типы, записываемые экземпляром этого правила

Если нужно вычилить тип вершины:

- создаются экземпляры правил, которые могут повлиять на тип этой вершины
- производится топологическая сортировка по зависимостям
- экземпляры правил запускаются в правильном порядке

Операции над типами

- Операция объявляется отделено от реализации
- Реализации могут добавляться в существующую операцию

Операции над типами

- Операция объявляется отделено от реализации
- Реализации могут добавляться в существующую операцию

Пример: операция box в Java

Объявление операции

```
java.box :: primitiveType -> referenceType
```

Операции над типами

- Операция объявляется отделено от реализации
- Реализации могут добавляться в существующую операцию

Пример: операция box в Java

Объявление операции

```
java.box :: primitiveType -> referenceType
```

Объявление реализаций

```
java.box(int)  = Integer
java.box(byte) = Byte
.....................
java.box(char) = Character
```

Операции над типами

- Операция объявляется отделено от реализации
- Реализации могут добавляться в существующую операцию

Пример: операция box в Java

Объявление операции

```
java.box :: primitiveType -> referenceType
```

Объявление реализаций

```
java.box(int)  = Integer
java.box(byte) = Byte
.....................
java.box(char) = Character
```

Расширяющая реализация

```
java.box(dict) = Map<String, Object>
```

Операции над типами

С помощью операций над типами можно определить отношения:

```
java.isSubtype :: type -> type -> boolean
```

В работе реализованы операции:

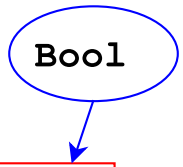
- box
- unbox

Реализованы отношения:

- identity conversion
- widening primitive conversion
- widening reference conversion
- narrowing primitive conversion
- narrowing reference conversion
- capture conversion
- assignment conversion
- unchecked conversion

Реализация унификации

- Унификация объявляется как бинарная операция над типами
- Унификация запускается при повторной записи типа для одной и той же вершины



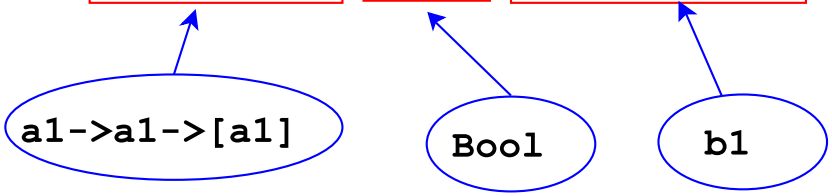

```
data Bool = True | False

listPair :: ∀ a => a -> a -> [a]
listPair x y = [x, y]

undefined :: ∀ b => b
undefined = undefined

main = listPair True undefined
```

```
a1 == b1 == Bool
```



Заключение

Разработан интерфейс взамодействия IDE и системы типов, основанный на репозитории типов

Разработан предметно-ориентированный язык описания правил типизации

- Написана виртуальная машина, запускающая правила в нужном порядке

На этом языке написана система типов языка Haskell

На этом языке описаны элементы системы типов Java

Заключение

В описании системы типов Haskell правила отражают уравнения, описанные в спецификации системы типов

В описании системы типов Java отношения и операции над типами записываются подобно их описанию в спецификации языка

Возможности для дальнейшего развития

Усилить интеграцию с MPS

- использовать разработанную систему для подсказок IDE
- использовать её при генерации кода в MPS

Улучшить синтаксис языка описания систем типов

- синтаксис языка в MPS описывается независимо от семантики, и может дорабатываться отдельно

Добавить поддержку других платформ метапрограммирования

Использовать разработанную систему при создании новых языков

Спасибо за внимание!