Start by downloading spark locally

http://spark.apache.org/downloads.html

Spark is built on Scala 2.10. To build Spark and its example programs, enter directory and run:

./sbt/sbt assembly

This takes a long time.

To run the logistic regression example

./bin/run-example mllib.JavaLR data/mllib/lr\_data.txt 1 100

Standalone application for AMIDST

First we need to install maven and make sure that it builds. If you already have done this, you can skip the next step.

Install Maven: http://maven.apache.org/download.cgi

Put in ~/.profile:

source $HOME/workspace/core/doc/readMe/maven\_startup.sh

It is also important to put the hugin code in a certain place. I recommend to put it on core, (BUT DO NOT ADD to git repository as I am not sure that Hugin wants the jar file to be on the open source code.

Copy hgapi82\_amidst-64.jar to workspace/core, so Maven can find it (ref. pom.xml). Remember to add to project structure in IntelliJ.

Add hgapi82\_amidst-64.jar to your local .gitignore so that the file is not shown as untracted.

Create package, ignoring tests that fail:

mvn -Dmaven.test.skip=true package

I recommend that you all do not submit failing tests to core. Keep this on local projects till test are not failing. In essence, code should “always” build on master.

The final command to run JavaLR2 in the amidst project is:

spark-submit --class "eu.amidst.core.sparkExamples.JavaLR2" --master local[4] target/core-1.0-SNAPSHOT.jar

MAVEN

mvn clean compile assembly:single

java -cp "./lib/\*:./target/\*" -Djava.library.path="./lib/" eu.amidst.examples.ParallelTANDemo

mvn install:install-file -Dfile=path-to-non-maven.jar \

-DgroupId=some-id \

-DartifactId=some-artifact \

-Dversion=some-version \

-Dpackaging=jar

SET UP SERVER

Intall maven

wget <http://mirror.olnevhost.net/pub/apache/maven/binaries/apache-maven-3.2.2-bin.zip>

and unzip

Install java 8 on Ubuntu

sudo add-apt-repository ppa:webupd8team/java

sudo apt-get update

sudo apt-get install oracle-java8-installer

CHECK GITHUB HISTORY TO RECOVER PREVIOUS VERSIONS:

Add the following to the .gitconfig file in your $HOME directory:

[alias]

hist = log --pretty=format:\"%h %ad | %s%d [%an]\" --graph --date=short

> git hist

> git checkout e6fab8e

> git checkout @{14.days.ago}

> git checkout master %to get back to original/last version without committing anything.

If log needs to be updated:

> git fetch %and then checkout to the desired version

GO BACK TO PREVIOUS COMMIT:

# Resets index to former commit; replace '56e05fced' with your commit code

git reset 56e05fced

# Moves pointer back to previous HEAD

git reset --soft HEAD@{1}

git commit -m "Revert to 56e05fced"

# Updates working copy to reflect the new commit

git reset --hard