Major Technical Project
2020-2021

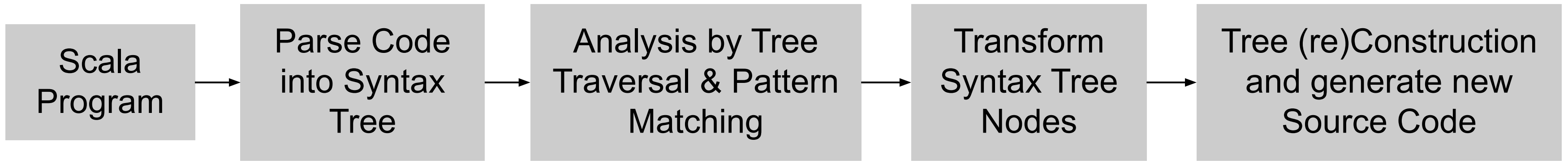
Identifying Refactoring Opportunities that promote Functional Design Patterns in Scala

Namrata Malkani
B17096

Advisor:
Dr. Manas Thakur

Problem Introduction

- Code Refactoring - Restructuring existing computer code—changing the factoring—without changing its external behavior.
- Analyse a Scala program and suggest code improvements (refactorings) on code snippets which can be/need to be implemented with a functional design pattern.
- How (analysis)? Ans. Program syntax tree nodes analysis and manipulation to achieve desired result.


Scala Program
Parse Code into Syntax Tree
Analysis by Tree Traversal & Pattern Matching
Transform Syntax Tree Nodes
Tree (re)Construction and generate new Source Code

Problem Motivation

- Code refactoring comes under the umbrella of problems like code quality, programmability, paradigm and structural shift to help developers write better programs problems the industry finds quite useful.
- This particular problem is motivated by a leading healthcare-analytics company based at Chennai, and is popular in the software engineering domain.
- Certain programming problems are better solved by conforming to the functional design pattern.
- As an example - higher order functions (filter, map, etc.) are mutable, pure, easily parallelized, compared to ‘loops’ that may perform the same task.

Language and Tools

- Scala: Well known Functional language, combines OO and Functional Paradigms well.
  - Lightweight syntax to define anonymous tasks.
  - Supports first-order operations, allows nested functions, and supports curry.

- Scalameta library
  - The foundation library for meta programming in Scala with a powerful parser for Scala code.
  - Industry-wide employed to explore and manipulate Scala code structurally and excellent choice for a static analysis tool.

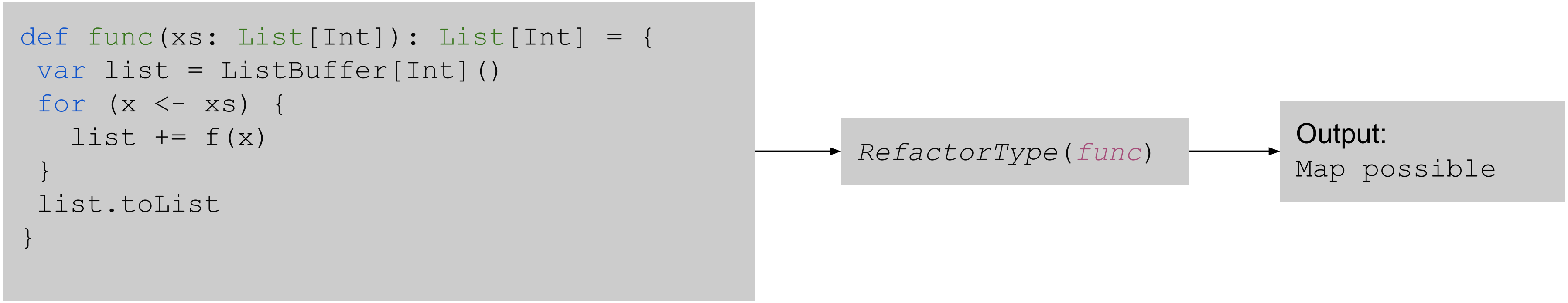
- IntelliJ IDEA
  - IDE for development and the ideal platform for plugin deployment.

- Sbt-idea-plugin: A Github repository maintained by JetBrains.
  - Develop IntelliJ plugins with Scala and SBT.

Initial Research and Groundwork

- **Courses:** CS-302 Paradigms of Programming, CS-502 Compiler Design
- **Papers:** 'Crossing the Gap from Imperative to Functional Programming through Refactoring',
  'Identifying Refactoring Opportunities for Replacing Type Code with Subclass and State'
- **Documentations:** Scala Documentation: Documentation
  IntelliJ Platform SDK—IntelliJ Platform Plugin SDK
  Scalameta · Library to read, analyze, transform and generate Scala programs
- **Tutorials:** Scala Programming for beginners (various tutorials)
  'Code Real World App Using Purely Functional Techniques (in Scala)' by Coding Tech
  'Busy plugin developers series' by JetBrains TV
- **Online Talks:** Scala Meta Live Coding Session by Pathikrit Bhowmick, Scala Days Conferences
  Inside the IntelliJ Scala Plugin by ScalaSphere
  Building IntelliJ IDEA plugins in Scala by Igal Tabachnik: Scala in the City Conference

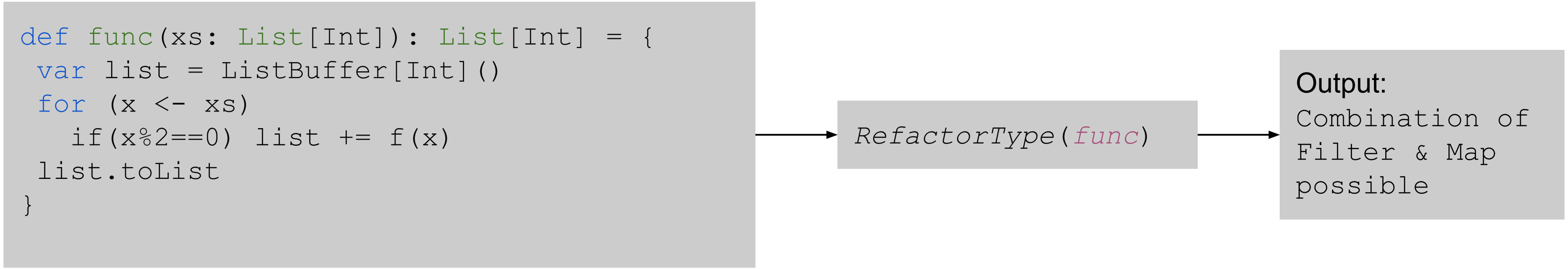
Work Done: Identify the type of Refactoring



```scala
def func(xs: List[Int]): List[Int] = {
 var list = ListBuffer[Int]()
 for (x <- xs) {
   list += f(x)
 }
 list.toList
}
```

RefactorType(*func*)

Output:
Map possible



```scala
def func(xs: List[Int]): List[Int] = {
 var list = ListBuffer[Int]()
 for (x <- xs)
   if(x%2==0) list += f(x)
 list.toList
}
```

RefactorType(*func*)

Output:
Combination of Filter & Map possible

Work Done: Transforming the Program

```scala
def func(xs: List[Int]): Int = {
 var sum = 0
 for (x <- xs) {
   sum += x
 }
 sum
}
```

$\rightarrow$ *Refactor(func)* $\rightarrow$

```scala
def func(xs: List[Int]): Int = {
 val sum = xs.reduce((x,y) => x+y)
 sum
}
```

```scala
def func(xs: List[Int]): List[Int]
= {
 var list = ListBuffer[Int]()
 for (x <- xs) {
   if(x%2==0) list += x
 }
 list.toList
}
```

$\rightarrow$ *Refactor(func)* $\rightarrow$

```scala
def func(xs: List[Int]): List[Int] =
{
 var list = List(0)
 List = xs.filter(_%2 == 0)
 list
}
```

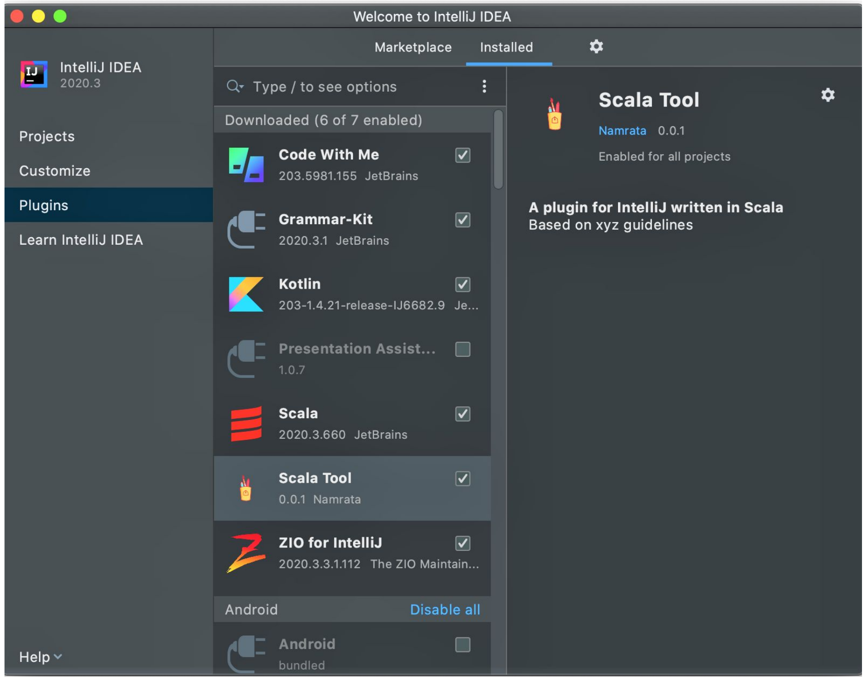
Next Steps

Current Focus

- ❖ IntelliJ IDEA Plugin Development in Scala: Code-Plugin Integration till Feb'21.
- ❖ SBT-based structure configured-

Next Semester

- ❖ Add more refactorings and integrate in the plugin after consulting Scala Developers in the Industry.
- ❖ Evaluation on real-world code.



Downloaded (6 of 7 enabled)

Code With Me ✓
203.5981.155 JetBrains

Grammar-Kit ✓
2020.3.1 JetBrains

Kotlin ✓
203-1.4.21-release-IJ6682.9 Je...

Presentation Assist...
1.0.7

Scala ✓
2020.3.660 JetBrains

Scala Tool ✓
0.0.1 Namrata

ZIO for IntelliJ ✓
2020.3.3.1.112 The ZIO Maintain...

Android

Android
bundled

Scala Tool
Namrata 0.0.1
Enabled for all projects

A plugin for IntelliJ written in Scala
Based on xyz guidelines