

Qinyun Lin – SEC01 (NUID 001582464)
Big Data System Engineering with Scala
Spring 2022
Assignment No. 6



Task

1. Implement 3 TODOs in WebCrawler.scala
2. Implement 2 TODOs in MonadOps.scala
3. Suggest how to improve the web crawler

Solution

1. Implement 3 TODOs in WebCrawler.scala

```
// 16 points.
def getURLs(ns: Node): Seq[Try[URL]] = for(anchor <- ns \\ "a") yield createURL(Option(u), anchor \\@ "href")

// 9 points.
for(st <- getURLContent(u); r <- MonadOps.asFuture(getLinks(st))) yield r

// 15 points. Implement the rest of this, based on us2 instead of us.
Future.sequence( for (u <- us2) yield MonadOps.sequence(wget(u)) )
// Future.sequence( us2.map(wget).map(MonadOps.sequence(_)) )
```

2. Implement 2 TODOs in MonadOps.scala

```
// 6 points.
def mapFuture[X](xfs: Seq[Future[X]])(implicit executor: ExecutionContext): Seq[Future[Either[Throwable, X]]] =
  for(xf <- xfs) yield sequence(xf)

// 7 points.
def sequence[X](xe: Either[Throwable, X]): Option[X] = xe match {
  case Left(_) => None
  case Right(x) => Some(x)
}
```

3. Suggest how to improve the web crawler

(1) In this assignment, we process the URLs one by one. Actually we can transform it to parallel processing.

In the wget(us), we perform wget(u) and MonadOps.sequence() to each url. I think we can make it parallel here: split the URLs to N parts(N is number of threads), for each thread, only process one of the parts.

(2) The result of wget(us) is Future[Seq[...]], which means if any URLs of the sequence don't response correctly, we can't get other results, too. Maybe we don't have to use Future.sequence to transform the former result of it, just keep it Seq[Future[...]].

Unit Test Screenshot

The screenshot shows the IntelliJ IDEA IDE with the following details:

- Project Structure:** The project is named "assignment-web-crawler". The source code is located in "src/main/scala/edu.neu.coe.csye7200.asstwc". The test code is in "src/test/scala/edu.neu.coe.csye7200.asstwc/WebCrawlerSpec.scala".
- Code Editor:** The file "WebCrawlerSpec.scala" is open. It contains Scala code for testing the "WebCrawler" class. The code includes imports for "org.scalatest.concurrent.ScalaFutures", "org.scalatest.matchers.should.Matchers", and "org.scalatest.wordspec.AnyWordSpec". The test suite "WebCrawlerSpec" extends "AnyWordSpec" and includes several test cases for the "WebCrawler" class.
- Run Tab:** The "Run" tab shows the results of the test suite "WebCrawlerSpec". It indicates that 7 tests passed in 4 seconds (759 ms). The tests include "getURLContent", "wget(URL)", "filterAndFlatten", and "crawler(Seq(URL))".
- Calendar:** A calendar widget on the right side of the IDE shows the date "2022年4月3日 三月初三" (March 3, 2022).

The screenshot shows the IntelliJ IDEA IDE with the following details:

- Project Structure:** The project is named "assignment-web-crawler". The source code is located in "src/main/scala/edu.neu.coe.csye7200.asstwc". The test code is in "src/test/scala/edu.neu.coe.csye7200.asstwc/MonadOpsSpec.scala".
- Code Editor:** The file "MonadOpsSpec.scala" is open. It contains Scala code for testing the "MonadOps" class. The code includes imports for "org.scalatest.concurrent.ScalaFutures", "org.scalatest.matchers.should.Matchers", and "org.scalatest.wordspec.AnyWordSpec". The test suite "MonadOpsSpec" extends "AnyWordSpec" and includes several test cases for the "MonadOps" class.
- Run Tab:** The "Run" tab shows the results of the test suite "MonadOpsSpec". It indicates that 21 tests passed in 126 milliseconds. The tests include "LiftFuture", "AsFuture", "SequenceForgivingWithLogging", "SequenceForgiving", "LiftTry", "zipOption", "zipTry", "zipFuture", "OptionToTry", "Sequence", "LiftOption", "Map2", and "Flatten".
- Calendar:** A calendar widget on the right side of the IDE shows the date "2022年4月3日 三月初三" (March 3, 2022).

Project Source

<https://github.com/MrNiro/CSYE7200/tree/Spring2022/assignment-web-crawler>