

Performance measurements of the Wonder system

The following is an evaluation of query performance using the WONDER system. The evaluation results are measured as the time (milliseconds) elapsed from the moment the user requests the execution of a query within the WONDER interface, to the moment WONDER displays the first results for his query. This time comprises: query translation, query execution by the Reasoner, plus communication protocol and GUI manipulation overhead.

Comparison: In order to measure the performance impact of adding WONDER to a existing OBDA layer, we compared WONDER's query execution performance to that of a simple Java-based DIG client that communicates with the Reasoner directly. Again, the time measured for the Java-DIG client is the time elapsed from the moment the client sends the "Ask" request with the query to the DIG Reasoner, to the moment it receives the first result set.

Specs: DIG-QuOnto and the Java-DIG client were running in a Linux PC with an Intel Core Duo (2.5 Ghz) processor and 4 Gb of RAM. The HGT database was hosted in an Oracle 10g installation on a Windows XP PC with an Intel Core Duo (1.66 Ghz) processor and 2 Gb of RAM.

TESTER	Time 1	Time 2	Time 3	Time 4	Time 5	Average*	Std. Dev.*
A	2052	2009	2525	1980	2125	2138.2	223.01
B	41076	59202	64905	45000	49588	51954.2	9906.42
C	40500	60234	56611	49245	50166	51351.2	7585.5
D	42338	54148	48440	49956	46617	48299.8	4339.9
WONDER	Time 1	Time 2	Time 3	Time 4	Time 4	Average*	Std. Dev.*
A	2959	2020	1882	2439	4832	2826.4	1197.11
B	42129	50605	49432	48049	51967	48436.4	3810.92
C	41006	50426	96564	63524	52318	60767.6	21551.36
D	52539	49593	50714	48976	50992	50562.8	1375.76

Synopsis of the analysis of the results. In contrast to intuition that queries executed within the WONDER system would always be slightly slower than queries executed using a simple DIG client, we obtained results (Query B) in which the time elapsed for WONDER was smaller than that of the simple DIG client. We attribute this result to the high Standard Deviation observed during these tests. We will update these results as soon as we isolate the causes of this high Standard Deviation and re-run the tests. Among the reasons we are considering are the low specifications and load of the machine running the database engine, which is not ideal for obtaining optimal performance, but instead is a very realistic user scenario. To address this issue, we will re-run the tests by both moving to an isolated high-end Linux server and run performance tests remotely with the domain experts in Spain and the Netherlands. In any case, the response times are good and within the acceptable range according to the domain experts.

Queries: queries A-D of Section 2 translate into the following SPAQL queries:

Query A

```
SELECT $13
WHERE { $1 rdf:type 'Organism' .
        $2 rdf:type 'OrganismInfo' .
        $5 rdf:type 'HGTPredictionGene' .
        $13 rdf:type 'Gene' .
        $1 :OrganismHasOrganismInfo $2 .
        $13 :GeneHasOrganism $1 .
        $13 :GeneHasHGTPredictionGene $5 .
        $5 :Prediction 'hgt' .
        $2 :OrganismName 'Bordetella_pertussis'
      }
```

Query B

```
SELECT q1.e, q1.f FROM sparqltable
  (SELECT $e $f
    WHERE
      { $c rdf:type 'GeneFunction' .
        $e rdf:type 'Organism' .
        $1 rdf:type 'Gene' .
        $1 :GeneHasGeneFunction $c .
        $1 :GeneHasOrganism $e .
        $c :PATH $f
      }
  ) q1
WHERE ( q1.f LIKE '%ko03010%' )
```

Query C

```
SELECT q1.bh FROM sparqltable
  (SELECT $bh $h $l $bb
    WHERE
      { $b rdf:type 'Organism' .
        $c rdf:type 'Taxonomy' .
        $f rdf:type 'GeneIDInfo' .
        $g rdf:type 'HGTPredictionGene' .
        $bh rdf:type 'Gene' .
        $b :OrganismHasTaxonomy $c .
        $bh :GeneHasOrganism $b .
        $bh :GeneHasGeneIDInfo $f .
        $bh :GeneHasHGTPredictionGene $g .
        $c :Classification $h .
        $f :GeneName $l .
        $g :Prediction $bb
      }
  ) q1 WHERE ( q1.l = 'dnaA' AND
               q1.bb = 'hgt' AND
               q1.h LIKE '%Bacillus%' )
```

Query D

```
SELECT q1.b, q1.bf FROM sparqltable
( SELECT $b $bf $h $l
  WHERE
    { $b rdf:type 'Organism' .
      $c rdf:type 'Taxonomy' .
      $be rdf:type 'GCstatsGene' .
      $be rdf:type 'GCstats' .
      $bf rdf:type 'Gene' .
      $b :OrganismHasTaxonomy $c .
      $bf :GeneHasOrganism $b .
      $bf :GeneHasGCstatsGene $be .
      $c :Classification $h .
      $be :GC3 $l
    }
) q1 WHERE ( q1.h LIKE '%Firmicutes%' AND
            q1.l > '80' )
```