Замена парсера Scala в Scala-плагине при помощи генераторов синтаксических анализаторов

И. Шугаепов
Руководитель: А. Козлов

СПбАУ РАН

8 сентября 2016

Цели и задачи

Цель: замена рукописного парсера в Scala-plugin на автоматически сгенерированный.

Задачи:

- Интеграция сгенерированного парсера в Scala-plugin
- Сравнение производительности рукописного и сгенерированного парсеров

Интеграция парсера (Grammar Kit)

Основные проблемы Grammar Kit:

- работает с PEG грамматиками, а не с CFG (!)
- не умеет бороться ни с direct left recursion, ни с mutually left recursive правилами
- проблема с ε-правилами
- генерирует свой PSI

Вывод: Из-за описанных выше проблем, было принято решение использовать другой генератор синтаксических анализаторов.

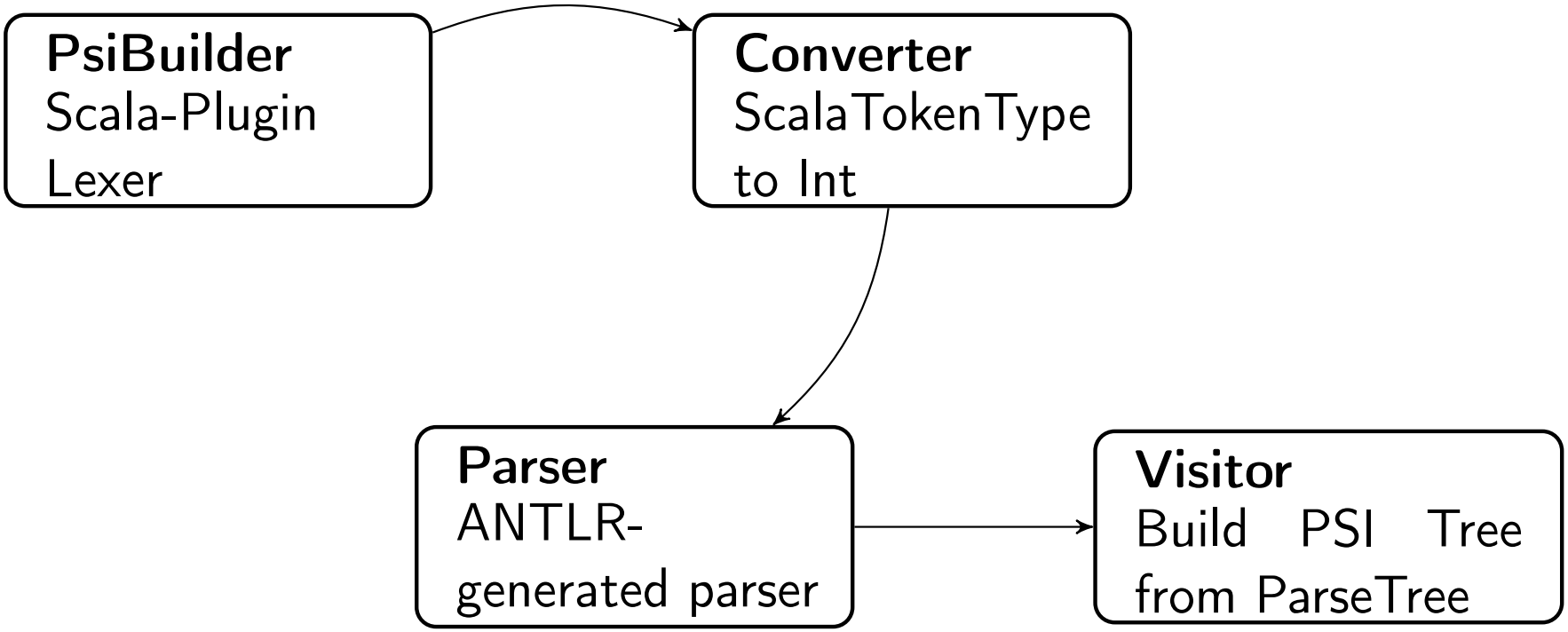
Интеграция парсера (ANTLR)

Основные проблемы:

- парсер в Scala-plugin распознает язык отличный от пораженного официальной грамматикой Scala
- ANTLR не умеет бороться с mutually left recursive правилами
- CRLF не является токеном для PsiBuilder, но CRLF присутствует в правилах грамматики (!)
- стандартная стратегия обработки ошибок отличается от стратегии в рукописном парсере (!)

Интеграция парсера (ANTLR)

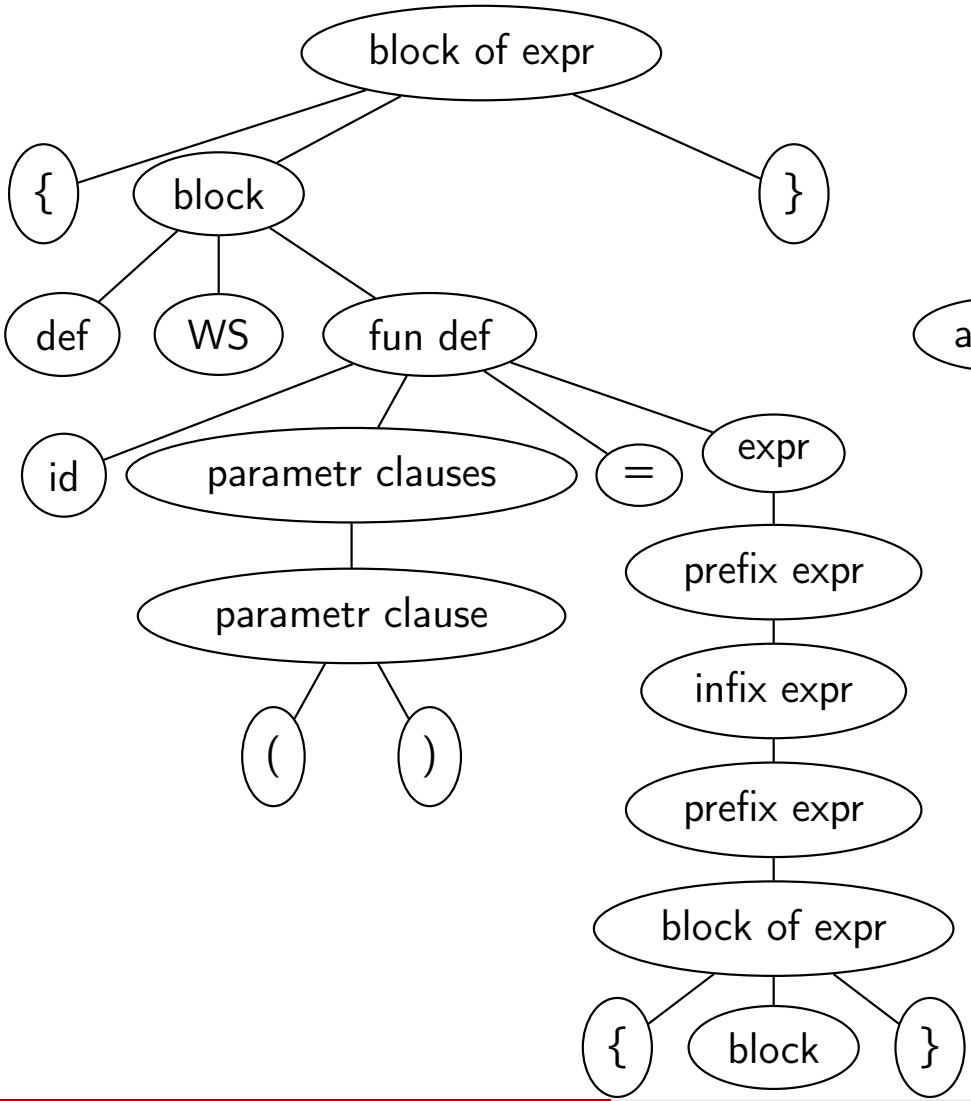
Упрощенная схема:

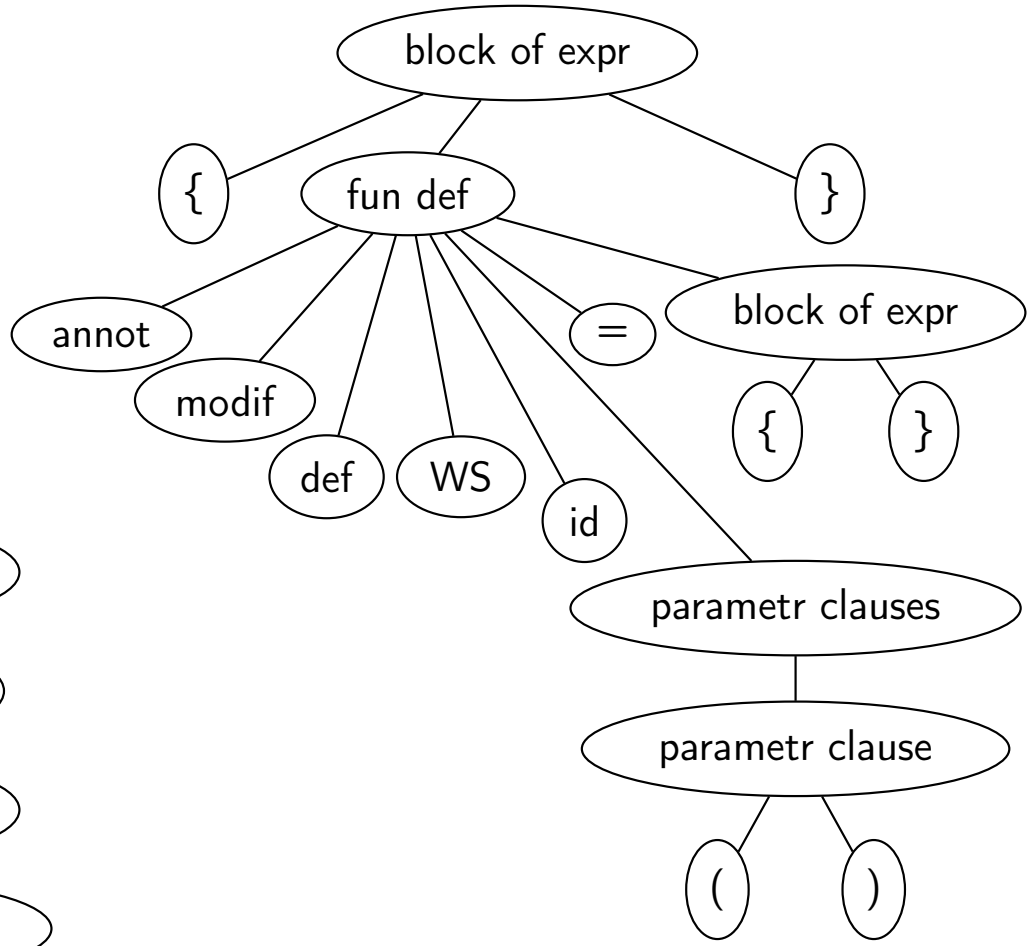


Visitor: Обход дерева разбора и расстановка маркеров с помощью PsiBuilder.

Интеграция парсера (ANTLR)

Разница между структурой дерева разбора и PSI. Пример: `def f()={}`





Интеграция парсера (ANTLR)

Неудачная попытка заменить парсинг части конструкций языка. Например, парсинг типов.

```
type : typeType
     | infixType
     | existentialType
     | wildcardType ;
```

Причина: стандартная стратегия обработки ошибок.

Решение проблемы CRLF

Вариант №1. Удалить CRLF из грамматики и добавить semantic predicates and actions.

Изменение правил:

До
```
doStmt: 'do' expr (';' | Nl)? 'while' expr ;
```
После
```
doStmt: 'do' expr (';' | { isNl() } ?)? 'while' expr ;
```

Решение проблемы CRLF

Вариант №2. Добавить CRLF в грамматику везде, где это необходимо, и изменить Converter.

Изменение правил:

До

```
doStmt: 'do' expr (';' | Nl)? 'while' expr ;
```

После

```
doStmt: 'do' Nl* expr semi? 'while' Nl* expr ;
semi   : ';'
       | Nl+ ;
```

Результаты

- Сгенерированный парсер проходит все тесты с корректным вводом
- Performance тесты показали, что сгенерированный парсер в 2 раз медленее рукописного
- Сгенерированный парсер не проходит тесты с некорректным вводом из-за проблем, связанных со стандартной стратегией обработки ошибок

Исходный код

Репозиторий : `https://github.com/ilnurshug/intellij-scala`