Report ISW2 – Modulo Software Testing – Prof. G. De Angelis

Tummolo Gabriele 0283629

Introduzione

Il seguente documento descrive il lavoro svolto durante il corso di Ingegneria del Software 2; in particolare si riferisce al modulo sul Software Testing svolto con il Professore Guglielmo De Angelis. Questo progetto consiste nell'applicare delle tecniche di software testing a due progetti open-source Apache, valutando successivamente il lavoro svolto tramite l'ausilio di diversi frameworks. I progetti presi in analisi sono "Apache Bookkeeper" e "Apache Tajo".

Configurazione

Per la configurazione dei progetti in locale è stata effettuata prima la fork dalla repository Github, successivamente si è proceduto ad eliminare tutti i test presenti, in quanto non utili per lo scopo del progetto. Inoltre i progetti sono stati collegati con Travis CI e SonarCloud. Tajo non è stato integrato con SonarCloud.

Travis Ci si occupa dell'esecuzione della build del progetto e dell'esecuzione dei casi di test, mentre SonarCloud esegue un controllo qualitativo del progetto. I progetti sono stati scaricati localmente per poter iniziare a sviluppare il progetto; per poter effettuare la compliazione in locale è stato necessario aggiornare la variabile d'ambiente JAVA_HOME per i rispettivi progetti (Java8 per Tajo e Java11 per Bookkeeper).

Classi Bookkeeper

Per questo progetto sono state scelte le seguenti classi, che sono state prese dal modulo di bookkeeper-server:

- **ArrayGroupSort**: questa classe permette di ordinare un array di long
- **RoundRobinDistribuionSchedule**: che rappresenta uno specifico DistributionSchedule inserendo le voci in modalità RoundRobin; DistributionSchedule determina come vengono distribuite le entries tra i bookies.

Entries: contengono i dati scritti nei Ledgers insieme ad alcuni importanti Metadati.

Bookie: sono server di Bookkeeper che gestiscono i vari Ledgers.

Ledgers: sono un unità di archiviazione di Bookkeeper.

Queste informazioni sono state reperite dal link 1.

Classi Tajo

Per Tajo sono state scelte anche due classi che si occupano della gestione delle date, in particolare sono delle specializzazioni della classe Datum:

- **TimeDatum**
- **TextDatum**

Ogni classe permette di effettuare delle operazioni sulle date per tipologia di rappresentazione.

Category Partition

Nella creazione della Category Partition è stato utilizzato il metodo descritto durante il corso, partizionando il dominio di input includendo un elemento per ogni partizione.

Category Partition Bookkeeper

- **ArrayGroupSort:** come detto sopra questa classe permette di ordinare in modo crescente un array di long; per effettuare Category Partition è stato studiato in primo luogo i costruttore di questa classe. Public ArrayGroupSort **(**int keySize,int groupSize) che è il costruttore di questa classe ha come parametri 2 interi keySize e groupSize che definiscono rispettivamente fino a che punto deve essere fatto il controllo nel sottogruppo dell'array di long e la dimensione del sottogruppo. E' stato preso in considerazione il metodo public void sort(long[] array) che riordina l'array. Con le ipotesi sopra indicate sono stati presi i seguenti valori da esaminare:
    - Array di long[] ammissibile (con dimensione multipla a groupsize)
    - Array di long[] non ammissibile (con dimensione non multipla a groupsize)
    - Array di long[] vuoto
- **RoundRobilDistributionSchedule**: come fatto per la classe precedente è stato prima studiato il suo costruttore, che introduceva nuove variabili inizialmente non analizzate.Public RoundRobinDistributionSchedule (int writeQuorumSize, int ackQuorumSize, int ensembleSize) , in particolare dalla documentazione sono state ricavate le informazioni per la dichiarazione corretta di questa classe, infatti deve essere soddisfatta la seguente condizione ensembleSize >= writeQuorumSize >= ackQuorumSize. Successivamente si è pensato di testare il seguente metodo public BitSet getEntriesStripedToTheBookie(int bookieIndex, long startEntryId, long lastEntryId) che permette di ottenere un set di bit che rappresenta le voci che verrebbero trasferite al bookie con il rispettivo indice. Dopo queste considerazioni sono stati costruiti i vari domini:
    - bookieIndex { >=0 && <ensembleSize, >=0 && >=ensembleSize}
    - startEntryId {>=0,<0}
    - lastEntryId {>=0,<0}

    Come suggerito dai nomi degli ultimi 2 argomenti, devono essere presa in considerazione anche questa condizione { startEntryId <= lastEntryId}

Category Partition Tajo

- **TimeDatum:** per questa classe sono stati testati diversi metodi. Public boolean equals(Object obj), questo metodo permette verificare se una classe TimeDatum è uguale ad un'altra (cioè se rappresenta la stessa ora) restituendo un valore booleano, l'unico parametro essendo una classe Object il dominio individuate {null, variabile ammissibile, variabile non ammissibile}. Public int compareTo(Datum datum) in questo metodo l'unico argomento è di tipo Datum quindi il dominio è {null, Datum, not Datum}; per creare istanze Datum è stato utilizzato DatumFactory. Public equalsTo(Datum datum) questo metodo molto simile al precedete ha portato alla creazione dello stesso e identico dominio. Public Datum minus(Datum datum) che permette di effettuare una differenza tra 2 date e il dominio considerato è stato {null, datumTime, datumInterval, datum diverso da Time e Interval }. Public Datum plus(Datum datum) permette di effettuare la somma tra 2 datum , ha come i precedenti un solo argomento e il dominio considerato è {null, datumInterval, datumDate, DatumTimeStamp, Datum differente dai precedenti}.
- **TextDatum:** anche per questa classe sono stati testati più metodi che sono molto simili a quelli della classe precendente. Public Datum equalsTo(Datum datum) permette di verificare se 2 textDatum rappresentano la stessa data e per l'unico parametro il dominio è {null, DatumBool, Datum differente da

Text,char,blob}. Public boolean equals(Object obj) è molto simile al metodo precedente ma restituisce un valore booleano , il dominio per l'unico parametro è {null, TextDatum, Datum differente da TextDatum}. Public int compareTo(Datum datum) il cui dominio di partizione è {null,BoolDatum, datum differente da bool}.

Implementazione dei casi di test/Jacoco

Per implementare i test è stato utilizzato il framework JUnit, in particolare sono stati definiti dei metodi con l'annotazione @Test, che permette di identificare i vari metodi che implementano i Test.

Il primo passo è stato quello di scrivere dei metodi per implementare i Test seguendo le considerazioni fatte in precedenza. Dopo una prima scrittura dei casi di test è stato utilizzato il plugin Jacoco che mi ha permesso di ottenere informazioni riguardo l'adeguatezza dei Test, in particolare riguardo il criterio statement coverage e branch coverage. Dopo aver integrato Jacoco nel progetto è stato analizzato il report per verificare se secondo quei criteri i test precedentemente sviluppati sono soddisfacenti. Dopo aver preso in considerazione questo i test sono stati migliorati i test aumentando le percentuali di copertura dei vari criteri, questo è stato fatto aumentando il numero di Test effettuati testando anche, eventualmente, metodi non presi in considerazione per aumentare la copertura del test sull'intera classe e non limitando la copertura al singolo metodo preso in considerazione (o metodi in caso di Tajo).

Mutation Test

Per misurare l'adeguatezza e la robustezza dei Test implementati è stato utilizzato PIT che permette di effettuare mutation test, che è stato inizialmente integrato nel progetto modificando il pom.xml. Bisogna specificare le classi target per effettuare questa mutazione evitando cosi di fare mutazioni su altre classi, visualizzando il report si è provato a rendere il test più robusto modificando o aggiungendo dei test che hanno permesso di ridurre il numero di mutanti sopravvissuti. Mutation Test effettua delle modifiche alle classi indicate e successivamente testa le classi con i file di test indicati, i mutanti killati sono quei mutanti che non sopravvivono alle modifiche apportate, mentre i mutanti sopravvissuti sono appunto quelli che con le modifiche apportate non falliscono.

Molti mutanti sono stati da survived a killed integrando e modificando i test, ma questo non sempre ha avuto successo soprattutto nel progetto Bookkeeper in quanto le modifiche che fanno sopravvivere i mutanti sono molto complicate.

Immagini Jacoco

Figura 1 Jacoco Bookkeeper RoundRobin

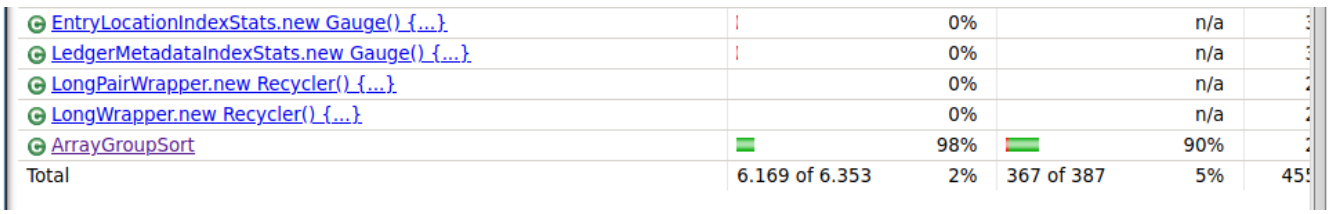

Figura 2 Jacoco Bookkeeper ArrayGroupSort

org.apache.tajo.datum

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed
IntervalDatum		34%		21%	89
Int8Datum		0%		0%	96
Float4Datum		0%		0%	96
Float8Datum		0%		0%	94
Int2Datum		0%		0%	94
Int4Datum		0%		0%	95
DatumFactory		7%		2%	114
Datum		6%		0%	68
DateDatum		4%		0%	41
TimestampDatum		17%		13%	38
BlobDatum		0%		0%	22
BooleanDatum		47%		0%	36
CharDatum		0%		0%	28
BitDatum		0%		0%	25
ProtobufDatumFactory		0%		0%	12
NullDatum		23%		0%	29
AnyDatum		0%		0%	11
ProtobufDatum		0%		0%	11
TextDatum		93%		100%	3
TimeDatum		98%		100%	1
NumericDatum		0%		n/a	1
IntervalDatum.DATE_UNIT		100%		n/a	0
Total	7.031 of 8.413	16%	927 of 996	6%	1.004

Figura 3 Jacoco Tajo

tajo-common

Element	Missed Instructions	Cov.	Missed Branches	Cov.
org.apache.hadoop.yarn.proto		0%		0%
org.apache.tajo.util		0%		0%
org.apache.tajo.util.datetime		39%		8%
org.apache.tajo.rpc.protocolrecords		0%		0%
org.apache.tajo		0%		0%
org.apache.tajo.datum		16%		6%
org.apache.tajo.common		12%		1%
org.apache.hadoop.security.proto		0%		0%
org.apache.tajo.tuple.memory		0%		0%
org.apache.tajo.error		40%		0%
org.apache.tajo.datum.protobuf		0%		0%
org.apache.tajo.exception		13%		3%
org.apache.tajo.type		8%		0%
org.apache.tajo.schema		0%		0%
org.apache.tajo.validation		12%		2%
org.apache.tajo.conf		70%		3%
org.apache.tajo.util.graph		0%		0%
org.apache.tajo.storage		0%		0%
org.apache.tajo.rule		0%		0%
org.apache.tajo.common.type		0%		0%
org.apache.tajo.json		0%		0%
org.apache.tajo.rule.base		0%		0%
org.apache.tajo.service		0%		0%
org.apache.tajo.auth		0%		0%
org.apache.tajo.tuple		0%		0%
org.apache.tajo.unit		0%		n/a
org.apache.tajo.ha		0%		n/a
Total	140.499 of 156.747	10%	15.717 of 15.925	1%

Figura 4 Jacoco Tajo Main

Immagini PIT

Mutations

1. changed conditional boundary → KILLED
2. negated conditional → KILLED
3. removed call to com/google/common/base/Preconditions::checkArgument → KILLED

1. changed conditional boundary → SURVIVED
2. negated conditional → KILLED
3. removed call to com/google/common/base/Preconditions::checkArgument → SURVIVED

1. changed conditional boundary → SURVIVED
2. negated conditional → KILLED
3. removed call to com/google/common/base/Preconditions::checkArgument → SURVIVED

1. removed call to org/apache/bookkeeper/bookie/storage/ldb/ArrayGroupSort::sort → KILLED

1. Replaced integer modulus with multiplication → KILLED
2. negated conditional → KILLED
3. removed call to com/google/common/base/Preconditions::checkArgument → SURVIVED

1. Replaced integer addition with subtraction → SURVIVED
2. Replaced integer subtraction with addition → KILLED
3. removed call to org/apache/bookkeeper/bookie/storage/ldb/ArrayGroupSort::quickSort → KILLED

1. changed conditional boundary → SURVIVED
2. negated conditional → KILLED

1. Replaced integer subtraction with addition → KILLED
2. removed call to org/apache/bookkeeper/bookie/storage/ldb/ArrayGroupSort::quickSort → SURVIVED

1. Replaced integer addition with subtraction → KILLED
2. removed call to org/apache/bookkeeper/bookie/storage/ldb/ArrayGroupSort::quickSort → KILLED

1. changed conditional boundary → SURVIVED
2. Replaced integer addition with subtraction → KILLED
3. negated conditional → KILLED

1. negated conditional → KILLED

1. removed call to org/apache/bookkeeper/bookie/storage/ldb/ArrayGroupSort::swap → KILLED

1. Replaced integer addition with subtraction → KILLED

1. removed call to org/apache/bookkeeper/bookie/storage/ldb/ArrayGroupSort::swap → KILLED

1. replaced int return with 0 for org/apache/bookkeeper/bookie/storage/ldb/ArrayGroupSort::partition → KILLED

1. changed conditional boundary → KILLED
2. Changed increment from 1 to -1 → KILLED
3. negated conditional → KILLED

1. Replaced integer addition with subtraction → KILLED

1. Replaced integer addition with subtraction → KILLED
2. Replaced integer addition with subtraction → KILLED

1. Replaced integer addition with subtraction → KILLED

1. changed conditional boundary → SURVIVED
2. Changed increment from 1 to -1 → KILLED
3. negated conditional → KILLED

1. Replaced integer addition with subtraction → KILLED

1. Replaced integer addition with subtraction → KILLED

1. changed conditional boundary → KILLED
2. negated conditional → KILLED

1. replaced boolean return with false for org/apache/bookkeeper/bookie/storage/ldb/ArrayGroupSort::isLess → KILLED

1. changed conditional boundary → KILLED
2. negated conditional → KILLED

1. replaced boolean return with true for org/apache/bookkeeper/bookie/storage/ldb/ArrayGroupSort::isLess → KILLED

1. replaced boolean return with true for org/apache/bookkeeper/bookie/storage/ldb/ArrayGroupSort::isLess → SURVIVED

Figura 5 PIT Bookkeeper ArrayGroupSort

Mutations

1. replaced return value with null for org/apache/bookkeeper/client/RoundRobinDistributionSchedule::getWriteSet → KILLED
1. replaced return value with null for org/apache/bookkeeper/client/RoundRobinDistributionSchedule::getEnsembleSet → NO_COVERAGE
1. removed call to org/apache/bookkeeper/client/RoundRobinDistributionSchedule$WriteSetImpl::access$000 → NO_COVERAGE
1. changed conditional boundary → NO_COVERAGE
2. negated conditional → NO_COVERAGE
1. replaced return value with null for org/apache/bookkeeper/client/RoundRobinDistributionSchedule::writeSetFromValues → NO_COVERAGE
1. replaced return value with null for org/apache/bookkeeper/client/RoundRobinDistributionSchedule::getAckSet → NO_COVERAGE
1. replaced return value with null for org/apache/bookkeeper/client/RoundRobinDistributionSchedule::getEnsembleAckSet → NO_COVERAGE
1. replaced return value with null for org/apache/bookkeeper/client/RoundRobinDistributionSchedule::getCoverageSet → NO_COVERAGE
1. replaced boolean return with false for org/apache/bookkeeper/client/RoundRobinDistributionSchedule::hasEntry → KILLED
2. replaced boolean return with true for org/apache/bookkeeper/client/RoundRobinDistributionSchedule::hasEntry → KILLED
1. removed call to org/apache/bookkeeper/client/DistributionSchedule$WriteSet::recycle → SURVIVED
1. changed conditional boundary → SURVIVED
2. changed conditional boundary → SURVIVED
3. changed conditional boundary → SURVIVED
4. changed conditional boundary → SURVIVED
5. changed conditional boundary → SURVIVED
6. negated conditional → KILLED
7. negated conditional → KILLED
8. negated conditional → KILLED
9. negated conditional → KILLED
10. negated conditional → KILLED
1. Replaced long subtraction with addition → SURVIVED
2. Replaced long addition with subtraction → SURVIVED
1. changed conditional boundary → KILLED
2. Replaced long addition with subtraction → KILLED
3. negated conditional → KILLED
1. Replaced long modulus with multiplication → KILLED
1. Replaced long addition with subtraction → KILLED
2. Replaced long subtraction with addition → KILLED
3. Replaced long modulus with multiplication → KILLED
1. changed conditional boundary → SURVIVED
2. negated conditional → KILLED
1. changed conditional boundary → SURVIVED
2. changed conditional boundary → KILLED
3. negated conditional → KILLED
4. negated conditional → KILLED
1. Replaced long subtraction with addition → KILLED
2. removed call to java/util/BitSet::set → KILLED
1. changed conditional boundary → KILLED
2. changed conditional boundary → NO_COVERAGE
3. negated conditional → KILLED
4. negated conditional → NO_COVERAGE
1. Replaced long subtraction with addition → KILLED
2. removed call to java/util/BitSet::set → KILLED
1. replaced return value with null for org/apache/bookkeeper/client/RoundRobinDistributionSchedule::getEntriesStripedToTheBookie → KILLED

Figura 6 PIT Bookkeper RoundRobin

Mutations

1. replaced short return with 0 for org/apache/tajo/datum/TextDatum::asInt2 → KILLED
1. replaced int return with 0 for org/apache/tajo/datum/TextDatum::asInt4 → KILLED
1. replaced long return with 0 for org/apache/tajo/datum/TextDatum::asInt8 → KILLED
1. replaced float return with 0.0f for org/apache/tajo/datum/TextDatum::asFloat4 → KILLED
1. replaced double return with 0.0d for org/apache/tajo/datum/TextDatum::asFloat8 → KILLED
1. replaced return value with null for org/apache/tajo/datum/TextDatum::asByteArray → KILLED
1. replaced return value with "" for org/apache/tajo/datum/TextDatum::asChars → KILLED
1. replaced return value with null for org/apache/tajo/datum/TextDatum::asUnicodeChars → NO_COVERAGE
1. replaced return value with null for org/apache/tajo/datum/TextDatum::asTextBytes → NO_COVERAGE
1. replaced int return with 0 for org/apache/tajo/datum/TextDatum::size → KILLED
1. replaced int return with 0 for org/apache/tajo/datum/TextDatum::compareTo → KILLED
1. replaced int return with 0 for org/apache/tajo/datum/TextDatum::compareTo → KILLED
1. negated conditional → KILLED
1. replaced boolean return with true for org/apache/tajo/datum/TextDatum::equals → KILLED
2. negated conditional → KILLED
1. replaced boolean return with true for org/apache/tajo/datum/TextDatum::equals → KILLED
1. negated conditional → KILLED
2. replaced return value with null for org/apache/tajo/datum/TextDatum::equalsTo → KILLED
1. replaced return value with null for org/apache/tajo/datum/TextDatum::equalsTo → KILLED
1. replaced int return with 0 for org/apache/tajo/datum/TextDatum::hashCode → NO_COVERAGE
1. replaced return value with "" for org/apache/tajo/datum/TextDatum::toString → KILLED

Figura 7 PIT Tajo TextDatum

Mutations

1. removed call to org/apache/tajo/util/datetime/DateTimeUtil::date2j → KILLED
1. replaced return value with null for org/apache/tajo/datum/TimeDatum::asTimeMeta → KILLED
1. replaced int return with 0 for org/apache/tajo/datum/TimeDatum::getHourOfDay → KILLED
1. replaced int return with 0 for org/apache/tajo/datum/TimeDatum::getMinuteOfHour → KILLED
1. removed call to org/apache/tajo/util/datetime/DateTimeUtil::date2j → KILLED
1. replaced int return with 0 for org/apache/tajo/datum/TimeDatum::getSecondOfMinute → KILLED
1. Replaced integer division with multiplication → KILLED
2. replaced int return with 0 for org/apache/tajo/datum/TimeDatum::getMillisOfSecond → KILLED
1. replaced return value with "" for org/apache/tajo/datum/TimeDatum::toString → KILLED
1. replaced long return with 0 for org/apache/tajo/datum/TimeDatum::asInt8 → KILLED
1. replaced return value with "" for org/apache/tajo/datum/TimeDatum::asChars → KILLED
1. replaced int return with 0 for org/apache/tajo/datum/TimeDatum::size → KILLED
1. replaced return value with null for org/apache/tajo/datum/TimeDatum::asByteArray → NO_COVERAGE
1. removed call to org/apache/tajo/util/datetime/TimeMeta::plusInterval → KILLED
1. replaced return value with null for org/apache/tajo/datum/TimeDatum::plus → KILLED
1. removed call to org/apache/tajo/util/datetime/TimeMeta::plusTime → KILLED
1. replaced return value with null for org/apache/tajo/datum/TimeDatum::plus → KILLED
1. removed call to org/apache/tajo/util/datetime/TimeMeta::plusTime → KILLED
1. replaced return value with null for org/apache/tajo/datum/TimeDatum::plus → KILLED
1. removed negation → SURVIVED
2. removed negation → KILLED
3. removed call to org/apache/tajo/util/datetime/TimeMeta::plusInterval → KILLED
1. replaced return value with null for org/apache/tajo/datum/TimeDatum::minus → KILLED
1. Replaced long subtraction with addition → KILLED
2. Replaced long division with multiplication → KILLED
3. replaced return value with null for org/apache/tajo/datum/TimeDatum::minus → KILLED
1. negated conditional → KILLED
1. negated conditional → KILLED
2. replaced return value with null for org/apache/tajo/datum/TimeDatum::equalsTo → KILLED
1. negated conditional → KILLED
1. replaced return value with null for org/apache/tajo/datum/TimeDatum::equalsTo → KILLED
1. negated conditional → KILLED
1. replaced int return with 0 for org/apache/tajo/datum/TimeDatum::compareTo → KILLED
1. negated conditional → KILLED
1. replaced int return with 0 for org/apache/tajo/datum/TimeDatum::compareTo → KILLED
1. negated conditional → KILLED
1. replaced boolean return with true for org/apache/tajo/datum/TimeDatum::equals → KILLED
2. negated conditional → KILLED
1. replaced boolean return with true for org/apache/tajo/datum/TimeDatum::equals → KILLED
1. replaced int return with 0 for org/apache/tajo/datum/TimeDatum::hashCode → KILLED

Figura 8 PIT Tajo TimeDatum

Conclusione

Entrambi i progetti sono stati integrati con Travis CI e Bookkeeper è stato integrato anche con Sonarcloud, per Apache Tajo non è stato possibile integrarlo con Sonarcloud in quanto da Gennaio 2021 SonarCloud non effettua una scansione se si utilizza Java inferiore alla verione 11.

Links

- **Bookkeeper**
  - **Github:** https://github.com/brielino/bookkeeper
  - **Travis CI:** https://www.travis-ci.com/github/brielino/bookkeeper
  - **SonarCloud:** https://sonarcloud.io/dashboard?id=brielino_bookkeeper
- **Tajo**
  - **Github:** https://github.com/brielino/tajo

- **Travis CI:** https://www.travis-ci.com/github/brielino/tajo

SonarCloud Aggiornamento Java: https://sonarcloud.io/documentation/appendices/end-of-support/