HAL
open science

Optimize First, Buy Later: Analyzing Metrics to Ramp-up Very Large Knowledge Bases

Paea Lependu, Natalya F. Noy, Clement Jonquet, Paul R. Alexander, Nigam H. Shah, Mark A. Musen

To cite this version:

Paea Lependu, Natalya F. Noy, Clement Jonquet, Paul R. Alexander, Nigam H. Shah, et al.. Optimize First, Buy Later: Analyzing Metrics to Ramp-up Very Large Knowledge Bases. 9th International Semantic Web Conference (ISWC 2010), Nov 2010, Shangai, China. pp.486-501. hal-00558036

HAL Id: hal-00558036

https://hal.science/hal-00558036v1

Submitted on 20 Jan 2011

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Optimize First, Buy Later: Analyzing Metrics to Ramp-up Very Large Knowledge Bases

Paea LePendu, Natalya F. Noy, Clement Jonquet, Paul R. Alexander, Nigam H. Shah, and Mark A. Musen

Stanford University, Stanford, California USA
{plependu,noy,jonquet,palexander,nigam,musen}@stanford.edu

Abstract. As knowledge bases move into the landscape of larger ontologies and have terabytes of related data, we must work on optimizing the performance of our tools. We are easily tempted to buy bigger machines or to fill rooms with armies of little ones to address the scalability problem. Yet, careful analysis and evaluation of the characteristics of our data—using metrics—often leads to dramatic improvements in performance. Firstly, are current scalable systems scalable enough? We found that for large or deep ontologies (some as large as 500,000 classes) it is hard to say because benchmarks obscure the load-time costs for materialization. Therefore, to expose those costs, we have synthesized a set of more representative ontologies. Secondly, in designing for scalability, how do we manage knowledge over time? By optimizing for data distribution and ontology evolution, we have reduced the population time, including materialization, for the NCBO Resource Index, a knowledge base of 16.4 billion annotations linking 2.4 million terms from 200 ontologies to 3.5 million data elements, from one week to less than one hour for one of the large datasets on the same machine.

1 Introduction

Researchers are using ontologies extensively to annotate their data, to drive decision-support systems, and to perform natural language processing and information extraction. As a result, we have an abundance of information across many domains making their way into knowledge-based systems. For example, annotation databases that link terms from biomedical ontologies to clinical data reach well into the tens of billions of records and help scientists discover new connections among genes and diseases, or drugs, treatments, and patient outcomes [11, 20].

At the same time, ontologies are diverse; they are evolving; and they are getting larger. Many have over 25,000 classes. A few have over 200,000 classes. Some change on a daily basis. As we move into this abundant landscape, we are tempted to meet the computational challenges either by scaling-up and purchasing bigger machines, or by scaling-out and renting armies of little ones from the various compute clouds. Here, we study how a careful analysis and evaluation of the characteristics of our ontologies and data—using metrics—leads to dramatic improvements in performance, without spending on new infrastructure.

We focus on the domain of biomedicine, which has some of the largest, actively used, and actively evolving ontologies today. In our laboratory, as part of the Na-

tional Center for Biomedical Ontology (NCBO), we have developed BioPortal [16]—the largest repository of publicly available biomedical ontologies. It currently contains more than 200 ontologies, which comprises over 2.4 million classes.

BioPortal includes the NCBO Resource Index [19], which is a searchable database of semantic annotations for biomedical resources using all BioPortal ontologies. In this context, a biomedical *resource* is a repository of elements that may contain patient records, gene expression data, scholarly articles, and so on. A *data element* is unstructured text describing elements in the resource. An *annotation*—a central component—links an ontology term to a data element, indicating that the element refers to the term. To generate the Resource Index, we use a concept-recognition tool to find ontology terms and their synonyms in data elements, and to store these associations in the index. The Resource Index currently includes 22 different data resources, comprising over 3.5 million data elements resulting in 16.4 billion annotations stored in a 1.5 terabyte MySQL database. We are ramping-up the system to include nearly 100 different data resources, 50 million data elements, and well over 100 billion annotations.

Many large-scale knowledge bases will pay an amortized penalty up-front by materializing inferences (e.g., forward chaining, materializing views, computing transitive closures) so that queries will run much faster, but at what cost? Knowing these tradeoffs and performance limitations helps us make critical decisions on which systems will work best for our needs, or when—and how—to build something entirely new. As one example of a critical problem we encountered for the Resource Index: what happens when ontology evolution outpaces materialization?

Therefore, we approach the scalability problem for a knowledge base of annotations, like the Resource Index, first by examining existing, scalable systems. Our goal is to incorporate a large variety of ontologies as well as a large amount of data. However, we found that benchmarks fall short in illuminating the fundamental tradeoffs between query-time and load-time costs precisely because they do not account for variability among ontologies. The size and depth of an ontology hierarchy significantly affects the cost-curve for materialization. Furthermore, ontologies are not stagnant. Hence, in building our own tool to handle a variety of ontologies and datasets, we found that optimizing primarily for data distribution and ontology evolution significantly improves the performance of the system.

Besides providing metrics for the most comprehensive set of ontologies and annotations available in biomedicine (Section 3), we offer the following contributions:

- we used clustering algorithms on the size and depth of ontologies to identify some characteristic distributions (4.2).
- we synthesized a set of representative ontologies as a new benchmark and demonstrate that these ontologies illuminate previously opaque materialization costs (4.3).
- we analyzed the distribution of annotations together with ontology evolution metrics to determine partitioning schemes that streamline our workflow (5.2).
- we improved the performance time of the Resource Index by several orders of magnitude by applying our analysis toward optimization strategies (5.3).

2 Related Work

Some of the related areas for this study include work on annotation indexes, benchmarking tools, ontology metrics, and the motivating biomedical uses cases.

Annotation indexes. An annotation assigns a tag to some media as a whole, but it also records the context in which the tag applies. Text annotations differ from multimedia annotations mainly in terms of dimensionality. For text strings, the context can be as simple as a position-offset value. For images or video (such as radiology x-rays, EKG data, CT scans, MRI images, or even YouTube videos), additional dimensions, including temporal offsets are required to localize the annotation context [18]. Large-scale annotation systems often resort to scale-out architectures. Annotation databases and ontology-based indexes often exemplify this: GoPubMed [5] indexes all of the PubMed[1] articles using terms from the Gene Ontology; Sindice [6] is another example of a more general ontology-based index; Yahoo! is adopting the Healthline[2] ontology-based search engine. Some of the underlying technologies include databases such as MySQL, indexing tools like Lucene, and frameworks like Hadoop and Solr.[3]

Benchmarking tools. Researchers use the Lehigh University Benchmark (LUBM) [7] to evaluate the scalability of knowledge bases. LUBM provides developers, engineers and architects with methods for quantifying the load-time and query-time for computationally intensive tasks such as computing massive RDF closures [23, 26]. Some researchers have indicated that the benchmark can be improved by reflecting real-world workloads based on various dimensions such as reasoning complexity [13], data distribution [24], and even ontology variation [21]. However, to our knowledge, still no benchmark takes the diversity of ontologies into account; therefore, materialization costs remain inadequately characterized.

Ontology metrics. Researchers aiming to characterize various dimensions of large numbers of ontologies in RDF(S), OWL or DAML format have performed several surveys of the Semantic Web landscape [4, 14, 21, 25]. Ontologies vary considerably along many dimensions: by size and shape [4, 25], by expressiveness and complexity [13, 25], by feature selection [21, 25], or by instance density [4].

Biomedical use cases. The NCBO annotations have been used to interpret high-throughput biomedical data (e.g., gene expression) using generalized (ontology neutral) enrichment analysis [22] to discover significant gene–disease relationships that are implicitly embedded in scientific literature. We have also used annotations to discover context-specific mappings among biomedical terminologies, which gives insights into the relationship between, for example, the liver (Minimal Anatomical terminology) and transduction (Gene Ontology) within the context of cancer (Human Disease Ontology).

One of the gaps in current research involves using multiple ontologies at once in large-scale knowledge base systems, such as the Resource Index, for scientific analysis. Not only must we keep pace with the growing abundance of biomedical data, we must also account for the number, variety and evolution of ontologies being used in practice.

[1] `http://www.ncbi.nlm.nih.gov/pubmed`
[2] `http://www.healthline.com`
[3] `http://www.apache.org`

3 Data

We have collected metrics for the most comprehensive set of publicly available ontologies and annotations in biomedicine. In this section, we outline what ontologies and annotations we used and how we gathered metrics on them. Our collected data represents a snapshot taken in May, 2010.

Ontologies. The BioPortal repository [16] stores biomedical ontologies developed in various formats—OWL, OBO, RRF, Lexgrid-XML and Protégé Frames—and ranging in subject matter from representation of anatomy and phenotype to diseases. Researchers in biomedicine actively contribute their ontologies to BioPortal. Users can submit new versions of their ontologies; visualize, browse and search them; make comments on and get notifications about ontology changes; or create mappings between terms from different ontologies.

BioPortal makes all data accessible programmatically via RESTful Web services. Almost all of the BioPortal ontologies, including earlier versions, can be downloaded.[4] We used these services to collect the ontologies used in our study. Of the 200 ontologies, we incorporated 145. We skipped the remainder due to limitations on what was available for parsing at the time.

Annotations. The Resource Index workflow, as illustrated by Figure 1, is composed of two main steps: First, *direct annotations* are created from the text metadata of a resource element using an off-the-shelf concept recognition tool, which in our case is MGREP [3]. Second, we use subclass relations to traverse ontology hierarchies to create new, *expanded annotations*.[5]

The ontology terms play a vital role because of the subsumption hierarchy. Users who search for a general term like "cancer" will find results for documents that have been annotated with, say, "melanocytic neoplasm" because it is defined as a kind of cancer in the NCI Thesaurus, one of the ontologies in BioPortal.

The Resource Index currently has 22 resources indexed. We use the following sample of 4 resources in this study because they are representative in terms of size, type of content, frequency of updates, and quantity of data per element:

Biositemaps (BSM) represent a mechanism that researchers in biomedicine use to publish and retrieve metadata about biomedical resources (1.5K elements).

ArrayExpress (AE) is a public repository of microarray data and gene-indexed expression profiles from a curated subset of experiments (10K elements).

ClinicalTrials.gov (CT) provides regularly updated information about federally and privately supported clinical trials (89K elements).

GRANTS combines three different funding databases: Research Crossroads, CRISP, and the Explorer of the NIH Reporter (1,400K elements).

[4] `http://www.bioontology.org/wiki/index.php/BioPortal_REST_services#Download_an_ontology_file`

[5] This workflow is also available as a web service called the NCBO Annotator [10], which provides researchers with an easy mechanism to employ ontology-based annotation using BioPortal ontologies in their respective pipelines.

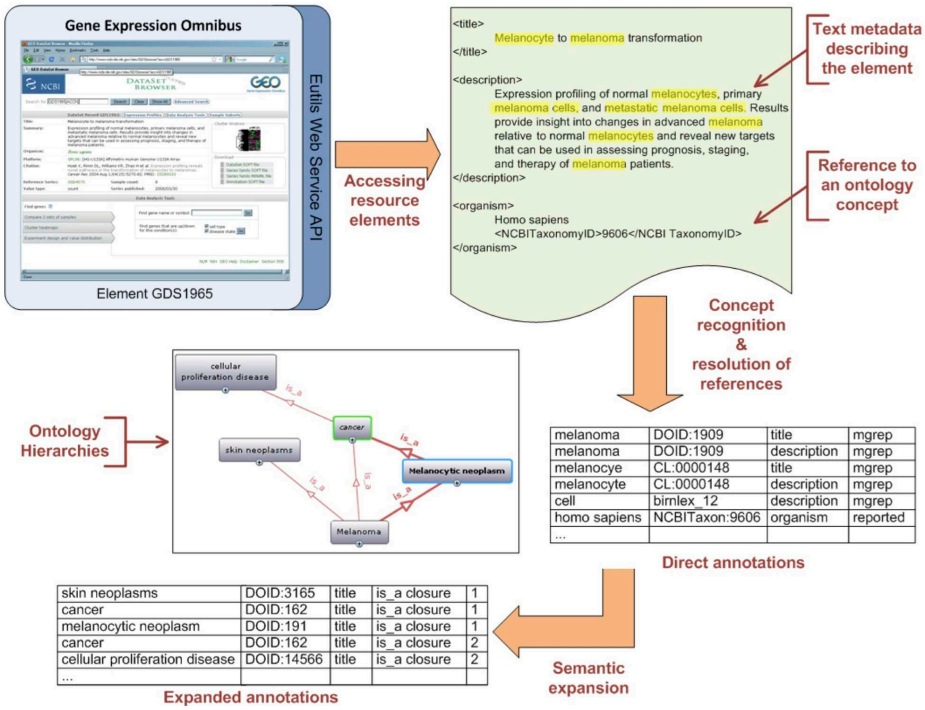

Fig. 1. Resource Index workflow: Using the Gene Expression Omnibus (GEO) data resource, we illustrate how direct annotations are associated with a data element, then we expand those annotations to account for the ontology hierarchy.

Metrics. BioPortal maintains a limited set of ontology metrics, such as the number of classes or siblings (Table 1), which we gathered using REST services. For every ontology in either OWL or OBO format, we also used the ontology download service and used the OWL API Metrics tool [8] to complement those statistics. We followed imports during all calculations and consider only the asserted hierarchy.

For annotation metrics, we directly download the statistical data on the number and kinds of annotations per resource and per ontology kept by the Resource Index. Table 1 lists the specific metrics we used in this study. All the metrics are available online.[6]

[6] `http://www.bioontology.org/wiki/index.php/Metrics_Study`

Table 1. Ontology and annotation metrics that we used.

Ontology Metrics	Annotation Metrics
Number of classes	Number of data elements
Number of versions	Number of direct annotations
Maximum depth of the class hierarchy	Number of expanded annotations
Maximum and average number of siblings	
Average number of adds, deletes and changes per version	

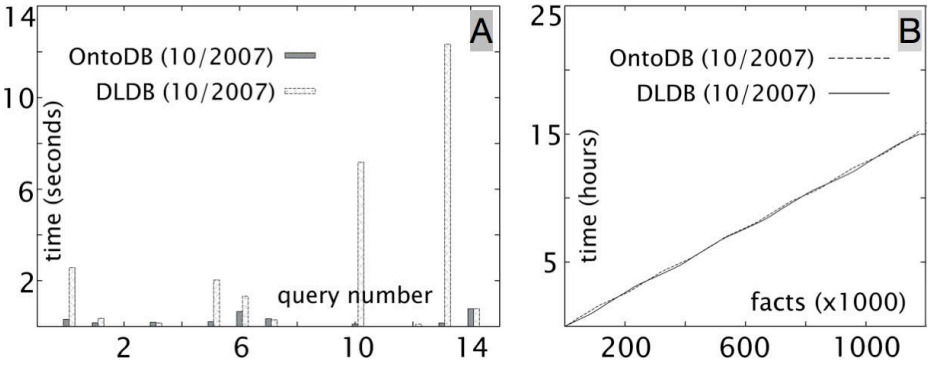

Fig. 2. Materialized versus non-materialized KBs: OntoDB outperforms DLDB on LUBM query-time performance (A). Yet, OntoDB *appears* to add no additional cost for load-time performance (B). (Note: we reported these figures previously [12].)

4 Are existing systems scalable enough?

The set of ontologies in BioPortal, which we use to generate the Resource Index, is extremely diverse, both in terms of size and depth of the class hierarchy. In order to analyze and improve the query performance of the Resource Index, we must first analyze the effects of the ontology characteristics on both load-time and query-time performance. We started by using popular ontology benchmarks to compare the performance of different approaches. However, as we show in this section, these benchmarks do not account for the diversity in size and depth of ontologies. We then discuss the complementary set of benchmarks that we synthesized based on the ontologies in our repository.

4.1 A Tale of Two KBs: Is Materialization Really Free?

Because materializing inferences is a large part of high-performance KBs, we would expect the obvious tradeoff: systems that perform materialization should obtain faster query time at the cost of slower load time. However, upon comparing two dichotomous KB systems, we were baffled to find that materialization seems to cost nothing at all.

DLDB [17] is a knowledge base system developed at Lehigh University that uses database views to assist with query answering on large sets of data. OntoDB [12] was developed at the University of Oregon using other database features. These two systems are very similar: they take an ontology and a set of instances as input; they create and load a relational database schema based on predicates from the ontology; they store the instances in database tables; and they use intrinsic database features to maintain the knowledge model and answer queries. OntoDB differs from DLDB by materializing inferences at load time—using triggers—rather than by unfolding views at query time.

Comparing OntoDB with DLDB using LUBM, we clearly see the expected gains in query-response time, as illustrated by Figure 2–A: materializing the inferences yields marked gains because of the pre-computation. However, suspiciously, the gains come at no apparent cost as shown in Figure 2–B: the slope of the overlapping lines indicates the

Table 2. EM clustering: Cluster 3 characterizes 38 percent of ontologies.

Cluster	Num. Classes	Max. Depth	Max. Siblings	Avg. Siblings
0 (8%)	19628 (+/- 29189)	13.7 (+/- 5.6)	249.9 (+/- 118.6)	39.5 (+/- 13.4)
1 (26%)	1264 (+/- 791)	10.7 (+/- 4.7)	68.8 (+/- 35.5)	5.1 (+/- 6.0)
2 (3%)	13338 (+/- 10483)	37.0 (+/- 5.3)	2252.9 (+/- 2335.9)	10.4 (+/- 10.0)
3 (38%)	**181 (+/- 170)**	**7.5 (+/- 3.4)**	**13.6 (+/- 8.4)**	**3.4 (+/- 2.6)**
4 (21%)	34401 (+/- 66037)	18.8 (+/- 11.0)	385.1 (+/- 380.9)	1.1 (+/- 0.5)
5 (2%)	344095 (+/- 158541)	29.7 (+/- 10.3)	9939 (+/- 100.4)	1 (+/- 18.0)
6 (3%)	45303 (+/- 20784)	21.2 (+/- 15.1)	2226.3 (+/- 790.2)	76.8 (+/- 49.8)

Table 3. K-means clustering: The characteristics of small, medium and large ontologies.

Cluster	Num. Classes	Max. Depth	Max. Siblings	Avg. Siblings
Small (80%)	4925	9.6	110.6	3.8
Medium (8%)	26062	14.7	654.7	55.8
Large (12%)	96502	33.2	2571.4	8.6

same *constant cost* per assertion for both systems. Would larger or deeper ontologies demonstrate the expected load-time cost for materialization?

4.2 Is the LUBM Ontology Too Small and Shallow?

To determine whether larger and deeper ontologies would expose the expected load-time costs, we need ontologies that vary in size and depth. Motivated by results that are practical and relevant to biomedicine, we analyzed the 145 ontologies from BioPortal by running clustering algorithms on the ontology metrics that we gathered for them (Section 3, Table 1). We specifically considered the number of classes, maximum depth and maximum and average number of siblings. We used two clustering algorithms: the Expectation-Maximization clustering algorithm (EM) with an unspecified number of clusters, which uses an iterative mechanism to find an optimal clustering distribution (Table 2); and the simple K-means clustering algorithm with 3 clusters, identifying small, medium, and large ontologies (Table 3).

We found that the most representative ontology has 181 classes and depth 8 (Table 2). In addition, the three clusters with the largest number of ontologies (3, 1, 4) cover 85% of the ontologies. The characteristics in the K-means results are highly skewed, with 80% of ontologies falling into the "small" category. The previous EM results (cluster 3, 38%) suggest that there are a significant number of clustered ontologies having smaller characteristics—so, we might consider introducing a smaller division than K-means suggests. Furthermore, the medium and large categories can be collapsed into cluster 4.

Reasoning in this way, we combined the results of EM clusters 3, 1 and 4 with the K-means clusters to extrapolate a division between small, medium and large ontologies: 100, 2,000 and 25,000 classes. Similarly, we defined the range of shallow, mid-range

Table 4. Ontologies: nine synthetic ontologies representing biomedicine.

	Ontology Parameters		Schema Load Time		Instance Load Time	
	Size	**Depth**		**Mean**		**Mean**
Small	78	5	6.95 ms	7.57 ms	3.65 ms	4.78 ms
	81	10	7.12 ms		4.99 ms	
	72	20	8.64 ms		5.69 ms	
Medium	1623	5	8.21 ms	8.61 ms	6.51 ms	10.55 ms
	1555	10	8.93 ms		9.45 ms	
	1827	20	8.68 ms		15.69 ms	
Large	19992	5	9.57 ms	9.82 ms	11.84 ms	24.93 ms
	22588	10	9.59 ms		22.80 ms	
	19578	20	10.28 ms		40.14 ms	

(mid) and deep ontologies as having depths of 5, 10, and 20. We use these results to generate a set of ontologies that account for the variety of depths and sizes.

4.3 Accounting for Size and Depth: New Benchmark Ontologies

The clustering results give us the characteristics of nine possible ontologies varying along the size and depth dimensions: small–medium–large and shallow–mid–deep. Our goal is to determine whether this benchmark produces enough variability to compare load times for use cases that rely on different ontologies.

We developed OntoGenerator, a tool that generates a synthetic ontology given the following parameters: a seed, the maximum number of classes, maximum number of siblings (i.e., span), density, and number of individuals. The density parameter introduces a degree of randomization in the fullness of the tree structure. It denotes the probability that the maximum number of siblings or the maximum depth will be reached along any path to a leaf node. We use the seed value to prime the randomizing function which allows us to reproduce the same ontology given the same parameters, or, conversely, to construct a new ontology (of the same kind) by using a different seed. Finally, the tool creates the given number of individuals as instances of randomly chosen classes in the ontology, distributing data uniformly at various depths in the hierarchy.

We used OntoGenerator to synthesize the nine different ontologies whose specific size and depth are outlined in Table 4. We also had OntoGenerator create one million individuals for each of the ontologies. The data is available online.[7]

4.4 Conclusion: Materialization Costs Depend on the Size and Depth

Our hypothesis that materialization costs will be exposed for larger and deeper ontologies was confirmed: the results demonstrate that cost is *not constant* per assertion, but it depends on ontology size and depth. We obtained the load-time results in Table 4 by populating OntoDB (the KB that explicitly materializes inferences) with the nine ontologies we generated, i.e., we loaded the ontology and the data. We measured load time in two phases: the time to transform the ontology into a schema and load it into the database (averaged per class), and the average time for loading a single instance assertion (taken 1,000 at a time).

[7] `http://www.bioontology.org/wiki/index.php/Metrics_Study`

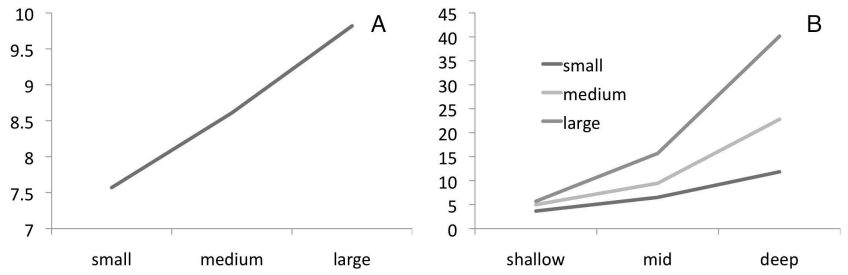

Fig. 3. New benchmark performance: For OntoDB, the average time in milliseconds to load a single assertion for small, medium, and large ontologies increases with larger-sized ontologies (A). Load time also increases with depth (B).

The positive slope of the lines displayed in Figure 3 show a clear cost dependency. Interestingly, size and depth have both super-linear and interactive effects on the cost. The crooked slope of each line indicates that size (and depth) independently yields a super-linear effect on load time. Furthermore, size and depth have a cumulative effect: the larger the ontology, the larger the role of depth (note increasing crookedness).

In conclusion, we should include ontologies of varying size and depth in KB benchmarking suites. We have proposed a set of ontologies that can be used to enhance the LUBM for characterizing materialization costs. This study clearly points out that, depending on their ontologies, system designers should worry about materialization.

5 Managing Large-scale Annotation Databases

Based on the results above (cf. Table 4), neither OntoDB nor DLDB could handle data on the scale of the Resource Index: they would take several hours just to create the schema (let alone process any data!) for the NCI Thesaurus, which has 74,646 classes. Furthermore, the Resource Index uses not just one ontology, but over 200 of them (over 2.4 million classes) for annotation purposes. Finally, not only must we abate the costs of materializing inferences for large-scale KBs, but we must also consider how to manage that knowledge over time for various, evolving ontologies.

Whereas ontology size and depth affect materialization costs in the stagnant scenarios described above, we demonstrate below that data distribution and ontology evolution significantly also affect how we manage a very large KB over time. By optimizing for these metrics, we have streamlined the Resource Index population workflow by several orders of magnitude: from taking over one week for loading to less than one hour for one of our larger datasets on the same machine.

5.1 The NCBO Resource Index

Annotation databases such as the NCBO Resource Index take the structure of an ontology into account to provide enhanced search and retrieval functionality for documents. As mentioned previously, a direct annotation “tags” a data element with a class from an ontology. If a document is annotated with a class from an ontology, then we infer that it is also annotated by the superclasses of that class (Figure 1).

Table 5. Annotations: A sample of four biomedical resources from the Resource Index shows the number of elements, the number of direct annotations, and the number of expanded annotations.

Resource	No. Elements	No. Direct Annotations	No. Expanded Annotations	Avg. I-Density
BSM	1.5 K	0.5 M	3 M	1.9
AE	11 K	13 M	115 M	68.6
CT	89 K	181 M	1,300 M	434.4
GRANTS	1,400 K	1,900 M	13,800 M	7621.6

In terms of load time versus query time, the tradeoff has to do with materializing those superclass annotations. We can perform the inference at query time by unwinding the hierarchical structure and issuing a union of sub-queries, one for each subclass (recursively), to retrieve annotations. However, unfolding queries is probably not a viable option because users expect split-second response times and the number of subclasses for a given class can reach into the thousands. The query-time results in Figure 2 testify to the slowness of query unfolding, even for a small ontology. But, as our benchmarking study further illustrates (Figure 3), we can expect that for very large and deep ontologies (e.g., the NCBI Organismal Classification Ontology, which has 513,248 classes) a user could potentially wait for several minutes to get answers on a very simple query as it unfolds into tens of thousands of sub-queries.

The other option is to materialize the expanded annotations by forward propagating (i.e., copying) them up the class hierarchy when they are created. As a result, no unfolding occurs during query time—we can directly look up the answers for each class quickly by using an index. However, materializing inferred annotations results in very large expanded annotation databases (Table 5). In general, the ratio of annotations to elements is 1000:1 and the ratio of expanded to direct annotations is 8:1.

Prior to optimization, the Resource Index population workflow took approximately one week to generate and materialize annotations for the CT resource (the largest processable resource), which has 89,000 data elements. We extrapolated that it would take several months to process a repository of a million documents (e.g., the GRANTS resource), which puts repositories serving nearly 20 million documents (e.g., PubMed) completely out of reach. Therefore, our goal is to reduce the load time by optimizing the workflow so that it will scale to handle these large data resources—and to keep up with them as each one grows.

5.2 Optimizing for Data Distribution and Ontology Evolution

Since our goal is to manage very large sets of annotations and expanded annotations, one obvious choice is to partition the data in a way that supports a streamlined and efficient execution. Ideally, a good partitioning mechanism will also distribute easily over multiple nodes (if necessary). In order to determine how to partition the data, we look at metrics on data distribution and ontology evolution:

Data Distribution. Instance density tells us how the data is distributed. If we treat annotations as a kind of data instance, the instance density (I-density) of an ontology

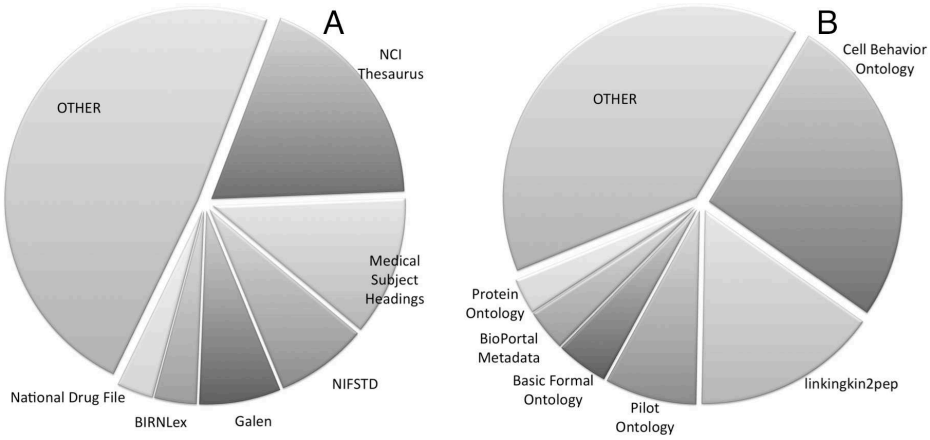

Fig. 4. Annotations per ontology: The number of annotations per ontology obviously preferences some of the larger ontologies like NCIT, MeSH and Galen (A). However, I-density (B) shows that small but generic ontologies have higher percentages of their terms used for annotation.

measures the number of instances for each class in an ontology [4]. As Table 5 shows, larger resources have larger I-densities because there is simply more data to be distributed. This indicates that resource size is an important consideration. By the same token, larger ontologies entail more annotations (Figure 4–A) because more terms are available for annotation matches, so ontology size is another factor. Interestingly, accounting for ontology size, the actual I-density of a resource per ontology preferences very small but very general ontologies such as the Cell Behavior Ontology (6 terms), Basic Formal Ontology (an upper ontology), and BioPortal Metadata Ontology (Figure 4–B) because there is a higher chance that a high percentage of terms are used in those ontologies.

Ontology Evolution. Ontologies evolve by growing in size and changing in structure. We used the structural differences (diffs) that BioPortal provides for 15 of its ontologies to understand their evolution. The diffs record the additions, deletions and changes to classes in the ontologies for consecutive versions. Figure 5 shows that users modify ontologies (change or delete terms) more often than they add new terms. This implies that materialized inferences must be updated regularly as well. Moreover, the number of versions per ontology in BioPortal indicates that the frequency of updates depends on the ontology in question: new versions of ontologies accumulate frequently for a few but rarely for most, following a power law distribution (Figure 6).

The metrics on data distribution and ontology evolution help to decide how to partition the database so that we can streamline the workflow. Our goal is not only to materialize the expanded annotations efficiently, but also to manage them as the ontologies evolve. In developing an optimization strategy, we need to determine which partitioning criteria are most important, in what order.

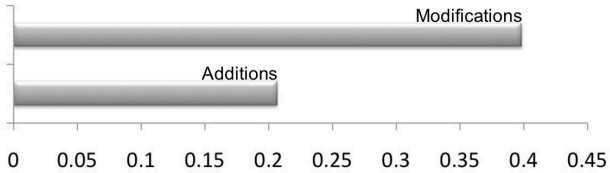

Fig. 5. Version differences: Based on a sample of 15 ontologies, normalized by ontology-size, the percentage of modifications (updates and deletes) are double the percentage of additions.

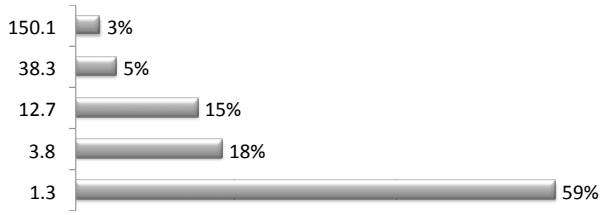

Fig. 6. Ontology versions: Ontologies loaded in BioPortal evolve according to a power law distribution: most ontologies (59%) are rarely modified (1.3 versions on average) but some (3%) are revised weekly (150 versions). The Gene Ontology changes the most (daily, 212 versions).

Partitioning the Database. First of all, we can see from Table 5 and Figure 4 that data is naturally distributed along resource size and ontology size by examining instance density metrics—therefore, those attributes make good partitioning candidates. We confirmed this by using information theory on the annotation metrics to determine feature selection: we ran a basic information gain algorithm (ID3). However, we still need more information to decide which should be the primary partitioning criteria: resource size or ontology size.

Hence, we looked at ontology evolution metrics next. Evolution influences how we keep the database up-to-date. Whereas data is always added, metrics confirm that ontologies are mostly (but infrequently) updated. From these evolution metrics, we concluded that resources should be the primary partitioning criterion because data additions will far outweigh ontology updates in importance.

Therefore, we first partition the database by resource, which means that each set of annotations for a resource will be kept in its own database file. Next, we partition again by ontology. Partitioning first by resource allows us to process data elements in a pipelined fashion: it enables adding new data elements over time extremely quickly by merely appending to the end of the file (minimizing disk seeks). Furthermore, adding a new resource is as simple as adding a new partition: it has no detrimental effect on existing annotations. Finally, as we show in the next section, this partitioning helps with expanding annotations very rapidly.

Sub-partitioning by ontology allows us to drop annotations efficiently for only the ontology that has changed: it has no effect on annotations associated with other ontologies. Moreover, since sub-partitioning subdivides the files into smaller chunks, we also gain speed in reading, indexing and updating annotations by ontology.

Table 6. Optimized workflow: We achieve highly scalable performance using partitioning and merge-join optimization based on density and evolution metrics.

Resource	No. Elements	Old Population Time	Optimized Population Time
BSM	1.5 K	<< 1 day	0.4 min
AE	11 K	~ 1 day	3 min
CT	89 K	~ 7 days	49 min
GRANTS	1,400 K	N/A	492 min

Speeding-up Expanded Annotations. Expanding annotations along the subsumption hierarchy is—by far—the most computationally intensive, storage expensive and time consuming phase of the population workflow. Partitioning by resource makes it possible to compute the expansions extremely efficiently. Again, the goal is to take a direct annotation using a class and to use the ontology hierarchy to materialize (i.e., expand) the annotations to superclasses of that class (cf. Figure 1). There are 2.4 million classes in BioPortal ontologies, which, if we compute the transitive closure on subclass assertions, result in 20.4 million subclass relationships to consider during expansion. However, the real challenge comes when we have to cross-reference these 20.4M subclass relationships against 2 billion direct-annotation records (e.g., the GRANTS data set, Table 5).

Stored in a database, we essentially have two tables to join: [term, superclass] and [term, data-element]. The former contains the subclass relationships and the latter contains the direct annotations. The efficient way to compute this join—called a merge-join—is to sort both tables first by the join condition (term) and then scan them both sequentially, outputting the join result in a linear fashion in the size of each table. However, sorting 2 billion records on disk is unrealistic: it would take days if not weeks; moreover, data is constantly being added, so maintaining sorted order would slow down inserts. Joining them using just indexes causes thrashing and unnecessary seeking and re-reading of records (billions of times over!). The database is unable to automatically optimize the join because it lacks the appropriate metrics.

The solution relies on two important facts: (1) the data grows rapidly but is partitioned by resource, and (2) the ontology subclass relations will not grow quickly beyond the 20.4 million (or, say, a 100 million) mark any time soon. Therefore, we stream annotations by resource into the [term, data-element] table as rapidly as the disk can write them (which is many thousands of records a second). We also force the [term, superclass] table into main memory using the MySQL MEMORY engine. (Forcing the annotations into main memory is not an option because they are simply too large.) By having the entire hierarchy in main memory, we can achieve the optimal performance of the simple merge-join query without sorting any records. The result of applying these key decisions yields remarkable performance gains.

5.3 Conclusion: Memory-based Merge-Join with Partitioning Performs Well

Table 6 shows that we have achieved extraordinary performance gains—on a single machine. In the worst case, we could repopulate the entire database (all resources and all ontologies) in less than a day. Extrapolating these results, we anticipate that indexing PubMed (20M articles) with all BioPortal ontologies will take only 5 days, as opposed to nearly a year based on previous estimates.

In conclusion, by analyzing how data is distributed, how it grows, and how it evolves, we can determine how to partition the data, how to streamline population speed, what to keep in main memory, and how to update data without slowing down population.

6 Discussion

We would like to see previous results on large-scale knowledge base systems [17, 23, 26] re-evaluated using the nine new benchmark ontologies we created. For example, we believe that anomalies described when evaluating the BibTeX benchmark [24]—a phenomenon similar to that in our "tale of two KBs"—would be explained by using a variety of ontologies, rather than merely a variety of data. Furthermore, we believe that differences in materialization heuristics will be easier to comprehend for systems that compute massive RDF closures [23, 26].

We would like to see metrics used more often for query optimization on Semantic Web queries [1]. As our study illustrates, database optimization techniques should be reused. Just as query optimizers use catalogs, which are metrics kept on the data distribution in each database table, so can metrics improve knowledge bases. For example, we can use ontology metrics for improving query federation over SPARQL endpoints: we can easily optimize a query intersecting two ontologies by taking the smaller one first. The same goes for annotations. What makes optimization in this area different from database optimization, is that ontologies are just like data—they are not treated as higher class citizens the way schemas are treated in databases.

As future work, we plan to develop better metrics on resource evolution: how frequently and how drastically do resources change. Likewise, we are working on computing the differences between all versions of all ontologies. What will be most interesting is to analyze those differences to come up with a viable change-propagation model that could increase the performance of our systems even further. Considering that only fractions of an ontology change during a revision, this approach should save a considerable amount time updating annotations.

Our future research directions include using metrics for SPARQL query optimization. Also, we can analyze differences between ontology revisions to determine an incremental, change-propagation model for updating materialized inferences as ontologies evolve. Finally, we will consider providing annotations as Linked Open Data in a scalable way.

7 Conclusion

We collected metrics for the most comprehensive ontologies available in biomedicine. By analyzing those ontology metrics, we have created a set of representative ontologies that improve upon existing benchmarks. These new ontologies help to illuminate previously opaque load time costs. By using these new evaluation tools, researchers can improve upon KB systems that use materialization strategies.

We followed up on our evaluation by studying additional metrics on instance density and ontology evolution to reduce load time for the NCBO Resource Index as much as possible. As a result of our analyses, we improved performance time by several orders

of magnitude, without investing in new infrastructure. In the worst case, the current database of 16.4 billion annotations can be re-materialized overnight. More importantly, now we can annotate resources that were previously impossible, such as PubMed.

Acknowledgments. We thank Matthew Horridge for assisting with the OWL API Metrics tool. Benchmarking work was performed at the University of Oregon under the direction of Professor Dejing Dou, supported in part by grant R01 EB007684 from the National Institutes of Health. This work was also supported largely in part by the National Center for Biomedical Ontology, under roadmap-initiative grant U54 HG004028 from the National Institutes of Health.

References

1. A. Bernstein, C. Kiefer, and M. Stocker. OptARQ: A SPARQL Optimization Approach based on Triple Pattern Selectivity Estimation. Technical report, University of Zurich, 2007.
2. O. Bodenreider, B. Smith, A. Kumar, and A. Burgun. Investigating Subsumption in SNOMED CT: An Exploration into Large Description Logic-Based Biomedical Ontologies. *Artificial Intelligence in Medicine*, 39(3):183–195, 2007.
3. M. Dai, N. Shah, and W. Xuan. An Efficient Solution for Mapping Free Text to Ontology Terms. In *AMIA Summit on Translational Bioinformatics*, 2008.
4. M. d'Aquin, C. Baldassarre, L. Gridinoc, S. Angeletou, M. Sabou, and E. Motta. Characterizing Knowledge on the Semantic Web with Watson. In *Evaluation of Ontology-based Tools Wkshp (ISWC)*, pages 1–10, 2007.
5. A. Doms and M. Schroeder. GoPubMed: exploring PubMed with the Gene Ontology. *Nucleic Acids Research*, 33(Web Server issue):783–786, July 2005.
6. G.Tummarello, E. Oren, and R. Delbru. Sindice.com: Weaving the Open Linked Data. In *Int'l Sem Web Conf (ISWC)*, pages 547–560, 2007.
7. Y. Guo, Z. Pan, and J. Heflin. LUBM: A Benchmark for OWL Knowledge Base Systems. *Journal of Web Semantics*, 3(2):158–182, 2005.
8. M. Horridge and S. Bechhofer. The OWL API: A Java API for Working with OWL 2 Ontologies. In *OWL Experiences and Directions Wkshp (OWLED)*, 2009.
9. C. Jonquet, M. Musen, and N. Shah. Building a Biomedical Ontology Recommender Web Service. *Biomedical Semantics*, 1(S1), 2010.
10. C. Jonquet, N. Shah, C. Youn, C. Callendar, M. Storey, and M. Musen. NCBO Annotator: Semantic Annotation of Biomedical Data. In *Int'l Sem Web Conf (ISWC)*, 2009.
11. M. Krallinger, F. Leitner, and A. Valencia. Analysis of biological processes and diseases using text mining approaches. In *Methods in Molecular Biology*, pages 341–382, 2010.
12. P. LePendu and D. Dou. Using Ontology Databases for Scalable Query Answering, Inconsistency Detection, and Data Integration. *Intelligent Info Sys (JIIS)*, 2010.
13. L. Ma, Y. Yang, Z. Qiu, G. Xie, Y. Pan, and S. Liu. Towards a Complete OWL Ontology Benchmark. In *European Sem Web Conf(ESWC)*, pages 125–139, 2006.
14. A. Magkanaraki, A. Sofia, V. Christophides, and D. Plexousakis. Benchmarking RDF Schemas for the Semantic Web. In *Int'l Sem Web Conf (ISWC)*, pages 132–146, 2002.
15. K. Marwah, D. Katzin, and G. Alterovitz. Context-Specific Ontology Integration. In *(under review) Pacific Symp on Biocomputing (PSB)*, 2011.
16. N. Noy, N. Shah, P. Whetzel, B. Dai, M. Dorf, N. Griffith, C. Jonquet, D. Rubin, M. A. Storey, C. Chute, and M. Musen. BioPortal: Ontologies and Integrated Data Resources at the Click of a Mouse. *Nucleic Acids Research*, 1(37), 2009.
17. Z. Pan and J. Heflin. DLDB: Extending Relational Databases to Support Semantic Web Queries. In *Practical and Scalable Sem. Web Systems Wkshp (ISWC)*, pages 109–113, 2003.

18. D. Rubin, K. Supekar, P. Mongkolwat, V. Kleper, and D. Channin. Annotation and Image Markup: Accessing and Interoperating with the Semantic Content in Medical Imaging. *IEEE Intelligent Systems*, 24(1):57–65, 2009.
19. N. Shah, C. Jonquet, A. Chiang, A. Butte, R. Chen, and M. Musen. Ontology-driven Indexing of Public Datasets for Translational Bioinformatics. *BMC Bioinformatics*, 10, 2009.
20. P. Stenson, E. Ball, K. Howells, A. Phillips, M. Mort, and D. Cooper. The Human Gene Mutation Database: providing a comprehensive central mutation database for molecular diagnostics and personalized genomics. *Hum Genomics*, 4(2):69–72, 2009.
21. C. Tempich and R. Volz. Towards a benchmark for Semantic Web reasoners - an analysis of the DAML ontology library. In *Evaluation of Ontology-based Tools Wkshp (ISWC)*, 2003.
22. R. Tirrell, U. Evani, A. Berman, S. Mooney, M. Musen, and N. Shah. An Ontology-Neutral Framework for Enrichment Analysis. In *AMIA Annual Symposium*, 2010.
23. J. Urbani, S. Kotoulas, E. Oren, and F. Harmelen. Scalable Distributed Reasoning using MapReduce. In *Int'l Sem Web Conf (ISWC)*, pages 634–649, 2009.
24. S. Wang, Y. Guo, A. Qasem, and J. Heflin. Rapid Benchmarking for Semantic Web Knowledge Base Systems. In *Int'l Sem Web Conf (ISWC)*, pages 758–772, 2005.
25. T. Wang, B. Parsia, and J. Hendler. A Survey of the Web Ontology Landscape. In *Int'l Sem Web Conf (ISWC)*, pages 682–694, 2009.
26. J. Weaver and J. Hendler. Parallel Materialization of the Finite RDFS Closure for Hundreds of Millions of Triples. In *Int'l Sem Web Conf (ISWC)*, pages 682–697, 2009.