Dependency Extraction of Hadoop

REVERSE 'EM ALL

Overview

- Include VS Understand
  - Extraction Process
  - Comparison Results
  - Qualitative Analysis
- IDEA VS Understand
  - Extraction Process
  - Comparison Results
  - Qualitative Analysis
- Alternative Dependency Tool : srcML
- Lessons learned
- Conclusion

Tools Used

INCLUDE.JAVA VS UNDERSTAND

Understand

- Understand is a static analysis tool focused on source code comprehension, metrics, and standards testing.
- Used for dependency extraction
- A file depends on another through:
  - Imports
  - Inheritance
  - Implementations
  - Throws
  - Method calls and object initializations
  - @ Java annotations
- Extracted the dependencies by exporting a CSV file
  - File A, File B

Include.java

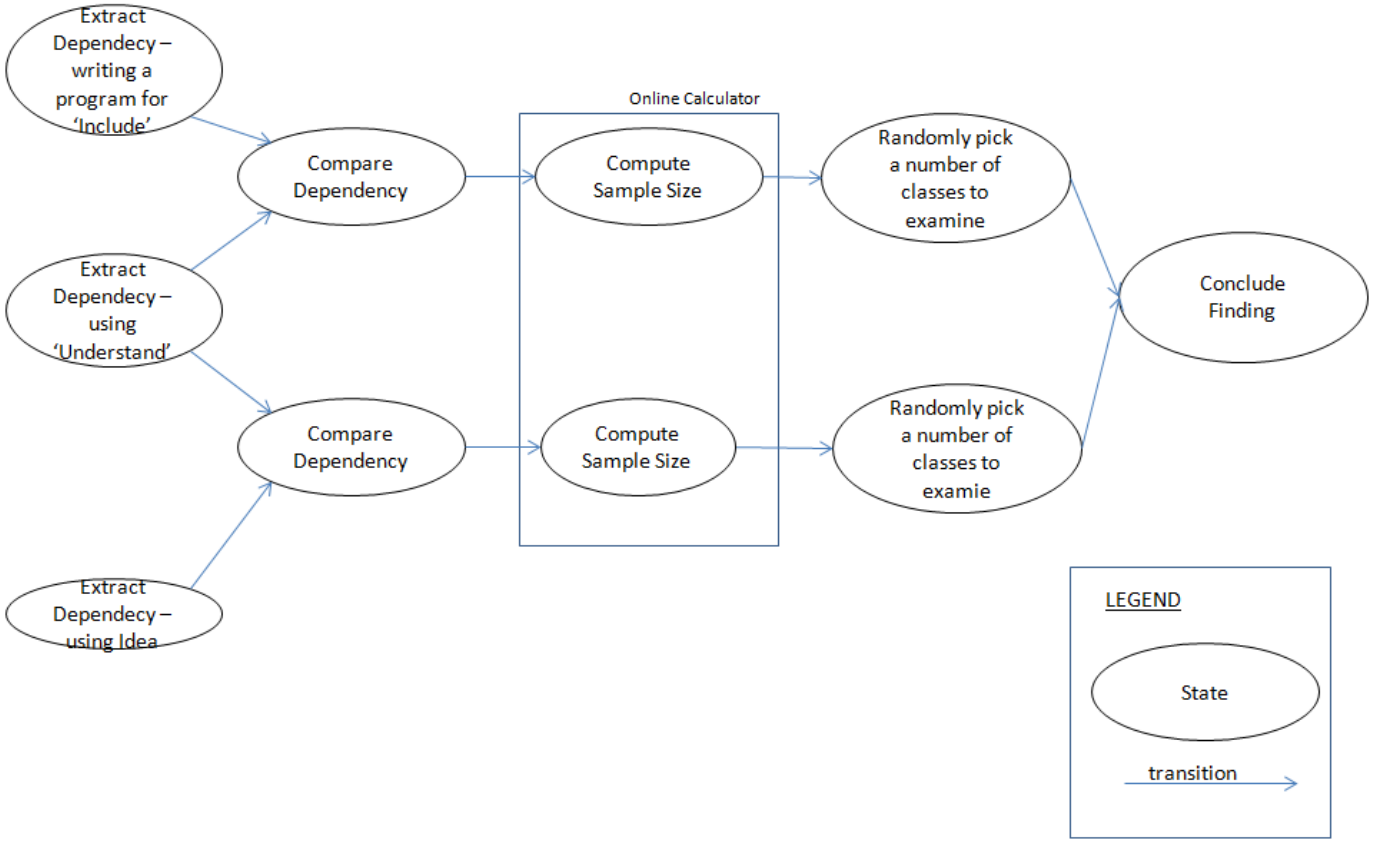
- Coded in Java
- Recursively goes through each folder in the Hadoop Project
- Searches for import statements ONLY
- Checks list of files to see if the pathway matches
- If so, it creates a dependency line

```
import org.apache.hadoop.hdfs.protocol.Block;
import java.io.*
```

Include VS Understand

QUANTITATIVE ANALYSIS

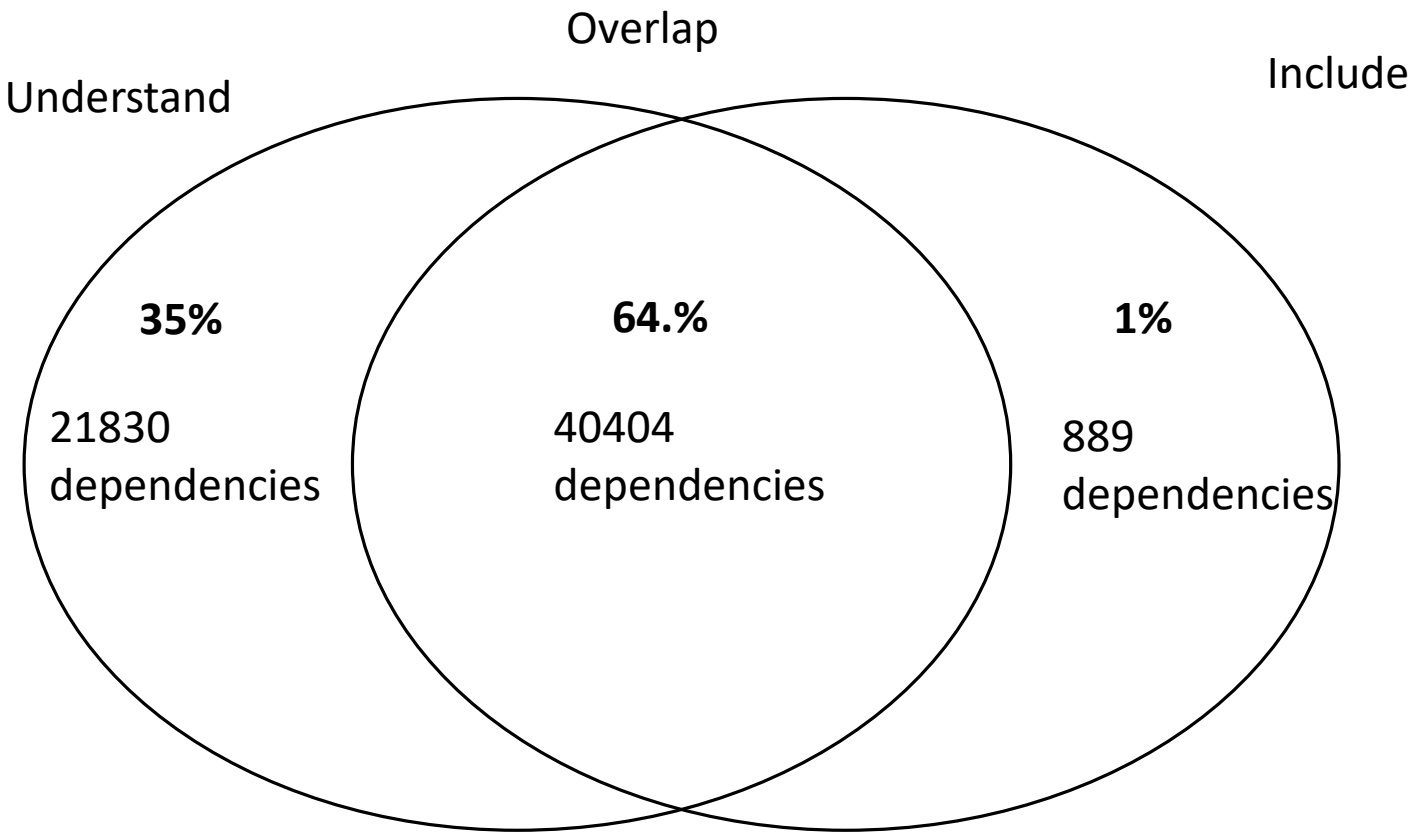
Extraction / Comparison process



Comparison Process

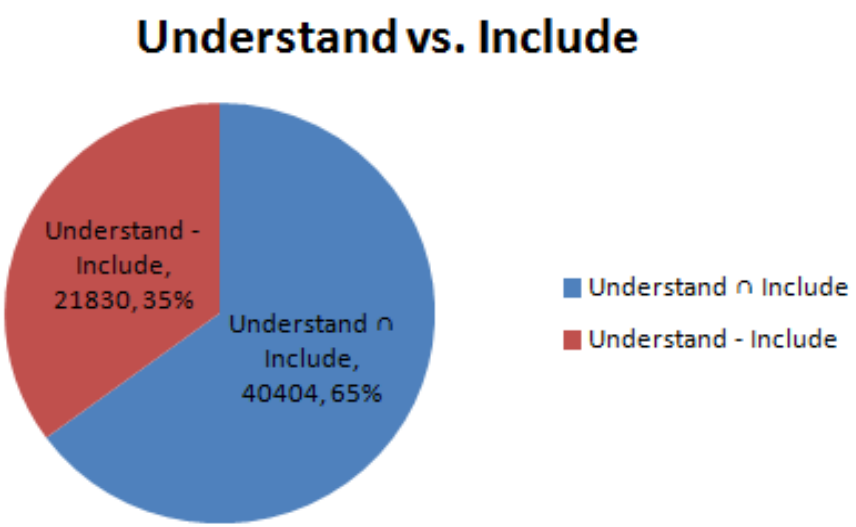
- Used a Java program Compare.java
- Made sure both files are in the same format ie Understand and IDEA/Include.java
- File A → File B
- Simply compared the two files line by line to see which is the same
- Stored the result in a text file

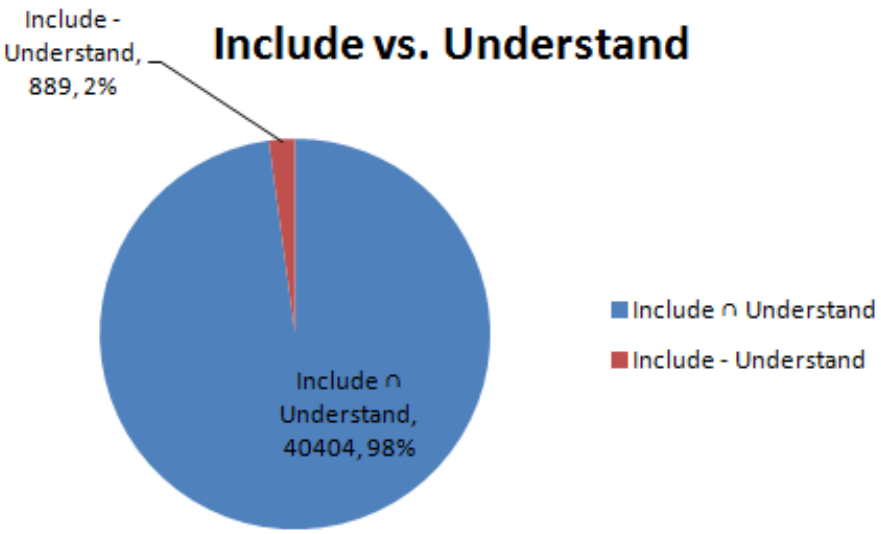
Venn Diagram of Understand VS Include



Statistics: Understand vs. Include

	Intersect	Excluding	Total
Undertand	40404	21830	62234
Include		889	41293
Total Dependencies	63123		



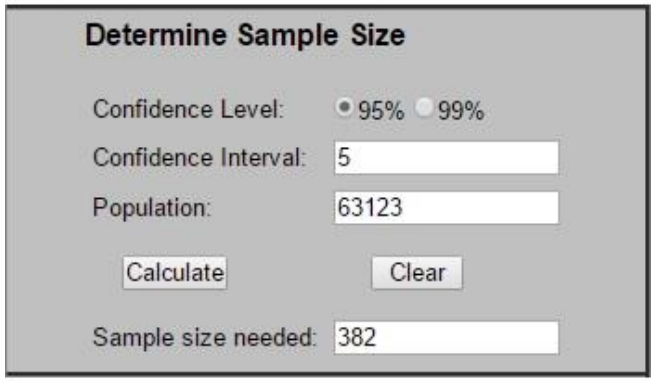


Sample Calculator for Include VS Understand

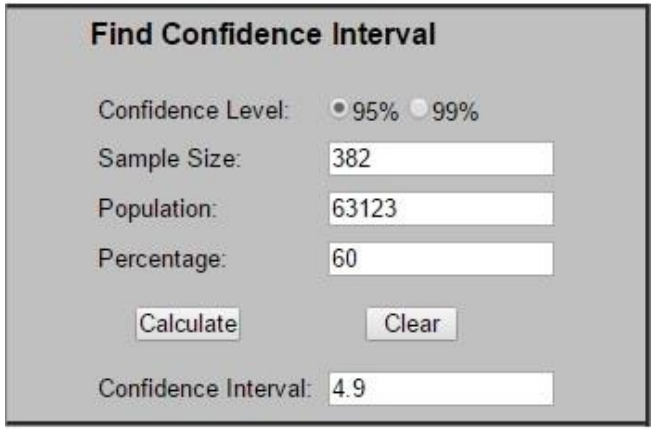
Confidence Level: 95%

Confidence Interval: 5

Population: 63123



Sample Size: 382



Comparison Result Summary – Understand versus Include

Most of the dependency Extracted by Include are captured by Understand

→ Understand extract dependency by looking at method-to-method class of every class, at which most of the time need import of the classes of the method used.

34.5% of the total dependencies is captured only by Understand but not by Include

→ Understand extract dependency by looking at the calling of methods to methods, and many of the 'supplier' classes are not imported but explicitly written it's full path namespace

è Undertand extract dependency by looking at the interfaces extended or classes inherited by every class

Example case(1):

From: hadoop-2.7.3-src\hadoop-mapreduce-project\hadoop-mapreduce-client\hadoop-mapreduce-client-core\...\mapred\lib\aggregate\ValueAggregator.java

To: hadoop-2.7.3-src\hadoop-mapreduce-project\hadoop-mapreduce-client\hadoop-mapreduce-client-core\...\mapreduce\lib\aggregate\ValueAggregator.java

Example case:
From: hadoop-2.7.3-src\hadoop-mapreduce-project\hadoop-mapreduce-client\hadoop-mapreduce-client-core\...\mapred\lib\aggregate\ValueAggregator.java
To: hadoop-2.7.3-src\hadoop-mapreduce-project\hadoop-mapreduce-client\hadoop-mapreduce-client-core\...\mapreduce\lib\aggregate\ValueAggregator.java

"mapred. ... .ValueAggregator" extends "mapreduce. ... .ValueAggregator"

CompareUnique.java | ValueAggregator.java

```
2⊕ * Licensed to the Apache Software Foundation (ASF) under one
18
19 package org.apache.hadoop.mapred.lib.aggregate;
20
21⊖ import org.apache.hadoop.classification.InterfaceAudience;
22 import org.apache.hadoop.classification.InterfaceStability;
23
24⊖ /**
25  * This interface defines the minimal protocol for value aggregators.
26  */
27 @InterfaceAudience.Public
28 @InterfaceStability.Stable
29 public interface ValueAggregator<E> extends
30     org.apache.hadoop.mapreduce.lib.aggregate.ValueAggregator<E> {
31 }
32
```

Comparison Result Summary – Understand versus Include

1% of total dependencies is capture only by Include and not by Understand

➔ Those capture by Include but not by Understand are those import used not for method invocation, but as variable/attribute type declaration

<u>Case Example(1):</u>

From: hadoop-2.7.3-src\hadoop-yarn-project\...\server\nodemanager \containermanager\localizer\event\ApplicationLocalizationEvent.java

To: hadoop-2.7.3-src\hadoop-yarn-project\...\server\nodemanager \containermanager\application\Application.java

<u>Case Example(2):</u>

From: hadoop-2.7.3-src\hadoop-yarn-project\...\yarn\server\resourcemanager\reservation\TestCapacityOverTimePolicy.java

To: hadoop-2.7.3-src\hadoop-yarn-project\...\yarn\server\resourcemanager\reservation\exceptions\ResourceOverCommitException.java

Comparison Result Summary – Understand versus Include

Case Example (1) - The application object is used as an attribute, and method of Application is not invoked

EECS 4314 Workplace - Java - Hadoop/src/main/resources/archetype-resources/src/hadoop-2.7.3-src/hadoop-yarn-project/hadoop-yarn/hadoop-yarn-server/hadoop-yarn-server-nodemanager/src/main/java/org/apache/hadoop/yarn/server/nodemanager/containermanager/loca

File Edit Source Refactor Navigate Search Project Run Window Help

CompareUnique.java | ApplicationLocalizationEvent.java

```
2  * Licensed to the Apache Software Foundation (ASF) under one
18
19 package org.apache.hadoop.yarn.server.nodemanager.containermanager.localizer.event;
20
21 import org.apache.hadoop.yarn.server.nodemanager.containermanager.application.Application;
22
23 public class ApplicationLocalizationEvent extends LocalizationEvent {
24
25   final Application app;
26
27   public ApplicationLocalizationEvent(LocalizationEventType type, Application app) {
28     super(type);
29     this.app = app;
30   }
31
32   public Application getApplication() {
33     return app;
34   }
35
36 }
37
```

Package Explorer | ModifiedFile.java | CompareTextFiles.java | Console | Outline

- org.apache.hadoop.yarn.server.nodemanager.containermanager.localizer.event
- ApplicationLocalizationEvent
  - app : Application
  - ApplicationLocalizationEvent(LocalizationEventType, Application)
  - getApplication() : Application

Case Example (2)

From: hadoop-2.7.3-src\hadoop-yarn-project\...\yarn\server\resourcemanager\reservation\TestCapacityOverTimePolicy.java

To: hadoop-2.7.3-src\hadoop-yarn-project\...\yarn\server\resourcemanager\reservation\exceptions\ResourceOverCommitException.java

ResourceOverCommitException –> Used only as expected Exception, but never use within any method implementation as a catch item

CompareUnique.java | TestCapacityOverTimePolicy.java

```java
import static org.mockito.Mockito.mock;
import static org.mockito.Mockito.when;

import java.io.IOException;
import java.util.Map;
import java.util.TreeMap;

import org.apache.hadoop.yarn.api.records.ReservationRequest;
import org.apache.hadoop.yarn.api.records.Resource;
import org.apache.hadoop.yarn.server.resourcemanager.reservation.exceptions.PlanningException;
import org.apache.hadoop.yarn.server.resourcemanager.reservation.exceptions.PlanningQuotaException;
import org.apache.hadoop.yarn.server.resourcemanager.reservation.exceptions.ResourceOverCommitException;

import org.apache.hadoop.yarn.server.resourcemanager.scheduler.QueueMetrics;
import org.apache.hadoop.yarn.util.resource.DefaultResourceCalculator;
import org.apache.hadoop.yarn.util.resource.ResourceCalculator;
import org.junit.Assert;
import org.junit.Before;
import org.junit.Test;

public class TestCapacityOverTimePolicy {

  long timeWindow;
  long step;
  float avgConstraint;
  float instConstraint;
  long initTime;
```

```java
            ReservationSystemTestUtil.getNewReservationId(), null, "u" + i,
            "dedicated", initTime, initTime + f.length,
            ReservationSystemTestUtil.generateAllocation(initTime, step, f),
            res, minAlloc)));
    }
  }

  @Test(expected = ResourceOverCommitException.class)
  public void testMultiTenantFail() throws IOException, PlanningException {
    // generate allocation from multiple tenants that exceed tot capacity
    int[] f = generateData(3600, (int) Math.ceil(0.25 * totCont));
    for (int i = 0; i < 5; i++) {
      assertTrue(plan.toString(),
          plan.addReservation(new InMemoryReservationAllocation(
              ReservationSystemTestUtil.getNewReservationId(), null, "u" + i,
              "dedicated", initTime, initTime + f.length,
              ReservationSystemTestUtil.generateAllocation(initTime, step, f),
              res, minAlloc)));
    }
```

org.apache.hadoop.yarn.server.resourcemanager.
TestCapacityOverTimePolicy
- timeWindow : long
- step : long
- avgConstraint : float
- instConstraint : float
- initTime : long
- plan : InMemoryPlan
- mAgent : ReservationAgent
- minAlloc : Resource
- res : ResourceCalculator
- maxAlloc : Resource
- totCont : int
- setup() : void
- generateData(int, int) : int[]
- testSimplePass() : void
- testSimplePass2() : void
- testMultiTenantPass() : void
- testMultiTenantFail() : void
- testInstFail() : void
- testInstFailBySum() : void
- testFailAvg() : void
- testFailAvgBySum() : void

- initTime : long
- plan : InMemoryPlan
- mAgent : ReservationAgent
- minAlloc : Resource
- res : ResourceCalculator
- maxAlloc : Resource
- totCont : int
- setup() : void
- generateData(int, int) : int[]
- testSimplePass() : void
- testSimplePass2() : void
- testMultiTenantPass() : void
- testMultiTenantFail() : void
- testInstFail() : void
- testInstFailBySum() : void
- testFailAvg() : void
- testFailAvgBySum() : void

Understand VS Include

OVERLAP

- **8297**: …\hadoop\hdfs\server\namenode**FSDirSymlinkOp.java** → …\hadoop\fs\permission**PermissionStatus**.java
- Both have an import statement
- Understand contains method parameters and method calls

```
PermissionStatis dirPerms;

dirPerms.getUserName;
```

Understand Only

- **44087**…\java\org\apache\hadoop\security**TestNetgroupCache.java** → …\main\java\org\apache\hadoop\security**NetgroupCache.java**
- No import statement in TestNetgroupCache.java
- Uses a method from NetGroupCache, both in same Hadoop project so import not needed
- Static reference is allowed in Java

```
NetgroupCache.add(GROUP1, users)
```

Include Only

- **62615**:…\hadoop\yarn\server\nodemanager\containermanager\container**ContainerImpl.java** →… \hadoop\yarn\server\nodemanager\containermanager\application\ **ApplicationContainerFinishedEvent.java**
- Import statement in file but not outputted in Understand csv file
- Understand Dependency browser lists it as a dependency
- File does create a new Application class

```
eventHandler.handle(new
ApplicationContainerFinishedEvent(containerId))
```

Qualitative Overview (IncVUnd)

It is generally seen that the dependents not overlapping retrieved by Understand is due to children being imported but parents being used and data types being used without any imports at all. Thus having more dependencies that would not be found by Include as it only finds imports of the direct class

The Include tool finds some dependencies that are not found by Understand and those are generally imported classes used as parameters or links, or inherited classes that add features

The precision of the Include tool is 40404/41293 ~ 0.98

The recall of the Include tool is 40404/65650 ~ 0.64

Intellij IDEA vs. Understand

QUANTITATIVE ANALYSIS

Extracting info from IDEA


```xml
<?xml version="1.0" encoding="UTF-8"?>
<root isBackward="false">
  <file path="$PROJECT_DIR$/hadoop-common-project/hadoop-annotations/src/ma
    <?xml version="1.0" encoding="UTF-8"?>
    <root isBackward="false">
      <file path="hadoop-2.7.3-src/hadoop-common-project/hadoop-annotations/src/m
        <dependency path="hadoop-2.7.3-src/hadoop-common-project/hadoop-annotatio
      </file>
  </f  <file path="hadoop-2.7.3-src/hadoop-common-project/hadoop-annotations/src/m
  <fi    <dependency path="hadoop-2.7.3-src/hadoop-common-project/hadoop-annotatio
      </file>
      <file path="hadoop-2.7.3-src/hadoop-common-project/hadoop-annotations/src/m
        <dependency path="hadoop-2.7.3-src/hadoop-common-project/hadoop-annotatio
        <dependency path="hadoop-2.7.3-src/hadoop-common-project/hadoop-annotatio
  </f  </file>
  <fi  <file path="hadoop-2.7.3-src/hadoop-common-project/hadoop-annotations/src/m
         <dependency path="hadoop-2.7.3-src/hadoop-common-project/hadoop-annotatio
         <dependency path="hadoop-2.7.3-src/hadoop-common-project/hadoop-annotatio
       </file>
       <file path="hadoop-2.7.3-src/hadoop-common-project/hadoop-annotations/src/m
         <dependency path="hadoop-2.7.3-src/hadoop-common-project/hadoop-annotatio
```

Extracting info from IDEA

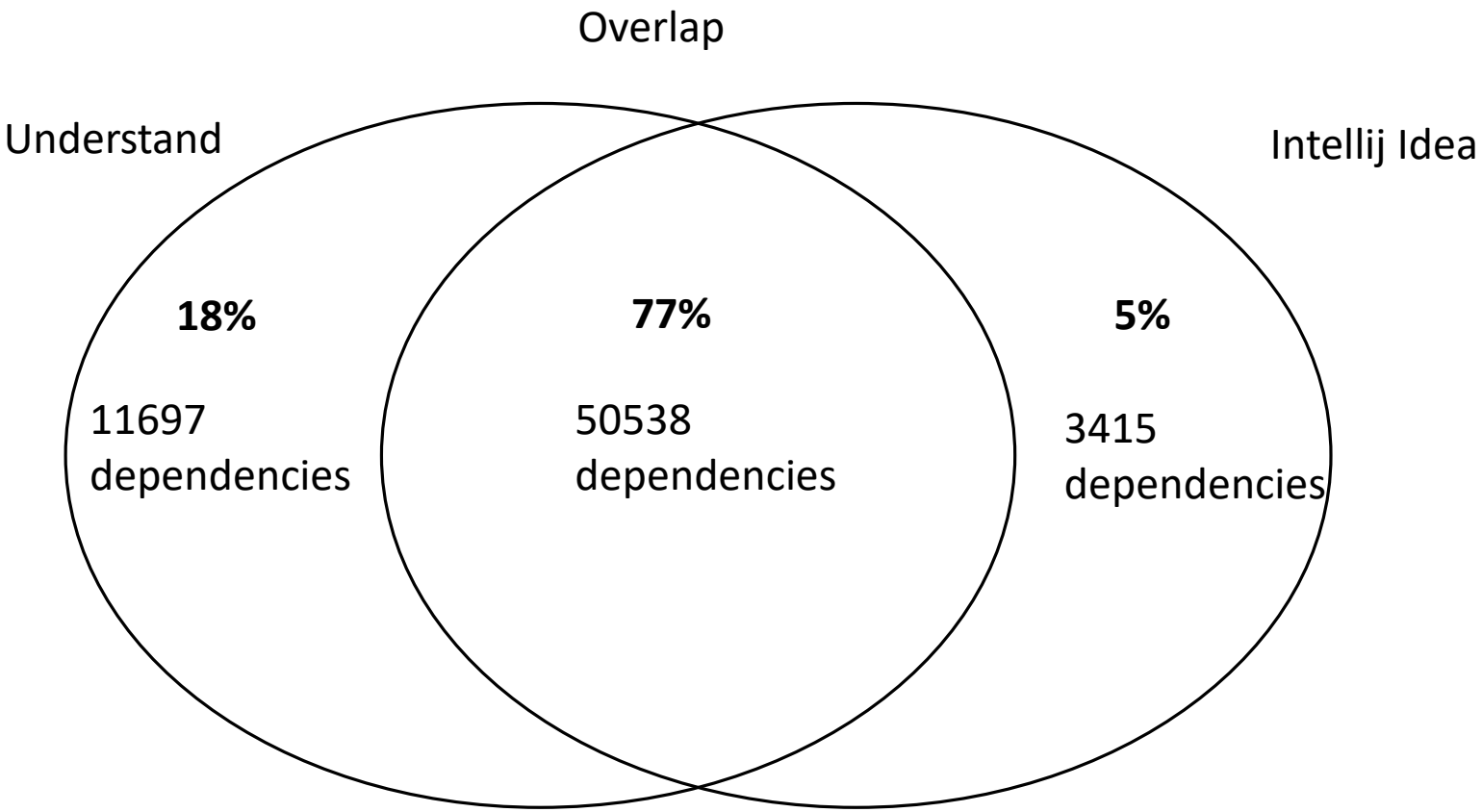
```
public class XMLParsing_idea {

    public
        7.3-src\hadoop-common-project\hadoop-annotations\src\main\java\org\apache\hadoop\classificatic
        7.3-src\hadoop-common-project\hadoop-annotations\src\main\java\org\apache\hadoop\classificatic
        7.3-src\hadoop-common-project\hadoop-annotations\src\main\java\org\apache\hadoop\classificatic
        7.3-src\ha he/hadoop/classification/tools/IncludePublicAnnotationsStandardDoclet.java,
        7.3-src\ha he/hadoop/classification/tools/package-info.java,hadoop-2.7.3-src/hadoop-co
        7.3-src\ha he/hadoop/classification/tools/RootDocProcessor.java,hadoop-2.7.3-src/hadoo
        7.3-src\ha he/hadoop/classification/tools/RootDocProcessor.java,hadoop-2.7.3-src/hadoo
        7.3-src\ha he/hadoop/classification/tools/RootDocProcessor.java,hadoop-2.7.3-src/hadoo
        7.3-src\ha he/hadoop/classification/tools/StabilityOptions.java,hadoop-2.7.3-src/hadoo
        7.3-src\ha ache/hadoop/security/authentication/examples/WhoClient.java,hadoop-2.7.3-sr
        7.3-src\ha oop/security/authentication/client/AuthenticatedURL.java,hadoop-2.7.3-src/ha
        7.3-src\ha oop/security/authentication/client/AuthenticatedURL.java,hadoop-2.7.3-src/ha
        7.3-src\ha oop/security/authentication/client/AuthenticatedURL.java,hadoop-2.7.3-src/ha
        7.3-src\ha oop/security/authentication/client/AuthenticatedURL.java,hadoop-2.7.3-src/ha
        7.3-src\ha oop/security/authentication/client/AuthenticatedURL.java,hadoop-2.7.3-src/ha
        7.3-src\ha oop/security/authentication/client/Authenticator.java,hadoop-2.7.3-src/hadoo
        7.3-src\ha oop/security/authentication/client/Authenticator.java,hadoop-2.7.3-src/hadoo
        7.3-src\ha oop/security/authentication/client/Authenticator.java,hadoop-2.7.3-src/hadoo
        7.3-src\ha oop/security/authentication/client/KerberosAuthenticator.java,hadoop-2.7.3-s
        7.3-src\ha oop/security/authentication/client/KerberosAuthenticator.java,hadoop-2.7.3-s
        7.3-src\ha oop/security/authentication/client/KerberosAuthenticator.java,hadoop-2.7.3-s
        7.3-src\ha oop/security/authentication/client/KerberosAuthenticator.java,hadoop-2.7.3-s
                oop/security/authentication/client/KerberosAuthenticator.java,hadoop-2.7.3-s
                oop/security/authentication/client/KerberosAuthenticator.java,hadoop-2.7.3-s
                oop/security/authentication/client/KerberosAuthenticator.java,hadoop-2.7.3-s
```

Extraction philosophy

IDEA

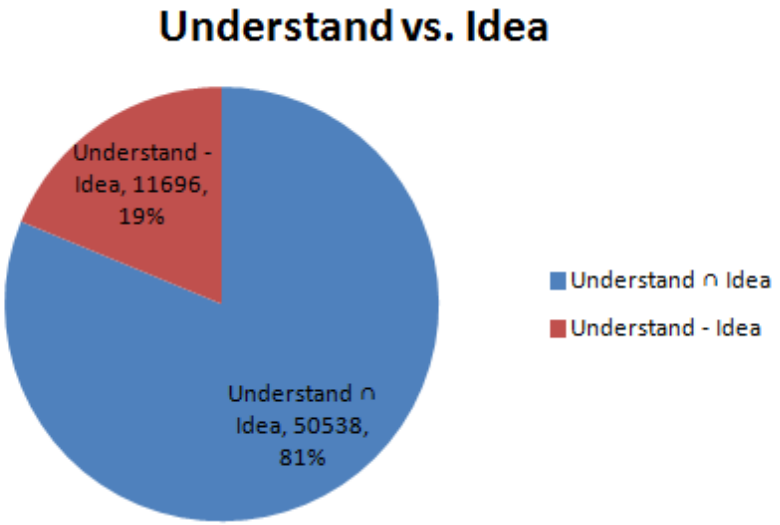
- Include .xml dependency
- It grabs dependency relation in a different way than Understand
- Implicit dependency such as Extend/implement is evaluated
- Extract dependency base on parameters of methods of every class

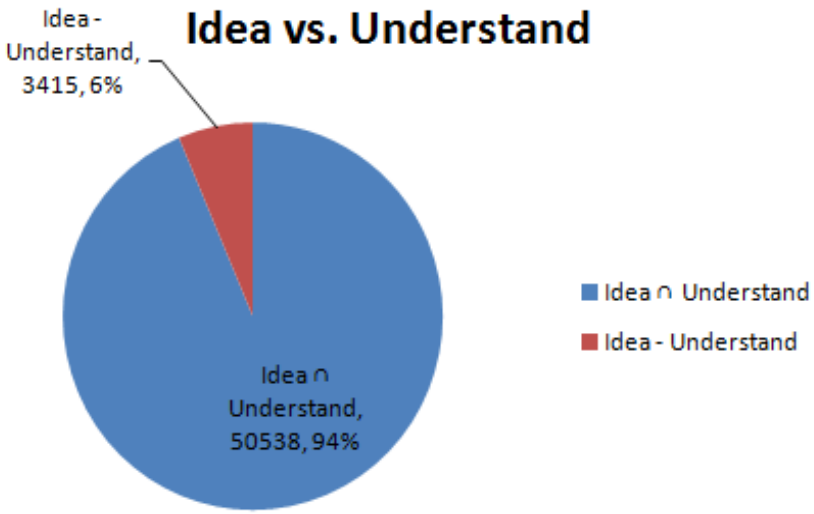
Venn Diagram of Understand VS Intellij IDEA



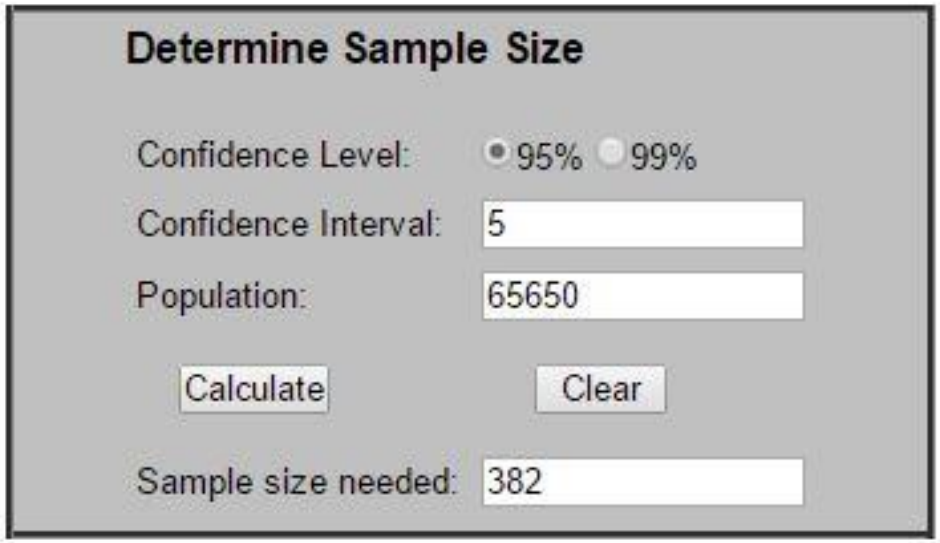
Statistics: Understand vs. IDEA

	Intersect	Excluding	Total
Undertand	50538	11696	62234
Idea		3415	53953
Total Dependencies	63123		





Sample Calculator for IDEA VS UNDERSTAND



Intellij IDEA vs. Understand

QUALITATIVE ANALYSIS

Non-Java dependency

Will be excluded from our research scope.

Dependency relation is defined in .xml file for namespace concern.

...\hadoop-common-project\hadoop-auth-examples\src\main\webapp\WEB-INF**web.xml** dependent

...\hadoop-common-project\hadoop-auth\src\main\java\org\apache\hadoop\security\authentication\server**AuthenticationFilter.java** dependee

```
<filter>
  <filter-name>requestLoggerFilter</filter-name>
  <filter-class>org.apache.hadoop.security.authentication.exampl
</filter>
```

IDEA only

62251:

...\src\main\java\org\apache\hadoop\fs**FileUtil.java** dependent

...\src\main\java\org\apache\hadoop\fs**FSDataInputStream.java** dependee

```
public static boolean copy(FileSystem srcFS, FileStatus srcStatus,
                           FileSystem dstFS, Path dst,
        in = srcFS.open(src);
```

```
public abstract class FileSystem extends Configured implements Closeable {

        public abstract FSDataInputStream open(Path f, int bufferSize)
          throws IOException;
```

IDEA only

62303:

…\src\main\java\org\apache\hadoop\io**AbstractMapWritable**.java dependent

…\src\main\java\org\apache\hadoop\io**NullWritable**.java dependee

```
addToMap(NullWritable.class,
    Byte.valueOf(Integer.valueOf(-119).byteValue()));
```

IDEA only

62357:

dependent

...\src\main\java\org\apache\hadoop\security\token\delegation\ZKDelegationTokenSecretManager.java

...\src\main\java\org\apache\hadoop\io\Writable.java dependee

```
private void processKeyAddOrUpdate(byte[] data) throws IOException {
  ByteArrayInputStream bin = new ByteArrayInputStream(data);
  DataInputStream din = new DataInputStream(bin);
  DelegationKey key = new DelegationKey();
```

```
public class DelegationKey implements Writable {
```

IDEA only

62652 :

...\src\main\java\org\apache\hadoop\hdfs\server\namenode**NameNode**.java dependent

...\src\main\java\org\apache\hadoop\hdfs\server\namenode**Namesystem**.java dependee

```
protected FSNamesystem namesystem;
```

```
public class FSNamesystem implements Namesystem, FSNamesystemMBean,
```

Understand only

50864:

...\src\main\java\org\apache\hadoop\security\authentication\client**KerberosAuthenticator**.java dependent

...\src\test\java\org\apache\hadoop\crypto**TestCryptoCodec**.java dependee

Seems to be error.

\main folder may not need to invoke a function in \test folder.

Found no clues in dependent class

Understand only

58802:

...\src\main\java\org\apache\hadoop\yarn\server\nodemanager\recovery**NMLeveldbStateStoreService**.java dependent

...\src\main\java\org\apache\hadoop\yarn\server\records\impl\pb**\VersionPBImpl.**java dependee

```
import org.apache.hadoop.yarn.server.records.impl.pb.VersionPBImpl;
```

```
Version version =
    new VersionPBImpl(VersionProto.parseFrom(data));
```

```
byte[] data =
    ((VersionPBImpl) state).getProto().toByteArray();
```

Understand only

62206:

...\src\test\java\org\apache\hadoop\yarn\server\webproxy**TestAppReportFetcher**.java dependent

...\src\main\java\org\apache\hadoop\yarn\api**ApplicationBaseProtocol**.java dependee

TestAppReportFetcher.testHelper Calls ApplicationBaseProtocol.getApplicationReport

IDEA doesn't detect this, doesn't list most dependencies from file.

Overlap

15972:

…\src\test\java\org\apache\hadoop\fs**TestHDFSFileContextMainOperations**.java dependent

…\src\main\java\org\apache\hadoop\hdfs**DistributedFileSystem.**java dependee

```
import org.apache.hadoop.hdfs.DistributedFileSystem;

DistributedFileSystem fs = cluster.getFileSystem();
```

Overlap

29322:

…\src\main\java\org\apache\hadoop\mapreduce\counters**CounterGroupBase**.java dependent

…\src\main\java\org\apache\hadoop\mapreduce**\Counter.**java dependee

```
import org.apache.hadoop.mapreduce.Counter;
```

```
public interface CounterGroupBase<T extends Counter>
```

Qualitative Overview (IdVInc)

Similarly like the Include tool the dependencies missed by the IDEA tool are children that are imported but methods implemented by a parent

Unlike Include, the IDEA tool missed some dependencies that would have otherwise been found by Include

IDEA tool finds dependencies that aren't found by Understand, these are found to be implemented classes (may not be an actual dependency)

The precision of the IDEA tool is 50538/53954 ~ 0.94

The recall of the IDEA tool is 50538/65650 ~ 0.77

Alternative Tool: srcML

Other Alternative Tool Explore - srcML

About ScrML

- Convert source code file into srcML format, which is an XML representation of source code. As xml representation, markup tags identify elements of the abstract syntax for the language.
- It currently support the parsing of C, C++, C# and Java.
- The transformation is lossless. No changes is made to the original source code file

How ScrML work

- Once in srcML format, XML tools and technologies can be used for such things as extraction and transformation.
- Using its query command ‘xpath’, it is capable of executing various fact extraction.

Initial Aim

- extract dependency by listing the types of variables declared by every class.

```
srcml --xpath "//src:decl_stmt/src:decl/src:type"
hadoop-2.7.3-src\hadoop-common-project
>decl_stmt_common.xml

srcml --xpath "//src:decl_stmt/src:decl/src:type"
hadoop-2.7.3-src\hadoop-hdfs-project
>decl_stmt_hdfs.xml
```

(due to the large number of instances, run the query by subdirectory)

Challenges

- Need to map every type to its associated file path and requires running additional query for list of all classes
- Need to tackle type of generic class, and hence require further parsing of the parameter types
- Need to map with associated packages, and requires running additional query for list of import statements (which makes this exercise almost similar to Include.java)

Other Alternative Tool Explore - srcML

Open | BloomFilterCommonTester.java | Windows (C:) \srcML 0.9.5\bin\hadoo...st\java\org\apache\hadoop\util\bl... | Save

```
29
30 import org.junit.Assert;
31 import org.apache.hadoop.io.DataInputBuffer;
32 import org.apache.hadoop.io.DataOutputBuffer;
33 import org.apache.hadoop.util.hash.Hash;
34 import org.apache.log4j.Logger;
35
36 import com.google.common.collect.ImmutableList;
37 import com.google.common.collect.ImmutableSet;
38
39 public class BloomFilterCommonTester<T extends Filter> {
40
41   private static final double LN2 = Math.log(2);
42   private static final double LN2_SQUARED = LN2 * LN2;
43
44   private final int hashType;
45   private final int numInsertions;
46
47   private final ImmutableList.Builder<T> builder = ImmutableList.builder();
48
49   private ImmutableSet<BloomFilterTestStrategy> filterTestStrateges;
50
51   private final PreAssertionHelper preAssertionHelper;
52
53   static int optimalNumOfBits(int n, double p) {
54     return (int) (-n * Math.log(p) / LN2_SQUARED);
55   }
56
57   public static <T extends Filter> BloomFilterCommonTester<T> of(int hashId,
58       int numInsertions) {
59     return new BloomFilterCommonTester<T>(hashId, numInsertions);
60   }
61
```

Type of generic class

Lessons Learned

- Powerful programs like Understand can make errors
- There is a lot of digging to find dependencies between classes, especially in a sophisticated Java IDE like Intellij IDEA
- Many Approaches at extracting Dependency (listing include, method-to-method, extending interface/inheriting classes, method's parameter types)

  Focusing just at one approach is not advisable
  - Understand is more inclusive, but comprise many redundant non-inter-subcomponent's dependencies --> confusing and might lead to wrong conclusion
  - Include offers the most compact information, however it might omit important dependency by excluding the listing of its 'supplier' classes
  - Idea provides dependency based on type of parameter of every class methods. These reveals those classes as 'helper methods'. However, cross study with back dependency is required to fully utilize this information
- Studying Dependency reveals a substantial part of the design of the whole system

Conclusion

- We used 3 different extraction techniques
  - Include
  - Understand
  - Intellij IDEA
- Quantitative and Qualitative Analysis of Include VS Understand
- Quantitative and Qualitative Analysis of IDEA VS Understand
- Alterantive tool: srcML
- Lessons Learned