ICT Seventh Framework Programme (ICT FP7)

Grant Agreement No: 318497

Data Intensive Techniques to Boost the Real – Time Performance of Global Agricultural Data Infrastructures

D4.4b: Experimental Report on Projected Datasets

Deliverable Form	
Project Reference No.	ICT FP7 318497
Deliverable No.	D4.4b
Relevant Workpackage:	WP4: Rigorous Experimental Testing
Nature:	R
Dissemination Level:	PU
Document version:	2.0
Date:	2/3/2016
Authors:	NCSR-D, UAH
Document description:	This report documents experiments conducted on datasets contributed and prepared in WP2 using the research components developed in WP3.

Document History

Version	Date	Author (Partner)	Remarks
Draft v0.1	16/10/2015	NCSR-D, UAH	First draft circulated to consortium
Draft v0.2	30/10/2015	NCSR-D, UAH	Pre-final draft
Draft v0.9	3/11/2015	SWC	Internal review
Final v1.0	26/11/2015	NCSR-D, UAH	Submitted as D4.4
Draft v1.1	22/02/2016	NCSR-D	Section 2.3 and Table 2 extended with the experimental results and relevant discussion, for FedX/HiBiSCuS on SemaGrow data
Draft v1.2	29/02/2016	SWC	Internal review
Final v2.0	4/03/2016	NCSR-D	Submitted as D4.4b

EXECUTIVE SUMMARY

This report documents rigorous testing experiments using the *SemaGrow* research prototypes and based on the *SemaGrow* use cases and datasets.

These experiments go beyond the technical testing of the project's prototype components carried out in the WP3 tasks: the experiments reported here test the overall methodology of the system, its underlying assumptions, and well as the individual components in the context of the overall system.

Besides evaluating *SemaGrow* methods, these experiments also contribute to the development of the *automatic rigorous testing* components that periodically and automatically test deployments of the *SemaGrow Stack*.

TABLE OF CONTENTS

1. INTRODUCTION

1.1 Purpose and Scope

This report documents rigorous testing experiments using the *SemaGrow* research prototypes and based on the *SemaGrow* use cases and datasets.

These experiments go beyond the technical testing of the project's prototype components carried out in the WP3 tasks: the experiments reported here test the overall methodology of the system, its underlying assumptions, and well as the individual components in the context of the overall system.

1.2 Relation to other Work Packages and Deliverables

Work in this task receives:

- The experimental methodology (D4.1)
- Prototypes and infrastructure (WP3, WP5)
- Test data (WP2)

in order to provide:

- Evaluation of SemaGrow technologies

1.3 Big Data Aspects

This deliverable reports on experiments on both SemaGrow piloting datasets and on publicly available benchmarks. Specifically, besides the previously used FedBench (cf. D3.4), we have now experimented with its LargeRDFBench extension that increases the volume of data involved to more than 1 Gtriple (Table 1). Details about the experimental setup and the results can be found in the sections indicated on the first column.

Table 1: Dataset sizes

Collection	#triples	#subjects	#predicates	#objects
Reactive Resource Discovery (Sect 2)	289m	50m	1910	70m
Reactive Data Analysis (Sect 2)	271m	47m	1302	70m
LargeRDFBench (Sect 3)	1131m	215m	2757	367m
FedBench (previously reported in D3.4)	168m	19m	1659	54m

2. EXPERIMENTS ON SEMAGROW DATASETS

2.1 Objectives and Requirements

In this experiment, we compare the SemaGrow Stack engine with state-of-the-art SPARQL federation systems on the workloads produced during the pilots on the specific datasets. The objective of this experiment is to technically evaluate the performance of SemaGrow Stack engine in real-word workloads produced by project-specific requirements.

In the remainder of this chapter, we will document the experimental setup developed for this test, present and analyse results.

2.2 Experimental Setup

This experiment is based on the query logs collected during the second round of SemaGrow pilots.

The *Reactive Data Analysis* pilot (cf. Section 3, D6.2.2) consists of the federation of two databases: (a) the AGRIS bibliographic database on agricultural research and technology (cf. Section 3, Deliverable 2.2) and (b) the AGRIS crawler database that is populated periodically by crawling the web and automatically annotating documents with relevant AGROVOC terms. The main query of FAO's pilot case is to retrieve the most relevant crawled documents that share the same AGROVOC terms with a certain AGRIS document. The result of such queries can be consumed by a recommender system that can then enable the recommendation of most relevant documents with respect to a certain AGRIS document to users through a web widget. To achieve that, the recommender uses a SemaGrow federation of the aforementioned databases and issues queries consecutively produced by the following template:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dct: <http://purl.org/dc/terms/>

SELECT DISTINCT ?s (COUNT(DISTINCT ?o) as ?NELEMENTS)
WHERE {
  <http://agris.fao.org/aos/records/$id> dct:subject ?o .
  ?s dct:subject ?o.
  ?s rdf:type <http://semagrow.eu/rdf#CrawledDocument>.
}
GROUP BY ?s
ORDER BY DESC(?NELEMENTS)
LIMIT 6
```

The *Reactive Resource Discovery* pilot (cf. Section 4, D6.2.2) produced simpler queries that combine data from various federated sources. The federation includes (a) the AGRIS database that contains agricultural publications, (b) the AGROVOC database that contains the thematic taxonomy of AGROVOC terms, (c) the IFPRI database, (d) the Natural Europe dataset and (e) a database that is used for data cleansing purposes and in particular contain publication years for a subset of AGRIS publications. A typical query that is used during this pilot is the following:

```
SELECT ?s ?a  WHERE {
  ?s <http://www.semagrow.eu/rdf/year> ?a.
  ?s <http://purl.org/dc/terms/type> "Image".
  FILTER (?a < "2011"^^<http://www.w3.org/2001/XMLSchema#integer>).
  FILTER (?a > "2000"^^<http://www.w3.org/2001/XMLSchema#integer>).
}
LIMIT 10
```

The workload is produced via the AKstem microsite environment where customers can construct queries using a UI editor or freely from a SPARQL editor. We have randomly selected queries from the query logs produced by real usage during the pilots and compiled two experimental setups over the federations shown in Table 1.

We compare SemaGrow Stack with FedX [1], since the FedX federator is the one that outperforms most of the existing state-of-the-art systems [2]. For each experiment we use the live endpoints, which are (with the exception of AGRIS Dates) in remote servers on the internet. Both federation services can access the data sources via the SPARQL protocol. All experiments were performed on a Linux Desktop PC (Ubuntu 14.04 LTS) with Intel(R) Core(TM) i7-4790 CPU, 8 GB RAM.

Metadata used by SemaGrow was created using ELEON, a visual tool that allows data annotators to describe the content and technical specifications of the data sources that of a SemaGrow federation. On the other hand, FedX uses no statistics for its evaluation process.

2.3 Experimental Results

Table 3 shows the execution times of the first run and the average execution times of three consecutive runs for both SemaGrow and FedX on 20 randomly selected queries from the *Reactive Data Analysis* pilot workload. SemaGrow outperforms FedX with all queries. The main reason behind this vast difference of execution times is that SemaGrow constructs a more efficient execution plan than FedX. The integration of SemaGrow with the HiBiSCuS source selector gives more opportunities to SemaGrow's query planner to construct more efficient execution plans. In particular, HiBiSCuS' ability to perform join-aware source selection, prunes more efficiently sources that will not eventually provide tuples to the final result. However, the integration of FedX with HiBiSCuS does not yield any substantial performance increase. This is because HiBiSCuS improves source selection but does not improve any other aspect of the efficiency of the execution plan.

Table 4 shows the execution times of the first run and the average execution times of three runs for both SemaGrow and FedX, for all queries of the AgroKnow first pilot workload. In order to make the presentation of the results more manageable, we partitioned the full workload into 10 query sets, each one of them consisted of 10-12 queries. Each line corresponds to the average execution time of three runs of each query set. Even if the query have similar structure, the execution times can be vastly different between queries. However, both SemaGrow and FedX perform similarly, producing similar execution plans. One can notice that in simple and fast queries FedX might perform slightly better than SemaGrow, while, on the other hand, when the queries are more demanding in terms of execution, SemaGrow outperforms FedX.

Table 2 Datasets that participate in each federation experiment and their statistics

Collection	Dataset	#triples	#subjects	#predicates	#objects
Reactive Resource Discovery	AGRIS	253277276	42409056	1298	67889810
	AGROVOC	6145893	733449	129	1511748
	AGRIS Dates	28566199	6612999	11	242
	IFPRI	728392	71422	266	11769
	Natural Europe	643241	75853	206	208770
	SUM	**289m**	**50m**	**1910**	**70m**
Reactive Data Analysis	AGRIS	253277276	42409056	1298	67889810
	AGRIS Crawler	17320363	4330090	4	1640284
	SUM	**271m**	**47M**	**1302**	**70m**

Table 3: Query execution times (in milliseconds) for the Reactive Data Analysis workload

	First Run Execution Time			Average Execution Time		
Query	SemaGrow/HiBiSCuS	FedX	FedX/HiBiSCuS	SemaGrow/HiBiSCuS	FedX	FedX/HiBiSCuS
1	2880	656246	671197	2457	632329	671458
2	1189	T/O	T/O	1343	T/O	T/O
3	495	323507	325689	491	321881	325694
4	4784	T/O	T/O	3614	T/O	T/O
5	3130	722397	779817	2891	732046	784007
6	1529	658232	684106	1284	643393	692355
7	2939	463123	499339	2805	439603	498465
8	3210	426288	526669	2906	428229	542796
9	1834	208305	229537	1619	209956	229392
10	799	87327	89148	704	86431	88958
11	1537	167525	184582	1583	168960	184502
12	1320	275996	305384	1428	281624	305729
13	2688	489504	562083	2644	497729	574168
14	261	625	524	254	261	185
15	947	T/O	T/O	906	T/O	T/O
16	4625	403828	406371	3736	411202	402844
17	1450	173496	200890	1240	204292	200709
18	2603	335210	399287	2708	336520	397103
19	8334	576882	442134	6788	615381	445466
20	3636	286741	338756	3327	321712	338769

Table 4: Query execution times (in milliseconds) for the Reactive Resource Discovery workload

	Average Execution Time	
Query Set	SemaGrow	FedX
1	1080	1054
2	4098	3994
3	11225	18732
4	630	481
5	6271	9959
6	1037	836
7	13243	15253
8	46537	56748
9	15234	20225
10	1544	1253

3. EXPERIMENTS UNDER NETWORK DELAYS

3.1 Objectives and Requirements

The objective of this experiment is to compare the behaviour of the SemaGrow Stack with other state-of-the-art systems, in a more realistic environment where data sources can be subject to network delays. For that reason, we have developed a benchmark based on query sets from FedBench and LargeRDF Benchmark where we introduce simulated delays in each of the underlying data sources which is more appropriate for a real-life environment.

3.2 Experimental Setup

In this experiment we use the Cross-domain (CD) and Life Science (LS) queries of the FedBench benchmark and the Complex queries of the LargeRDFBench [4]. FedBench [3] is commonly used to evaluate the performance of SPARQL query federation systems. The benchmark is explicitly designed to represent SPARQL query federation on real-world datasets. The FedBench suite contains two domains: the Cross-domain (CD) and Life Science (LS). The benchmark queries resemble typical requests on these datasets and their structure ranges from simple star and chain queries to more complex graph patterns. Table 5 Query characteristics presents the various query characteristics.

Table 5 Query characteristics

Collection	Query	Type	# Triple Patterns	# Results
Cross Domain	CD1	Complex	3	90
	CD2	Star	3	1
	CD3	Chain-Star	5	2
	CD4	Complex	5	1
	CD5	Chain-Star	4	2
	CD6	Chain-Star	4	11
	CD7	Chain-Star	4	1
Life Sciences	LS1	Complex	2	1159
	LS2	Chain	3	333
	LS3	Chain-Star	5	9054
	LS4	Complex	7	3
	LS5	Complex	6	393
	LS6	Complex	5	28
	LS7	Complex	5	144
Complex	C1	Complex	8	1000
	C2	Complex	8	4
	C3	Complex	8	9
	C4	Complex	12	50
	C5	Complex	8	500
	C6	Complex	9	148
	C7	Complex	9	112
	C8	Complex	11	3067
	C9	Complex	9	100
	C10	Complex	10	102

We compare the SemaGrow Stack with FedX [1], since the FedX federator is the one that outperforms most of the existing state-of-the-art systems [2]. The endpoints of the experiment are deployed locally in different Virtuoso servers. All the federation engines can access the data sources via the SPARQL protocol. The statistics of each dataset used in the evaluation is presented in Table 6. SemaGrow uses VoID metadata that is generated by extracting statistics directly from the actual dataset. The VoID descriptions referred only to property and class partitions of the corresponding datasets. The federation in each case contains only the data sources that can potentially contribute to the queries of the collection.

Table 6: Dataset statistics used in FedBench and LargeRDFBench collection of queries.

Collection	Dataset	#triples	#subjects	#predicates	#objects	#types	#links
Cross Domain	Dbpedia subset	43.6M	9.50M	1063	13.6M	248	61.5k
	GeoNames	108M	7.48M	26	35.8M	1	118k
	LinkedMDB	6.15M	694k	222	2.05M	53	63.1k
	Jamendo	1.05M	336k	26	441k	11	1.7k
	New York Times	335k	21.7k	36	192k	2	31.7k
	SW Dog Food	104k	12K	118	37.5k	103	1.6k
Life Sciences	KEGG	1.09M	34.3k	21	939k	4	30k
	CheBI	7.33M	50.5k	28	772k	1	-
	Drugbank	767k	19.7k	119	276k	8	9.5k
	Dbpedia subset	43.6M	9.50M	1063	13.6M	248	61.5k
Complex	Dbpedia subset	43.6M	9.50M	1063	13.6M	248	61.5k
	GeoNames	108M	7.48M	26	35.8M	1	118k
	LinkedMDB	6.15M	694k	222	2.05M	53	63.1k
	Jamendo	1.05M	336k	26	441k	11	1.7k
	New York Times	335k	21.7k	36	192k	2	31.7k
	SW Dog Food	104k	12K	118	37.5k	103	1.6k
	KEGG	1.09M	34.3k	21	939k	4	30k
	CheBI	7.33M	50.5k	28	772k	1	N/A
	Drugbank	767k	19.7k	119	276k	8	9.5k
	LinkedTCGA-M	415M	83M	6	166M	1	-
	LinkedTCGA-E	344M	54.4M	7	84.4M	1	-
	LinkedTCGA-A	35M	5.7M	384	8.3M	23	251.3k
	Affymetrix	44.20M	44.2M	105	13.2M	3	246.3k
SUM		**1131M**	**215M**	**2757**	**367M**	**642**	**1070k**

In order to simulate the network delay of each of the above endpoints, we use the iprelay tool[1], which works by acting as a TCP proxy and can be used to shape the TCP traffic forwarded through it to a specified bandwidth. We are using the statistics available on endpoints registered in DataHub to calibrate this simulator. The actual statistics (if available) are shown in the first column of the following table and correspond to the run time of a cold JOIN query of 1000 results[2].

The corresponding iprelay parameters were calculated as follows: Firstly, we calculated the average size in bytes of a single result by dividing the size in bytes of all results of FedBench by the total number of the triples of the FedBench result, and by multiplying with 1000 we have the estimated size of the result set of the join queries. Therefore, the calculated bandwidth of each endpoint can be shown in the second column of the following table, which are the corresponding iprelay parameters. For the non-available statistics we use the average bandwidth.

3.3 Experimental Results

In the first experiment we calibrate the data source endpoints to act as if there is a limited throughput available and then execute FedBench query sets (CD, LS) and the Complex query set from LargeRDFBench. The results are depicted in Table 8. In some cases, both systems experience timeouts since the data sources that needed to be queried have delays that exceed the allowed time limit. In the simpler Cross Domain (CD) query set the performance of SemaGrow and FedX is comparable. In the Life Science (LS) query set, SemaGrow performs slightly better in cases where plans are equivalent, and vastly better when plans are different (e.g. LS7).

We now consider a single query of the whole query set and experiment with different throughputs to observe different execution times. We select LS3 since it is one query where all systems construct very similar plans, i.e. the order of the data sources to be contacted is the same.

Table 9 presents consecutive executions of the query LS3 for various setups of the throughput of the data source that is first contacted by the federators. SemaGrow and FedX, that both use a multithreading mechanism in order to continue do useful work in presence of slow data sources, exhibit similar behaviour. On the other hand, SPLENDID exhibits the worst performance of the three systems since its execution engine supports only pipeline parallelization and not support intra-operator parallelization. Therefore, a slow data source will block the remaining execution.

Table 7 Datasets and their corresponding throughput retrieved by sparqles service.

Dataset	Roundtrip latency (ms)	Bandwidth (bytes/s)
Dbpedia subset	112.71	1150000
GeoNames	N/A	
LinkedMDB	2.48	69172
Jamendo	0.53	25626
New York Times	N/A	
SW Dog Food	N/A	
KEGG	0.28	79310
CheBI	0.99	139394
Drugbank	6.8	1581
LinkedTCGA-M	N/A	
LinkedTCGA-E	N/A	
LinkedTCGA-A	N/A	
Affymetrix	19.81	8954

[1] Please cf. http://www.stewart.com.au/ip_relay/

[2] Please cf. http://sparqles.ai.wu.ac.at/

Table 8 Comparison of First Run and Average Overall Execution Time (in milliseconds)

		First Run Execution Time			Average Execution Time		
Collection	**Query**	**SemaGrow**	**FedX**	**SPLENDID**	**SemaGrow**	**FedX**	**SPLENDID**
Cross Domain	CD1	2306	630	44167	1031	493	16744
	CD2	280	255	19551	174	154	6879
	CD3	951	534	144309	665	633	50497
	CD4	1284	538	146572	1015	757	52196
	CD5	597	447	66478	485	414	23862
	CD6	12960	11211	431794	12044	11260	235722
	CD7	5985	6339	447988	5216	5813	205718
Life Sciences	LS1	156786	159211	174694	152659	156712	162563
	LS2	60622	68498	161539	57633	68628	97252
	LS3	T/O	T/O	T/O	T/O	T/O	T/O
	LS4	6579	11548	8699	3088	4644	6200
	LS5	275506	272649	T/O	267169	268644	T/O
	LS6	12830	T/O	T/O	8109	T/O	T/O
	LS7	2919	267353	476055	1011	278419	345332
Complex	C1	781812	1031945		853886	1031945	
	C2	284723	240437		275965	264622	
	C3	T/O	1107141		T/O	1103719	
	C4	20782	364832		20782	364832	
	C5	T/O	T/O		T/O	T/O	
	C6	T/O	T/O		T/O	T/O	
	C7	4310	14145		2791	12844	
	C8	92760	180706		89680	167685	
	C9	T/O	T/O		T/O	T/O	
	C10	145269	2900277		142915	271985	

Table 9 Execution times (in milliseconds) over different throughputs of the first data source

Throughput (bytes/sec)	**SemaGrow**	**FedX**	**SPLENDID**
1150000	45528	53027	963153
115000	47936	53900	953486
11500	154891	162213	1057264
1150	1352836	1375868	>3000000

4. REFERENCES

1. A. Schwarte, P. Haase, K. Hose, R. Schenkel, and M. Schmidt, “FedX: a federation layer for distributed query processing on linked open data,” Semantic Web Res. Appl., pp. 481–486, 2011.
2. M. Saleem, Y. Khan, A. Hasnain, I. Ermilov, and A. N. Ngomo, “A Fine-Grained Evaluation of SPARQL Endpoint Federation Systems,” vol. 1, pp. 1–5, 2009.
3. P. Haase, T. Mathäß, and M. Ziller, “An evaluation of approaches to federated query processing over linked data,” Proc. 6th Int. Conf. Semant. Syst. - I-SEMANTICS ’10, p. 1, 2010.
4. LargeRDFBench: https://code.google.com/p/bigrdfbench/