A METHOD FOR EVALUATION OF INFORMATIONAL SERVICES - STEP 3: RANK-BASED MULTIPLE COMPARISON

Krassimira Ivanova, Krassimir Markov, Stefan Karastanev

***Abstract**: Enhancing the hardware power does not cause linear enhancing of the informational services' performance. To discover the value of growth one has to test both source and enhanced systems running equal or similar services. If we need to discover the growth of services' performance for different computers' configurations we have to have common basis for comparing one software service with those of other systems which are tested on different computer configurations. In papers [Ivanova et al 2016a; Ivanova et al 2016b] the first and second steps of a method for solving such problem were presented. In this paper we outline the third step of the method. This step consists of the analysis of experiments: rank-based multiple comparison. All examples in the paper are based on results from real experiments presented in the [Markov et al, 2015].*

***Keywords**: Evaluation of informational services; Analysis of experiments: Rank-based multiple comparison.*

***ACM Classification Keywords**: H.3.4 Systems and Software - Performance evaluation (efficiency and effectiveness); H.3.5 Online Information Services.*

Introduction

Enhancing the hardware power does not cause linear enhancing of the informational services' performance. To discover the value of growth one has to test both source and enhanced systems running equal or similar services. If we need to discover the growth of services' performance for different computers' configurations we have to have common basis for comparing one software service with those of other systems which are tested on different computer configurations. In papers [Ivanova et al 2016a; Ivanova et al 2016b] the first and second steps of a method for solving such problem were presented. In this paper we outline the third step of the method. This step consists of the analysis of experiments: rank-based multiple comparison. All examples in the paper are based on results from real experiments presented in the [Markov et al, 2015].

The problem which has to be solved is to discover the growth of software performance for different computers' configurations if we have common basis for comparing one software system with same or other systems which are tested on different computer configurations. Enhancing the hardware power does not cause linear enhancing of the software performance. To discover the value of growth one has to test both source and enhanced systems running equal or similar software. Practically, the computers have different characteristics and operational systems. In addition, the target computers and operational systems may be not available for experiments but some benchmarks may be published.

As running example we use the problem to compare loading times for given datasets for different software systems in the next conditions:

- Program system **X** is tested on two computer configurations: **U** and **W**, where **W** is enhanced configuration in respect of **U**; program system **Y** is tested on different computer configuration **V** of the same class and similar characteristics as **U**. We have testing couples **(X,U)**, **(X,W)**, and **(Y,V)**;
- Computer configurations **U** and **W** are not available for testing and all work has to be done on computer configuration **V**;
- **X** has published results from tests on **U** by dataset **S1** with |**S1**| instances and on **W** with similar dataset **S2** with |**S2**| instances; **Y** is tested on configuration **V** by datasets **S1** and **S2**;
- Loading times are respectively: $\mathbf{L_{(X,U,S1)}}$, $\mathbf{L_{(X,W,S2)}}$, $\mathbf{L_{(Y,V,S1)}}$, $\mathbf{L_{(Y,V,S2)}}$.

The problem we have to solve is: "What will be the loading time of system **Y** if it will be run on computer configuration **W** with dataset **S2**?" i.e. $\mathbf{L_{(Y,W,S2)} = ?}$.

The methodology for solving this problem consists of three steps:

1. Computing the hardware proportionality constants;
2. Computing the software systems' performance and proportionality constants;
3. Analysis of experiments: Rank-based multiple comparison.

Further in this paper we describe the third point of methodology - Analysis of experiments: Rank-based multiple comparison. The experiments are based on real systems, data sets, and real as well as published benchmarks.

Experiments

We will compare a real RDF-data storing system **R** [RDFArM, 2015] with three other similar real RDF-stores:

- **V** [Virtuoso, 2013];
- **J** [Jena, 2016];
- **S** [Sesame, 2015],

Systems **V**, **J**, and **S** are tested by Berlin SPARQL Bench Mark (BSBM) team and connected to it research groups [Becker, 2008; BSBMv2, 2008; BSBMv3, 2009].

We provided experiments with *middle-size RDF-datasets*, based on selected real datasets from DBpedia [DBpedia, 2007a; DBpedia, 2007b] and artificial datasets created by BSBM Data Generator [BSBM DG, 2013; Bizer & Schultz, 2009].

The real middle-size RDF-datasets which we used consist of DBpedia's homepages and geocoordinates datasets with minor corrections [Becker, 2008]:

- ***H.nt*** (200,036 instances; 24 MB) Based on DBpedia's homepages.nt dated 2007-08-30 [DBpedia, 2007a]. 3 URLs that included line breaks were manually corrected (fixed for DBpedia 3.0);
- ***G.nt*** (447,517 instances; 64 MB) Based on DBpedia's geocoordinates.nt dated 2007-08-30 [DBpedia, 2007b]. Decimal data type URI was corrected (DBpedia bug #1817019; resolved).

The RDF stores feature different indexing behaviors: **S** automatically indexes after each import, while SDB and **V** allow for selective index activation which cause corresponded limitations or advantages. In order to make load times comparable, the data import by [Becker, 2008] was performed as follows:

- ***H.nt*** was imported with indexes enabled;
- ***G.nt*** was imported with indexes enabled.

In the case with **R** no parameters are needed. The data sets were loaded directly from the source N-triple files.

The artificial middle-size RDF-datasets are generated by BSBM Data Generator [BSBM DG, 2013] and published in N-triple as well as in Turtle format [BSBMv1, 2008; BSBMv2, 2008; BSBMv3, 2009]. We converted Turtle format in N-triple format using “rdf2rdf” program developed by Enrico Minack [Minack, 2010].

We have use four BSBM datasets – 50K, 250K, 1M, and 5M. Details about these datasets are summarized in following Table 1.

Table 1. Details about used artificial middle-size RDF-datasets

Name of RDF-dataset:	B50K	B250K	B1M	B5M
Exact Total Number of Instances:	50,116	250,030	1,000,313	5,000,453
File Size Turtle (unzipped)	14 MB	22 MB	86 MB	*1,4 GB*

Loading of B50K

R has loaded all **50116** instances from **B50K** for about **113 seconds** (112851 ms) or average time of **2.3 ms** per triple.

Number of Subjects in this dataset was S=4900; number of relations R=40; and number of objects O=50116.

This means that practically we had 40 layers with 4900 NL-locations (containers) which contain 50116 objects. The loading time' results from our experiment and [Bizer & Schultz, 2008] are given in Table 2.

Computer configurations and corresponded to them coefficients were given in [Ivanova et al, 2016a].

Benchmark configuration used by [Bizer&Schultz, 2008] is **Configuration B**.

Our benchmark configuration is **Configuration K**.

The loading times proportionality formula is

$$L_{(R,B,S2)} = L_{(R,K,S2)} * R_{R,K,B}, \text{ and } R_{R,K,B} = \mathbf{0.025729};$$

and we compute final loading time as follow: 113 * 0.025729= **2.91 sec**.

Table 2. Benchmark results for B50K

system	loading time in seconds
S	3
J SDB	5
V	2
R	**3**

From Table 2 we may conclude that **V** has the best loading time for **B50K. R** has same loading time as **S** and 40% better performance than **J**.

Loading of H.nt

R has loaded all **200036** instances from ***H.nt*** for about **727 seconds** (727339 ms) or average time of **3.6 ms** per triple. More detailed information is given in Table 3. Every row of this table contains data for storing of one hundred thousand instances. Total stored instances were 200036 and Table 3 contains three rows.

Table 3. Results for loading times of H.nt by **R**

part	instances stored	ms for all	ms for one	Subjects	Relations	Objects
1	100000	360955	3.6	100000	1	100000
2	100000	366275	3.7	100000	1	100000
3	36	109	3.0	36	1	36
Total:	**200036**	**727339**	**3.6**	**200036**	**1**	**200036**

Number of Subjects in this dataset was S=200036; number of relations R=1; and number of objects O=200036.

This means that practically we had only one layer with 200036 NL-locations (containers) which contain the same number of objects. The loading time' results from our experiment and [Becker, 2008] are given in Table 4.

Benchmark configuration used by [Becker, 2008] is **Configuration A**.

Our benchmark configuration is **Configuration K**.

The loading times proportionality formula is

$$L_{(R,A,S2)} = H_{AK} * L_{(R,K,S2)}, \text{ where } H_{AK} = 3.125;$$

and we compute final loading time as follow: 727 x 3.125 = **2271.875** sec.

Table 4. Benchmark results for H.nt

system	loading time in seconds
V	1327
J V1	5245
J V2	3557
J V3	9681
S	2404
R	**2272**

From Table 4 we may conclude that **V** has the best time (about 42% better result than **R**); **R** has about 5% better time than **S** and 36% better time than **J** (we take in account only the best results of compared systems, in this case – **J**).

Loading of B250K

R has loaded all **250030** instances from **B250K** for about **575 seconds** (575069 ms) or average time of **2.3 ms** per triple.

More detailed information is given in 0. Every row of this table contains data for storing of one hundred thousand instances. Total stored instances were 250030 and 0 contains three rows.

Table 5. Results for loading times of B250K by **R**

part	instances stored	ms for all	ms for one	Subjects	Relations	Objects
1	100000	238525	2.4	19854	6	100000
2	100000	228854	2.3	26505	22	100000
3	50030	107690	2.1	14525	22	50030
Total:	**250030**	**575069**	**2.3**	**60884**	**22**	**250030**

Number of Subjects in this dataset was S=60884; number of relations R=22; and number of objects O=250030.

This means that practically we had 22 layers with 60884 NL-locations (containers) which contain 250030 objects. The loading time' results from our experiment and [BSBMv2, 2008] are given in Table 6.

Benchmark configuration used by [BSBMv2, 2008] is **Configuration B**.

Our benchmark configuration is **Configuration K**.

The loading times proportionality formula is

$$\mathbf{L_{(R,B,S2)}} = \mathbf{L_{(R,K,S2)}} * \mathbf{R_{R,K,B}}, \text{ and } \mathbf{R_{R,K,B}} = \mathbf{0.025729};$$

and we compute final loading time as follow: 575 x 0.025729= **14.79 sec**.

Table 6. Benchmark results for B250K

system	loading time in seconds
S	19
J TDB	13
V TS	05
V RDF views	09
V SQL	09
R	**14.79**

From Table 6 we may conclude that **V** has 66% and **J** has 12% better performance than **R**. **R** has 22% better performance than **S**.

Loading of G.nt

R has loaded all 447517 instances from **G.nt** for about 1110 seconds (1110415 ms) or average time of 2.5 ms per triple.

More detailed information is given in Table 7 Every row of this table contains data for storing of one hundred thousand instances. Total stored instances were 447517 and Table 7 contains five rows.

Table 7. Results for loading times of G.nt by **R**

part	instances stored	ms for all	ms for one	Subjects	Relations	Objects
1	100000	244453	2.4	34430	6	100000
2	100000	246747	2.5	34909	6	100000
3	100000	245530	2.5	33863	6	100000
4	100000	248198	2.5	33678	6	100000
5	47517	47517	2.6	16095	6	47517
Total:	**447517**	**1110415**	**2.5**	**152975**	**6**	**447517**

Number of Subjects in this dataset was S=152975; number of relations R=6; and number of objects O=447517.

This means that practically we had six layers with 152975 NL-locations (containers) which contain 447517 objects, i.e. some containers in some layers are empty. The loading time' results from our experiment and [Becker, 2008] are given in **Error! Reference source not found.** and **Error! Reference source not found.**.

Benchmark configuration used by [Becker, 2008] is **Configuration A**.

Our benchmark configuration is **Configuration K**.

The loading times proportionality formula is

$$L_{(R,A,S2)} = H_{AK} * L_{(R,K,S2)}, \text{ where } H_{AK} = \mathbf{3.125};$$

and we compute final loading time as follow: 1110 x 3.125= **3468.75 sec**.

Table 8. Benchmark results for G.nt

system	loading time in seconds
V	1235
J V1	6290
J V2	3305
J V3	9640
S	2341
R	**3469**

From Table 8 we may conclude that **R** has the worst performance (we take the best time of **J**). **V** has 64%, **S** has 33%, and **J** has 5% better performance.

Loading of B1M

R has loaded all **1000313** instances from **B1M** for about **2349 seconds** (2349328 ms) or average time of **2.3 ms** per triple.

More detailed information is given in Table 9. Every row of this table contains data for storing of one hundred thousand instances. Total stored instances were 1000313 and Table 9 contains 11 rows. This table has new structure. It contains number of stored instances to corresponded part including it and in separate columns the time for storing the last 100000 instances and average time for one triple from this part.

Table 9. Results for loading times of B1M by **R**

part	instances stored	ms for all	ms for one	ms for last 100000	ms for one	Subjects	Relations	Objects
1	100000	241099	2.4	241099	2.4	6859	22	100000
2	200000	480265	2.4	239166	2.4	14363	29	200000
3	300000	714453	2.4	234188	2.3	24365	29	300000
4	400000	962994	2.4	248541	2.5	34366	29	400000
5	500000	1194344	2.4	231350	2.3	44368	29	500000
6	600000	1423665	2.4	229321	2.3	54370	29	600000
7	700000	1655420	2.4	231755	2.3	64324	40	700000
8	800000	1892074	2.4	236654	2.4	73799	40	800000
9	900000	2116590	2.4	224516	2.2	83269	40	900000
10	1000000	2348501	2.3	231911	2.3	92729	40	1000000
11	1000313	2349328	2.3	827	2.6	92757	40	1000313

Number of Subjects in this dataset was S=92757; number of relations R=40; and number of objects O=1000313.

This means that practically we had 40 layers with 92757 NL-locations (containers) which contain 1000313 objects. The loading time' results from our experiment and [BSBMv2, 2008; BSBMv3, 2009] are given in Table 10.

Benchmark configuration used by [BSBMv2, 2008; BSBMv3, 2009] is **Configuration B**.

Our benchmark configuration is **Configuration K**.

The loading times proportionality formula is

$$L_{(R,B,S2)} = L_{(R,K,S2)} * R_{R,K,B}, \text{ and } R_{R,K,B} = \mathbf{0.025729};$$

and we compute final loading time as follow: 2349 x 0.025729 = **60.437421 sec**.

Table 10. Benchmark results for B1M

system	loading time in min:sec	
	(a) [BSBMv2, 2008]	(b) [BSBMv3, 2009]
S	02:59	03:33
J TDB	00:49	00:41
J SDB	02:09	-
V TS	00:23	00:25
V RV	00:34	00:33
V SQL	00:34	00:33
R	**01:00**	**01:00**

From Table 10 we may conclude that **V** has 62% and **J** has 32% better performance than **R**. **R** has 67% better performance than **S**.

Loading of B5M

R has loaded all **5000453** instances from **B5M** for about **11704 sec.** (11704116 ms) or average time of **2.3 ms** per triple.

Number of Subjects in this dataset was S=458142; number of relations R=55; and number of objects O=5000453.

This means that practically we had 55 layers with 458142 NL-locations (containers) which contain 5000453 objects. The loading time' results from our experiment and [Bizer & Schultz, 2008] are given in Table 12.

More detailed information is given in Table 11. Every row of this table contains data for storing of one hundred thousand instances. Total stored instances were 5000453 and Table 11 contains 51 rows.

This table contains number of stored instances to corresponded part including it and in separate columns the time for storing the last 100000 instances and average time for one triple from this part.

Table 11. Results for loading times of B5M by **R**

part	instances stored	ms for all	ms for one	ms for last 100000	ms for one	Subjects	Relations	Objects
1	100000	250023	2.5	250023	2.5	5463	22	100000
2	200000	506660	2.5	256637	2.6	7973	22	200000
3	300000	751254	2.5	244594	2.4	10471	22	300000
4	400000	983196	2.5	231942	2.3	12974	22	400000
5	500000	1227104	2.5	243908	2.4	22353	29	500000
6	600000	1468063	2.4	240959	2.4	32357	29	600000
7	700000	1708663	2.4	240600	2.4	42360	29	700000
8	800000	1956034	2.4	247371	2.5	52363	29	800000
9	900000	2190644	2.4	234610	2.3	62366	29	900000
10	1000000	2430043	2.4	239399	2.4	72369	29	1000000
11	1100000	2666041	2.4	235998	2.4	82372	29	1100000

12	1200000	2910230	2.4	244189	2.4	92375	29	1200000
13	1300000	3143529	2.4	233299	2.3	102377	29	1300000
14	1400000	3371618	2.4	228089	2.3	112381	29	1400000
15	1500000	3605136	2.4	233518	2.3	122384	29	1500000
16	1600000	3838139	2.4	233003	2.3	132387	29	1600000
17	1700000	4070830	2.4	232691	2.3	142390	29	1700000
18	1800000	4298155	2.4	227325	2.3	152393	29	1800000
19	1900000	4527367	2.4	229212	2.3	162396	29	1900000
20	2000000	4758030	2.4	230663	2.3	172399	29	2000000
21	2100000	4985698	2.4	227668	2.3	182402	29	2100000
22	2200000	5212742	2.4	227044	2.3	192405	29	2200000
23	2300000	5439692	2.4	226950	2.3	202408	29	2300000
24	2400000	5685347	2.4	245655	2.5	212043	40	2400000
25	2500000	5922328	2.4	236981	2.4	221512	40	2500000
26	2600000	6155331	2.4	233003	2.3	230972	40	2600000
27	2700000	6391610	2.4	236279	2.4	240447	40	2700000
28	2800000	6630417	2.4	238807	2.4	249912	40	2800000
29	2900000	6855511	2.4	225094	2.3	259371	40	2900000

30	3000000	7078545	2.4	223034	2.2	268831	40	3000000
31	3100000	7305979	2.4	227434	2.3	278290	40	3100000
32	3200000	7533928	2.4	227949	2.3	287754	40	3200000
33	3300000	7773608	2.4	239680	2.4	297240	40	3300000
34	3400000	8006782	2.4	233174	2.3	306704	40	3400000
35	3500000	8239629	2.4	232847	2.3	316145	40	3500000
36	3600000	8464536	2.4	224907	2.2	325609	40	3600000
37	3700000	8693202	2.3	228666	2.3	335077	40	3700000
38	3800000	8919248	2.3	226046	2.3	344557	40	3800000
39	3900000	9150254	2.3	231006	2.3	354009	40	3900000
40	4000000	9383912	2.3	233658	2.3	363472	40	4000000
41	4100000	9616120	2.3	232208	2.3	372924	40	4100000
42	4200000	9850090	2.3	233970	2.3	382383	40	4200000
43	4300000	10073842	2.3	223752	2.2	391847	40	4300000
44	4400000	10305832	2.3	231990	2.3	401308	40	4400000
45	4500000	10536619	2.3	230787	2.3	410763	40	4500000
46	4600000	10769997	2.3	233378	2.3	420233	40	4600000

47	4700000	11004030	2.3	234033	2.3	429699	40	4700000
48	4800000	11242836	2.3	238806	2.4	439169	40	4800000
49	4900000	11474107	2.3	231271	2.3	448643	40	4900000
50	5000000	11702852	2.3	228745	2.3	458099	40	5000000
51	5000453	11704116	2.3	1264	2.8	458142	55	5000453

Benchmark configuration used by [Bizer & Schultz, 2008] is **Configuration B**.

Our benchmark configuration is **Configuration K**.

The loading times proportionality formula is

$$L_{(R,B,S2)} = L_{(R,K,S2)} * R_{R,K,B}, \text{ and } R_{R,K,B} = \mathbf{0.025729};$$

and we compute final loading time as follow: 11704 * 0.025729= **301.13** sec.

Table 12. Benchmark results for B5M

system	loading time in seconds
S	1988
J	1053
V	609
R	**301**

From Table 12 we may conclude that **R** has best loading time (better about 85% than **S**, 71% than **J**, and 51% than **V**).

Experiments with large datasets

We provided experiments with real large datasets which were taken from DBpedia's homepages [DBpedia, 2007c] and Billion Triple Challenge (BTC) 2012 [BTC, 2012].

The real dataset from DBpedia's ***l.nt*** (15,472,624 instances; 2.1 GB) is based on DBpedia's *infoboxes.nt* dated 2007-08-30 [DBpedia, 2007c]. 166 instances from the original set were excluded because they contained excessively large URIs *(> 500 characters)* that caused importing problems with **V** (DBpedia bug #1871653). **R** has no such limitation. **l.nt** was imported with indexes initially disabled in **V**. Indexes were then activated and the time required for index creation time was factored into the import time. In the case with **R** no parameters are needed. The datasets were loaded directly from the source file.

The RDF stores feature different indexing behaviors: **S** automatically indexes after each import, while SDB and **V** allow for selective index activation.

Artificial large datasets are taken from Berlin SPARQL Bench Mark (BSBM) [Bizer & Schultz, 2009; BSBMv3, 2009; BSBMv5, 2009; BSBMv6, 2011]. Details about the benchmark artificial datasets are summarized in the following Table 13:

Table 13. Details about artificial large RDF-datasets

Number of Instances	25M	100M
Exact Total Number of Instances	25000244	100000112
File Size Turtle (unzipped)	*2.1 GB*	8.5 GB

Loading of l.nt

R has loaded all 15 472 624 instances from **l.nt** for about 43652 seconds (43652528 ms) or average time of 2.8 ms per triple.

Detailed information is not given here because of the size of the table with results. Every row of this table contains data for storing of one hundred thousand instances and total number of stored instances is 15,472,624 and table contains 155 rows.

Number of Subjects in this dataset was S=1354298; number of relations R=56338; and number of objects O=15472624.

This means that practically we had 56338 layers with 1354298 NL-locations (containers) which contain 15472624 objects, i.e. some containers in some layers are empty. The loading time' results from our experiment and [Becker, 2008] are given in Table 14.

Benchmark configuration used by [Becker, 2008] is **Configuration A**.

Our benchmark configuration is **Configuration K**.

The loading times proportionality formula is
$L_{(\mathbf{R},A,S2)} = H_{AK} * L_{(\mathbf{R},K,S2)}$, where H_{AK} =3.125;
and we compute final loading time as follow: 43652 x 3.125= **136412.5** sec.

Table 14. Benchmark results for I.nt

system	loading time in seconds
V	7017
J Variant 1	70851
J Variant 2	73199
J Variant 3	734285
S	21896
R	**136412**

From Table 14 we may conclude that **R** has the worst loading time. **V** is 95%, **S** is 84%, and **J** is 48% better than **R** (we take in account only the best results of compared systems).

Loading of B25M

R has loaded all **25000244** instances from **B25M** for about **56488** seconds (56488509ms) or average time of **2.3** ms per triple.

Number of Subjects in this dataset was S=2258132; number of relations R=112; and number of objects O=25000244.

This means that practically we had 112 layers with 2258132 NL-locations (containers) which contain 25000244 objects, i.e. some containers in some layers are empty.

The loading time' results from our experiment and [Bizer & Schultz, 2009; BSBMv3, 2009] are given in Table 15.

Benchmark configuration used by [Bizer & Schultz, 2009; BSBMv3, 2009] is **Configuration B**. Our benchmark configuration is **Configuration K**.

The loading times proportionality formula is

$$L_{(R,B,S2)} = L_{(R,K,S2)} * R_{R,K,B}, \text{ and } R_{R,K,B} = \mathbf{0.025729}.$$

We compute final loading time as follow: 56488*0.025729= **1453.38** sec.

Table 15. Benchmark results for B25M

system	loading time in seconds
S	44225
J TDB	1013
J SDB	14678
V TS	2364
V RV	1035
V SQL	1035
R	**1453**

From Table 15 we may conclude that **J** (with 30%) and **V** (with 29%) are better than **R**. **R** has 97% better performance than **S**.

Loading of B100M and BSBM 200M

R has loaded all **100000112** instances from **B100M** for about **229344** seconds (229343807 ms) or average time of **2.3** ms per triple.

Number of Subjects in this dataset was S=9034046; number of relations R=341; and number of objects O=100000112.

This means that practically we had 341 layers with 9034046 NL-locations (containers) which contain 100000112 objects, i.e. some containers in some layers contain more than one object. The loading time' results from our experiment and [Bizer & Schultz, 2009; BSBMv3, 2009] are given in Table 16.

Benchmark configuration used by [Bizer & Schultz, 2009; BSBMv3, 2009] is **Configuration B**.

Our benchmark configuration is **Configuration K**.

The loading times proportionality formula is

$$\mathbf{L_{(R,B,S2)}} = \mathbf{L_{(R,K,S2)}} * \mathbf{R_{R,K,B}}, \text{ and } \mathbf{R_{R,K,B}} = \mathbf{0.025729}.$$

We compute final loading time as follow:

229344 * 0.025729 = **5900.79** sec.

Table 16. Benchmark results for B100M

system	loading time in seconds
S	282455
J TDB	5654
J SDB	139988
V TS	28607
V RV	3833
V SQL	3833
R	**5901**

From Table 16 we may conclude that **V** is 35% better than **R** and **J** is 4% better than **R**. **R** is 98% better than **S**.

Analysis of experiments: Rank-based multiple comparison

We have provided experiments with middle-size and large RDF-datasets, based on selected datasets from DBpedia's homepages and Berlin SPARQL Bench Mark (BSBM) to make comparison with published benchmarks of known RDF triple stores. Result from Rank-based multiple comparison is discussed below.

We used the Friedman test to detect statistically significant differences between the systems [Friedman, 1940]. The Friedman test is a non-parametric test, based on the ranking of the systems on each dataset. It is equivalent of the repeated-measures ANOVA [Fisher, 1973]. We used Average Ranks ranking method, which is a simple ranking method, inspired by Friedman's statistic [Neave & Worthington, 1992]. For each dataset the systems are ordered according to the time measures and are assigned ranks accordingly. The best system receives rank 1, the second – 2, etc. If two or more systems have equal value, they receive equal rank which is mean of the virtual positions that had to receive such number of systems if they were ordered consecutively each by other.

Let n is the number of observed datasets; k is the number of systems.

Let i_{rj} be the rank of system j on dataset i. The average rank for each system is calculated as

$$R_j = \frac{1}{n}\sum_{i=1}^{k} r_j^i .$$

The null-hypothesis states that if all the systems are equivalent than their ranks R_j should be equal. When null-hypothesis is rejected, we can proceed with the Nemenyi test [Nemenyi, 1963] which is used when all systems are compared to each other. The performance of two systems is significantly different if the corresponding average ranks differ by at least the critical difference

$$CD = q_\alpha \sqrt{\frac{k(k+1)}{6N}}$$

where critical values q_α are based on the Studentized range statistic divided by $\sqrt{2}$. Some of the values of q_α are given in Table 17 [Demsar, 2006].

Table 17. Critical values for the two-tailed Nemenyi test

systems	2	3	4	5	6	7	8	9	10
$q_{0.05}$	1.960	2.343	2.569	2.728	2.850	2.949	3.031	3.102	3.164
$q_{0.10}$	1.645	2.052	2.291	2.459	2.589	2.693	2.780	2.855	2.920

The results of the Nemenyi test are shown by means of critical difference diagrams.

Experiments which we will take in account were presented in corresponded tables of above as follow (Table 18):

Table 18. Information about tests and results

test No:	results
1	Table 2
2	Table 4
3	Table 6
4	Table 8
5a	Table 10
5b	Table 10
6	Table 12
7	Table 14
8	Table 15
9	Table 16

Benchmark values from our 12 experiments and corresponded published experimental data from BSBM team are given in Table 19 Published results do not cover all table, i.e. we have no values for some cells. To solve this problem we will take in account only the best result for given system on concrete datasets (Table 20). **S** had no average values for tests 10a and 10b. Because of this we will not use these test in our comparison. They were useful to see the need of further refinement of **R** for big data.

The ranks of the systems for the ten tests are presented below in Table 21.

Table 19. Benchmark values for middle size datasets

system	**TEST**									
	1	**2**	**3**	**4**	**5a**	**5b**	**6**	**7**	**8**	**9**
R	3	2272	14.79	3469	60	60	301	136412	1453	5901
S	3	2404	19	2341	179	213	1988	21896	44225	282455
V	2	1327		1235			609	7017		
V TS			05		23	25			2364	28607
V RDF			09							
V SQL			09		34	33			1035	3833
V RV					34	33			1035	3833
J SDB	5		13		129		1053		14678	139988
J TDB					49	41			1013	5654
J V1		5245		6290				70851		
J V3		3557		3305				73199		
J V2		9681		9640				734285		

Table 20. Chosen benchmark values for middle size datasets

	TEST									
system	**1**	**2**	**3**	**4**	**5a**	**5b**	**6**	**7**	**8**	**9**
R	3	2272	14.79	3469	60	60	301	136412	1453	5901
S	3	2404	19	2341	179	213	1988	21896	44225	282455
V	2	1327	05	1235	23	25	609	7017	1035	3833
J	5	3557	13	3305	49	41	1053	70851	1013	5654

Table 21. Ranking of tested systems

system	**ranks for the tests**										**average rank**
	1	**2**	**3**	**4**	**5a**	**5b**	**6**	**7**	**8**	**9**	
R	2.5	2	3	4	3	3	1	4	3	3	**2.85**
S	2.5	3	4	2	4	4	4	2	4	4	**3.35**
V	1	1	1	1	1	1	2	1	2	1	**1.2**
J	4	4	2	3	2	2	3	3	1	2	**2.6**

All average ranks are different. The null-hypothesis is rejected and we can proceed with the Nemenyi test. Following [Demsar, 2006], we may compute the critical difference by formula:

$$CD = q_{\alpha}\sqrt{\frac{k(k+1)}{6N}}$$

where q_{α} we take as $q_{0.10}$ = 2.291 (from Table 17 [Demsar, 2006; Table 5a]); *k* will be the number of systems compared, i.e. *k*=4; N will be the number of datasets used in benchmarks, i.e. N=10. This way we have:

$$CD_{0.10} = 2.291 * \sqrt{\frac{4*5}{6*10}} = 2.291 * \sqrt{\frac{20}{60}} = 2.291 * 0.577 = 1.322$$

We will use for critical difference $CD_{0.10}$ the value 1.322.

At the end, average ranks of the systems and distance to average rank of the first one are shown in Table 22.

Table 22. Average ranks of systems and distance to average rank of the first one

place	system	average rank	Distance between average rank of the every system and average rank of the first one
1	**V**	**1.2**	**0**
2	**J**	**2.6**	**1.4**
3	**R**	**2.85**	**1.65**
4	**S**	**3.35**	**2.15**

The visualization of Nemenyi test results for tested systems is shown on Figure 1.

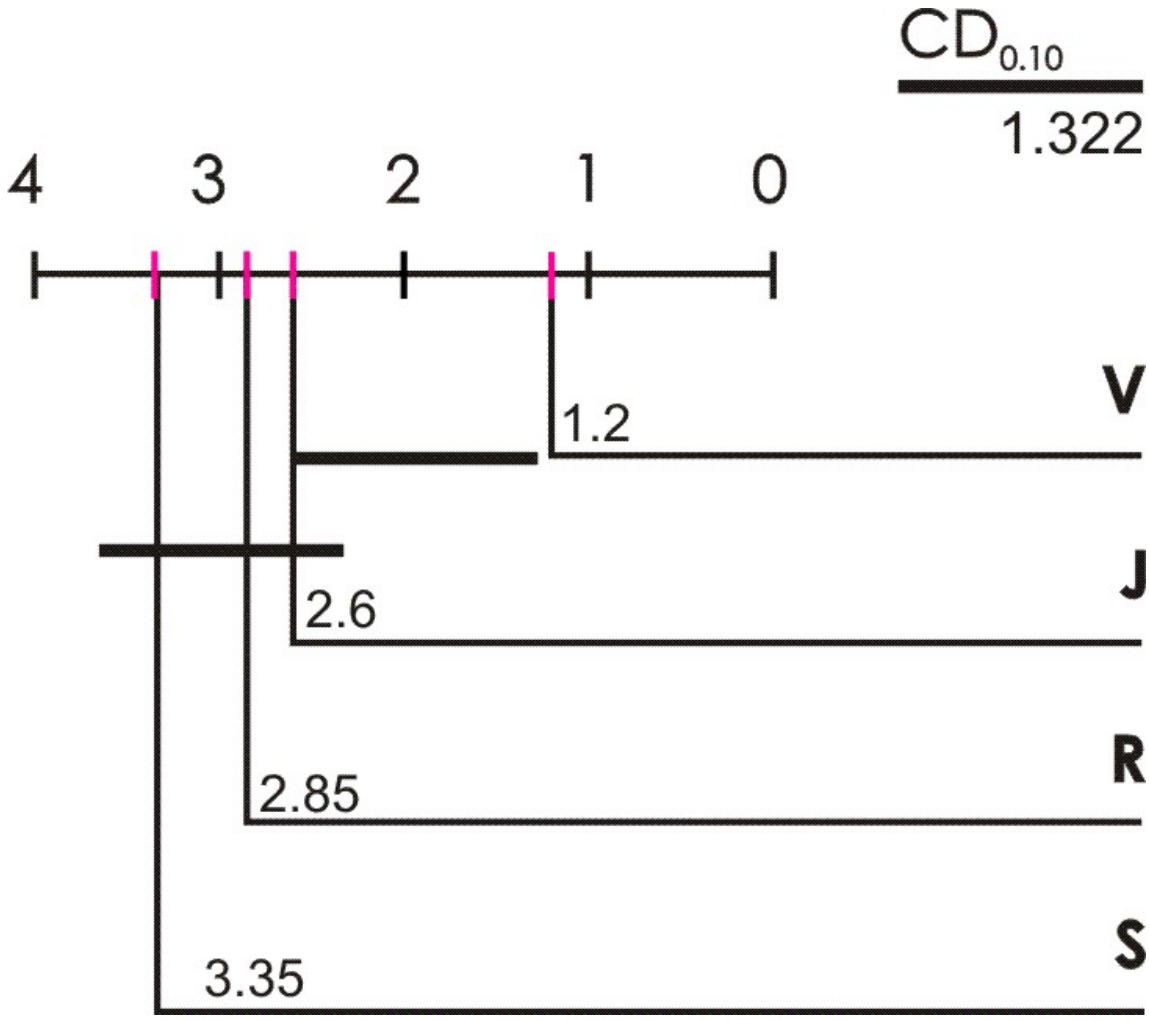

Figure 1. Visualization of Nemenyi test results

Analyzing these experiments we may conclude that **R** is at critical distances to **J** and **S**. **R** is nearer to **J** than to **S**. **R**, **J**, and **S** are significantly different from **V**.

Conclusion

We have presented results from series of experiments which were needed to estimate the storing time of NL-addressing for middle-size and very large RDF - datasets.

We described the experimental storing models and special algorithm for NL-storing RDF instances. Estimation of experimental systems was provided to make different configurations comparable. Special proportionality constants for hardware and software were proposed. Using proportionality constants, experiments with middle-size and large datasets become comparable.

Experiments were provided with both real and artificial datasets. Experimental results were systematized in corresponded tables. For easy reading visualization by histograms was given.

Experimental results will be analyzed in the next chapter.

The goal experiments for NL-storing of middle-size and large RDF-datasets were to estimate possible further development of NL-ArM. We assumed that its "software growth" will be done in the same grade as one of the known systems like **V**, **J**, and **S**. In the next chapter we will analyze what will be the place of NL-ArM in this environment but already we may see that NL-addressing have good performance and NL-ArM has similar results as **J** and **S**.

Acknowledgement

The paper is published with partial support by the Project "Methods for modeling and evaluation of informational services" of the University of Telecommunications and Posts, Sofia, Bulgaria.

Bibliography

[Becker, 2008] Christian Becker, "RDF Store Benchmarks with Dbpedia", Freie Universität Berlin, 2008, http://wifo5-03.informatik.uni-mannheim.de/benchmarks-200801/ (accessed: 05.04.2013)

[Bizer & Schultz, 2009] Christian Bizer, Andreas Schultz, "The Berlin SPARQL Benchmark", In: International Journal on Semantic Web & Information Systems, Vol. 5, Issue 2, Pages 1-24, 2009, http://wifo5-03.informatik.uni-mannheim.de/bizer/pub/Bizer-Schultz-Berlin-SPARQL-Benchmark-IJSWIS.pdf; see also http://wifo5-03.informatik.uni-mannheim.de/bizer/berlinsparqlbenchmark/ (accessed: 31.07.2013)

[BSBM DG, 2013] Data Generator and Test Driver, In: Berlin SPARQL Benchmark (BSBM) - Benchmark Rules, http://wifo5-03.informatik.uni-mannheim.de/bizer/berlinsparqlbenchmark/spec/BenchmarkRules/index.html#datagenerator (accessed: 31.07.2013)

[BSBMv1, 2008] Berlin SPARQL Benchmark Results, V1, 2008, http://wifo5-03.informatik.uni-mannheim.de/bizer/berlinsparqlbenchmark/V1/results/index.html (accessed: 31.07.2013)

[BSBMv2, 2008] Berlin SPARQL Benchmark Results, V2 2008, http://wifo5-03.informatik.uni-mannheim.de/bizer/berlinsparqlbenchmark/results/V2/index.html (accessed: 31.07.2013)

[BSBMv3, 2009] Berlin SPARQL Benchmark Results, V3, 2009, http://wifo5-03.informatik.uni-mannheim.de/bizer/berlinsparqlbenchmark/results/V3/index.html (accessed: 31.07.2013)

[BSBMv6, 2011] Berlin SPARQL Benchmark Results, V6, 2011, http://wifo5-03.informatik.uni-mannheim.de/bizer/berlinsparqlbenchmark/results/V6/index.html (accessed: 31.07.2013)

[DBpedia, 2007a] DBpedia dataset “homepages.nt” dated 2007-08-30, http://wifo5-03.informatik.uni-mannheim.de/benchmarks-200801/H.nt.gz (accessed: 31.07.2013)

[DBpedia, 2007b] DBpedia dataset “geocoordinates.nt” dated 2007-08-30, http://wifo5-03.informatik.uni-mannheim.de/benchmarks-200801/G.nt.gz (accessed: 31.07.2013)

[Dujmovi'c, 1996] Jozo Dujmovi'c, “A Method for Evaluation and Selection of Complex Hardware and Software Systems”, In: CMG 96 Proceedings, 1996, pp. 368-378 http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.48.4388 (accessed: 31.07.2013)

[i7 950, 2009] Intel i7 950 @ 3.07GHz (quadcore); CPU Launched: 2009; http://www.cpubenchmark.net/cpu.php?cpu=Intel+Core+i7+950+%40+3.07GHz&id=837 (accessed: 31.07.2013)

[Ivanova et al, 2016a] Krassimira Ivanova, Emiliya Saranova, Krassimir Markov, Stefan Karastanev, “A Method for Evaluation of Informational Services - Step 1: Computing the Hardware Proportionality Constants”, International Journal "Information Technologies & Knowledge" Volume 10, Number 2, 2016, ISSN 1313-0455 (printed), ISSN 1313-048X (online). pp. 103- 110.

[Ivanova et al, 2016b] Krassimira Ivanova, Ivan Ivanov, Mariyana Dimitrova, Krassimir Markov, Stefan Karastanev, “A Method for Evaluation of Informational Services - Step 2: Computing the Informational Services’ Performance Proportionality Constants”, International Journal "Information Models and Analyses" Volume 5, Number 3, 2016, ISSN 1314-6416 (printed), ISSN 1314-6432 (Online)pp. 203 – 214.

[Jena, 2016] Apache Jena, https://jena.apache.org/about_jena/about.html (accessed: 23.02.2016)

[LDIF Benchmarks, 2013] LDIF - Benchmark Results, http://ldif.wbsg.de/benchmark.html (accessed: 31.07.2013)

[LDIF, 2013] LDIF – Linked Data Integration Framework, http://ldif.wbsg.de/ (accessed 09.04.13).

[Minack, 2010] Enrico Minack, “RDF2RDF converter”, http://www.l3s.de/~minack/rdf2rdf/ 2010, (accessed: 31.07.2013).

[Pentium Dual, 2008] Intel Pentium Dual Core CPU @ 2.8 GHz; CPU Launched: 2008; http://www.cpubenchmark.net/cpu.php?cpu=Intel+Pentium+D+2.80GHz&id=1126 (accessed: 31.07.2013)

[Q9450, 2008] Intel Core 2 Quad Q9450 @ 2.66GHz, CPU Launched: 2008; http://www.cpubenchmark.net/cpu.php?cpu=Intel+Core2+Quad+Q9450+%40+2.66GHz&id=1046 (accessed: 31.07.2013)

[RDFArM, 2015] Krassimira Ivanova. RDFArM - A System For Storing Large Sets Of Rdf Triples And Quadruples By Means Of Natural Language Addressing. International Journal "Information Models and Analyses", Vol. 3, Number 4, 2014, ISSN 1314-6416 (printed), 1314-6432 (online), pp. 303 - 322.

[Sesame, 2015] Sesame, OpenRDF, https://bitbucket.org/openrdf/sesame (accessed: 01.12.2015)

[T9550, 2009] Intel® Core 2 Duo CPU T9550 @ 2.66GHz; CPU Launched: 2009, http://www.cpubenchmark.net/cpu.php?cpu=Intel+Core2+Duo+T9550+%40+2.66GHz&id=1011 (accessed: 31.07.2013)

[Virtuoso, 2013] OpenLink Virtuoso Universal Server: Documentation, http://Virtuoso.openlinksw.com/ (accessed: 23.11.2015)

Authors' Information

Krassimira Ivanova *– Assoc. prof. Dr.; University of Telecommunications and Posts, Sofia, Bulgaria; Institute of Mathematics and Informatics, BAS, Bulgaria; e-mail: krasy78@mail.bg ;*

Major Fields of Scientific Research: *Software Engineering, Business Informatics, Data Mining, Multidimensional multi-layer data structures in self-structured systems*

Krassimir Markov *– Institute of Mathematics and Informatics, BAS, Acad. G.Bontchev St., bl.8, Sofia-1113, Bulgaria; e-mail: markov@foibg.com*

Major Fields of Scientific Research*: General theoretical information research, Multi-dimensional information systems; Software Engineering, Business Informatics, Data Mining.*

Stefan Karastanev *– Assist. prof.; Institute of Mechanics, BAS, Bulgaria; e-mail: stefan@imbm.bas.bg*

Major Fields of Scientific Research: *Software Engineering, Data Processing and Mining, Data structures in information systems.*