

LUBM and Virtuoso

Introduction

This article discusses Virtuoso's performance of the Lehigh University Benchmark (LUBM) at different scales and in different configurations.

We analyze the performance of Virtuoso in both single-server and clustered configurations for loading and querying a derivative of the LUBM data set.

The goal of this article is to give a general understanding of Virtuoso's triple-store performance and governing factors.

The Data Set

We use the unmodified LUBM synthetic data set at different scales. The basic query timing is done with the qualification data set of about 100K triples. Tests with concurrent queries are performed at the scales of 800, 8000 and 160,000 universities, corresponding roughly to 100M, 1G and 20G triples.

We have adapted the original LUBM benchmark queries to Virtuoso and verified that when applied to the qualification data set the correct answers are produced.

Aside storage and query, the LUBM benchmark involves simple inference on RDF data. Some of this inference can be done at either load or query time. Some of the inference must be done after loading because this is not supported at query time.

- We always materialize the transitive suborganization relation, so that a suborganization triple is always present between all direct and indirect super/suborganization pairs.
- Inverse relation inference is not tested. The queries are rewritten to avoid reliance on inverse relation inference.
- Subclass and subproperty inference may be done either at query or load time. We experiment with both.
- The benchmark does not involve owl:same-as. Virtuoso's query-time support for owl:same-as is not used.

The Queries and Metric

The queries exist in three variants:

1. Open Coded Inference. All combinations of subclass and subproperty relations are expressed as unions.
2. Inference using Virtuoso's query-time support of subclass and subproperty.

3. Materialized data where all triples implied by subclass and subproperty relations are physically present.

The original LUBM involves a composite metric of speed and completeness of inference. We produce complete results in all cases but vary the time and mode of inference.

We measure load rates as Kt/s, with 1Kt/s being 1000 triples per second of real time.

- We give times for single user query execution against the one university qualification database. These times are in milliseconds.
- For concurrent query load, we have defined a query mix consisting of the 14 LUBM queries, modified when necessary so as not to return excessive volumes of data. The metric is queries per second at scale, where scale is the number of universities. Each query belonging to a completed query mix is counted as one query. Only queries from query mixes completed during the measurement interval are counted.

All queries are modified so as to be scoped to a single university. The rationale is that queries that read through the whole database will be by far the longest in duration and hence the benchmark would measure only these if these were included in the mix. A mix with orders of magnitude between longest and shortest might as well not include the shorter queries.

Query Mix

The adapted query mix is shown below. We only show the version that runs against the materialized data. The other mixes are similarly modified. The mixes run against the qualification database are listed in all three variants in the appendix.

```
/* Q1 */
select * from <lubm where
{{ ?x rdf:type ub:GraduateStudent . ?x ub:takesCourse <%s> }};
/* Q2 */
select * from <lubm where
  {{ ?x a ub:GraduateStudent . ?z a ub:Department . ?x
ub:memberOf ?z . ?z ub:subOrganizationOf <%s> . ?x
ub:undergraduateDegreeFrom <%s> }}
/* Q3 */
select * from <lubm where
  {{ ?x a ub:Publication . ?x ub:publicationAuthor <%s> }}
/* Q4 */
select * from <lubm where
  {{ ?x a ub:Professor . ?x ub:worksFor <%s> . ?x ub:name ?y1 . ?
x ub:emailAddress ?y2 . ?x ub:telephone ?y3 . }};
/* Q5 */
select * from <lubm where
  {{ ?x a ub:Person . ?x ub:memberOf <%s> }};
/* Q6 */
select * from <lubm where
  {{ ?x a ub:Student . ?x ub:memberOf <%s> }};
/* Q7 */
select * from <lubm where
```

```

    {{ ?x a ub:Student . ?y a ub:Course . <%s> ub:teacherOf ?y . ?x
ub:takesCourse ?y . }};
    /* Q8 */
select * from <lubm where
    {{ ?x a ub:Student . ?y a ub:Department . ?x ub:memberOf ?y . ?
y ub:subOrganizationOf <%s> . ?x ub:emailAddress ?z }};
    /* Q9 */
select * from <lubm where
    {{ ?x a ub:Student . ?y a ub:Faculty . ?z a ub:Course . ?x
ub:advisor ?y . ?x ub:takesCourse ?z . ?y ub:teacherOf ?z . ?x
ub:memberOf <%s> . }}
    /* Q10 */
select * from <lubm where
    {{ ?x a ub:Student . ?x ub:takesCourse <%s> . }};
    /* Q11 */
select * from <lubm where
    { ?x a ub:ResearchGroup . ?x ub:subOrganizationOf <%s> . };
    /* Q12 */
select * from <lubm where
    {{ ?x a ub:Professor . ?y a ub:Department . ?x ub:headOf ?y . ?
y ub:subOrganizationOf <%s> . }};
    /* Q13 */
select count (*) from <lubm where
    { ?x a ub:Person . ?x ub:degreeFrom <%s> . }
    /* Q14 */
select * from <lubm where
{ ?x a ub:UndergraduateStudent . ?x ub:memberOf ?z . ?z
ub:subOrganizationOf <%s> . };

```

The %s is substituted with a randomly selected IRI of the appropriate type. Q13 is modified to return a count because otherwise it would take all the time of the benchmark since it returns up to 48K rows.

Database Layout

The tests are run against Virtuoso with default triple storage layout. All the data except for schema data is loaded in a single graph. The quads are indexed as GSPO and OGPS, where the latter is a bitmap index with all values of S for a given OGP combination represented as a bitmap. All URI and object ids are 32-bit. An O that is an IRI or short scalar is stored inline in the O column of the quad table. Long string-valued Os are assigned an id and referenced using this id from the O of the quad table.

Loading

We have experimented with different ways of loading RDF data using different multithreading schemes.

1. Loading on a single thread, with the same thread running the parser and translating the URIs to URI id and inserting these into the quad table
2. One thread parsing a file and feeding a queue from which worker threads pick triples. The

worker threads then translate the URIs to IDs and insert the triples.

3. Cluster loading with optimization on message passing.

All loading is done without locking, transaction rollback possibility and with no roll-forward logging. This is reasonable since this is a bulk-load activity. The loads are hardened by a database checkpoint.

We have found that with a single server process, the best performance is obtained by running one single threaded load function on each core, thus with 4 concurrent loads proceeding at all times on a 4-core machine. With 8 cores, the optimum is around 6 streams.

If only a single load stream is available, some performance gain is obtained by having up to 3 worker threads for processing the output of a single parser thread.

In all cases, we avoid having threads repeatedly hit the same last page of data by giving each thread a small pool of URI ids to allocate. Thus two threads do not generally try to write the same page at the same time.

Tuning

On all systems, the count of database cache buffers was selected so that the Virtuoso process, after reaching steady memory consumption, took about three-quarters the available physical memory. In this way, a database buffer counts for about 9.5K.

When separate disks were available, if running a single server, the database was striped across all disks. When running multiple server processes in cluster mode, each had its own disk when available.

Besides this, no other special configuration measures were taken.

Systems Tested

The systems tested were:

```
System A: 2 x Xeon 5130 2GHz, 8G RAM, 6 x 160G SATA2 disk  
System B: 2 x Xeon 5330 2GHz, 6 x 250G SATA2 disks
```

Large Load Rate

8000 Universities

```
System A: 29.7 Kt/s, single server, 4 streams  
System B: 36.9 Kt/s
```

Qualification Database

We ran all versions of the queries against the qualification database of 1 university, about 100K triples to verify that all query versions produce the same data. The total run times of the queries, one stream at a time, warm cache, are stated below, tested on system A:

1. Unions: 1917 ms
2. Inference: 1029 ms
3. Materialized: 724 ms

With the materialized run, the set of queries performs 77,600 single row retrievals, a rate of 98,000 rows per second. This does not include lookups which find no rows. The single row lookup rate is about 300,000 per second if there is no other query logic, for example when joining one index of the quad table to another index on full equality, i.e. checking that the two indices have the same content.

All three modes access the same amount of data on a warm cache. The difference is only due to the different length of execution path in the SQL run time.

Concurrent Query Rate

We filled a database with 8000 universities and ran different numbers of clients on different fractions of the database. Each query is scoped to a single university picked at random from the n first universities of the 8000 university database; this ensures the total volume is the same but the working set varies.

The queries considered here are using the built-in inference of subclasses and subproperties. The only materialized inference is the suborganization property.

The timing results are obtained when the server has reached a steady state with the selected number of universities. Steady state is here defined as either (1) having less than 1% of real time in disk i/o or (2) having filled all disk cache buffers after starting with an empty cache.

The results are reported per query, taking a sample of the test driver's output.

The numbers are, for example in:

```
-- Q1                2 / 40 / 299          3451
0% 85 times

Query
| shortest/average /longest msec
| total msec
| percentage of total run time spent in this query
| count of times the query was run in the reported interval
```

100 Universities

The queries were applied against the 100 first universities of a database of 8000. This measures memory-based performance.

```
1 client: 11 qps
4 clients: 31 qps
8 clients: 33.1 qps
```

CPU at approx 360% of 400%, less than 0.04 threads waiting for disk

Sample output from run with 4 clients:

```
-- Q1          1 / 2 / 12          29
0% 10 times
-- Q2          7 / 9 / 11          92
0% 10 times
-- Q3          2 / 10 / 24         104
0% 10 times
-- Q4          8 / 47 / 153        476
2% 10 times
-- Q5          0 / 6 / 16          62
0% 10 times
-- Q6          7 / 18 / 28         185
1% 10 times
-- Q7          6 / 12 / 31         120
0% 10 times
-- Q8         299 / 431 / 546       4311
23% 10 times
-- Q9          70 / 88 / 123        888
4% 10 times
-- Q10         2 / 4 / 11           44
0% 10 times
-- Q11         5 / 6 / 8            67
0% 10 times
-- Q12         3 / 8 / 13           82
0% 10 times
-- Q13        822 / 911 / 1023      9110
50% 10 times
-- Q14         83 / 170 / 275       1700
9% 10 times
```

1000 Universities

The measurement was done with 8 concurrent clients feeding the query mix against the 1000 first universities of the 8000 university set.

```
6.7 qps
CPU 89% of 400%, disk 6.9 of 8 threads waiting on average
```

Sample:

```
-- Q1          1 / 18 / 33          186
0% 10 times
-- Q2          8 / 115 / 640        1150
0% 10 times
-- Q3          7 / 75 / 278         756
```

```

0% 10 times
-- Q4          39 / 117 / 182          1175
0% 10 times
-- Q5          4 / 7 / 25            78
0% 10 times
-- Q6          9 / 28 / 98           280
0% 10 times
-- Q7          5 / 72 / 326          724
0% 10 times
-- Q8          347 / 13939 / 30077    139397
83% 10 times
-- Q9          170 / 364 / 763        3648
2% 10 times
-- Q10         3 / 18 / 45           186
0% 10 times
-- Q11         6 / 105 / 406         1058
0% 10 times
-- Q12         6 / 102 / 721         1025
0% 10 times
-- Q13         691 / 930 / 1260       9302
5% 10 times
-- Q14        101 / 589 / 1690        5898
3% 10 times

```

8000 Universities

4.8 qps

CPU at 20% of 400%, 7.7 threads waiting for disk on the average.

Sample:

```

-- Q1          20 / 71 / 219          710
0% 10 times
-- Q2          28 / 110 / 484         1106
0% 10 times
-- Q3          48 / 83 / 172          830
0% 10 times
-- Q4          121 / 205 / 364         2056
1% 10 times
-- Q5          4 / 40 / 133           403
0% 10 times
-- Q6          73 / 129 / 224         1298
0% 10 times
-- Q7          77 / 170 / 323         1706
0% 10 times
-- Q8          8169 / 15293 / 25299    152930
75% 10 times
-- Q9          234 / 629 / 988         6294
3% 10 times
-- Q10         12 / 36 / 69           363
0% 10 times
-- Q11        255 / 411 / 617         4116
2% 10 times
-- Q12         7 / 600 / 1027         6007

```

2%	10 times						
--	Q13	15	/	303	/	1154	3035
1%	10 times						
--	Q14	958	/	1897	/	2706	18979
9%	10 times						

Comments: Q13 is low because there in fact are universities in the generated set from which nobody has a degree.

Even though the performance is totally I/O-bound, all indices of the database have a hit rate of over 99%. This means less than 1 read per 100 successfully retrieved rows.

Analysis

We see that the database size has little effect on query-times as long as the working set fits in memory. The single query stream rate with 100K triples is 14 qps at 100K triples and 11 qps at 1G triples. We also note that while staying in memory, contention between processor cores does not severely affect performance: from 11 qps with 1 stream to 33 qps with 8 streams with 4 cores.

As expected, we get a severe drop in performance when going out of purely memory-based working set. This emphasizes the need for a memory-efficient storage format. This has been addressed in Virtuoso 6, which stores twice as many triples in the same space.

All disk-access is done on-demand, one page at a time. The workload does not have many opportunities for exploiting sequentiality in disk access. The starting point of a navigational query is typically a bitmap, such as the bitmap of all subjects of a given type or all suborganizations of a university. These often fit on a single page but for larger bitmaps read ahead is beneficial and should be used.

Bitmap intersections are frequent: for example in Q13, where we have an intersection of all subcases of person with all types of graduates from a given university. Thus we have a loop iterating over the types of persons, a nested loop iterating over the types of degrees and then a bitmap intersection counting how many of the persons intersect with the bitmap of graduates of the given type from the university.

The bitmap intersection is about twice as efficient as the equivalent loop join, even in the worst case, i.e. a short bitmap (all doctoral graduates of university 1) with a large bitmap (all associate professors in the database). If the bitmaps are about the same size the gain from a bitmap merge join is still greater.

Otherwise the access method is loop joining, most often using the OGPS bitmap index. This is preferred because it is only about 1/3 the size of the GSPO index with the same data. Loop joins with random access offer little opportunity for optimizing disk-access.

Hash joins do not occur in the execution plans, which is for the best. The cases of joining a small set to a large one on equality of a key are covered by bitmap intersections.

Conclusions and Future Work

We have here presented intermediate results following a review of the LUBM query workload and some consequent optimizations. All results are measured on Virtuoso 5.0.4, as of February 1, 2008.

We thank the authors of the LUBM benchmark for their work in defining the test data and the workload. A point-by-point run-through this and the issues this presented resulted in an improvement of over 30% in our performance of this workload. This serves to demonstrate that benchmarks are always useful.

As we have stated many times before, RDF benchmarking needs to evolve to more varied workloads, specifically analytics with aggregation and grouping. This is where a lot of the action in the relational space is and where RDF also may find uses as a data integration medium.

A query performance metric should have the right mix of frequent and infrequent queries. Also, the queries likely to run in an interactive application and those run in batch mode should be differently weighted and with different frequencies or should have their own benchmark and metric. Due to this the queries-per-second metric presented here is not representative of any specific type of application.

While LUBM has served us well indeed, it is time to define a new benchmark with a metric for concurrent performance and a more complex and varied workload.

As future work, we intend to define a new RDF database benchmark drawing on the social web as a use-case and featuring a more varied workload with a well-defined metric for concurrent query and update.

At the time of writing, we are running the same tests on Virtuoso 6.0 in single-machine and cluster configurations and hope to publish results in due course. As a preview, we can say that performance there is higher due to improved storage density.

Appendix A Query Text

This appendix contains the text of the queries adapted to Virtuoso. Three variants are presented: one with unions, one with run-time subclass and subproperty inference and one with all entailed triples materialized. The script text can be run with the Virtuoso *isql* utility.

Entailment

For all scripts, the `ub:chair` property was materialized. The statement for this is:

```
sparql prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#>
insert into graph <lubm> { ?x ub:subOrganizationOf ?z } from
<lubm> where { ?x ub:subOrganizationOf ?y . ?y
```

```
ub:subOrganizationOf ?z . };
```

Additionally, for getting the correct results with the materialized script, the following statements were run:

```
sparql prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-  
bench.owl#>  
insert into graph <lubm> { ?x a ub:Professor }  
where {  
    { ?x a ub:AssistantProfessor } union  
    { ?x a ub:AssociateProfessor } union  
    { ?x a ub:FullProfessor } union  
    { ?x a ub:VisitingProfessor }  
};
```

```
sparql prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-  
bench.owl#>  
insert into graph <lubm> { ?x a ub:Faculty }  
where {  
    { ?x a ub:Professor } union  
    { ?x a ub:PostDoc } union  
    { ?x a ub:Lecturer }  
};
```

```
sparql prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-  
bench.owl#>  
insert into graph <lubm> { ?x a ub:Student }  
where {  
    { ?x a ub:UndergraduateStudent } union  
    { ?x a ub:GraduateStudent } union  
    { ?x a ub:ResearchAssistant }  
};
```

```
sparql prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-  
bench.owl#>  
insert into graph <lubm> { ?x a ub:AdministrativeStaff }  
where {  
    { ?x a ub:ClericalStaff } union  
    { ?x a ub:SystemsStaff }  
};
```

```
sparql prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-  
bench.owl#>  
insert into graph <lubm> { ?x a ub:Employee }  
where {  
    { ?x a ub:Faculty } union  
    { ?x a ub:AdministrativeStaff }  
};
```

```
sparql prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-  
bench.owl#>  
insert into graph <lubm> { ?x a ub:Person }  
where {  
    { ?x a ub:Chair } union
```

```

        { ?x a ub:Dean } union
        { ?x a ub:Director } union
        { ?x a ub:Employee } union
        { ?x a ub:Student } union
        { ?x a ub:TeachingAssistant }
};

sparql prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#>
insert into graph <lubm> { ?x a ub:Course }
where {
    { ?x a ub:GraduateCourse }
};

sparql prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#>
insert into graph <lubm> { ?x ub:worksFor ?z }
where {
    { ?x ub:headOf ?z }
};

sparql prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#>
insert into graph <lubm> { ?x ub:memberOf ?z }
where {
    { ?x ub:worksFor ?z }
};

sparql prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#>
insert into graph <lubm> { ?x ub:degreeFrom ?z }
where {
    { ?x ub:doctoralDegreeFrom ?z } union
    { ?x ub:mastersDegreeFrom ?z } union
    { ?x ub:undergraduateDegreeFrom ?z }
};

```

Query Text with Unions

```

set autocommit on;

-- Q1
sparql prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#>
select * from <lubm>
where { ?x rdf:type ub:GraduateStudent . ?x ub:takesCourse
<http://www.Department0.University0.edu/GraduateCourse0> };

-- Q2
sparql prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#>
select * from <lubm>
where { ?x a ub:GraduateStudent . ?y a ub:University . ?z a
ub:Department . ?x ub:memberOf ?z . ?z ub:subOrganizationOf ?y .

```

```

?x ub:undergraduateDegreeFrom ?y };

-- Q3
sparql prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#>
select * from <lubm>
where { ?x a ub:Publication . ?x ub:publicationAuthor
<http://www.Department0.University0.edu/AssistantProfessor0> };

-- Q4
sparql prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#>
select * from <lubm>
where
{
{ ?x a ub:AssociateProfessor . ?x ub:worksFor
<http://www.Department0.University0.edu> . ?x ub:name ?y1 . ?x
ub:emailAddress ?y2 . ?x ub:telephone ?y3 . }
union
{ ?x a ub:AssistantProfessor . ?x ub:worksFor
<http://www.Department0.University0.edu> . ?x ub:name ?y1 . ?x
ub:emailAddress ?y2 . ?x ub:telephone ?y3 . }
union
{ ?x a ub:FullProfessor . ?x ub:worksFor
<http://www.Department0.University0.edu> . ?x ub:name ?y1 . ?x
ub:emailAddress ?y2 . ?x ub:telephone ?y3 . }
};

-- Q5
sparql prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#>
select distinct * from <lubm>
where
{
{ ?x a ub:AssociateProfessor . ?x ub:memberOf
<http://www.Department0.University0.edu> } union
{ ?x a ub:FullProfessor . ?x ub:memberOf
<http://www.Department0.University0.edu> } union
{ ?x a ub:AssistantProfessor . ?x ub:memberOf
<http://www.Department0.University0.edu> } union
{ ?x a ub:Lecturer . ?x ub:memberOf
<http://www.Department0.University0.edu> } union
{ ?x a ub:UndergraduateStudent . ?x ub:memberOf
<http://www.Department0.University0.edu> } union
{ ?x a ub:GraduateStudent . ?x ub:memberOf
<http://www.Department0.University0.edu> } union
{ ?x a ub:TeachingAssistant . ?x ub:memberOf
<http://www.Department0.University0.edu> } union
{ ?x a ub:ResearchAssistant . ?x ub:memberOf
<http://www.Department0.University0.edu> } union
{ ?x a ub:AssociateProfessor . ?x ub:worksFor
<http://www.Department0.University0.edu> } union

```

```

{ ?x a ub:FullProfessor . ?x ub:worksFor
<http://www.Department0.University0.edu> } union
{ ?x a ub:AssistantProfessor . ?x ub:worksFor
<http://www.Department0.University0.edu> } union
{ ?x a ub:Lecturer . ?x ub:worksFor
<http://www.Department0.University0.edu> } union
{ ?x a ub:UndergraduateStudent . ?x ub:worksFor
<http://www.Department0.University0.edu> } union
{ ?x a ub:GraduateStudent . ?x ub:worksFor
<http://www.Department0.University0.edu> } union
{ ?x a ub:TeachingAssistant . ?x ub:worksFor
<http://www.Department0.University0.edu> } union
{ ?x a ub:ResearchAssistant . ?x ub:worksFor
<http://www.Department0.University0.edu> } union

```

```

{ ?x a ub:AssociateProfessor . ?x ub:headOf
<http://www.Department0.University0.edu> } union
{ ?x a ub:FullProfessor . ?x ub:headOf
<http://www.Department0.University0.edu> } union
{ ?x a ub:AssistantProfessor . ?x ub:headOf
<http://www.Department0.University0.edu> } union
{ ?x a ub:Lecturer . ?x ub:headOf
<http://www.Department0.University0.edu> } union
{ ?x a ub:UndergraduateStudent . ?x ub:headOf
<http://www.Department0.University0.edu> } union
{ ?x a ub:GraduateStudent . ?x ub:headOf
<http://www.Department0.University0.edu> } union
{ ?x a ub:TeachingAssistant . ?x ub:headOf
<http://www.Department0.University0.edu> } union
{ ?x a ub:ResearchAssistant . ?x ub:headOf
<http://www.Department0.University0.edu> }

```

```
};
```

```
-- Q6
```

```
sparql prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#>
```

```
select distinct * from <lubm> where {
  { ?x a ub:UndergraduateStudent . }
  union
  { ?x a ub:ResearchAssistant . }
  union
  { ?x a ub:GraduateStudent . }
};
```

```
-- Q7
```

```
sparql prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#>
```

```
select distinct * from <lubm>
where
{
  { ?x a ub:UndergraduateStudent . ?y a ub:Course .
<http://www.Department0.University0.edu/AssociateProfessor0>
ub:teacherOf ?y . ?x ub:takesCourse ?y . }
  union

```

```

{ ?x a ub:UndergraduateStudent . ?y a ub:GraduateCourse .
<http://www.Department0.University0.edu/AssociateProfessor0>
ub:teacherOf ?y . ?x ub:takesCourse ?y . }
union
{ ?x a ub:ResearchAssistant . ?y a ub:Course .
<http://www.Department0.University0.edu/AssociateProfessor0>
ub:teacherOf ?y . ?x ub:takesCourse ?y . }
union
{ ?x a ub:ResearchAssistant . ?y a ub:GraduateCourse .
<http://www.Department0.University0.edu/AssociateProfessor0>
ub:teacherOf ?y . ?x ub:takesCourse ?y . }
union
{ ?x a ub:GraduateStudent . ?y a ub:Course .
<http://www.Department0.University0.edu/AssociateProfessor0>
ub:teacherOf ?y . ?x ub:takesCourse ?y . }
union
{ ?x a ub:GraduateStudent . ?y a ub:GraduateCourse .
<http://www.Department0.University0.edu/AssociateProfessor0>
ub:teacherOf ?y . ?x ub:takesCourse ?y . }
}
;

```

-- Q8

```

sparql prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#>
select distinct * from <lubm>
where
{
  { ?x a ub:UndergraduateStudent . ?y a ub:Department . ?x
ub:memberOf ?y . ?y ub:subOrganizationOf
<http://www.University0.edu> . ?x ub:emailAddress ?z }
  union
  { ?x a ub:UndergraduateStudent . ?y a ub:Department . ?x
ub:worksFor ?y . ?y ub:subOrganizationOf
<http://www.University0.edu> . ?x ub:emailAddress ?z }
  union
  { ?x a ub:UndergraduateStudent . ?y a ub:Department . ?x
ub:headOf ?y . ?y ub:subOrganizationOf
<http://www.University0.edu> . ?x ub:emailAddress ?z }
  union
  { ?x a ub:ResearchAssistant . ?y a ub:Department . ?x
ub:memberOf ?y . ?y ub:subOrganizationOf
<http://www.University0.edu> . ?x ub:emailAddress ?z }
  union
  { ?x a ub:ResearchAssistant . ?y a ub:Department . ?x
ub:worksFor ?y . ?y ub:subOrganizationOf
<http://www.University0.edu> . ?x ub:emailAddress ?z }
  union
  { ?x a ub:ResearchAssistant . ?y a ub:Department . ?x ub:headOf
?y . ?y ub:subOrganizationOf <http://www.University0.edu> . ?x
ub:emailAddress ?z }
  union
  { ?x a ub:GraduateStudent . ?y a ub:Department . ?x ub:memberOf
?y . ?y ub:subOrganizationOf <http://www.University0.edu> . ?x
ub:emailAddress ?z }
}

```

```

union
{ ?x a ub:GraduateStudent . ?y a ub:Department . ?x ub:worksFor
?y . ?y ub:subOrganizationOf <http://www.University0.edu> . ?x
ub:emailAddress ?z }
union
{ ?x a ub:GraduateStudent . ?y a ub:Department . ?x ub:headOf ?y
. ?y ub:subOrganizationOf <http://www.University0.edu> . ?x
ub:emailAddress ?z }
}
;

-- Q9
sparql prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#>
select distinct * from <lubm>
where
{
{ ?x a ub:ResearchAssistant . ?y a ub:Lecturer . ?z a ub:Course
. ?x ub:advisor ?y . ?x ub:takesCourse ?z . ?y ub:teacherOf ?z .
} union
{ ?x a ub:ResearchAssistant . ?y a ub:PostDoc . ?z a ub:Course
. ?x ub:advisor ?y . ?x ub:takesCourse ?z . ?y ub:teacherOf ?z .
} union
{ ?x a ub:ResearchAssistant . ?y a ub:VisitingProfessor . ?z a
ub:Course . ?x ub:advisor ?y . ?x ub:takesCourse ?z . ?y
ub:teacherOf ?z . } union
{ ?x a ub:ResearchAssistant . ?y a ub:AssistantProfessor . ?z a
ub:Course . ?x ub:advisor ?y . ?x ub:takesCourse ?z . ?y
ub:teacherOf ?z . } union
{ ?x a ub:ResearchAssistant . ?y a ub:AssociateProfessor . ?z a
ub:Course . ?x ub:advisor ?y . ?x ub:takesCourse ?z . ?y
ub:teacherOf ?z . } union
{ ?x a ub:ResearchAssistant . ?y a ub:FullProfessor . ?z a
ub:Course . ?x ub:advisor ?y . ?x ub:takesCourse ?z . ?y
ub:teacherOf ?z . } union

{ ?x a ub:ResearchAssistant . ?y a ub:Lecturer . ?z a
ub:GraduateCourse . ?x ub:advisor ?y . ?x ub:takesCourse ?z . ?y
ub:teacherOf ?z . } union
{ ?x a ub:ResearchAssistant . ?y a ub:PostDoc . ?z a
ub:GraduateCourse . ?x ub:advisor ?y . ?x ub:takesCourse ?z . ?y
ub:teacherOf ?z . } union
{ ?x a ub:ResearchAssistant . ?y a ub:VisitingProfessor . ?z a
ub:GraduateCourse . ?x ub:advisor ?y . ?x ub:takesCourse ?z . ?y
ub:teacherOf ?z . } union
{ ?x a ub:ResearchAssistant . ?y a ub:AssistantProfessor . ?z a
ub:GraduateCourse . ?x ub:advisor ?y . ?x ub:takesCourse ?z . ?y
ub:teacherOf ?z . } union
{ ?x a ub:ResearchAssistant . ?y a ub:AssociateProfessor . ?z a
ub:GraduateCourse . ?x ub:advisor ?y . ?x ub:takesCourse ?z . ?y
ub:teacherOf ?z . } union
{ ?x a ub:ResearchAssistant . ?y a ub:FullProfessor . ?z a
ub:GraduateCourse . ?x ub:advisor ?y . ?x ub:takesCourse ?z . ?y
ub:teacherOf ?z . } union
{ ?x a ub:UndergraduateStudent . ?y a ub:Lecturer . ?z a

```

```

ub:Course . ?x ub:advisor ?y . ?x ub:takesCourse ?z . ?y
ub:teacherOf ?z . } union
  { ?x a ub:UndergraduateStudent . ?y a ub:PostDoc . ?z a
ub:Course . ?x ub:advisor ?y . ?x ub:takesCourse ?z . ?y
ub:teacherOf ?z . } union
  { ?x a ub:UndergraduateStudent . ?y a ub:VisitingProfessor . ?z
a ub:Course . ?x ub:advisor ?y . ?x ub:takesCourse ?z . ?y
ub:teacherOf ?z . } union
  { ?x a ub:UndergraduateStudent . ?y a ub:AssistantProfessor . ?
z a ub:Course . ?x ub:advisor ?y . ?x ub:takesCourse ?z . ?y
ub:teacherOf ?z . } union
  { ?x a ub:UndergraduateStudent . ?y a ub:AssociateProfessor . ?
z a ub:Course . ?x ub:advisor ?y . ?x ub:takesCourse ?z . ?y
ub:teacherOf ?z . } union
  { ?x a ub:UndergraduateStudent . ?y a ub:FullProfessor . ?z a
ub:Course . ?x ub:advisor ?y . ?x ub:takesCourse ?z . ?y
ub:teacherOf ?z . } union

  { ?x a ub:UndergraduateStudent . ?y a ub:Lecturer . ?z a
ub:GraduateCourse . ?x ub:advisor ?y . ?x ub:takesCourse ?z . ?y
ub:teacherOf ?z . } union
  { ?x a ub:UndergraduateStudent . ?y a ub:PostDoc . ?z a
ub:GraduateCourse . ?x ub:advisor ?y . ?x ub:takesCourse ?z . ?y
ub:teacherOf ?z . } union
  { ?x a ub:UndergraduateStudent . ?y a ub:VisitingProfessor . ?z
a ub:GraduateCourse . ?x ub:advisor ?y . ?x ub:takesCourse ?z . ?
y ub:teacherOf ?z . } union
  { ?x a ub:UndergraduateStudent . ?y a ub:AssistantProfessor . ?
z a ub:GraduateCourse . ?x ub:advisor ?y . ?x ub:takesCourse ?z .
?y ub:teacherOf ?z . } union
  { ?x a ub:UndergraduateStudent . ?y a ub:AssociateProfessor . ?
z a ub:GraduateCourse . ?x ub:advisor ?y . ?x ub:takesCourse ?z .
?y ub:teacherOf ?z . } union
  { ?x a ub:UndergraduateStudent . ?y a ub:FullProfessor . ?z a
ub:GraduateCourse . ?x ub:advisor ?y . ?x ub:takesCourse ?z . ?y
ub:teacherOf ?z . } union
  { ?x a ub:GraduateStudent . ?y a ub:Lecturer . ?z a ub:Course .
?x ub:advisor ?y . ?x ub:takesCourse ?z . ?y ub:teacherOf ?z . }
union
  { ?x a ub:GraduateStudent . ?y a ub:PostDoc . ?z a ub:Course .
?x ub:advisor ?y . ?x ub:takesCourse ?z . ?y ub:teacherOf ?z . }
union
  { ?x a ub:GraduateStudent . ?y a ub:VisitingProfessor . ?z a
ub:Course . ?x ub:advisor ?y . ?x ub:takesCourse ?z . ?y
ub:teacherOf ?z . } union
  { ?x a ub:GraduateStudent . ?y a ub:AssistantProfessor . ?z a
ub:Course . ?x ub:advisor ?y . ?x ub:takesCourse ?z . ?y
ub:teacherOf ?z . } union
  { ?x a ub:GraduateStudent . ?y a ub:AssociateProfessor . ?z a
ub:Course . ?x ub:advisor ?y . ?x ub:takesCourse ?z . ?y
ub:teacherOf ?z . } union
  { ?x a ub:GraduateStudent . ?y a ub:FullProfessor . ?z a
ub:Course . ?x ub:advisor ?y . ?x ub:takesCourse ?z . ?y
ub:teacherOf ?z . } union
  { ?x a ub:GraduateStudent . ?y a ub:Lecturer . ?z a

```

```

ub:GraduateCourse . ?x ub:advisor ?y . ?x ub:takesCourse ?z . ?y
ub:teacherOf ?z . } union
  { ?x a ub:GraduateStudent . ?y a ub:PostDoc . ?z a
ub:GraduateCourse . ?x ub:advisor ?y . ?x ub:takesCourse ?z . ?y
ub:teacherOf ?z . } union
  { ?x a ub:GraduateStudent . ?y a ub:VisitingProfessor . ?z a
ub:GraduateCourse . ?x ub:advisor ?y . ?x ub:takesCourse ?z . ?y
ub:teacherOf ?z . } union
  { ?x a ub:GraduateStudent . ?y a ub:AssistantProfessor . ?z a
ub:GraduateCourse . ?x ub:advisor ?y . ?x ub:takesCourse ?z . ?y
ub:teacherOf ?z . } union
  { ?x a ub:GraduateStudent . ?y a ub:AssociateProfessor . ?z a
ub:GraduateCourse . ?x ub:advisor ?y . ?x ub:takesCourse ?z . ?y
ub:teacherOf ?z . } union
  { ?x a ub:GraduateStudent . ?y a ub:FullProfessor . ?z a
ub:GraduateCourse . ?x ub:advisor ?y . ?x ub:takesCourse ?z . ?y
ub:teacherOf ?z . }
};

```

-- Q10

```

sparql prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#>
select * from <lubm>
where
{
{ ?x a ub:ResearchAssistant . ?x ub:takesCourse
<http://www.Department0.University0.edu/GraduateCourse0> . }
union
{ ?x a ub:UndergraduateStudent . ?x ub:takesCourse
<http://www.Department0.University0.edu/GraduateCourse0> . }
union
{ ?x a ub:GraduateStudent . ?x ub:takesCourse
<http://www.Department0.University0.edu/GraduateCourse0> . }
};

```

-- Q11

```

sparql prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#> select * from <lubm> where { ?x a ub:ResearchGroup .
?x ub:subOrganizationOf <http://www.University0.edu> . };

```

-- Q12

```

sparql prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#> select * from <lubm> where
  {
    { ?x a ub:FullProfessor . ?y a ub:Department . ?x
ub:headOf ?y . ?y ub:subOrganizationOf
<http://www.University0.edu> . }
    union
    { ?x a ub:AssistantProfessor . ?y a ub:Department . ?x
ub:headOf ?y . ?y ub:subOrganizationOf
<http://www.University0.edu> . }
    union
    { ?x a ub:AssociateProfessor . ?y a ub:Department . ?x
ub:headOf ?y . ?y ub:subOrganizationOf
<http://www.University0.edu> . }
  }

```

```
};
```

```
-- Q13
```

```
sparql prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-bench.owl#> select * from <lubm> where
```

```
{  
  { ?x a ub:AssociateProfessor . ?x ub:doctoralDegreeFrom  
<http://www.University0.edu> . }  
  union  
  { ?x a ub:FullProfessor . ?x ub:doctoralDegreeFrom  
<http://www.University0.edu> . }  
  union  
  { ?x a ub:AssistantProfessor . ?x ub:doctoralDegreeFrom  
<http://www.University0.edu> . }  
  union  
  { ?x a ub:Lecturer . ?x ub:doctoralDegreeFrom  
<http://www.University0.edu> . }  
  union  
  { ?x a ub:UndergraduateStudent . ?x ub:doctoralDegreeFrom  
<http://www.University0.edu> . }  
  union  
  { ?x a ub:GraduateStudent . ?x ub:doctoralDegreeFrom  
<http://www.University0.edu> . }  
  union  
  { ?x a ub:TeachingAssistant . ?x ub:doctoralDegreeFrom  
<http://www.University0.edu> . }  
  union  
  { ?x a ub:ResearchAssistant . ?x ub:doctoralDegreeFrom  
<http://www.University0.edu> . }  
  union  
  
  { ?x a ub:AssociateProfessor . ?x ub:mastersDegreeFrom  
<http://www.University0.edu> . }  
  union  
  { ?x a ub:FullProfessor . ?x ub:mastersDegreeFrom  
<http://www.University0.edu> . }  
  union  
  { ?x a ub:AssistantProfessor . ?x ub:mastersDegreeFrom  
<http://www.University0.edu> . }  
  union  
  { ?x a ub:Lecturer . ?x ub:mastersDegreeFrom  
<http://www.University0.edu> . }  
  union  
  { ?x a ub:UndergraduateStudent . ?x ub:mastersDegreeFrom  
<http://www.University0.edu> . }  
  union  
  { ?x a ub:GraduateStudent . ?x ub:mastersDegreeFrom  
<http://www.University0.edu> . }  
  union  
  { ?x a ub:TeachingAssistant . ?x ub:mastersDegreeFrom  
<http://www.University0.edu> . }  
  union  
  { ?x a ub:ResearchAssistant . ?x ub:mastersDegreeFrom  
<http://www.University0.edu> . }  
}
```

```

union
{ ?x a ub:AssociateProfessor . ?x ub:undergraduateDegreeFrom
<http://www.University0.edu> . }
union
{ ?x a ub:FullProfessor . ?x ub:undergraduateDegreeFrom
<http://www.University0.edu> . }
union
{ ?x a ub:AssistantProfessor . ?x ub:undergraduateDegreeFrom
<http://www.University0.edu> . }
union
{ ?x a ub:Lecturer . ?x ub:undergraduateDegreeFrom
<http://www.University0.edu> . }
union
{ ?x a ub:UndergraduateStudent . ?x ub:undergraduateDegreeFrom
<http://www.University0.edu> . }
union
{ ?x a ub:GraduateStudent . ?x ub:undergraduateDegreeFrom
<http://www.University0.edu> . }
union
{ ?x a ub:TeachingAssistant . ?x ub:undergraduateDegreeFrom
<http://www.University0.edu> . }
union
{ ?x a ub:ResearchAssistant . ?x ub:undergraduateDegreeFrom
<http://www.University0.edu> . }

}

;

-- Q14
sparql prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#> select * from <lubm> where { ?x a
ub:UndergraduateStudent . };

```

Query Text With Inference Options

```

set autocommit on;

-- Q1
sparql
define input:inference 'inft'
prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#>
select * from <lubm>
where { ?x rdf:type ub:GraduateStudent . ?x ub:takesCourse
<http://www.Department0.University0.edu/GraduateCourse0> };

-- Q2
sparql
define input:inference 'inft'
prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#>

```

```

select * from <lubm>
where { ?x a ub:GraduateStudent . ?y a ub:University . ?z a
ub:Department . ?x ub:memberOf ?z . ?z ub:subOrganizationOf ?y .
?x ub:undergraduateDegreeFrom ?y };

-- Q3
sparql
define input:inference 'inft'
prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#>
select * from <lubm>
where { ?x a ub:Publication . ?x ub:publicationAuthor
<http://www.Department0.University0.edu/AssistantProfessor0> };

-- Q4
sparql
define input:inference 'inft'
prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#>
select distinct * from <lubm>
where { ?x a ub:Professor . ?x ub:worksFor
<http://www.Department0.University0.edu> . ?x ub:name ?y1 . ?x
ub:emailAddress ?y2 . ?x ub:telephone ?y3 . };

-- Q5
sparql
define input:inference 'inft'
prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#>
select distinct * from <lubm>
where { ?x a ub:Person . ?x ub:memberOf
<http://www.Department0.University0.edu> };

-- Q6
sparql
define input:inference 'inft'
prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#>
select distinct * from <lubm> where { ?x a ub:Student . };

-- Q7
sparql
define input:inference 'inft'
prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#>
select distinct * from <lubm>
where { ?x a ub:Student . ?y a ub:Course .
<http://www.Department0.University0.edu/AssociateProfessor0>
ub:teacherOf ?y . ?x ub:takesCourse ?y . };

-- Q8: XXX
sparql
define input:inference 'inft'
prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#>

```

```

select distinct * from <lubm>
where { ?x a ub:Student . ?y a ub:Department . ?x ub:memberOf ?y
. ?y ub:subOrganizationOf <http://www.University0.edu> . ?x
ub:emailAddress ?z };

-- Q9: XXX
sparql
define input:inference 'inft'
prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#>
select distinct * from <lubm>
where { ?x a ub:Student . ?y a ub:Faculty . ?z a ub:Course . ?x
ub:advisor ?y . ?x ub:takesCourse ?z . ?y ub:teacherOf ?z . };

-- Q10
sparql
define input:inference 'inft'
prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#>
select * from <lubm>
where { ?x a ub:Student . ?x ub:takesCourse
<http://www.Department0.University0.edu/GraduateCourse0> . };

-- Q11
sparql
define input:inference 'inft'
prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#> select * from <lubm> where { ?x a ub:ResearchGroup .
?x ub:subOrganizationOf <http://www.University0.edu> . };

-- Q12
sparql
define input:inference 'inft'
prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#> select * from <lubm> where { ?x a ub:Professor . ?y a
ub:Department . ?x ub:headOf ?y . ?y ub:subOrganizationOf
<http://www.University0.edu> . };

-- Q13
sparql
define input:inference 'inft'
prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#> select * from <lubm> where { ?x a ub:Person . ?x
ub:degreeFrom <http://www.University0.edu> . };

-- Q14
sparql
define input:inference 'inft'
prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#> select * from <lubm> where { ?x a
ub:UndergraduateStudent . };

```

Appendix C Query Text With Materialized Entailed Triples

```

set autocommit on;

-- Q1
sparql prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#>
select * from <lubm>
where { ?x rdf:type ub:GraduateStudent . ?x ub:takesCourse
<http://www.Department0.University0.edu/GraduateCourse0> };

-- Q2
sparql prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#>
select * from <lubm>
where { ?x a ub:GraduateStudent . ?y a ub:University . ?z a
ub:Department . ?x ub:memberOf ?z . ?z ub:subOrganizationOf ?y .
?x ub:undergraduateDegreeFrom ?y };

-- Q3
sparql prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#>
select * from <lubm>
where { ?x a ub:Publication . ?x ub:publicationAuthor
<http://www.Department0.University0.edu/AssistantProfessor0> };

-- Q4
sparql prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#>
select * from <lubm>
where { ?x a ub:Professor . ?x ub:worksFor
<http://www.Department0.University0.edu> . ?x ub:name ?y1 . ?x
ub:emailAddress ?y2 . ?x ub:telephone ?y3 . };

-- Q5
sparql prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#>
select * from <lubm>
where { ?x a ub:Person . ?x ub:memberOf
<http://www.Department0.University0.edu> };

-- Q6
sparql prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#>
select * from <lubm> where { ?x a ub:Student . };

-- Q7
sparql prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#>
select * from <lubm>
where { ?x a ub:Student . ?y a ub:Course .
<http://www.Department0.University0.edu/AssociateProfessor0>
ub:teacherOf ?y . ?x ub:takesCourse ?y . };

-- Q8
sparql prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#>

```

```

select * from <lubm>
where { ?x a ub:Student . ?y a ub:Department . ?x ub:memberOf ?y
. ?y ub:subOrganizationOf <http://www.University0.edu> . ?x
ub:emailAddress ?z };

-- Q9
sparql prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#>
select * from <lubm>
where { ?x a ub:Student . ?y a ub:Faculty . ?z a ub:Course . ?x
ub:advisor ?y . ?x ub:takesCourse ?z . ?y ub:teacherOf ?z . };

-- Q10
sparql prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#>
select * from <lubm>
where { ?x a ub:Student . ?x ub:takesCourse
<http://www.Department0.University0.edu/GraduateCourse0> . };

-- Q11
sparql prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#> select * from <lubm> where { ?x a ub:ResearchGroup .
?x ub:subOrganizationOf <http://www.University0.edu> . };

-- Q12
sparql prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#> select * from <lubm> where { ?x a ub:Professor . ?y a
ub:Department . ?x ub:headOf ?y . ?y ub:subOrganizationOf
<http://www.University0.edu> . };

-- Q13
sparql prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#> select * from <lubm> where { ?x a ub:Person . ?x
ub:degreeFrom <http://www.University0.edu> . };

-- Q14
sparql prefix ub: <http://www.lehigh.edu/~zhp2/2004/0401/univ-
bench.owl#> select * from <lubm> where { ?x a
ub:UndergraduateStudent . };

```

Appendix B Configuration

Single process, 8G RAM.

The following lines were changed in the default virtuoso.ini file:

```

NumberOfBuffers = 550000
MaxCheckpointRemap = 2000000
Striping = 1

[Striping]
# One file per disk, with distinct IO queue
Segment1          = 100G /disk1/db1-1.db q1, /disk2/db1-2.db q2 #

```

and so on
