Smart city RDF Benchmark

Pierfrancesco Bellini, Paolo Nesi
Distributed Systems and Internet Technology Lab, DISIT, http://www.disit.org
Department of Information Engineering, DINFO, University of Florence, Florence, Italy
pierfrancesco.bellini@unifi.it, paolo.nesi@unifi.it

I. Dataset of the Smart City RDF Benchmark

The data used for the evaluation is based on the Km4City knowledge base [1]. The Km4City models many aspects of a smart city. Some of them are static (or quasi-static) data such as (i) the *road graph* modeling the roads, the public administrations, etc. (ii) the “services” that are present in the city (e.g., restaurants, hotels, cycle paths, …) that are associated with the road graph and organized in an hierarchy, (iii) the bus stops, bus lines of the local transportation, (iv) the road sensors that are present on the roads. Moreover, dynamic information that change over time is also modeled, such as: (i) the weather forecasts for the different municipalities, (ii) the status/position of the bus with eventual forecasts for the arrival at the bus stops, (iii) the status of the parking lots (e.g., number of free places), (iv) the readings of the traffic sensors, (v) the events defined on the city. The testing datasets, comprised of triples, have been generated on the basis of Km4City model by using data from the Florence smart city service.

Three different datasets has been generated. They share the same ‘static’ information and differ for the dynamic part, having one, two or three months of history, respectively, in the past of the dynamic information. In Table I, the number of quadruples that are present for the different parts of the Km4City ontology are reported. It can be seen that the dynamic parts grows from 22% to 48.5% mostly derived from the AVM (automatic vehicle monitoring, of the ITS) that it is generated from the data coming for three bus lines, while the static part is mostly based by the structural data as road graph with 34.5M triples.

TABLE I. Dataset distribution

	1 month		**2 months**		**3 months**	
Type	***quadruples***	***%***	***quadruples***	***%***	***quadruples***	***%***
AVM	8.4M	19%	18M	33%	28M	43.1%
Parking	413k	0.9%	976k	1.8%	1.4M	2.1%
Sensors	900k	2%	1.7M	3.1%	2.2M	3.3%
Meteo	15k	0%	23k	0%	23k	0%
Total dynamic	**9.7M**	**22%**	**21M**	**38%**	**32.5M**	**48.5%**
Road graph	33.5M	75%	33.5M	60.3%	33.5M	50%
Services	681k	1.5%	681k	1.2%	681k	1%
Other static	286k	0.6%	286k	0.5%	286k	0.4%
Total static	**34.5M**	**78%**	**34.5M**	**62%**	**34.5M**	**51.4%**
Total	**44.2M**	**100%**	**55.6M**	**100%**	**67.5M**	**100%**

A. SPARQL Queries of the Smart City RDF Benchmark

The queries performed over the dataset are mainly those used in http://servicemap.disit.org, and thus a live solution can be accessed. It should be noted that the SPARQL recommendation does not cover the geo-spatial queries and neither the full-text queries. Therefore, in order to support those features, RDF store builder/vendor implemented the feature with a specific syntax. For this reason for some queries there is not a unique formulation and the query has to be adapted for each RDF store under test (they can be accessed from the web page of the proposed benchmark http://www.disit.org/smartcityrdfbenchmark). In Table II, the semantic queries at the basis of the *Smart City RDF Benchmark* are briefly described indicating if the query uses inferred information or not.

TABLE II. QUERY LIST

Query	Description	inference
Find-address	given the latitude and longitude position retrieves the nearest address within 100m.	No
Municipalities-florence	retrieves the list of municipalities within the province of Florence.	No
Bus-lines	retrieve the list of bus lines.	No
Bus-stops-of-line	given the bus line retrieves the bus stops list of the line.	No
Lines-of-bus-stop	given a bus stop retrieves the lines passing from the same bus stop.	No
Bus-stop-latlng	given a position and a radius finds the bus stops that are within the radius.	No
Bus-stop-florence	retrieves all the bus stops in the municipality of Florence.	No
Bus-stop-forecast	given a bus stop finds the next forecasts for the lines crossing at the bus stop.	No
AVM-distribution	retrieves for each day the count of AVM records received.	No
Service-florence	retrieves all the services in the municipality of Florence.	Yes
Service-Acc-Clt-Trs-W&F-florence	retrieves all the services in the Accomodation, Clutural Activity, TourismService and Wine&Food classes within the municipality of Florence.	Yes
Service-Htl-B&B-florence	retrieves all the services in the Hotel and Bed&Breakfast classes within the municipalityof Florence.	Yes
Service-latlng	retrieves the services within a radius from a latitude, longitude position.	Yes
Service-Acc-Clt-Trs-W&F-latlng	retrieves all the services in the Accomodation, Clutural Activity, TourismService and Wine&Food classes within a radius from a position.	Yes
Service-Htl-B&B-latlng	retrieves all the services in the Hotel and Bed&Breakfast classes within a radius from a position.	Yes
Service-text-florence	retrieves all services in the municipality of Florence matching a keyword.	Yes
Service-text-latlng	retrieves all services matching a keyword given a position and a radius.	Yes
Sensor-florence	retrieves all the sensors within the municipality of Florence.	No
Sensor-latlng	retrieves all sensors within a radius from a position.	No
Sensor-status	retrieves the latest information associated with a sensor.	No
Sensor-distribution	finds for each day the count of sensor status updates received.	No
Parking-status	retrieves the latest information associated with a parking lot.	No
Parking-distribution	retrieves for each day the count of parking status records acquired.	No
Weather-florence	retrieves the latest forecast available for the municipality of Florence.	No
Weather-distribution	retrieves for each day the count of weather forecasts acquired.	No

II. PERFORMANCE ASSESSMENT

In Table III, the results for the load time are reported for the different time horizons of one, two and three months, respectively. GraphDB is about three times slower than Virtuoso due to the fact that GraphDB performs inference at load time while Virtuoso at query time. And also the number of triples indexed in GraphDB (106M) is 36% bigger than those of Virtuoso (69M). For Virtuoso, the increment of triples stored with respect to those stated (2.1M for the 3 months case) is only due to transform the geo:lat and geo:long triples in a geo:geometry with POINT() to enable the geo-spatial indexing. While in the same case, for GraphDB the increment of 39M triples is due to the materialization of inference.

In Table IV and V, the results for the query execution time are reported for the different time horizons of one, two and three months, respectively, for GraphDB and Virtuoso. Table V reports the performances for non spatial queries and Table VI for spatial queries. The queries were tested performing a pseudo-random sequence of 500 queries repeated two times with some pseudo-random arguments in order to reduce the caching effect. The sequence of queries performed is the same for each execution in order to test the same sequence on different systems. From the query results, when no spatial and full text search and inference are involved, the performance is quite comparable, and in some cases GraphDB is better ranked. When inference is needed (e.g., in the test cases *Service-florence*, *Service-Acc-Clt-Trs-W&F-florence*, *Service-Htl-B&B-florence*) in the case of Virtuoso the inference should be enabled on the single constraint involving a general class (e.g., all services in the Accommodation class). While if the inference is enabled, generally on the query, the internal automated query rewrite takes a very long time (perhaps due to the size of the ontologies used). For example, for query *Service-Acc-Clt-Trs-W&F-florence* in Virtuoso the time grows from an average of 2.62s to an average of 24.5s (on the 3 months dataset) while GraphDB takes about 11.45s.

TABLE III. RDF STORES PERFORMANCE OF DATA LOADING

	Triples load time	***Stated triples***	***Stored triples***	***Size (of which: fulltext index size, spatial index size)***
GraphDB – 1 month	4h 27m	44,274,820	73,529,571	9.1GB (365MB, 64MB)
GraphDB – 2 months	6h 21m	55,619,789	89,839,143	12GB (445MB, 67MB)
GraphDB – 3 months	8h 12m	67,084,661	106,393,968	14GB (525MB, 69MB)
Virtuoso – 1 month	1h 15m	44,274,820	46,259,439	2.6GB (NA, NA)
Virtuoso – 2 months	1h 58m	55,619,789	57,669,629	3.4GB (NA, NA)
Virtuoso – 3 months	3h 22m	67,084,661	69,200,459	3.9GB (NA,NA)

TABLE IV. RDF STORES PERFORMANCE OF NON-SPATIAL QUERIES (IN BOLD THE BETTER PERFORMANCES)

	GraphDB			**Virtuoso**			
Query	***1 month (ms)***	***2 months (ms)***	***3 months (ms)***	***1 month (ms)***	***2 months (ms)***	***3 months (ms)***	***#results***
Municipalities-florence	10	12	12	12	10	11	46
Bus-lines	14	11	10	**6**	**6**	**6**	88
Bus-stops-of-line	**18**	**21**	**19**	48	39	33	99 (max)
Lines-of-bus-stop	**16**	**14**	**15**	22	26	22	7 (max)
Bus-stop-florence	118	127	122	**100**	**100**	**102**	1108
Bus-stop-forecast	884	1185	2229	**410**	**555**	**597**	30 (max)
AVM-distribution	1113	2582	3995	**33**	**50**	**62**	89 (max)
Service-florence	8641	6322	6730	**1434**	**1476**	**1286**	3259
Service-Acc-Clt-Trs-W&F-florence	5947	10403	11452	**2538**	**2365**	**2622**	1179
Service-Htl-B&B-florence	4680	2151	1023	**1078**	**545**	**946**	234
Service-text-florence	1808	1854	2271	**81**	**91**	**75**	51 (max)
Sensor-florence	24	25	25	**21**	**18**	**18**	65
Sensor-status	916	1804	2313	**75**	**96**	**120**	1
Sensor-distribution	1145	2485	3065	**177**	**324**	**404**	78 (max)
Parking-status	125	323	424	**125**	**151**	**172**	1
Parking-distribution	562	1400	1976	**72**	**133**	**197**	83 (max)
Weather-florence	**29**	**29**	**37**	58	80	80	5
Weather-distribution	10	13	13	**7**	**7**	**7**	38 (max)

When considering the spatial indexing we found in Virtuoso various problems, using the *st_intersection* function. In some cases, Virtuoso returns an error, in other cases providing a lower number of results with respect to the correctly expected and providing different results for the same query on the three different datasets that do not differ for the part considered in the query. On the other hand, in Virtuoso, if the *st_distance* function is used, all the obtained results have been verified to be correct, apart from few cases on the border (due to the numerical computation in measuring distances). The usage of the distance function for Virtuoso is good solution in most cases for example query *Service-latlng(5km)* retrieving all services within 5km from a gps position on the 3 months datasets takes 1.5s on virtuoso using st_distance function while it takes 9.7s on GraphDB, but reducing the distance to 200m Virtuoso takes 248ms while GraphDB only 153ms. Using the st_distance function on Virtuoso seems that the query optimizer to do not exploit the spatial index. This fact may be deduced from comparing that a same query (*Find-address*) by using *st_distance* function takes about 6s while using the *st_intersect* function takes about 0.3s. Another aspect to be considered is the mixing of spatial query with text search query (for example for query *Service-text-latlng(500m)*). With GraphDB, we registered very long execution time hitting in some cases the timeout of one hour. In this case of mixing spatial and text search for Virtuoso, the intersect function returned an error while using the distance function takes only 157 ms. Regarding the analytic queries (*Weather-distribution, AVM-distribution, Sensor-distribution, Parking-distribution*) that count the daily number of records of the weather forecasts, bus, sensor data, parking status for the three datasets Virtuoso is better ranked, it has an execution time less than 404ms while GraphDB is less than 3s. Moreover Virtuoso presents a lower growing factor with respect to GraphDB.

TABLE V. RDF STORES PERFORMANCE OF SPATIAL QUERIES (IN BOLD THE BETTER PERFORMANCES)

	GraphDB			**Virtuoso (intersect)**			**Virtuoso (distance)**			
Query	***1 month (ms)***	***2 months (ms)***	***3 months (ms)***	***1 month (ms)***	***2 months (ms)***	***3 months (ms)***	***1 month (ms)***	***2 months (ms)***	***3 months (ms)***	***#results***
Find-address	999	480	397	**327**	**285**	**275**	8625	6093	6489	1
Bus-stop-latlng(100m)	**7**	**8**	**27**	2298	1488	1542	29	32	31	1
Bus-stop-latlng(200m)	**19**	**16**	**46**	2331	1508	1551	25	33	31	2
Bus-stop-latlng(500m)	40	47	173	--	--	--	**32**	**39**	**39**	20
Bus-stop-latlng(1km)	99	132	504	--	--	--	**30**	**29**	**34**	93 (max)
Bus-stop-latlng(2km)	222	322	1116	--	--	--	**44**	**43**	**45**	252
Bus-stop-latlng(5km)	571	729	2389	--	--	--	**121**	**122**	**125**	1004 (max)
Service-latlng(100m)	**88**	**80**	**82**	210	243	244	226	258	237	41 (max)
Service-latlng(200m)	**182**	**145**	**153**	--	--	--	251	269	248	130 (max)
Service-latlng(500m)	983	902	921	--	--	--	**367**	**415**	**376**	784 (max)
Service-latlng(2km)	5113	4398	4616	--	--	--	**1030**	**1067**	**1027**	3720 (max)
Service-latlng(5km)	15191	10598	9690	--	--	--	**1605**	**1669**	**1582**	6660 (max)
Service-Acc-Clt-Trs-W&F-latlng(100m)	**137**	**105**	**107**	2403	2135	2358	932	854	688	19 (max)
Service-Acc-Clt-Trs-W&F-latlng(200m)	**374**	**221**	**236**	--	--	--	896	825	787	113 (max)
Service-Acc-Clt-Trs-W&F-latlng(500m)	1784	948	1092	--	--	--	**903**	**781**	**759**	424 (max)
Service-Acc-Clt-Trs-W&F-latlng(1km)	4510	2848	3319	--	--	--	**1209**	**1062**	**961**	1555 (max)
Service-Acc-Clt-Trs-W&F-latlng(2km)	8610	4910	5893	--	--	--	**1434**	**1299**	**1191**	2256 (max)
Service-Acc-Clt-Trs-W&F-latlng(5km)	17589	8857	10509	--	--	--	**1409**	**1554**	**1353**	3102 (max)
Service-Htl-B&B-latlng(100m)	**83**	**46**	**50**	1164	1115	--	418	393	419	7 (max)
Service-Htl-B&B-latlng(200m)	**141**	**62**	**78**	--	--	--	358	374	359	16 (max)
Service-Htl-B&B-latlng(500m)	652	479	512	--	--	--	**404**	**376**	**405**	151 (max)
Service-Htl-B&B-latlng(1km)	1899	1128	1360	--	--	--	**448**	**471**	**447**	363 (max)
Service-Htl-B&B-latlng(2km)	3560	1904	2327	--	--	--	**528**	**544**	**515**	488
Service-Htl-B&B-latlng(5km)	7815	4063	4401	--	--	--	**537**	**522**	**484**	607(max)
Service-text-latlng(500m)	1501370	1693348	1882047	--	--	--	**77**	**75**	**157**	21 (max)
Sensor-latlng(100m)	**16**	**9**	**11**	3785	2255	2310	11	11	12	0
Sensor-latlng(200m)	36	16	13	3797	2273	2333	**13**	**14**	**11**	0
Sensor-latlng(500m)	149	55	65	--	--	--	**12**	**16**	**16**	5
Sensor-latlng(1km)	617	179	247	--	--	--	**13**	**19**	**15**	15
Sensor-latlng(2km)	1092	361	497	--	--	--	**13**	**14**	**12**	32
Sensor-latlng(5km)	2082	757	1000	--	--	--	**13**	**17**	**18**	59

ACKNOWLEDGMENT

The authors would like to thanks ONTOTEXT to have provided access to a trial version of their RDF store. This works has been developed in the context of Km4City activity for Sii-Mobility Smart City National project.

REFERENCES

[1] P. Bellini, M. Benigni, R. Billero, P. Nesi, N. Rauch, “Km4City Ontology Bulding vs Data Harvesting and Cleaning for Smart-city Services”, International Journal of Visual Language and Computing, Elsevier, 2014