



IGUANA v0.3.0  
Manual

November 6, 2015

# Contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
1.1	Motivation . . . . .	3
1.2	Goals . . . . .	3
<b>2</b>	<b>Architecture</b>	<b>4</b>
2.1	Core . . . . .	4
2.2	LogClusterer . . . . .	4
2.2.1	FEASIBLE . . . . .	5
2.3	DataGenerator . . . . .	5
2.4	Testcase . . . . .	5
2.4.1	StressTestcase . . . . .	6
2.4.1.1	Properties . . . . .	6
2.4.1.2	Result Metrics . . . . .	8
2.4.2	FederatedStressTestcase . . . . .	8
2.4.2.1	Properties . . . . .	9
2.4.2.2	Result Metrics . . . . .	9
2.4.3	UploadShellTestcase . . . . .	9
2.4.3.1	Properties . . . . .	9
2.4.3.2	Result Metrics . . . . .	9
<b>3</b>	<b>Configuration</b>	<b>10</b>
3.1	databases . . . . .	10
3.2	logclustering . . . . .	10
3.3	suite . . . . .	11
3.3.1	randomfunction . . . . .	12
3.3.2	test-db . . . . .	12
3.3.3	testcases . . . . .	13
3.4	Example . . . . .	13
<b>4</b>	<b>Write your own ...</b>	<b>18</b>
4.1	Testcase . . . . .	18
4.2	LogClusterer . . . . .	19
<b>5</b>	<b>Further information</b>	<b>21</b>
5.1	Links . . . . .	21
5.2	License . . . . .	21
<b>A</b>	<b>LGPL</b>	<b>23</b>

# 1 Introduction

In this manual we describe IGUANA - A generic Benchmark Framework for SPARQL and SPARQL UPDATE Endpoints. First we describe why we think IGUANA is necessary in todays world. Then we'll describe the architecture and how you can configure IGUANA. Finally we'll show you how to write your own modules for IGUANA.

## 1.1 Motivation

Triplestores are the backbone of the semantic web, but to determine which of the stores is the perfect one for your datasets and the user given queries you should benchmark the triplestores which fits to your application. While there are several Benchmarks and some serves an execution platform none of them are generic. We provide with IGUANA a solution which handles SPARQL Queries and SPARQL UPDATES for any SPARQL endpoint supported by the jena remote driver. It doesn't care which dataset you have nor queries nor what you want to test. We provide a mighty execution framework which can handle SPARQL Benchmarks you desire.

## 1.2 Goals

IGUANA should be...

- ... easy to use
- ... easy to configure
- ... generic
- ... well documented
- ... nearly completely changeable

IGUANA should can ...

- ... use user defined testcases
- ... log clustering
- ... data generation
- ... work with different testcases
- ... test SPARQL and UPDATE workers
- ... upload testcases
- ... Shell scripting

## 2 Architecture

In this section we'll describe IGUANAs Architecture, it's core, the log clustering function, the data generator and the testcase interface as well as the implemented testcases.

### 2.1 Core

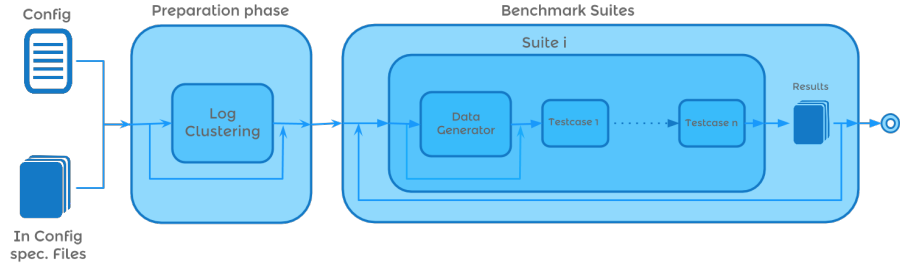


Figure 1: IGUANA Architecture v.0.3.0

As shown in Figure 1 IGUANA starts with a config file and the files/folder specified in this config file. We'll describe later on how the configuration file should look. IGUANA will parse the config file and start its preparation phase where it will cluster a given log file with a user specified log clusterer. This step is optional. If the preparation phase finished the Benchmark suites starts to begin. IGUANA will now start an already implemented in the suite user chosen DataGenerator (this step is optional too) and then all the testcases which were specified in the suite. After the suite finished the results will be saved in a folder called *results\_i* while i is the i'th suite.

The testcases will be executed for all specified and in the current suite referenced connections and all specified datasets.

Be aware that you can change the suites in the config file which will follow the last one while the previous suite is running. Also the testcases can be implemented ones as well as user written ones. If every Benchmark suite is finished IGUANA will be executed and if the user wishes send an email to the user.

### 2.2 LogClusterer

As previous told IGUANA can handle log clustering. While there is only one LogClusterer implemented in IGUANA, called FeasibleClusterer, the user can write an own LogClusterer which only needs to implement the interface *Clusterer.java* in the package *de.uni.leipzig.iguana.clustering.clusterer*. The LogClusterer can have several properties specified in the config. It should cluster a

given log file to a query file with either query templates or queries itself.

### 2.2.1 FEASIBLE

As explaining Feasible would blow the workload we link to the homepage to Feasible [<http://aksw.org/Projects/FEASIBLE.html>] and just tell what parameters you can change in the FeasibleClusterer which will use Feasible.

The parameters are:

name	description	default value	optional
draw-voronoi-diagram		true	yes
feature-filter		""	yes
clause-filter		""	yes
ask		false	yes
describe		false	yes
select		false	yes
construct		false	yes
triple-patterns-count		true	yes
result-size		true	yes
join-vertices		true	yes
mean-join-vertices-degree		true	yes
mean-triple-pattern-selectivity		true	yes
bgps		true	yes
union		true	yes
filter		true	yes
optional		true	yes
distinct		true	yes
orderby		true	yes
groupby		true	yes
limit		true	yes
regex		true	yes
offset		true	yes
run-time		true	yes

## 2.3 DataGenerator

As the DataGenerator is currently implemented but not supported (it will be in future work) as it needs too much memory and too much time for even small datasets we decided to not describe how to use it and simply link to the paper how the datagenerator should be if it's finished. [1]

## 2.4 Testcase

The testcase interface is designed so the users can define their own testcases as well as use implemented ones. Testcases are a part of a benchmark, while all defined testcases represent the whole benchmark or at least a big part of a benchmark. Testcases needs to implement several functions. These are a

simple *start* method, a Method to get the results called *getResults()*, a Function to add Results which were derived previously for other connections, called *addCurrentResults(Collection;ResultSet;)*, further on to set the properties specified in the config *setProperty(Properties)*, the current Connection *setConnection(Connection)*, setting the XML-Node where the connections are specified *setConnectionNode(Node, String)*, set the current connection name *setCurrentDBName(String)* and the current dataset *setCurrentPercent(String)* and at last if it's a test which doesn't separate the results in datasets *isOneTest()* (this is needed for example in the *UploadShellTestcase*).

### 2.4.1 StressTestcase

The StressTestcase is designed to test one Connection at a time with several SPARQL workers and several UPDATE workers, those will simulate a real test-case scenario where several Users will request the connection with SPARQL queries while in the background some users will update the connections dataset. This testcase should simulate a real world scenario.

#### 2.4.1.1 Properties

name	description	default value	optional
sparql-users	number of sparql workers(can be 0)	-	no
queries-path	path or filename of the sparql queries	-	no
is-pattern	are the sparql queries templates/patterns or real queries	-	no
timelimit	time in ms the stresstest should run	-	no
latency-strategy[0-9]+	Latency strategy between queries/updates	-	yes
latency-amount[0-9]+	only with latency-strategy[0-9]+	-	yes

*sparql-users* is an integer which tells you how many sparql worker should be used. If this is greater than 0 than you need the *queries-path* which is the filename or path to the sparql queries. *is-pattern* is a boolean which tells iguana if the queries specified in *queries-path* are templates/pattern or query instances. The file needs to have one query per line. Every query will be validated. If a query can't be validated it will not be tested.

Query Templates can have variables which must be structure like `%%v([0-9]?)%%`. For example:

```
PREFIX dbo: <http://dbpedia.org/ontology/> ASK FROM <urn:graph> {
    ?x dbo:name %%v1%% ; dbo:country %%v2%%}
```

The variables in the template will be exchanged through variables. For example `%%v9%%` will be exchanged to `?v9`. After this the Prefixes will be cut out and the head will be removed, the template will look as following:

```
{ ?x dbo:name %%v1%% ; dbo:country %%v2%%}
```

Now the prefixes will be added again and a head with SELECT DISTINCT jall variables which should be exchanged; FROM jgraph-uri; Will be added. The user defined limit (default=5000) will be added/replaced and the final query will look like following:

```
PREFIX dbo: <http://dbpedia.org/ontology/> SELECT DISTINCT ?v1 ?
v2 FROM <urn:graph> { ?x dbo:name %%v1%% ; dbo:country %%v2
%%} LIMIT 5000
```

This query will now be requested against the reference connection. And all instances will be saved in one File.

*timelimit* tells IGUANA how long the testcase should run (in ms). Be aware that the testcase will run longer if the last queries/updates take longer. In other words: the last queries/update will be executed no matter what.

If you want to simulate network latency or any kind of latency you can add several latencies to the testcase. These will be defined as *latency-strategy*. Their are 3 *latency-strategys*. NONE, VARIABLE and FIXED. NONE is as simple as not being there. It is like a placeholder. FIXED will wait a given time for every query. VARIABLE will get a random number out of a given or calculated intervall every time before a query/update will be executed. This number will be wait. Everything is in ms. If there are more than one latencies, the sumation of all will be waited. You can add latencies by adding the property *latency-strategy0*. if you want more than one latencies simply add the property *latency-strategy1*, *latency-strategy2*, ... To set an amount or intervall, or let be calculate an intervall simply put for every *latency-strategy* you defined a *latency-amount* property with either an integer or an intervall as follows: "[n;m]" where n is the lower bound and m the upper bound. If you defined the strategy VARIABLE but simply put an integer in the related *latency-amount* an intervall will be calculated as follows assumed x is the related *latency-amount*:

$$[x - \lfloor \sqrt{x} \rfloor; x + \lfloor \sqrt{x} \rfloor] \quad (1)$$

For every *latency-strategy* you need a *latency-amount*. They are related by the follwing number. For example: *latency-strategy8* is related to *latency-amount8*.

name	description	default value	optional
update-users	number of update workers(can be 0)	-	no
update-strategy		NONE	yes
update-path		-	no
linking-strategy		DI	yes
worker-strategy[0-9]+		NEXT	yes

while you do not only have sparql-users and general properties there is also *update-users* which basically tells IGUANA how many update workers should be used. While they do need a path with update files you need to assign a *update-path*. This path can be relative.

The files in the *update-path* should have the following formats. First a 6 digit

number which tells the order of the files. Than if it's an insertion file: added or if it's a deletion file: removed. You can have related deletion and insertion files. For example a path can look as following:  
00001.added.nt, 000002.added.nt, 000002.removed.nt. The latter two files are related. This will be important for the *linking-strategy*.  
Further on you can define an *update-strategy* which is either NONE, FIXED or VARIABLE. Exactly the same as latency-strategy with one difference. The update-strategy uses the following equation as x:

$$x = \frac{timelimit}{|updatepath|} \quad (2)$$

You can't define an own amount.

Further on you can define a linking-strategy which can be DI, ID, D or I. D means it will first update every deletion files and then every insertion files. I means it will first update every insertion files than every deletion files, DI will update first deletion than its related insertion file. Sorted by their numbers. ID will do the same other wise around.

The *worker-strategy[0-9]+* tells every worker (worker0 has the worker strategy worker-strategy0 and so on) which files it should upload. Either it'll only update ADDED or REMOVED or it'll simply take the NEXT file it can get which is still not updated.

#### 2.4.1.2 Result Metrics

It will produce several results, with the metrics failed queries, succeded queries, Queries per second, Total time of queries, Query Mixes per TimeLimit for every worker and also their SPARQL, as well as UPDATE summation, as their means. Queries Per Second will measure for every query how many queries per second the connection can handle. Total time of Queries will sum up for every query how much time all request of this query took. Query Mixes per TimeLimit will measure how many queries the connection can handle in the given timelimit. Failed and succeded queries measures for every query the number of succeded and failed queries.

#### 2.4.2 FederatedStressTestcase

Same as StressTestcase but the update workers can work on different connections, so a Federated Connection can be tested without any problem. The actual endpoint will be the QueryEngine in front of the federated system and the update worker will get the connections behind the federated system. For example if a federated system has localhost:8080/sparql as the sparql endpoint for all connections and a virtuoso and a blazegraph connection is behind this federated system the update workers need to know into which of them they should upload. This you can specify



#### 2.4.2.1 Properties

Nearly as same as StressTestcase but with one more option and one difference

name	description	default value	optional
worker[0-9]+	worker0 tells the connection id into which the update worker0 needs to upload to	-	no
update-path[0-9]+	Tells the worker0 which update-path is assigned to the worker	-	no

Be aware that the update-path[0-9]+ must be the same path for every worker with the same connection id. For example a worker with the connection id "fuseki" and a different worker also with the connection id "fuseki" needs to have the same update-path!

#### 2.4.2.2 Result Metrics

See the result metrics at StressTestcase with one difference. The summation and mean of the UPDATE workers will be seperated. It will get the summation and the mean for every different connection.

#### 2.4.3 UploadShellTestcase

This testcase is there to give you the option to upload several datasets to the current connection via shell script. The script can have several arguments called "%%PERCENT%%", "%%DBNAME%%", "%%FILE%%" which will be replaced by the current percentage, the current connection name and the current file/path to upload to.

These arguments needs to stand in the script-name argument. For example: script-name="./upload.sh %%DBNAME%% %%FILE%%"

##### 2.4.3.1 Properties

name	description	default value	optional
file	Tells the file which should be uploaded	-	no
script-name	tells the name of the script which should be executed	-	no

##### 2.4.3.2 Result Metrics

This will produce only one resultset for the whole suite. It'll measure the time the shell script took for each percent and each connection.

## 3 Configuration

In this section we'll describe what you can configure in IGUANA and after this an examples The root element looks like follow:

```
<?xml version="1.0" encoding="UTF-8"?>
<iguana xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
...
</iguana>
```

### 3.1 databases

The databases element consists out of several database elements. It must exist 1 and only 1 time in the config

The database elements consist of two attributes, the id and the type. The type should always be *impl* It consists out of a required endpoint element and optional update endpoint, user and password. The parent is the root tag.

```
...
<!-- minimal occurs: 1 -->
<!-- maximal occurs: 1 -->
<databases main="">
  <!-- minimal occurs: 1 -->
  <!-- maximal occurs: n -->
  <database id="sparqlendpoint1" type="impl">
    <!-- required -->
    <endpoint uri="localhost:8080/sparql" />
    <!-- optional -->
    <update-endpoint uri="localhost:8080/update" />
    <!-- optional -->
    <user value="dba" />
    <!-- optional -->
    <pwd value="dba" />
  </database>
  ...
</databases>
...
```

### 3.2 logclustering

The LogClustering element needs 3 attributes, the class name, the path or the name of the log file and the name of the output queries file. Also it consists out of several properties with the attributes name and value. the parent is the root tag

```

...
<!-- mininmal occurs: 1 -->
<!-- maximal occurs: 1 -->
<log-clustering class="org.example.clusterer.Clusterer"
  path="logFile" output-file="queries.txt">
  <property name="ask" value="false"/>
  <property name="union" value="false"/>
  <property name="clause-filter"
    value="(OPTIONAL AND DISTINCT) OR (UNION)"/>
  <property name="feature-filter"
    value="ResultSize >= 5 AND ResultSize
      <= 100 AND BGPs >= 2 AND
      TriplePatterns <=10"/>
</log-clustering>
...

```

### 3.3 suite

The *suites* elements are the elements who contains all the information for your specific benchmark. You have several elements in the suite element. The number-of-triples element tells IGUANA if files which will be uploaded and removed should be splitted into files with the specified number of triples in it. These will be uploaded seperated. Be aware that IGUANA will only measure the time of each file to upload/remove and not the splitting etc. also. The graph-uri element tells IGUANA in which graph datasets should be loaded in. the parent is the root tag

```

...
<!-- mininmal occurs: 1 -->
<!-- maximal occurs: n -->
<suite>
  <!-- mininmal occurs: 0 -->
  <!-- maximal occurs: 1 -->
  <number-of-triples value="9"/>
  <!-- mininmal occurs: 1 -->
  <!-- maximal occurs: 1 -->
  <graph-uri name="http://dbpedia.org" />
  <!-- mininmal occurs: 1 -->
  <!-- maximal occurs: 1 -->
  <warmup time="5" file-name="warmup.txt" />
  <!-- mininmal occurs: 1 -->
  <!-- maximal occurs: 1 -->
  <random-function ...>
    ...
  </random-function>

```

```

        <!-- mininmal occurs: 1 -->
        <!-- maximal occurs: 1 -->
        <test-db ...>
            ...
        </test-db>
        <!-- mininmal occurs: 1 -->
        <!-- maximal occurs: 1 -->
        <testcases>
            ...
        </testcases>
    </suite>
...

```

### 3.3.1 randomfunction

the parent is the suite tag

```

...
    <!-- mininmal occurs: 1 -->
    <!-- maximal occurs: 1 -->
    <random-function type="RandomTriple" generate="false">
        <!-- mininmal occurs: 1 -->
        <!-- maximal occurs: n -->
        <percent value="1.0" file-name="dbpedia2/" />
        ...
    </random-function>
...

```

### 3.3.2 test-db

the parent is the suite tag

```

...
    <!-- mininmal occurs: 1 -->
    <!-- maximal occurs: 1 -->
    <test-db type="choose" reference="ref">
        <!-- mininmal occurs: 1 -->
        <!-- maximal occurs: n -->
        <db id="owlim" />
        ...
    </test-db>
...

```

### 3.3.3 testcases

the parent is the suite tag

```
...
    <!-- minimal occurs: 1 -->
    <!-- maximal occurs: 1 -->
    <testcases testcase-pre="pre.sh" testcase-post="post.sh">
        ...
        <!-- minimal occurs: 1 -->
        <!-- maximal occurs: n -->
        <testcase class="de.uni_leipzig.iguana.testcases.
            StressTestcase">
            <property name="xyz"
                value="abc"/>
            ...
        </testcase>
        ...
    </testcases>
...
```

### 3.4 Example

```
<?xml version="1.0" encoding="UTF-8"?>
<iguana xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <databases main="dbpedia">
    <database id="owlim" type="impl">
      <endpoint
        uri="localhost:8080/openrdf-workbench/repositories/owlim-
          lite/query" />
      <update-endpoint
        uri="localhost:8080/openrdf-workbench/repositories/owlim-
          lite/update" />
    </database>
    <database id="fuseki" type="impl">
      <endpoint uri="localhost:3030/tdb/sparql" />
      <update-endpoint uri="localhost:3030/tdb/update" />
    </database>
    <database id="blazegraph" type="impl">
      <endpoint uri="localhost:9999/bigdata/sparql" />
    </database>
    <database id="virtuoso" type="impl">
      <endpoint uri="localhost:8890/sparql-auth" />
      <user value="dba" />
      <pwd value="dba" />
    </database>
  </databases>
</iguana>
```

```

</database>
<database id="ref" type="impl">
  <endpoint uri="dbpedia.org/sparql" />
</database>
</databases>

<suite>
  <drop-db value="false" />
  <query-diversity value="2000" />
  <graph-uri name="http://dbpedia.org" />
  <random-function type="RandomTriple" generate="false">
    <percent value="1.0" file-name="dbpedia2/" />
  </random-function>
  <warmup time="20" file-name="warmup.txt" />
  <test-db type="choose" reference="ref">
    <db id="owlim" />
    <db id="blazegraph" />
    <db id="virtuoso" />
    <db id="fuseki" />
  </test-db>
  <testcases testcase-pre="./testcasePre.sh %DBID% %PERCENT% %
    TESTCASEID%"
    testcase-post="./testcasePost.sh %DBID% %PERCENT% %
    TESTCASEID%">
    <testcase class="de.uni_leipzig.iguana.testcases.
      StressTestcase">
      <property name="sparql-user" value="1" />
      <property name="update-user" value="0" />
      <property name="latency-amount0" value="20" />
      <property name="latency-strategy0" value="VARIABLE" />
      <property name="queries-path" value="queries-175.txt" />
      <property name="is-pattern" value="false" />
      <property name="timelimit" value="3600000" />
    </testcase>
    <testcase class="de.uni_leipzig.iguana.testcases.
      StressTestcase">
      <property name="sparql-user" value="2" />
      <property name="update-user" value="0" />
      <property name="latency-amount0" value="20" />
      <property name="latency-strategy0" value="VARIABLE" />
      <property name="queries-path" value="queries-175.txt" />
      <property name="is-pattern" value="false" />
      <property name="timelimit" value="3600000" />
    </testcase>
    <testcase class="de.uni_leipzig.iguana.testcases.
      StressTestcase">

```

```

    <property name="sparql-user" value="4" />
    <property name="update-user" value="0" />
    <property name="latency-amount0" value="20" />
    <property name="latency-strategy0" value="VARIABLE" />
    <property name="queries-path" value="queries-175.txt" />
    <property name="is-pattern" value="false" />
    <property name="timelimit" value="3600000" />
</testcase>
<testcase class="de.uni_leipzig.iguana.testcases.
    StressTestcase">
    <property name="sparql-user" value="8" />
    <property name="update-user" value="0" />
    <property name="latency-amount0" value="20" />
    <property name="latency-strategy0" value="VARIABLE" />
    <property name="queries-path" value="queries-175.txt" />
    <property name="is-pattern" value="false" />
    <property name="timelimit" value="3600000" />
</testcase>
<testcase class="de.uni_leipzig.iguana.testcases.
    StressTestcase">
    <property name="sparql-user" value="16" />
    <property name="update-user" value="0" />
    <property name="latency-amount0" value="20" />
    <property name="latency-strategy0" value="VARIABLE" />
    <property name="queries-path" value="queries-175.txt" />
    <property name="is-pattern" value="false" />
    <property name="timelimit" value="3600000" />
</testcase>
<testcase class="de.uni_leipzig.iguana.testcases.
    StressTestcase">
    <property name="sparql-user" value="1" />
    <property name="update-user" value="1" />
    <property name="latency-amount0" value="20" />
    <property name="latency-strategy0" value="VARIABLE" />
    <property name="queries-path" value="queries-175.txt" />
    <property name="is-pattern" value="false" />
    <property name="linking-strategy" value="ID" />
    <property name="update-path" value="ld" />
    <property name="worker-strategy0" value="ADDED" />
    <property name="update-strategy" value="VARIABLE" />
    <property name="timelimit" value="3600000" />
</testcase>
<testcase class="de.uni_leipzig.iguana.testcases.
    StressTestcase">
    <property name="sparql-user" value="2" />
    <property name="update-user" value="1" />

```

```

<property name="latency-amount0" value="20" />
<property name="latency-strategy0" value="VARIABLE" />
<property name="queries-path" value="queries-175.txt" />
<property name="is-pattern" value="false" />
<property name="linking-strategy" value="ID" />
<property name="update-path" value="ld" />
<property name="worker-strategy0" value="ADDED" />
<property name="update-strategy" value="VARIABLE" />
<property name="timelimit" value="3600000" />
</testcase>
<testcase class="de.uni_leipzig.iguana.testcases.
    StressTestcase">
  <property name="sparql-user" value="4" />
  <property name="update-user" value="1" />
  <property name="latency-amount0" value="20" />
  <property name="latency-strategy0" value="VARIABLE" />
  <property name="queries-path" value="queries-175.txt" />
  <property name="is-pattern" value="false" />
  <property name="linking-strategy" value="ID" />
  <property name="update-path" value="ld" />
  <property name="worker-strategy0" value="ADDED" />
  <property name="update-strategy" value="VARIABLE" />
  <property name="timelimit" value="3600000" />
</testcase>
<testcase class="de.uni_leipzig.iguana.testcases.
    StressTestcase">
  <property name="sparql-user" value="8" />
  <property name="update-user" value="1" />
  <property name="latency-amount0" value="20" />
  <property name="latency-strategy0" value="VARIABLE" />
  <property name="queries-path" value="queries-175.txt" />
  <property name="is-pattern" value="false" />
  <property name="linking-strategy" value="ID" />
  <property name="update-path" value="ld" />
  <property name="worker-strategy0" value="ADDED" />
  <property name="update-strategy" value="VARIABLE" />
  <property name="timelimit" value="3600000" />
</testcase>
<testcase class="de.uni_leipzig.iguana.testcases.
    StressTestcase">
  <property name="sparql-user" value="16" />
  <property name="update-user" value="1" />
  <property name="latency-amount0" value="20" />
  <property name="latency-strategy0" value="VARIABLE" />
  <property name="queries-path" value="queries-175.txt" />
  <property name="is-pattern" value="false" />

```



```
<property name="linking-strategy" value="ID" />
<property name="update-path" value="ld" />
<property name="worker-strategy0" value="ADDED" />
<property name="update-strategy" value="VARIABLE" />
<property name="timelimit" value="3600000" />
</testcase>
</testcases>
</suite>
</iguana>
```

## 4 Write your own ...

In this section we'll describe how you can write additional Modules in IGUANA without changing the code itself.

### 4.1 Testcase

We'll show you how you can write your own Testcase. The following testcase will test one in the properties given query 1000 times and measures the time. It will calculate the mean of the time of the queries as well as the total time.

```
import org.bio_gene.wookie.connection.Connection;
import de.uni_leipzig.iguana.utils.ResultSet;

public class OneQueryThousandTimesTestcase implements Testcase{

    private Collection<ResultSet> results = new LinkedList<ResultSet>();
    private String conName;
    private Connection con;
    private String percent;
    private String query;
    private int count = 1000;

    public void start() throws IOException{
        long time=0L;
        for(int i=0;i<count;i++){
            time+=con.selectTime(query);
        }
        ResultSet total = new ResultSet();
        total.setFileName("Total time of query");
        total.setTitle("Total time of query");
        total.setyAxis("time in ms");
        total.setxAxis("query");
        %TODO
    }

    public Collection<ResultSet> getResults(){
        return this.results;
    }

    public void addCurrentResults(Collection<ResultSet> currentResults){
        Iterator<ResultSet> it = results.iterator();
        for(ResultSet res : currentResults){
            ResultSet res2 = it.next();
            while(res.hasNext()){
                res2.addRow(res.next());
            }
        }
    }

    public void setProperties(Properties p){
        String query = p.getProperty("query");
    }
}
```

```

public void setConnection(Connection con){
    this.con = con;
}

public void setCurrentDBName(String name){
    this.conName = name;
}

public void setCurrentPercent(String percent){
    this.percent = percent;
}

public Boolean isOneTest(){
    return false;
}
}

```

## 4.2 LogClusterer

We'll show you how you can write your own Log Clusterer with the example of BorderFlow [<http://borderflow.sourceforge.net/>]

```

public class BorderFlowClusterer implements Clusterer{

    private String harden;
    private Double threshold;
    private Boolean testOne;
    private Boolean heuristic;
    private Boolean caching;
    private Integer minNodes;

    @Override
    public String cluster(String logPath, String queriesFile){

        String clusterOutput = "cluster.log";

        String sortedFreqFile = getSortedFrequency(logPath);
        String simFile = getSimilarities(sortedFreqFile);

        borderFlow(harden,
            threshold,
            testOne,
            heuristic,
            caching,
            minNodes,
            sortedFreqFile,
            simFile,
            clusterOutput,
            queriesFile);

        return queriesFile;
    }

    @Override
    public void setProperties(Properties p){

```

```

    harden = p.getProperty("harden");
    threshold = Double.valueOf(p.getProperty("threshold"));
    testOne = Boolean.valueOf(p.getProperty("test-one"));
    heuristic = Boolean.valueOf(p.getProperty("heuristic"));
    caching = Boolean.valueOf(p.getProperty("caching"));
    minNodes = Integer.valueOf(p.getProperty("min-nodes"));
}

private void borderFlow(String clusterHarden, double connThreshold,
    boolean testOne, boolean heuristic,
    boolean caching, Integer minNodes,
    String inputQueries, String input,
    String clusterOutput, String output)
    throws IOException{

    //This is the main class of the borderflow jar
    Main.borderFlowDemo(input,
        clusterOutput,
        connThreshold,
        testOne,
        heuristic,
        caching,
        HardenStrategy.valueOf(clusterHarden));
    rankAndChoose(inputQueries, clusterOutput, output, minNodes);
}

private void rankAndChoose(String input, String cluster,
    String output, String minNodes){

    /*
     * DO: rank the given input queries with their cluster bigger
     * than minNodes as you wish and write the choosen final
     * queries in the output file.
     */

}

private String getSortedFrequence(String logPath){

    /*
     * DO: Calculate Frequences of the queries in the logPath
     * Then sort them after their frequency
     */

}

private String getSimilarities(String sortedFreqFile){

    /*
     * DO: Calculate Similarity between the given queries
     * in the sortedFreqFile
     */

}

}

```

## 5 Further information

### 5.1 Links

Beware that these following information can be outdated

website	<a href="https://aksw.github.io/IGUANA/">https://aksw.github.io/IGUANA/</a>
distribution	<a href="https://github.com/AKSW/IGUANA/tree/master">https://github.com/AKSW/IGUANA/tree/master</a>
javadoc	<a href="https://aksw.github.io/IGUANA/javadoc">https://aksw.github.io/IGUANA/javadoc</a>
source code	<a href="https://github.com/AKSW/IGUANA">https://github.com/AKSW/IGUANA</a>
issue tracker	<a href="https://github.com/AKSW/IGUANA/issues">https://github.com/AKSW/IGUANA/issues</a>
contact information	<a href="mailto:mail2cpg@studserv.uni-leipzig.de">mail2cpg@studserv.uni-leipzig.de</a>
Border Flow (Clustering algorithm)	<a href="http://borderflow.sourceforge.net">http://borderflow.sourceforge.net</a>
DBpedia SPARQL Benchmark	<a href="http://aksw.org/Projects/DBPSB.html">http://aksw.org/Projects/DBPSB.html</a>
adjusted wookieConnection	<a href="https://github.com/AKSW/IGUANA/blob/develop/de.uni_leipzig.iguana/src/main/resources/lib/connection-0.0.1-SNAPSHOT.jar">https://github.com/AKSW/IGUANA/blob/develop/de.uni_leipzig.iguana/src/main/resources/lib/connection-0.0.1-SNAPSHOT.jar</a>

### 5.2 License

IGUANA itself is licensed under LGPL [A], but be aware that there are libraries we use which are licensed under different licenses.

## References

- [1] S. Duan, A. Kementsietsidis, K. Srinivas, and O. Udrea. Apples and oranges: A comparison of rdf benchmarks and real rdf datasets. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data*, SIGMOD '11, pages 145–156, New York, NY, USA, 2011. ACM.

## A LGPL

### GNU LESSER GENERAL PUBLIC LICENSE

Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <<http://fsf.org/>> Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

This version of the GNU Lesser General Public License incorporates the terms and conditions of version 3 of the GNU General Public License, supplemented by the additional permissions listed below.

#### 0. Additional Definitions.

As used herein, "this License" refers to version 3 of the GNU Lesser General Public License, and the "GNU GPL" refers to version 3 of the GNU General Public License.

"The Library" refers to a covered work governed by this License, other than an Application or a Combined Work as defined below.

An "Application" is any work that makes use of an interface provided by the Library, but which is not otherwise based on the Library. Defining a subclass of a class defined by the Library is deemed a mode of using an interface provided by the Library.

A "Combined Work" is a work produced by combining or linking an Application with the Library. The particular version of the Library with which the Combined Work was made is also called the "Linked Version".

The "Minimal Corresponding Source" for a Combined Work means the Corresponding Source for the Combined Work, excluding any source code for portions of the Combined Work that, considered in isolation, are based on the Application, and not on the Linked Version.

The "Corresponding Application Code" for a Combined Work means the object code and/or source code for the Application, including any data and utility programs needed for reproducing the Combined Work from the Application, but excluding the System Libraries of the Combined Work.

#### 1. Exception to Section 3 of the GNU GPL.

You may convey a covered work under sections 3 and 4 of this License without being bound by section 3 of the GNU GPL.

## 2. Conveying Modified Versions.

If you modify a copy of the Library, and, in your modifications, a facility refers to a function or data to be supplied by an Application that uses the facility (other than as an argument passed when the facility is invoked), then you may convey a copy of the modified version:

a) under this License, provided that you make a good faith effort to ensure that, in the event an Application does not supply the function or data, the facility still operates, and performs whatever part of its purpose remains meaningful, or

b) under the GNU GPL, with none of the additional permissions of this License applicable to that copy.

## 3. Object Code Incorporating Material from Library Header Files.

The object code form of an Application may incorporate material from a header file that is part of the Library. You may convey such object code under terms of your choice, provided that, if the incorporated material is not limited to numerical parameters, data structure layouts and accessors, or small macros, inline functions and templates (ten or fewer lines in length), you do both of the following:

a) Give prominent notice with each copy of the object code that the Library is used in it and that the Library and its use are covered by this License.

b) Accompany the object code with a copy of the GNU GPL and this license document.

## 4. Combined Works.

You may convey a Combined Work under terms of your choice that, taken together, effectively do not restrict modification of the portions of the Library contained in the Combined Work and reverse engineering for debugging such modifications, if you also do each of the following:

a) Give prominent notice with each copy of the Combined Work that the Library is used in it and that the Library and its use are covered by this License.

b) Accompany the Combined Work with a copy of the GNU GPL and this license document.

c) For a Combined Work that displays copyright notices during execution, include the copyright notice for the Library among these notices, as well as a reference directing the user to the copies of the GNU GPL and this license doc-



ument.

d) Do one of the following:

0) Convey the Minimal Corresponding Source under the terms of this License, and the Corresponding Application Code in a form suitable for, and under terms that permit, the user to recombine or relink the Application with a modified version of the Linked Version to produce a modified Combined Work, in the manner specified by section 6 of the GNU GPL for conveying Corresponding Source.

1) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (a) uses at run time a copy of the Library already present on the user's computer system, and (b) will operate properly with a modified version of the Library that is interface-compatible with the Linked Version.

e) Provide Installation Information, but only if you would otherwise be required to provide such information under section 6 of the GNU GPL, and only to the extent that such information is necessary to install and execute a modified version of the Combined Work produced by recombining or relinking the Application with a modified version of the Linked Version. (If you use option 4d0, the Installation Information must accompany the Minimal Corresponding Source and Corresponding Application Code. If you use option 4d1, you must provide the Installation Information in the manner specified by section 6 of the GNU GPL for conveying Corresponding Source.)

## 5. Combined Libraries.

You may place library facilities that are a work based on the Library side by side in a single library together with other library facilities that are not Applications and are not covered by this License, and convey such a combined library under terms of your choice, if you do both of the following:

a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities, conveyed under the terms of this License.

b) Give prominent notice with the combined library that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

## 6. Revised Versions of the GNU Lesser General Public License.

The Free Software Foundation may publish revised and/or new versions of the GNU Lesser General Public License from time to time. Such new versions

will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library as you received it specifies that a certain numbered version of the GNU Lesser General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that published version or of any later version published by the Free Software Foundation. If the Library as you received it does not specify a version number of the GNU Lesser General Public License, you may choose any version of the GNU Lesser General Public License ever published by the Free Software Foundation.

If the Library as you received it specifies that a proxy can decide whether future versions of the GNU Lesser General Public License shall apply, that proxy's public statement of acceptance of any version is permanent authorization for you to choose that version for the Library.