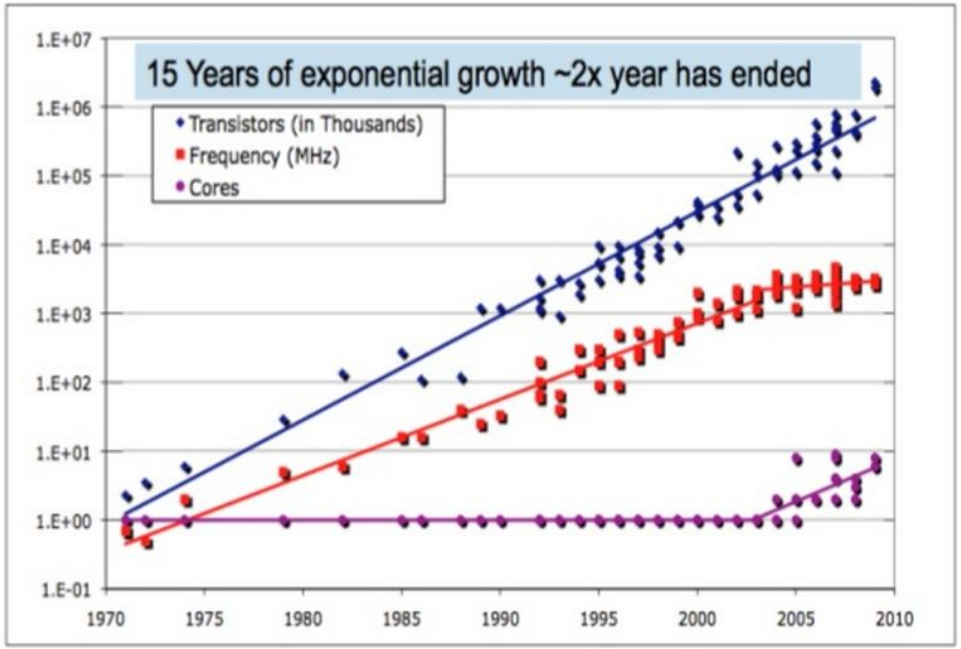
Scala

Scalable Language

Moore’s Law (Scalability)

- Moore’s law now achieved by increasing # of cores, not clock cycles
- We need Concurrency and Parallelism



Data from Kunle Olukoten

Concurrency and Parallelism

- Parallel Programming
  - Execute programs faster on parallel hardware
- Concurrent Programming
  - Manage concurrent execution threads explicitly.
- Both are too Hard
- The speedup of a program using multiple processors in parallel computing is limited by the time needed for the sequential fraction of the program. [Amdahl's Law]

The Root of The Problem

- Non-determinism caused by concurrent threads accessing shared mutable state.

```
var x = 0
async { x = x + 1 }
async { x = x * 2 }
print(x)
// can give 0, 1, 2
```

non-determinism = parallel processing + mutable state

- heisenberg problem (bug goes away when we introduce instrumentation)

The Root of The Problem

- Non-determinism caused by concurrent threads accessing shared mutable state.

```
var x = 0
async { x = x + 1 }
async { x = x * 2 }
print(x)
// can give 0, 1, 2
```

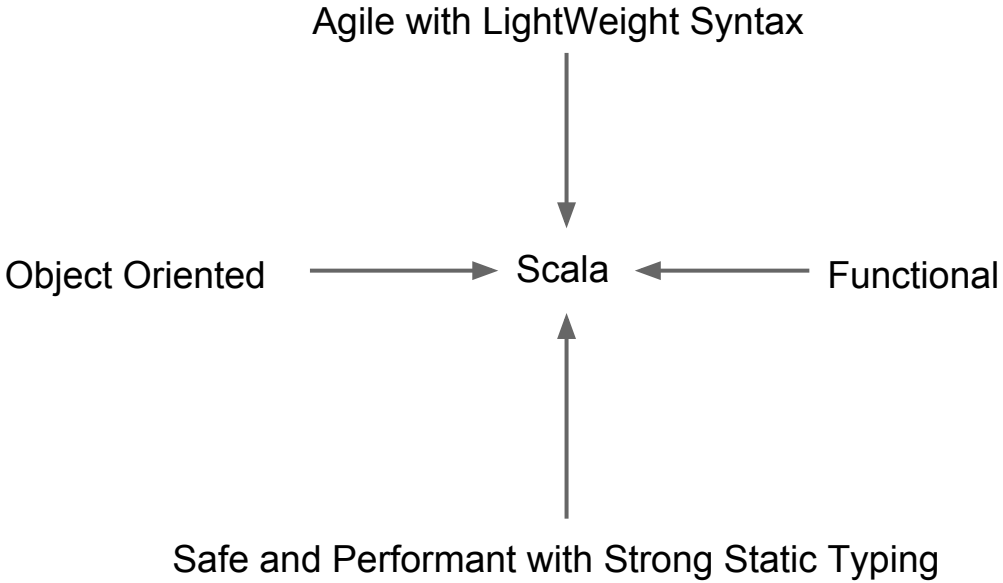
non-determinism = parallel processing + mutable state

- heisenberg problem (bug goes away when we introduce instrumentation)
- To get deterministic processing, avoid the mutable state.
- Avoid mutable state means program functionally.

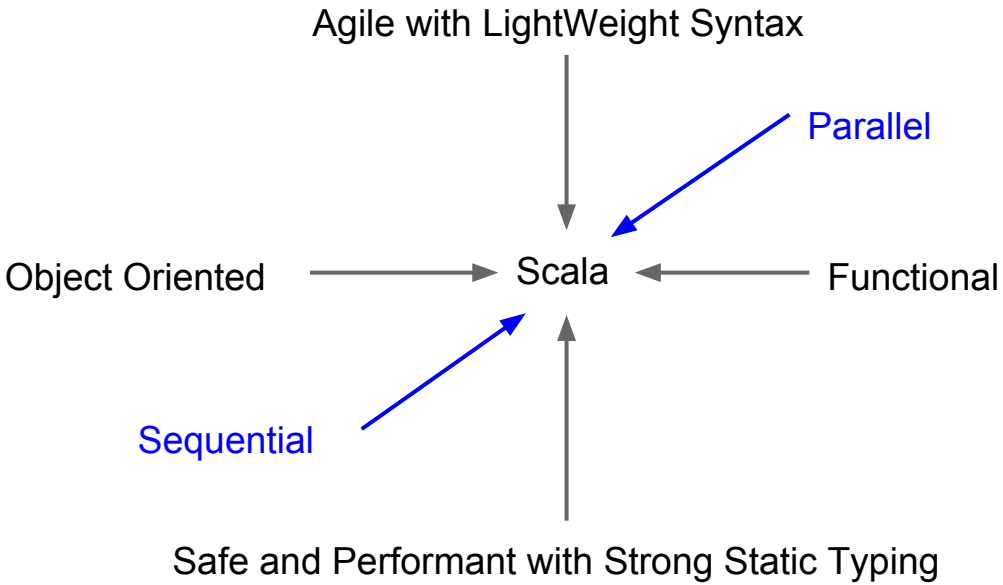
Why Scala

- Scala is by far the only prominent strongly typed language that provide functional style and great concurrency support. [Akka Essentials: Munish K, Gupta]

How to deal with it in Scala?



How to deal with it in Scala?



Scala Toolbox

- Parallelism
  - Collections
    - Parallel Collections
    - Distributed Collections
- Concurrency
  - Akka
    - Message passing framework
    - Actors
    - Futures

Lets see an Example

- Java
- C++
- Scala

What's Scala

- Pure Object Oriented Language
- Supports both Imperative and functional style.
- Provides Parallelism using collections
- Provides Concurrency using actors
- Statically Typed Language With Type Inference
- Runs on JVM. Intermixes well with Java
- Scala Web Frameworks
  - Play
  - Spray
- Highly scalable i.e. takes less code to create high performing applications

Course Contents

1. The Basics (A1)
2. Control Structures and Functions (A1)
3. Arrays (A1)
4. Maps and Tuples (A1)
5. Classes (A1)
6. Objects (A1)
7. Packages and Imports (A1)
8. Inheritance (A1)
9. Files and Regular Expressions (A1)

The Basics

- The Scala Interpreter
- Declaring Values and Variables
- Commonly Used Types
- Arithmetic and Operator overloading
- Calling Functions and Methods
- The apply Method
- Scaladoc

Scala Environment Setup

- Java Setup (JDK)
- sbt Setup (An open source build tool for Scala and Java. Just like Maven and Ant)
- Intellij (Can also use Eclipse)
- (https://class.coursera.org/progfun-003/wiki/ToolsSetup)

Scala Interpreter

- Read-Evaluate-Print-Loop (REPL)
- Every value is an object
- Tab Completion
- Scala Worksheet

Declaring Values and Variables

- val vs var
- Type Inference
- Multiple values declaration

Commonly Used Types

- Byte, Char, Short, Int, Long, Float, Double, Boolean
- All types are classes
- No distinction between primitive types and class types.
  - 1.toString()
  - 1.to(10)

Arithmetic and Operator Overloading

- Arithmetic operators in Scala work just as in Java or C++
- However, these operators are actually methods.
  - a + b is a shorthand for a.+(b)

Calling Functions and Methods

- Scala provides functions in addition to methods
- scala.math library
  - sqrt(2)
  - pow(2, 4)
  - min(2, Pi)

The apply Method

- In Scala, it is common to use a syntax that looks like a function call.
  - s(i) where s is a string. It will return ith character of string. In C++, it is s[i]. in java it is s.charAt(i)
  - s(i) is a shorthand for s.apply(i)

Testing in Scala

- Testing Frameworks
  - ScalaTest
  - Specs2
- Browser-UI test support
  - ScalaTest includes a DSL for writing browser-based tests using Selenium.

Scaladoc

http://www.scala-lang.org/api/current/#package

Scala is scalable

- What does it mean?
- It means, scala is great for
  - scripting stuff
  - writing applications
  - writing monster "enterprise" (for the want of a word) applications
- Lets see the code in action

Level of Expertise in Scala

Application Programmer	Library Designer	Overall Scala Level
Beginning (A1)		Beginning
Intermediate (A2)	Junior (L1)	Intermediate
Expert (A3)	Senior (L2)	Advanced
	Expert (L3)	Expert

Control Structure and Functions(A1)

- Conditional Expressions
- Statement Termination
- Block Expressions and Assignments
- Input and Output
- Loops
- Advanced for Loops and for Comprehensions
- Functions
- Default and Named Arguments
- Variable Arguments
- Procedures
- Lazy Values
- Exceptions