Large-scale Reasoning with a Complex Cultural Heritage Ontology (CIDOC CRM)

Vladimir Alexiev, Dimitar Manov, Jana Parvanova, Svetoslav Petrov

Ontotext Corp, Sofia, Bulgaria
`Email: {first.last}@ontotext.com`

Abstract. The CIDOC Conceptual Reference Model (CRM) is an important ontology in the Cultural Heritage (CH) domain. CRM is intended mostly as a data integration mechanism, allowing reasoning and discoverability across diverse CH sources represented in CRM. CRM data comprises complex graphs of nodes and properties. An important question is how to search through such complex graphs, since the number of possible combinations is staggering. One answer is the "Fundamental Relations" (FR) approach that maps whole networks of CRM properties to fewer FRs, serving as a "search index" over the CRM semantic web.

We present performance results for an FR Search implementation based on OWLIM. This search works over a significant CH dataset: almost 1B statements resulting from 2M objects of the British Museum. This is an exciting demonstration of large-scale reasoning with real-world data over a complex ontology (CIDOC CRM). We present volumetrics, hardware specs, compare the numbers to other repositories hosted by Ontotext, performance results, and compare performance of a SPARQL implementation.

Keywords: CIDOC CRM, cultural heritage, semantic search, Fundamental Relations, OWLIM, semantic repository, inference, performance, benchmark

1 Introduction

The CIDOC Conceptual Reference Model (CRM)[1] is an important ontology in the Cultural Heritage (CH) domain. CRM is intended mostly as a data integration mechanism, allowing reasoning and discoverability across diverse CH sources represented in CRM. CRM data comprises complex graphs of nodes and properties. An important question is how to search through such complex graphs, since the number of possible combinations is staggering. The "Fundamental Relations" (FR) approach [2,3] "compresses" the semantic network by mapping whole networks of CRM properties to fewer FRs that serve as a "search index" over the CRM semantic web and allow the user to use a simpler query vocabulary.

[1] http://www.cidoc-crm.org

In [1] we published an implementation of CRM FRs created within the ResearchSpace project[2] using OWLIM[3] [4], and presented preliminary performance results. Here we present a revised implementation, provide volumetrics and hardware specs, performance results over the full CRM repository comprising over 2M CH objects of the British Museum (BM), compare the numbers to other repositories hosted by Ontotext, and compare the performance to one based on SPARQL.

2 Specifics

2.1 Implemented FRs

We implemented the following FRs. Compared to [1] these are adjusted after initial experimentation and gained user experience in RS. Each FR has domain Thing and range indicated in parentheses. rso:E55_Technique is a subclass of crm:E55_Type that we use for focused searching of Techniques. The last 5 FRs (17-23) are special extensions:

1. rso:FR92i_created_by (crm:E39_Actor): Thing (or part/inscription thereof) was created or modified/repaired by Actor (or group it is member of, e.g. Nationality)
2. rso:FR15_influenced_by (crm:E39_Actor): Thing's production was influenced/motivated by Actor (or group it is member of). E.g.: Manner/ School/ Style of; or Issuer, Ruler, Magistrate who authorised, patronised, ordered the production.
3. rso:FR52_current_owner_keeper (crm:E39_Actor): Thing has current owner or keeper Actor
4. rso:FR51_former_or_current_owner_keeper (crm:E39_Actor): Thing has former or current owner or keeper Actor, or ownership/custody was transferred from/to actor in Acquisition/Transfer of Custody event
5. rso:FR67_about_actor (crm:E39_Actor): Thing depicts or refers to Actor, or carries an information object that is about Actor, or bears similarity with a thing that is about Actor
6. rso:FR12_has_met (crm:E39_Actor): Thing (or another thing it is part of) has met actor in the same event (or event that is part of it)
7. rso:FR67_about_period (crm:E4_Period): Thing depicts or refers to Event/Period, or carries an information object that is about Event, or bears similarity with a thing that is about Event
8. rso:FR12_was_present_at (crm:E4_Period): Thing was present at Event (eg exhibition) or is from Period
9. rso:FR92i_created_in (crm:E53_Place): Thing (or part/inscription thereof) was created or modified/repaired at/in place (or a broader containing place)
10. rso:FR55_located_in (crm:E53_Place): Thing has current or permanent location in Place (or a broader containing place)

[2] http://www.researchspace.org
[3] http://www.ontotext.com/owlim

11. rso:FR12_found_at (crm:E53_Place): Thing was found (discovered, excavated) at Place (or a broader containing place)
12. rso:FR7_from_place (crm:E53_Place): Thing has former, current or permanent location at place, or was created/found at place, or moved to/from place, or changed ownership/custody at place (or a broader containing place)
13. rso:FR67_about_place (crm:E53_Place): Thing depicts or refers to a place or feature located in place, or is similar in features or composed of or carries an information object that depicts or refers to a place
14. rso:FR2_has_type (crm:E55_Type): Thing is of Type, or has Shape, or is of Kind, or is about or depicts a type (e.g. IconClass or subject heading)
15. rso:FR45_is_made_of (crm:E57_Material): Thing (or part thereof) consists of material
16. rso:FR32_used_technique (rso:E55_Technique): The production of Thing (or part thereof) used general technique
17. luc:myIndex (rdfs:Literal): The full text of the thing's description (including thesaurus terms and textual descriptions) matches the given keyword. FTS using Lucene built into OWLIM.
18. rso:FR108i_82_produced_within (rdfs:Literal): Thing was created within an interval that intersects the given interval or year.
19. rso:FR1_identified_by (rdfs:Literal): Thing (or part thereof) has Identifier. Exact-match string
20. rso:FR138i_has_representation (xsd:boolean): Thing has at least one image representation. Used to select objects that have images
21. rso:FR138i_representation (crm:E38_Image): Thing has image representation. Used to fetch all images of an object
22. rso:FR_main_representation (crm:E38_Image): Thing has main image representation. Used to display object thumbnail in search results
23. rso:FR_dataset (rdfs:Literal): Thing belongs to indicated dataset. Used for faceting by dataset

2.2 OWLIM Rules

We used OWLIM Rules to implement the FRs: a total of 120 rules:

- 14 rules implement RDFS reasoning, a small subset of OWL (owl:TransitiveProperty, owl:inverseOf) and ptop:transitiveOver from the PROTON ontology[4]. These are copied from standard rulesets, as described in [5]
- 106 rules implement FRs. We use a method of decomposing the network of an FR in pieces [1]: conjunctive (e.g. checking the type of a node), disjunctive (parallel), serial (property path), transitive. We implement each piece as a sub-FR and use it to build up bigger pieces.

To deal with the complexity of implementation, we used several approaches:

[4] http://www.ontotext.com/proton-ontology

- A rule shortcut syntax that renders each rule on one line, instead of a line for each premise and conclusion
- A literate programming style, where rule definitions are interspersed with diagrams, discussion and justification in a wiki
- Checking that only known properties and classes are used in the rules (the dependency graph in the next section helped for this)
- Checking that rule variables are used in a linear way (premise variables make a chain, and the conclusion uses the ends of the chain), or in type checks. E.g.

```
x <rdf:type> <rso:FC70_Thing>; x <crm:P46_is_composed_of> y  => x <rso:FRT_46_106_148> y
x <rso:FRT_46_106_148> y; y <crm:P46_is_composed_of> z => x <rso:FRT_46_106_148> z
p <ptop:transitiveOver> q; x p y; y q z => x p z
```

(a) First rule: x is used in a type check, and x-y=>x-y is a linear chain.
(b) Second rule: x-y;y-z=>x-z is a linear chain.
(c) Third rule: p and q are **not** used in a linear way. These variables are in "property" position, and our check skips such variables

A lot more implementation details can be found at the ResearchSpace wiki[5]. The following OWLIM reasoning features[6] were important for the implementation:

- Custom rule-sets. The standard semantics that OWLIM supports (RDFS, RDFS Horst, OWL RL, QL and DL) are also implemented as rulesets.
- Fully-materializing forward-chaining reasoning. Rule consequences are stored in the repository and query answering is very fast.
- sameAs optimization that allows fast cross-collection search using coreferenced values (e.g. Agent URIs)
- Incremental retraction: when a triple is deleted, OWLIM removes all inferred consequences that are left without support (recursively). In order to facilitate this, OWLIM rules have a simple syntax, so they can be checked in "reverse".
- Incremental insert: when a triple is inserted (even an ontology triple), all rules are checked. If a rule fires, the new conclusion is also checked against the rules, etc.
- Efficient rule execution: rules are compiled to Java and executed quickly. For example, we decided late in the game that we want FR45 "Thing is made of Material" to be transitive over the "broader" hierarchy. We added the 2 triples below, and 1M new triples were inferred within 10 minutes (see the implementation of ptop:transitiveOver in (c) above).

```
rso:FR45_is_made_of ptop:transitiveOver skos:broader, crm:P127_has_broader_term.
```

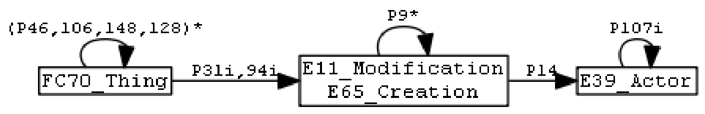
OWLIM rules also have their disadvantages, as described in [1] section 5.3. Chief among them is inflexibility: if the ruleset is changed, the OWLIM server needs to be restarted. Furthermore, if the ruleset should infer different conclusions from the exist-

[5] https://confluence.ontotext.com/display/ResearchSpace/FR+Implementation
[6] http://owlim.ontotext.com/display/OWLIMv53/OWLIM-SE+Reasoner

ing triples, the repository needs to be reloaded. But newly added triples are checked against the rules, as shown in the previous example.

2.3 Example: FR92i_created_by

As an example, let's consider FR92i_created_by "Thing created by Actor", which we define as "Thing (or part/inscription thereof) was created or modified/repaired by Actor (or a group it is a member of)":



This FR includes the following source properties:

- P46_is_composed_of, P106_is_composed_of, P148_has_component: navigates object part hierarchy
- P128_carries: to transition from object to Inscription carried by it
- P31i_was_modified_by (includes P108i_was_produced_by), P94i_was_created_by: Modification/Production of physical thing, Creation of conceptual thing (Inscription)
- P9_consists_of: navigates event part hierarchy (BM models uncorrelated production facts as sub-events)
- P14_carried_out_by, P107i_is_current_or_former_member_of: agent and groups he's member of

This FR uses a previously defined sub-FR FRT_46_106_148_128 (the first loop) and defines another sub-FR:

- FRX92i_created := (FC70_Thing) FRT_46_106_148_128* / (P31i | P94i) / P9*

The sub-FR extends to the Modification/Creation node including the P9 loop and is implemented with 5 rules:

```
x <rdf:type> <rso:FC70_Thing>; x <crm:P31i_was_modified_by> y => x <rso:FRX92i_created> y
x <rdf:type> <rso:FC70_Thing>; x <crm:P94i_was_created_by>  y => x <rso:FRX92i_created> y
x <rso:FRT_46_106_148_128> y; y <crm:P31i_was_modified_by> z => x <rso:FRX92i_created> z
x <rso:FRT_46_106_148_128> y; y <crm:P94i_was_created_by>  z => x <rso:FRX92i_created> z
x <rso:FRX92i_created> y; y <crm:P9_consists_of> z => x <rso:FRX92i_created> z
```

Finally, the FR uses the sub-FR (which also reused in another FR!), and is implemented with 2 rules:

- FR92i_created_by := FRX92i_created / P14 / P107i*

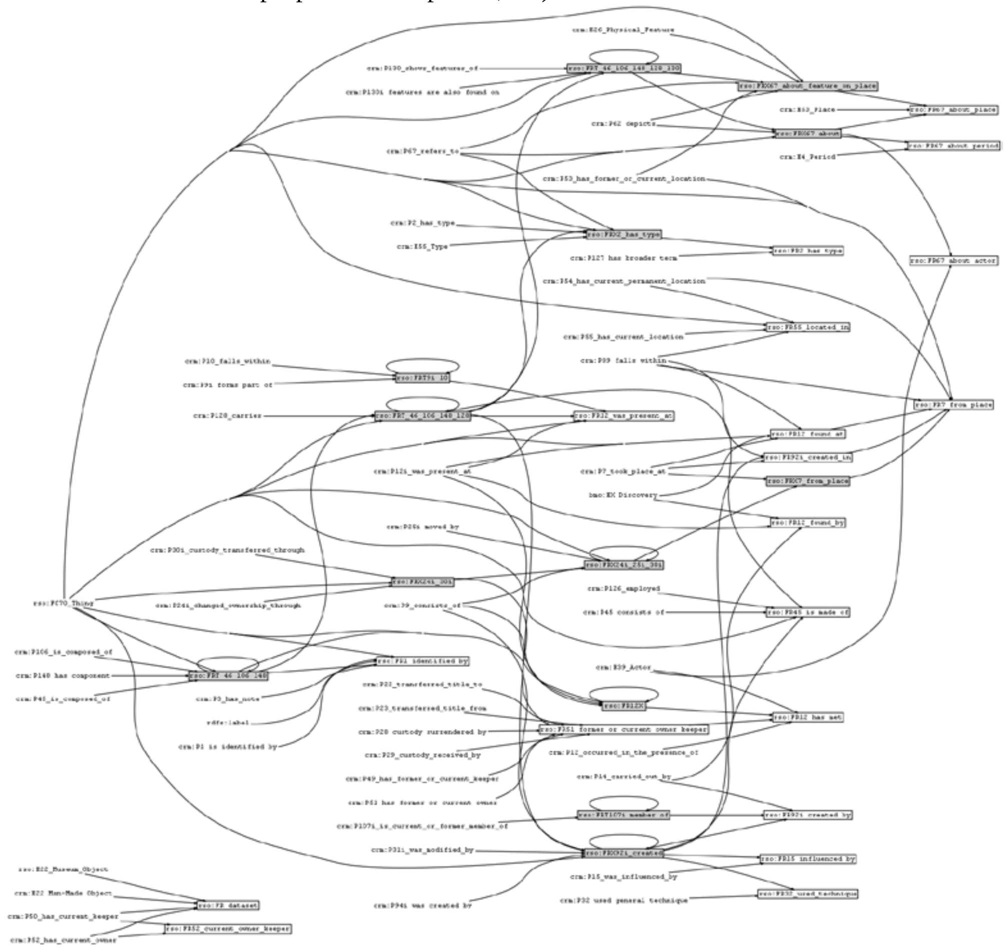
```
x <rso:FRX92i_created> y; y <crm:P14_carried_out_by> z => x <rso:FR92i_created_by> z
x <rso:FRX92i_created> y; y <crm:P14_carried_out_by> z; z <rso:FRT107i_member_of> t =>
  x <rso:FR92i_created_by> t
```

2.4 Sub-FRs and Dependency Graph

The dependency graph of our implementation is shown below, a zoomable version is also available[7]. It has:

- 51 source classes/properties, shown as plain text
- 13 intermediate sub-FRs, shown as filled rectangles. These sub-FRs are used by several FRs to simplify the implementation
- 19 target FRs, shown as rectangles

The diagram illustrates the complexity of the implementation. We used it to verify the implementation as OWLIM rules (e.g. that there are no disconnected properties, each FR uses all source properties as expected, etc).



[7] https://www.dropbox.com/s/6lz48qfdbitkvlk/FR-graph.png

2.5 Hardware Specification

- CPU: Intel Xeon E5-2650 2.00GHz, 20M Cache, 32 cores
- RAM: 128 GB RDIMM, 1600 MHz
- Solid-State Disks: 4*200GB SSD, SATA.
- Hard Disks: 3*300GB, SAS 6Gbps, 2.5-in, 15K RPM.
- Server cost: under $10k.

Large-scale OWLIM deployments are recommended to use SSD for faster disk speed, and the zFS compressing file-system for better SSD utilization and even faster speed. zFS is native for Solaris, but we now have successful deployments on Linux as well. This system has a lot of spare capacity: the hard disks and zFS are currently not used.

2.6 Volumetrics

Some numeric data of our implementation, with discussion:

- Museum objects: 2,051,797 (entities with type rso:FC70_Thing). Most of these are from the British Museum. We are currently completing the ingest of Yale Center for British Art objects into ResearchSpace.
- Thesaurus entries: 415,509 (type skos:Concept). All kinds of "fixed" values that are used for search: object types, materials, techniques, people, places, … (a total of 90 ConceptSchemes)
- Explicit statements: 195,208,156. We estimate that of these, 185M are for objects (90 statements/object) and 9M are for thesaurus entries (22 statements/term).
- Total statements: 916,735,486. The expansion ratio is **4.7x** (i.e. for each statement, 3.7 more are inferred). This is considerably higher compared to the typical expansion for general datasets (e.g. DBpedia, GeoNames, FactForge) that is 1.2 - 2x, and is due to the complexity described below.
- Nodes (unique URLs and literals): 53,803,189. (We don't use blank nodes)
- Repository size: 42 Gb, object full-text index: 2.5 Gb, thesaurus full-text index (used for search auto-complete): 22Mb.
- Loading time (including all inferencing): 22.2h on RAM drive; 32.9h on hard-disks.

2.7 Complexity: Classes

CIDOC CRM is a complex ontology. The deepest branch of the class hierarchy[8] is 10 levels: E1>E77>E70>E71>E28>E90>E73>E36>E37>E34_Inscription. Furthermore, multiple inheritance is used extensively, e.g. E33 is also a super-class of E34_Inscription. For each inscription, 12 type statements are inferred. We use the Erlangen CRM mapping to OWL[9] because it provides inverse and transitive proper-

[8] http://www.cidoc-crm.org/cidoc_graphical_representation_v_5_1/class_hierarchy.html
[9] http://erlangen-crm.org

ties. But it includes a lot of owl:Restriction anonymous classes, e.g. (in Manchester notation)

```
E30_Right SubClassOf: P104i_applies_to some E72_Legal_Object
```

These anonymous classes are useless to us, so we wrote a tool that derives simpler profiles of Erlangen CRM. Even with this simplification, type statements alone are 302,149,587 or 37% of the total. The number of types is 238. We counted statements per type with this query and present some of the top types:

```
select ?t (count(*) as ?c) {?o a ?t} group by ?t
```

Class	**Statements**
owl:Thing	36485904
E1_CRM_Entity□	36485903
E77_Persistent_Item□	17408450
E70_Thing□	17339714
E71_Man-Made_Thing□	17216212
E72_Legal_Object□	17192518
E28_Conceptual_Object□	14776488
E90_Symbolic_Object□	14629292
E2_Temporal_Entity□	11924877
E4_Period□	11924877
E5_Event□	11922986
E7_Activity□	11796470
E63_Beginning_of_Existence□	6377421
E11_Modification□	6296015
E12_Production□	6295825
rso:FC70_Thing	2051797
skos:Concept	415509

Comments (look at the class hierarchy as well): we have 415k terms (skos:Concept) and 2M museum objects (FC70_Thing). These objects have 6.3M E12_Production records, which are repeated as the super-class E11_Modification; there are a few hundred Repairs mapped to E11, over and above the E12 number. E12 is also repeated as E63_Beginning_of_Existence; which has additional 100k records of Birth and Formation for the Person-Institution thesaurus. Another 5.4M E7_Activity records stand for Acquisition, Discovery, exhibition, etc. Each E7 is repeated as E5_Event, which is repeated as E4_Period (with an extra 19k historic Periods) and E2_Temporal_Entity; etc.

A lot of the higher-level classes are too abstract to be useful for querying (e.g. E1_CRM_Entity, E70_Thing, E77_Persistent_Item, E72_Legal_Object. But OWLIM materializes all inferences and unfortunately doesn't offer options for controlling which ones to materialize.

2.8 Complexity: Properties

(Note: the analysis below is based on a slightly older version of the repository with 806M statements instead of 917M statements. But the percentages and conclusions are approximately the same.)

We have a total of 339 properties. We analyzed the statement distribution per property with this query:

```
select ?p (count(*) as ?c) {?s ?p ?o} group by ?p
```

Properties	**Statements**	**Percent**
rdf:type	302149587	37.50%
Objects: CRM, rdfs:label	365430152	45.35%
Extensions: BMO, RSO	35903831	4.46%
FRs (70M=9%) and sub-FRs (26M=3%)	96526377	11.98%
Thesauri: BIBO, DC, DCT, FOAF, SKOS, QUDT, VAEM	5715250	0.71%
Ontology: RDF, RDFS, OWL	4159	0.00%
Total	**805729356**	**100.00%**
CRM inverses	149465596	18.55%

- **Type** statements take a significant proportion, analyzed in the previous section.
- Statements related to O**bjects** are the majority (365M or 45% of total). Chief amongst them are P3_has_note (10.10% of the Object statements), P2_has_type (6.49%), P12_occurred_in_the_presence_of (3.29%)
  - rdfs:label is also significant (13.9M or 3.81% of the Object statements). We estimate that 5M of rdfs:label statements are due to Thesauri and should be moved from row Objects to row Thesauri.
  - A lot of the CRM properties have inverses (79 properties in our system). They are useful when writing rules and queries, but create a significant number of duplicate statements (18.6% of total: included in row Objects, and shown separately on the last row)
- **Extensions** are sub-properties of CRM, following the CRM extensibility guidelines. CRM itself uses sub-properties extensively. The maximum depth of the property hierarchy is 4, e.g.: P12_occurred_in_the_presence_of> P11_had_participant> P14_carried_out_by> P22_transferred_title_to.

2.9 Comparison to Other Repositories

Below is a comparison of the RS CRM repository to some repositories hosted by Ontotext and PSNC and provided as SPARQL public services. In each cell we show the absolute number (in Millions, except for Expansion and Density) and the ratio compared to RS CRM. **Expansion**=Total statements/Explicit statements shows the intensity of inference. **Density**=Statements/Nodes shows the relative density of the graph. **Objects** is not defined for the last two repositories, since they cover broad domains and the objects are too heterogeneous.

Repo	Objects		Expl.st.		Ex.st/obj		Total st.		Expans.		Nodes		Density		Reasoning
CRM	2.0	1	195	1	90	1	916	1	4.7	1	54	1	17.0	1	rdfs+tr+FR
PSNC	3.1	1.5	234	1.2	75	0.83	535	0.58	2.3	0.49	60	1.1	8.9	0.52	rdfs-subCl
EDM	20.3	9.8	998	5.1	50	0.56	3798	4.1	3.8	0.8	266	4.9	14.3	0.84	owl-horst
FF			1673	8.6			3211	3.5	1.9	0.4	456	8.4	7.0	0.41	owl-horst
LLD			6706	34			10192	11	1.5	0.3	1554	29	6.6	0.38	rdfs+trans

- RS CRM: http://test.researchspace.com:8081/sparql, the subject of this paper
- PSNC: Polish Digital Library: http://dl.psnc.pl, national aggregation using FRBRoo and CRM. Subclass inference is disabled to avoid a proliferation of type statements (see section 2.7). See section 4.2 for more details about his repository.
- EDM: http://europeana.ontotext.com: Europeana data, snapshot of 14.9.2012.
- FF: http://www.factforge.net [6]: an RDF warehouse and Reason-able View including 10 of the most important LOD datasets of general interest: FreeBase, DBpedia, GeoNames, MusicBrainz, etc. FactForge reasoning is described in [7]
- LLD: http://linkedlifedata.com: a semantic data integration platform for the biomedical domain

Observations: The RS CRM repository is of moderate size compared to others (but is expected to grow as more partners join RS). CRM expresses objects in considerably more detail than all other repositories, even EDM. This can be seen in both ratios **Ex.st/obj** (explicit statements per object) and **Density** (total statements per node).

3 Performance

3.1 Performance of SPARQL Implementation

FRs can be implemented by composing straight SPARQL queries. For example, the query for FR92i_created_by (sec. 2.3) can be defined like this using SPARQL 1.1 Property Paths [8], and you can try it at the RS CRM endpoint[10]:

```
select ?obj $act {
 ?obj a rso:FC70_Thing;
  (crm:P46_is_composed_of|crm:P106_is_composed_of|crm:P148_has_component|crm:P128_carries)*/
  (crm:P31i_was_modified_by|crm:P94i_was_created_by) / crm:P9_consists_of* /
   crm:P14_carried_out_by / crm:P107i_is_current_or_former_member_of*
 $act
} limit 20
```

The first few objects returned are Rembrandt paintings from the RKD dataset. $act is bound to rkd-artist:Rembrandt, and also to groups that he belongs to: profession/draughtsman, profession/printmaker, nationality:Dutch (conversely, the user can search by such groups).

In the RS system, $act is bound to an input variable and ?obj is the output variable:

[10] http://test.researchspace.org:8081/sparql (ask the authors for login)

```
select distinct ?obj {
 ?obj a rso:FC70_Thing;
   (crm:P46_is_composed_of|crm:P106_is_composed_of|crm:P148_has_component|crm:P128_carries)*/
   (crm:P31i_was_modified_by|crm:P94i_was_created_by) / crm:P9_consists_of* /
    crm:P14_carried_out_by / crm:P107i_is_current_or_former_member_of*
 rkd-artist:Rembrandt
} limit 20
```

The endpoint takes over 15 minutes to answer the query. If you add more clauses, the performance is even worse. The query can be optimized a bit by using intermediate variables instead of property paths, but the performance is still untenable.

3.2 Performance of FR Implementation

The same query, using FR92i_created_by as defined in sec. 2.3, is trivial and has **sub-second response time**:

```
select distinct ?obj {?obj rso:FR92i_created_by rkd-artist:Rembrandt} limit 500
```

Currently RS imposes a limit of 500 results due to browser memory limitations of the used faceting system (Exhibit 2), but even the full set of 1418 objects is returned within a second.

Now let's add some complexity: let's find **drawings** by Rembrandt that are about **mammals**. We first need to find the corresponding thesaurus terms, e.g.

```
select * {?s rdfs:label "drawing"}
select * {?s rdfs:label "mammal"}
```

The query uses another FR from the list in sec. 2.1: FR2_has_type (which is used to relate to any E55_Type term, no matter whether it relates to the **isness** or **aboutness** of the object):

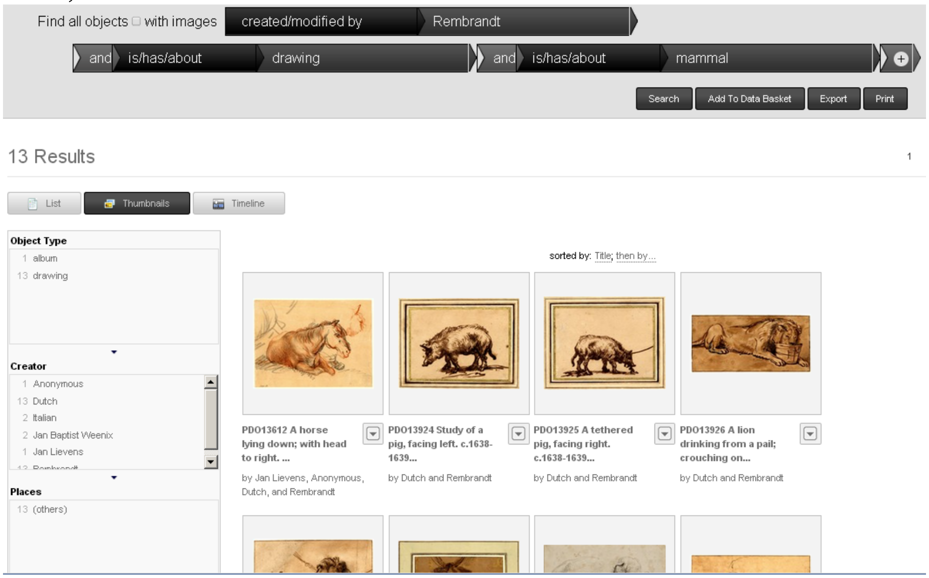
```
select distinct ?obj {
 ?obj rso:FR92i_created_by rkd-artist:Rembrandt;
      rso:FR2_has_type thes:x6544, thes:x12965
} limit 500
```

The query takes less than a second and returns 13 objects. None of them has subject "mammals" per se: they are about horses, pigs, lions, camels and an elephant (see next screen-shot). But the corresponding FR is defined as transitiveOver skos:broader, so it navigates the term hierarchy.

Materializing the FR triples adds 12% to the repository size (see sec. 2.8), which has negligible slow-down on basic querying speed. As shown in sec. 2.9, OWLIM has been used successfully on much bigger repositories, so this extra size is not a concern.

3.3 RS Semantic Search

RS uses the above to implement Semantic Search with controlled vocabularies and faceting. The user enters terms using auto-completion, RS restricts to FRs applicable to the specific term (e.g. created/modified is applicable to Agents, whereas is/has/about is applicable to concepts such as Object type, Subject, etc) and constructs a "search sentence". Here is a screen shot; you can view a video[11] of RS search in action, or ask the authors for a demo.


Find all objects with images created/modified by Rembrandt and is/has/about drawing and is/has/about mammal

13 Results

sorted by: Title; then by...

Object Type
1 album
13 drawing

Creator
1 Anonymous
13 Dutch
2 Italian
2 Jan Baptist Weenix
1 Jan Lievens

Places
13 (others)

PD013612 A horse lying down; with head to right. ...
by Jan Lievens, Anonymous, Dutch, and Rembrandt

PD013924 Study of a pig, facing left. c.1638-1639...
by Dutch and Rembrandt

PD013925 A tethered pig, facing right. c.1638-1639...
by Dutch and Rembrandt

PD013926 A lion drinking from a pail; crouching on...
by Dutch and Rembrandt

Note: the **Creator** facet is populated from FR92i_created_by, which includes not only individual creators but also groups they belong to. In this case "Dutch" is the Nationality of Rembrandt.

This search uses the query defined in the previous section. The search takes significantly longer than the query alone (4.5 seconds) because after obtaining up to 500 objects, it executes several more queries to fetch their display fields, facets, and images. Subsequent restrictions using the facets are much faster (sub-second response).

4 Conclusion

4.1 Summary

We presented performance results for the RS implementation [1] of FR Search as defined in [2,3]. This search works over a significant CH dataset (almost 1B statements), using a complex ontology (CIDOC CRM). Using a semantic repository is appropriate for this dataset because of its complexity, graph-oriented nature, diversity

[11] http://www.youtube.com/watch?v=HCnwgq6ebAs

of relations, and complexity of queries that users are interested in. This is an exciting demonstration of large-scale reasoning with real-world data:

- The well-structured nature of the data allows for expressive reasoning. The inferred knowledge makes good sense when reviewed by domain experts; unlike other combinations of RDF data gathered "from the wild" that often generate strange/ faulty results.
- This is one of the first examples of such expressive reasoning with large datasets. Previous examples work with 5-10M statements, and often use synthetic data
- Reasoning adds real value: it would be very hard to service the same complex queries without inference

4.2 Related Work

The RS repository is one of the largest CH datasets loaded in an RDF repository and provides valuable implementation experience. Some other large CH repositories include:

- The Europeana EDM repository (hosted by Ontotext) is bigger (see sec. 2.9), but is much less structured. Since most objects were converted from ESE, they include mostly literals: no controlled URIs and no links.
- CLAROS[12] (Classical Arts Research Online Services): in 2009 [9] reports 10M triples loaded in a Jena TDB triple-store. This has expanded, implementing offline indexing extensions (MILARQ) for better performance.
- The Poznan Supercomputing and Networking Center (PSNC) has implemented several national aggregations of museum and bibliographic data based on CIDOC CRM, also using OWLIM. [10] reports 600k publications converted to CRM/ FRBRoo as part of the SYNAT project. Krzysztof Sielski reported the numbers in section 2.9 at the CRMEX 2013 workshop, see the PSNC paper in this volume

4.3 Future Work

We would like to re-implement the FRs by using a lot of standard constructions and only a few OWLIM rules:

- Standard RDFS and OWL constructs: rdfs:subPropertyOf, owl:propertyChainAxiom, owl:inverseOf
- Additional properties: ptop:transitiveOver as generalization of owl:TransitiveProperty; conjunctive property definitions that are needed for FRs, as explained in [1] sec. 3.3
- Define the FR networks in RDF data

The benefits of such reimplementation are better flexibility (OWLIM rules are not flexible, see end of sec 2.2) and better portability to other repositories.

[12] http://www.clarosnet.org

We are exploring the opportunity to create a CH Benchmark using the above data and FRs under the auspices of the Linked Data Benchmarking Council (LDBC)[13] project, so that other vendors can implement the same reasoning and compare the performance of their implementations. LDBC seeks to empower users of semantic technologies by establishing significant and objective benchmarks that address real-world data and user needs.

4.4 Acknowledgements

This work is supported by the ResearchSpace project, executed by the British Museum and funded by the Andrew W. Mellon Foundation. We are grateful to Dominic Oldman, ResearchSpace principal investigator, for his excellent continuing collaboration. Dominic created the RS Semantic Search movie cited above.

We thank the anonymous reviewers for their constructive and specific comments.

5 References

1. Vladimir Alexiev: Implementing CIDOC CRM Search Based on Fundamental Relations and OWLIM Rules. Semantic Digital Archives workshop (SDA 2012), part of Theory and Practice of Digital Libraries conference (TPDL 2012). September 2012, Paphos, Cyprus. http://ceur-ws.org/Vol-912
2. Katerina Tzompanaki, Martin Doerr: A New Framework for Querying Semantic Networks. ICS-FORTH Technical Report TR-419, May 2011
3. Katerina Tzompanaki, Martin Doerr: Fundamental Categories and Relationships for intuitive querying CIDOC-CRM based repositories, ICS-FORTH Technical Report TR-429, April 2012, http://www.cidoc-crm.org/docs/TechnicalReport429_April2012.pdf
4. Barry Bishop, Atanas Kiryakov, Damyan Ognyanoff, Ivan Peikov, Zdravko Tashev, Ruslan Velkov, OWLIM: A family of scalable semantic repositories, Semantic Web Journal, Volume 2, Number 1, 2011.
5. Barry Bishop, Spas Bojanov. Implementing OWL 2 RL and OWL 2 QL rule-sets for OWLIM. OWL Experiences and Directions workshop (OWLED 2011), San Francisco, USA, June 5-6, 2011, CEUR-WS.org, ISSN 1613-0073
6. Barry Bishop, Atanas Kiryakov, Damyan Ognyanoff, Ivan Peikov, Zdravko Tashev, Ruslan Velkov. FactForge: A fast track to the web of data. Semantic Web Journal, V.2, N.2, 2011.
7. Barry Bishop, Atanas Kiryakov, Zdravko Tashev, Mariana Damova, Kiril Simov. OWLIM Reasoning over FactForge. Proceedings of OWL Reasoner Evaluation Workshop (ORE'2012), collocated with IJCAR 2012, Manchester, UK
8. SPARQL Property Paths, http://www.w3.org/TR/sparql11-property-paths
9. David Shotton, The Future of the Past: Using CIDOC CRM for CLAROS. Semantic Web and CIDOC CRM Workshop, co-located with ISWC 2009, Washington DC, http://www.semuse.org/index.php?title=Semantic_Web_and_CIDOC_CRM_Workshop
10. Cezary Mazurek, Krzysztof Sielski, Maciej Stroiński, Justyna Walkowska, Marcin Werla, Jan Węglarz. Transforming a Flat Metadata Schema to a Semantic Web Ontology: The Polish Digital Libraries Federation and CIDOC CRM Case Study. Studies in Computational Intelligence Volume 390, 2012, pp 153-177

[13] http://www.ldbc.eu/