Technical Report

TR-2007-014

NNexus: An automatic linker for collaborative web-based corpus

by

J. Gardner, A. Krowne, L. Xiong

Mathematics and Computer Science

EMORY UNIVERSITY

NNexus: An Automatic Linker for Collaborative Web-Based Corpora

James Gardner, Aaron Krowne, and Li Xiong

***Abstract*—In this paper, we introduce NNexus, a generalization of the automatic linking engine of Noosphere (at PlanetMath.org) and the first system that automates the process of linking disparate "encyclopedia" entries into a fully-connected conceptual network. NNexus facilitates the extension of this functionality to multiple knowledge bases of this sort (such as Wikipedia and MathWorld) and to web-based information environments in general. We discuss the challenges of the problem space of linking in collaborative corpora, the approaches taken by NNexus, some aspects of evaluation, and ongoing and future directions of research.**

***Index Terms*—E-Learning, Automatic Linking, Wiki, Semantic Web**

I. Introduction

Collaborative online encyclopedias or knowledge bases such as Wikipedia[1] and PlanetMath[2] are becoming increasingly popular because of their open access, comprehensive and interlinked content, rapid and continual updates, and community interactivity.

To understand a particular concept in these knowledge bases, a reader needs to learn about related and underlying concepts. Thus, it is critical that users of any online reference are able to easily "jump" to requisite concepts in the network in order to fully understand the current one. For full comprehension, these jumps should extend all the way "down" to the concepts that are evident to the reader's intuition.

To help users learn more quickly it is now generally accepted that knowledge bases should leverage each others content (or metadata) to increase the scope of the available learning materials. this is the reason for the development of Semantic Web standards such as OWL. NNexus utilizes OWL and a variety of novel computational and data management techniques to link between related concepts in near-real time, enabling users to learn from this dynamic content without having to wait for administrators and authors to make manual updates.

A. Existing Solutions

Most current online encyclopedias (including Wikipedia) require the author(s) or other contributors to an article to explicitly create links to other articles in order to build this semantic network. The perspective taken in our work is that this task is an unnecessary burden on contributors, since the knowledge management environment should "know" which concepts are present and how they should be cited. By contrast, authors will usually not be aware of all concepts which are already present within the system—especially for large or distributed corpora.

A more challenging problem with the manual linking strategy is that a growing, dynamic corpus will generally necessitate links from old entries to new entries as the collection becomes more complete. To attend to this reality would require continuous re-inspection of the entire corpus by writers or other maintainers, which is an $\mathcal{O}(n^2)$-scale problem (where the corpus contains n entries). To keep an evolving corpus fully-linked, it would be necessary for maintainers to search it upon each update (or at least periodically) to determine if the links in the constituent articles should be updated. When generalizing to inter-linkage across separate corpora, the task would potentially be even more laborious, as authors would have to search across multiple web sites to determine what new terms are available for linking into their entries.

The popularity of these encyclopedic knowledge bases has also brought about a situation where the availability of high-quality, canonical definitions and declarations of educationally useful concepts have outpaced their usage (or *invocation*) in other educational information resources on the web. Instead, the user must execute a new search (either online or offline) to look up an unknown term when it is encountered, if it is not linked to a definition. For example, blogs, research repositories, and digital libraries quite often do not link to definitions of the concepts contained in their texts and metadata, even when such definitions are available. This is generally not done because of the lack of appropriate software infrastructure and the extra work creating manual links entails. When such linking is actually done, it tends to be incomplete and is quite laborious.

B. Automatic Invocation Linking

To build this semantic network with minimal manual effort, we advocate *automatic invocation linking* between entries in online corpora [7]. For our purpose, a *collaborative online encyclopedia* is a kind of knowledge base containing "encyclopedic" (standardized) knowledge contributed by a large number of participants (typically but not necessarily in a volunteer capacity). Any article submitted by a user in such a collaborative corpus is an *entry*. We say *invocation* referring to a specific kind of semantic link: that of *concept invocation*.

J. Gardner and L. Xiong are with the Department of Mathematics and Computer Science, Emory University, Atlanta, GA email: jgardn3@emory.edu and lxiong@mathcs.emory.edu

A. Krowne is with Woodruff Library, Emory University, Atlanta, GA and PlanetMath.org email: akrowne@emory.edu

[1]http://www.wikipedia.org

[2]http://www.planetmath.org

ObjectId	Defines	MSC
1	triangle, right triangle, ...	51-00
2	planar, planar graph, ...	05C10
3	connected, ...	05C40
4	geometry, Euclidean geometry, ...	01A16
5	graph, graph theory, edge, ...	05C99
6	graph, function graph	03E20

A planar graph is a graph which can be drawn on a plane (a flat 2-d surface) or on a sphere, with no edges crossing. When drawn on a sphere, the edges divide its area in a number of regions called faces (or "countries", in the context of map coloring). Even if ... The terms underlined indicate terms that need to be linked based on the meta-data in the table.

Fig. 1. Example Document Corpora with Meta-data and Example Entry

Any statement in a language is composed of concepts represented by tuples of words. Such a statement invokes these concepts, as evidenced by the inclusion of word tuples that correspond to common labels for the concepts. We call these *concept labels*. A *link* is a hyperlink from one entry to another.

The following shows a list of entries (objects) in our corpus and an example of entry 1[3]) with links to concepts that are defined in the same corpus. We will use the example to explain the concepts discussed in this paper.

The optimal end product of an automatic invocation linking system should be a fully-connected network of articles that will enable readers to navigate and learn from the corpus almost as naturally as if was interlinked by painstaking manual effort. Without understanding the invoked concepts in a statement, the reader cannot attain a complete understanding of the statement, and by extension the entry it appears in. This is why node interlinkage is so important in hypertexts being used as knowledge bases, and why we believe an automated system is of such utility.

Such an automatic linking system would not only enable intra-linking collaborative encyclopedias, such as PlanetMath.org, but also allow for linking educational materials such as lecture notes, blogs, abstracts in research and educational digital libraries. Such usage could aid researchers and students in the better understanding of abstracts and full texts, and could also help them find related articles quickly. Automatic linking systems will likely also be useful as web services and/or plugins to document authoring systems.

While it is possible to extend our techniques for other types of linking such as links to articles with a similar or different point of view, it is our focus in this paper to study definitional or concept linking.

C. Challenges and Design Goals

Building an automatic invocation linking system for a collaborative online encyclopedia presents a number of computing challenges. We outline our design goals to address these challenges.

Linking Quality. The main analytic challenges lie in how to determine which terms or phrases to link and which entries to link to. Typical information retrieval and natural language processing issues such as plurality, homonyms, and polysemy are all relevant for the linking process and bear on the quality of linking. In addition, the task is doubly difficult in that both the link anchor and link destination are being identified and linked automatically.

In light of all these challenges, the analysis process is necessarily imperfect and so *linking errors* may be present. We characterize many such forms of errors as follows. Some of these errors take the form of links citing the incorrect homonym from a group of homonyms, while some take the form of linking when there should be no linking at all—a phenomenon which have termed *overlinking*. We use *mislinking* as a term to refer to any type of reduced *link precision* (the fraction of created links which are correct). From our example, if "graph" linked to object 6 instead of 5, then we have a mislink. If the term "even" were to link to any article in the corpus we would call this an overlink because "even" is not used in the mathematics sense.

An important goal of designing the automatic linking system is to improve the *linking precision* while maintaining high *link recall* (perfect link recall would mean a link is created for every concept label that can and should be linked given the present state of the corpus).

Linking across multiple sites. Online encyclopedias are typically organized into a classification hierarchy, and our experience has shown that this ontological knowledge can be utilized in order to dramatically increase the precision of automatic linking, largely solving the polysemy problem. Yet, this methodology presents problems when attempting to link across multiple sites (or *across domains*), as different knowledge bases may not use the same classification hierarchy. We discuss current and future efforts to solve this problem for automatic linking.

Dynamic Corpus. Most collaborative corpora change frequently, an automatic invocation linking system needs to efficiently update the links between entries that are related to newly defined or modified concepts in the corpus.

Efficiency and Scalability. In addition, a continually-changing corpus must be dealt in such a way that the analysis and processing of automatic links is tractable and scalable.

Ease of use and deployability. It is also necessary and important that an automatic linking system is easy to use for the adoption by a large user base and easy to setup for the widespread adoption for linking various materials across multiple sites.

D. Contributions

We designed and developed NNexus (Noosphere Networked Entry eXtension and Unification System), a system used to automate the process of automatically linking encyclopedia entries (or other definitional knowledge bases) into a semantic network of concepts. NNexus is an abstraction and generalization of the automatic linking component of the Noosphere system [7], which is the platform of PlanetMath

[3]Extracted from PlanetMath http://planetmath.org/encyclopedia/PlaneGraph.html

(planetmath.org), PlanetPhysics (planetphysics.org), and other Noosphere sites. To the best of our knowledge, it is the first automatic linking system that links articles and concepts with the use of a classification scheme, to make linking almost a "non-issue" for writers, and completely transparent to readers.

NNexus has a number of key features addressing the challenges we outlined above. First, it includes a customized information retrieval based automatic linking system coupled with a set of techniques such as ontology/subject driven link steering, and declarative linking priorities and clauses that are specifically designed to enhance the linking precision for a minority of "tough cases." Second, NNexus has mechanisms for efficiently updating the links between entries that are related to newly defined or modified concepts in the corpus. Third, NNexus achieves good efficiency and scalability by its efficient data structures and algorithm design. Finally, NNexus has a simple interface, which allows for an almost unlimited number of online corpora to interconnect for automatic linking.

In the rest of the paper we first explain the model behind NNexus and present some key technical details of its functioning. Then we discuss the interface to NNexus as a general, open source tool. Next we briefly discuss some evaluation and deployment results of NNexus. Finally, we discuss scenarios for applying NNexus beyond intra-linking in PlanetMath, including some we are working on.

II. NNexus Framework

In this section, we present the model behind NNexus and discuss key techniques and features in the NNexus framework.

A. Overview

Users of NNexus apply the following basic functionality to their corpus: When an entry is rendered (either at display time or during offline batch processing), the text is scanned for words (concept labels) that invoke concepts that have been defined in other entries. These words (or word tuples) are ultimately turned into hyperlinks to the corresponding entries in the output rendering.

In order to determine which entry to link to for a concept label, NNexus indexes the entries by building a *concept map* that maps all of the concept labels in the corpus to the entries which define these concepts (see Section II-B).

When an article is submitted, NNexus starts by pulling out unlinkable portions of text that need to be escaped (i.e., equations) and replaces them by special tokens. The engine then breaks the text of an entry into a single words/tokens array to iterate through. The tokens and token tuples (phrases) are then searched to determine candidate links using the concept map (see Section II-C). After the candidate links are determined they are filtered based on linking policies (see Section II-D). The candidates are then compared by "classification proximity" (see Section II-E.) The object with the closests classification is then the only object left in the match-candidates array (assuming a complete disambiguation). The "winning" candidates for each position are then substituted into the original text and the linked document is then returned.

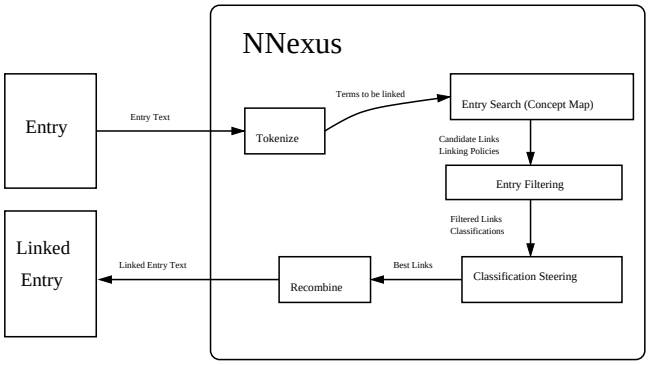


Fig. 2. Linking Diagram: When an entry is linked through NNexus the candidate links are found in the concept map. These candidates are then compared against the linking policies and sent through the classification module. The top candidate links are then recombined into the original text and returned to the user.

Figure 2 illustrates the conceptual flow of the automatic linking process.

In addition, when new concepts are added to the collection (or the set of concept labels otherwise changes), entries containing potential invocation of these concept labels can be *invalidated*. This allows entries to be re-scanned for links, either at invalidation time or before the next time they are displayed. NNexus uses a special structure called the *invalidation index* to facilitate this (see Section II-F).

This automatic system almost completely frees content authors from having to "think about links." It addresses the problems of both outgoing and incoming links, with respect to a new entry or new concepts.

However, it is not completely infallible, and in a epistemological sense, there is only so much that a system can infer without having a human-level understanding of the content. Because of this, the user can ultimately override the automatic linking, create their own manual links, or *steer* the automatic linker (all discussed in more detail later).

B. Indexing

NNexus indexes the entries by building a *concept map* that maps all of the concept labels in the corpus to the entries which define these concepts. The process of building the concept map follows. When adding a new object (entry) to NNexus a list of terms the object defines, synonyms, and a title are provided (the concept labels).

The concept labels are kept in a chained-hash index structure, called the *concept map*. This structure contains as keys the words that occur as the first word of some concept label. Following these words (retrieving the value for the key) leads to a list of full concept labels starting with that particular word. To facilitate efficient scanning of entry text to find concept labels, the map is structured as a chained hash, keyed by the first word of each phrase placed in it. This structure is shown graphically in Figure 3.

C. Entry Search

When searching for candidate links we are given an entry as an array of word tokens. This array form makes it easy to

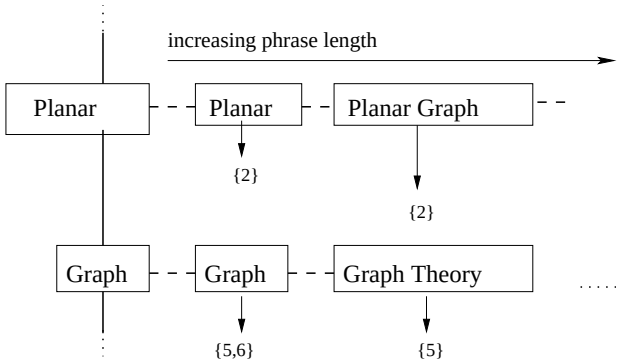


Fig. 3. Concept Map: a fast-access (chained-hash-based) structure filled with all the concept labels for all included corpora, used for determining available linking targets as the text is being scanned. This figure contains a subset that would be generated based on our example corpus.

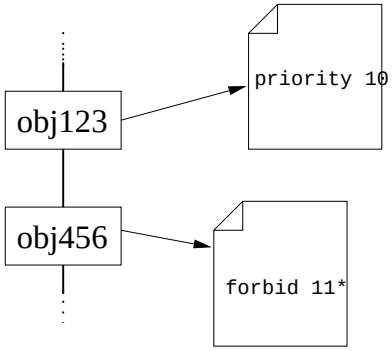


Fig. 4. The linking policy table stores a text chunk for each entry, containing optional user-supplied link-steering directives.

associate a particular word with a unique integer position. The now-tokenized text of the entry is then iterated over. If a word matches the start of an indexed concept label, the following words in the text are checked to see if they match the longest concept label starting with that word. If this fails, the next longest concept label is checked, and so on. When a matching concept label is found, it is included in the *match array*. In our example "graph", "plane", and "connected components" are all defined in the corpus. These terms (and phrases) are added to the match array.

D. Entry Filtering

One of the main contributions of NNexus to automatic linking is the classification steering and filtering techniques for entry selection.

Central to expressing linking restrictions is the *linking policy*, a set of directives controlling linking based on the subject classification system within the encyclopedia. NNexus allows authors to permit or forbid certain classes of articles from linking into their articles. The linking policy of an article describes, in terms of subject classes, to where links may be made or prohibited. For example, the linking policy for an entry on group theory might simply be that terms in the entry can only be linked to if the other object is also in the "group theory" class. Alternatively, an entry on set theory (because it is so elementary) might allow everyone to link to the terms it defines *except* restrict articles in the image processing class from linking to the term "image" (the word "image" has different meanings in the two areas).

For each object there is stored a text chunk representing the user-supplied linking policy (the linking policy is a series of directives which allow fine-tuned control of the linking behavior, mainly to resolve polysemous conflicts or prevent overlinking).

These policies are used during the final stage of processing an entry for linking, when it is being determined what the best source is for a concept label match, or whether the link should be created at all. The linking policy table is keyed by object ID. A diagram of the table is shown in Figure 4.

The linking policies were implemented to handle over-linking. The linking policies can be specified by the author but administrators also have the ability to modify the linking policies.

We also have a few efforts in progress exploring various ranking techniques by integrating multiple factors such as domain class, priority, pedagogical level, and reputation of the entries.

E. Classification Steering

Each object in the NNexus corpus may contain one or more classifications. The classification table maps entries (by object ID) to lists of classifications which have been assigned to them by users. NNexus uses classification to resolve ambiguous links (that is, links to concept labels which are polysemous). In our example "graph" has two possible link targets and the classification of our source article is MSC:05C40. We use the classification of the two possible targets (objects 5 and 6) to determine which is a better target. The classification hierarchy is represented as a weighted tree (see Figure 5). Each class is represented as a node in the tree. Edges represent parent/child relationships between the classes. NNexus compares the classes of the candidate objects to the classes of the source object and selects the object with the shortest distance in the classification tree. The distance between two classes is the shortest weighted path between the classes. NNexus uses Johnson's All Pairs Shortest Path algorithm to compute the distances between all classes at startup. NNexus supports any arbitrary weighting scheme for the edge weights, but we now discuss our recommended methods for assigning weights to the edges.

NNexus is bundled with a utility that converts an OWL formated ontology with `class` and `subClass` relationships into a weighted graph that is stored in the NNexus database. We first build a graph (usually a tree) structure of parent/child relationships.

Consider the classification hierarchy in Figure 5 and ignore the weights on the edges. Consider classes 05C40, 05-XX, and 05C10. We would like to determine whether class 05-XX of 05C10 is closer to 05C40 (when no weights are assigned the distances are equal). 05-XX, 05C10, and 05C40 correspond to Combinatorics, Topological Graph Theory, and

```
...
<owl:Class rdf:ID="root">
<rdfs:label>root</rdfs:label>
</owl:Class>
<owl:Class rdf:ID="01-XX">
  <rdfs:label>01-XX</rdfs:label>
  <rdfs:comment>History and biography</rdfs:comment>
   <rdfs:subClassOf rdf:resource="#root" />
</owl:Class>
<owl:Class rdf:ID="03-XX">
   <rdfs:label>03-XX</rdfs:label>
   <rdfs:comment>
     Mathematical logic and foundations
   </rdfs:comment>
   <rdfs:subClassOf rdf:resource="#root" />
</owl:Class>
<owl:Class rdf:ID="05-XX">
    <rdfs:label>05-XX</rdfs:label>
    <rdfs:comment>Combinatorics</rdfs:comment>
    <rdfs:subClassOf rdf:resource="#root" />
</owl:Class>
...
```

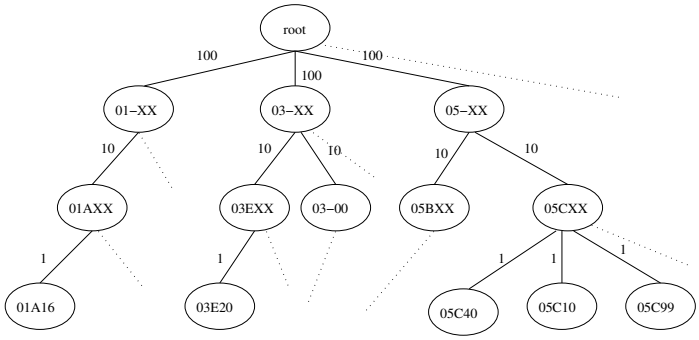


Fig. 5. Example Classification Tree: This is the MSC subject classification represented as an OWL formatted file and as a weighted graph. The weights are assigned with base 10.

Graph Connectivity, respectively. We then would hope that our system would determine that 05C10 is closer to 05C40. In general, classes at the same level and in the same subtree should be considered closer than classes at a higher level in the same subtree. It should also be noted that classes deeper in a subtree are more closely related than classes higher in the same subtree. E.g. 05C10 and 05C40 are more closely related than 05-XX and 03-XX. Based on these heuristics we define the weight of an edge in the graph as

$$w(e) = b^{height-i-1}$$

where b is the chosen base weight (default is 10), $height$ is the height of the tree (or in general the distance of the longest path from the designated root node), and i is the distance of the edge from the root.

Refering to our beginning example we link "graph" to object 5 because the distance from 05C99 is shorter in the weighted classification graph than 03E20 to 05C40.

At this point NNexus supports the MSC classification hierarchy. The MSC is broken down into over 5,000 two-, three-, and five-digit classfications, each corresponding to a discipline of mathematics (e.g., 11 = Number theory; 11B = Sequences and sets; 11B05 = Density, gaps, topology where 11 $\supset$ 11B $\supset$ 11B05)[4]. See Figure 5 for an example of the MSC classification structure represented as a weighted graph.

To address the general problem of inter-linking multiple corpora it is necessary to consider mapping (or otherwise combining) multiple, differing classification ontologies. We are currently investigating the techniques discussed in [14] and [15] and implementing this type of functionality in our system. [5]

F. Invalidation

When a new object is added, NNexus also utilizes an *invalidation index* to determine which articles may possibly link to the new object and need to be "invalidated." The invalidation index stores term and phrase *content* information for all entries in the corpus. It is an adaptive index in that longer phrases are only stored if they appear frequently in the collection. There is no limit to how long a stored phrase can be; however, very long phrases are extremely unlikely to appear (the falloff in occurrence count by phrase length follows a Zipf distribution).

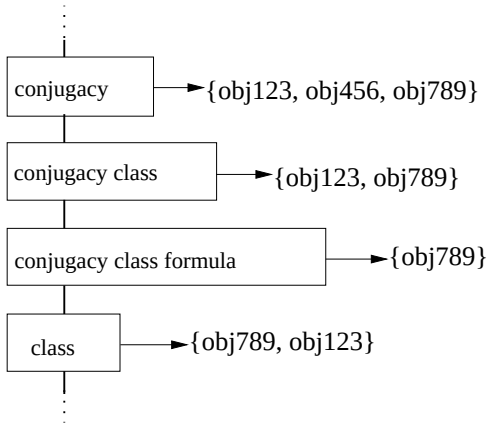


Fig. 6. Invalidation Index: an adaptive inverted index containing both words and phrases, used for determining which text objects are likely to need to be re-analyzed for linking after concept definition updates have occurred to the corpus. The structure is a chained hash, with words and phrases as hash keys, and an object identifier list for each. In the above example, if a definition for "conjugacy class formula" were added to the corpus, only object 789 would need to be invalidated.

The invalidation index is a variation on a standard text document inverted index structure and works in the usual way for lookups. However, instead of just being keyed on single-word terms, it is keyed on phrases (which are usually but not always single-word). For each term or phrase in the index, there is a list of objects which contain that term or phrase. These lists are called *postings lists*. A sketch of the invalidation index is shown in Figure 6.

The invalidation index has a special property that for every phrase indexed, all shorter prefixes of that phrase are also indexed for every occurrence of the longer phrase. This allows

[4]For more information see http://www.ams.org/msc/

[5]For more information on ontology mapping, we recommend the survey in [5].

us to guarantee that occurrences of the shorter phrases or single terms will be noticed if we do a lookup using these shorter tuples as keys. The importance of this will be made clear later.

The invalidation index exists for a single purpose: so that when concept labels are added to the collection (or when they change), we can determine which entries are highly likely to be effected by the change—that is, they likely link to the newly-added concept. The invalidation index allows us to do this in a way that never misses an entry that should be re-examined, but does not catch too many irrelevant entries (false positives).

When a lookup is done for a particular phrase in the invalidation index, the object IDs returned are updated (invalidated) in the cache table, which means they should be re-analyzed by the linker before being viewed.

G. Other Features and Characteristics

In this subsection we give a brief overview of a few other significant features and characteristics of NNexus:

- **Longest phrase match.** NNexus always performs longest phrase match at each location in the text. For example, if the writer mentions the phrase "orthogonal function" in their entry and links against a collection defining all of "orthogonal," "function," and "orthogonal function," then NNexus links to the latter. This is based on a nearly universally-consistent assumption of natural language, which is that longer phrases semantically subsume their shorter atoms.
- **Morphological invariance.** NNexus also performs some morphological transformations on concept labels in order to ensure they can be linked to in most typical usages. The first, and most important transformation, has the effect of invariance of pluralization. The second invariance is due to possessiveness. Another morphological invariance concerns international characters. When a token is checked into the index, NNexus will ensure that the token is singular and non-possessive, with a canonicalized encoding.
- **Link Suppression.** Automatic linking tends towards full recall, which produces *overlinking* in light of polysemy. One example of this is when a writer uses a word in a natural language sense (e.g. "even") which is also the title for an encyclopedia object (e.g. "even number"). In this case, automatic linking will turn that word into a hyperlink to the "offending" object. For this reason, users can escape certain words and phrases from being linked by NNexus using linking policies at the source.

III. NNexus API

NNexus was developed with Perl and was designed to have the minimum amount of dependencies necessary while still running efficiently. Thus, NNexus only requires a database system (currently MySQL is supported) and some Perl XML packages (available from CPAN). NNexus has been designed with an API so that it can be used with any document corpus and with client software written in any programming language.[6] Figure 7 shows a diagram of the overall NNexus system architecture.

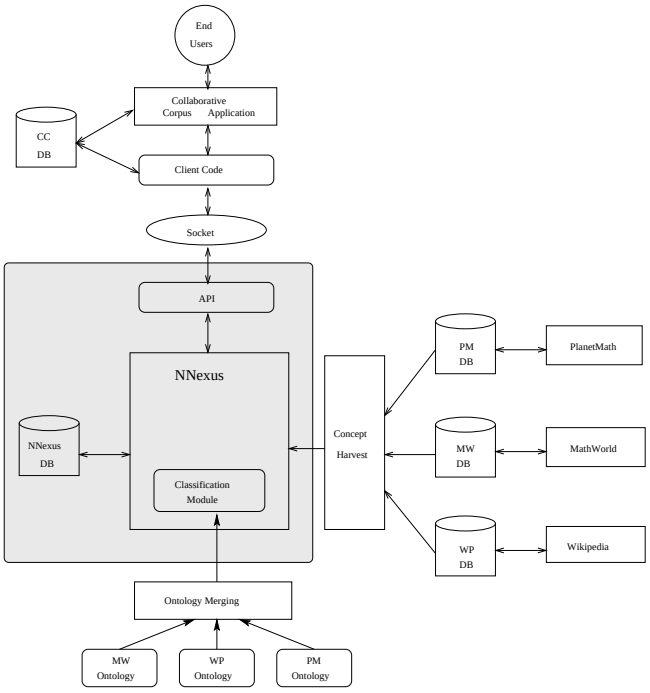


Fig. 7. NNexus System Architecture (in an example deployment): The shaded region denotes NNexus proper. The classification module provides classification-invariant link steering between multiple ontologies.

One of the design goals of NNexus was ease of deployability, programmability, and use. For this reason, NNexus uses simple XML formats for its communications and configuration. We give some examples below.

An example configuration file is given in Figure 8. Adding a new entry (along with the concepts it defines) to the corpus utilizes an XML command fragment akin to the example in Figure 9. The protocol allows adding multiple objects with one request, to facilitate batch loading. Figure 10 gives an example of linking an article with NNexus.

IV. Evaluation and Results

The core methods of NNexus have essentially proven their large-scale applicability in the PlanetMath[7] system, a collaborative and dynamic mathematics encyclopedia powered by our automatic invocation linking system. As of this writing it had more than 6,700 entries, declaring more than 11,000 concepts.

Below we also give some initial results examining NNexus in terms of the linking quality, efficiency and scalability, and its deployability to other applications.

A. Linking Quality

We define recall as the number of created ("retrieved") links over the number of possible links (given the concept labels declared in the knowledge-base) and precision as the number

[6] NNexus is released under an MIT/X11 style license.

[7] http://www.planetmath.org

```
<config>
    <domains>
        <domain>
            <name>planetmath.org</name>
            <link>http://link.to/xml/config/file</link>
            <urltemplate>
                http://planetmath.org/?op=getobj&from=objects&id=
            </urltemplate>
            <defaultscheme>msc</defaultscheme>
        </domain>
        <domain>
            <name>mathworld.com</name>
            <link>http://link.to/xml/config/file</link>
            <urltemplate></urltemplate>
            <defaultscheme>mw</defaultscheme>
        </domain>
    </domains>

    <database> ... </database>

    <server>
        <port>7070</port>
        <supported>  <!-- (classification schemes) -->
            <scheme>msc</scheme>
            <scheme>mw</scheme>
        </supported>
    </server>
</config>
```

Fig. 8. A sample NNexus config file, for linking to two math encyclopedias.

```
<request>
   <addobject>
     <entry>
        <title>same as above</title>
        <defines>thing</defines>
        <defines>widget</defines>
        <synonym>term3</synonym>
        <synonym>phrase of terms</synonym>
        <domain>planetmath.org</domain>
        <body>The body text</body>
        <objid>a3db</objid>
        <linkpolicy>permit 03A</linkpolicy>
        <author>1</author>
        <class>012A</class>
        <class>02ADD</class>
     </entry>
     <entry>
        ...
      </entry>
   </addobject>
</request>

<response>
   <invalid>ExternalID</invalid>
   <invalid>AnotherExternalID</invalid>
</response>
```

Fig. 9. An example protocol snippet of adding an object and concepts to NNexus and the response of invalid object IDs (from the invalidation index).

```
<request>
   <linkentry>
      <!-- on demand linking -->
      <body> full text of article </body>
      <class>03FA2</class>
       <!-- or -->
       <objid>objectid</objid>
       <domain>domain.org</domain>
   </linkentry>
</request>

<response>
   <body>full text of article with links added.</body>
   <links>[string of all links separated by commas]</links>
</response>
```

Fig. 10. An example protocol linking an object and the response from NNexus.

Statistic	Value
Targets before disambiguation	90342
Targets after disambiguation	67460
Links made	57761
Links made without disambiguation needed	38961
Links made with disambiguation needed	18800
Number of targets reduced by disambiguation	11762
Number of completely disambiguated links made	10648
% Reduced that needed disambiguation	62.6%
% Completely reduced that needed disambiguation	56.6%
% Completely reduced out of reduced	90.5%
% Links with only one target after disambiguation	85.9%

TABLE I
DISAMBIGUATION STATISTICS: LINKING ALL 4841 ENTRIES IN A SNAPSHOT OF THE PLANETMATH CORPUS.

of *correct* links over the number of created links. The Noosphere linking system was designed for near-perfect link recall. Link precision was not initially considered. However, with the general growth of the PlanetMath collection, it was found that precision began to fall, due to synonymy and various other problems which will be discussed in more detail. This is why we introduced linking policies that utilize classification-based filtering (see Section II-D and Section II-E).

In order to characterize the effects of this classification-based disambiguation, we performed a study to determine to what degree link targets are disambiguated based only on the disambiguate-classification-graph algorithm. For the study, we kept track of how many targets there were before disambiguation and how many targets after disambiguation. We say that a link was *reduced* if the number of targets after the disambiguation process is less than the number of targets before. We say that a link was *completely reduced* if there is only one target after disambiguation. % "Reduced that needed disambiguation" means the number of links that had more than one target before disambiguation. % "Completely reduced that needed disambiguation" is equal to the number of reduced links divided by the number of links made that needed disambiguation. % "Links with only one target after disambiguation" corresponds to the number of links that had only one target after disambiguation divided by the total number of links made.

We found that 85.9% of links had only one target after disambiguation. This verifies our hypothesis and real-world experience that disambiguation helps reduce the number of targets and as a result improves the precision of linking. See Table I for a list of all relevant statistics. A more thorough study of precision on a random subset of the collection follows.

We also performed a mislinking and overlinking study in June 2006 on the PlanetMath collection with and without linking policies. About 12% of links were mislinks and 7.9% of links were overlinks (thus 61.1% of the mislinks were overlinks). A similar, formative study had been performed in 2003 [7], and the results were consistent with the latest. Notably, these two studies span an increase in collection size of about 3,000-4,000 entries. This suggests that, as a general rule, about 12-15% mislinks can be expected in a real-world corpus with only lexical matching and classification steering.

However, based on these results, we believe linking preci-

Statistic	Before	After
number of links	156	145
good links	135	135
mislinks	21	10
overlinks	18	7
% mislink	13.4	6.9
% overlink	11.5	4.8
Precision	86.5	93.1

TABLE II
OVERLINKING STATISTICS: BEFORE AND AFTER UPDATING THE LINKING POLICIES FOR THE OFFENDING ENTRIES OF THE 5 RANDOM ENTRIES IN A RANDOM SUBSET OF 20.

sion with NNexus will typically approach or exceed 95% by adding linking policy capabilities and applying them to just a small subset of the collection.

To begin to explore this assumption we randomly selected 20 objects from the PlanetMath corpus and analyzed the linking quality, manually checking all links in the subset. This small corpus had 13.4% mislinks and 11.5% overlinks (that is, about 86% of mislinks were due to overlinks). We then randomly selected 5 of these objects and fixed all of their overlinks by creating new link policies (added to 8 problematic target objects). After eliminating all overlinks for these 5 objects, we resurveyed the initial 20 objects for linking quality. We found that the mislinking went down to 6.9% and the overlinking was reduced to 4.8%. See Table II for a before and after comparison. This provides compelling support for our hypothesis that overlinking, which represents at least two-thirds of the precision shortfall in our collection, can be largely eliminated by adding linking policies to a small subset of it.

A comparable system to Noosphere is Mediawiki (which powers Wikipedia). Mediawiki does not use automatic linking—links are manually-delimited by authors when the author invokes a concept that they believe should be in the collection. Thus, Wikipedia (and any similar wiki system) has near-perfect linking precision, but link *recall* is unknown. If an entry for a concept is present only by an alternate name, the link might fail to be connected. Links to non-existent entries are rendered specially, and the system makes it easy to create a new entry for that term. However, this is inherently somewhat distracting to those uninterested in creating a new entry.

A survey in [11] shows that about 97-99% of Wikipedia links are accurate. However, this study is not directly comparable to our survey because it relies on special "disambiguation nodes" (which are an additional distraction) and doesn't measure link recall (underlinking).

Most significantly from a usability and productivity standpoint, no formal comparison of the effort required for link maintenance in the manual vs. automatic paradigms has been made.

B. Scalability and Efficiency

To study the scalability and efficiency of our approach, we ran experiments on a Mac running Mac OS X with a 1.67 GHz PowerPC G4 and 1GB DDR2 SDRAM. We selected random subsets of size 100, 200, 500, 1000, 2000, and 4841 from the PlanetMath corpus and kept track of the number of seconds to link every object in the subset corpora.

Corpus Size	# of Links	Total Time	Time/Link
100	94	28.3	.301
200	517	72.4	.140
500	1886	223.1	.118
1000	4845	552.8	.114
2000	12306	2160.7	.176
4841	58077	8620.7	.148

TABLE III
SCALABILITY STUDY: RUNNING LINKING ON RANDOM SUBSETS OF OUR TEST CORPUS OF GRADUALLY INCREASING SIZE.

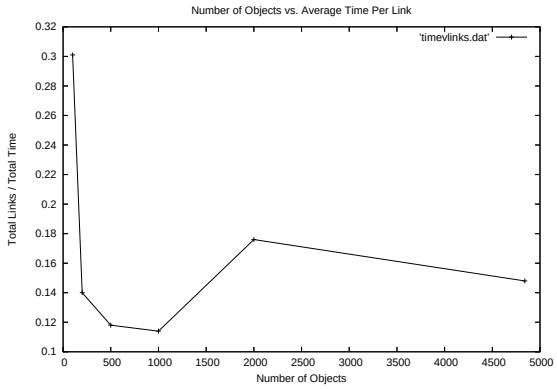


Fig. 11. Scalability study: time-per-link for progressively larger corpora, showing clearly that the automatic linking process is sub-linear in time complexity.

Table III and Figure 11 show the performance results for different corpus sizes. We can see that the time per link quickly falls off and then hovers around a constant value as the collection grows. This indicates that NNexus is not only efficient but also scalable to very large corpus sizes.

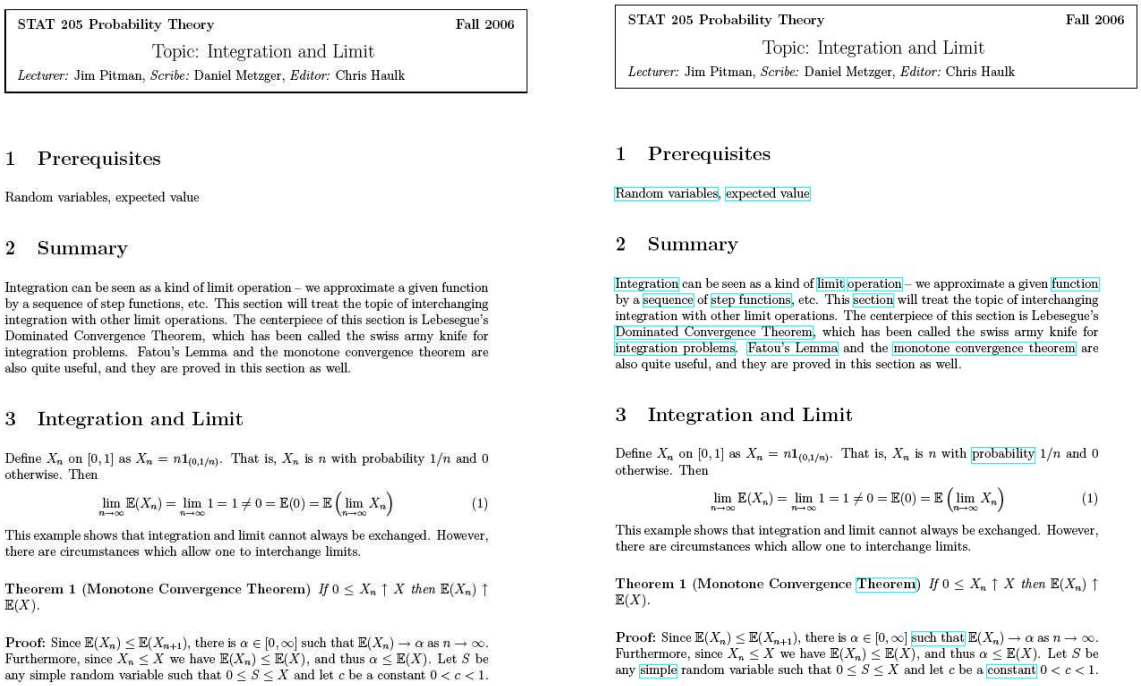
C. Deployability and Other Applications

In addition to enabling intra-linking in an single encyclopedic knowledge base such as PlanetMath, NNexus also provides an generalized automatic linking solution to a variety of potential applications.

One application of NNexus that we are currently pursuing is the linking of lecture notes to math encyclopedia sites (including PlanethMath and MathWorld, but potentially extending to others, such as Wikipedia, the Digitial Library of Mathematical Functions, and more). Figure 12 illustrates this application, showing screenshots of automatically-linked notes from a probabilities course taught by Jim Pitman at UC Berkeley, before and after automatic linking with NNexus (the links in this example are to both PlanetMath and MathWorld).

Due to the ease-of-use and success of linking lecture notes we are confident that we can extend NNexus to other applications with minimal additional effort. We are interested in the linking of abstracts in research and educational digital libraries. This would enable learners (students or researchers) to quickly find related articles and also would help the user better understand the underlying concepts in the abstracts.

We are also interested in applying automatic linking to educational blogs, which are of increasing prevalence and



STAT 205 Probability Theory Fall 2006

Topic: Integration and Limit

Lecturer: Jim Pitman, *Scribe:* Daniel Metzger, *Editor:* Chris Haulk

1 Prerequisites

Random variables, expected value

2 Summary

Integration can be seen as a kind of limit operation – we approximate a given function by a sequence of step functions, etc. This section will treat the topic of interchanging integration with other limit operations. The centerpiece of this section is Lebesegue's Dominated Convergence Theorem, which has been called the swiss army knife for integration problems. Fatou's Lemma and the monotone convergence theorem are also quite useful, and they are proved in this section as well.

3 Integration and Limit

Define X_n on $[0,1]$ as $X_n = n\mathbf{1}_{(0,1/n)}$. That is, X_n is n with probability $1/n$ and 0 otherwise. Then

$$\lim_{n\to\infty} \mathbb{E}(X_n) = \lim_{n\to\infty} 1 = 1 \neq 0 = \mathbb{E}(0) = \mathbb{E}\left(\lim_{n\to\infty} X_n\right) \quad (1)$$

This example shows that integration and limit cannot always be exchanged. However, there are circumstances which allow one to interchange limits.

Theorem 1 (Monotone Convergence Theorem) *If* $0 \leq X_n \uparrow X$ *then* $\mathbb{E}(X_n) \uparrow \mathbb{E}(X)$.

Proof: Since $\mathbb{E}(X_n) \leq \mathbb{E}(X_{n+1})$, there is $\alpha \in [0,\infty]$ such that $\mathbb{E}(X_n) \to \alpha$ as $n \to \infty$. Furthermore, since $X_n \leq X$ we have $\mathbb{E}(X_n) \leq \mathbb{E}(X)$, and thus $\alpha \leq \mathbb{E}(X)$. Let S be any simple random variable such that $0 \leq S \leq X$ and let c be a constant $0 < c < 1$.

Fig. 12. Screenshot of original (left) and automatically-linked (right) lecture notes using NNexus. The links in this example are to definitions on both MathWorld and PlanetMath, depending on which site had each particular definition available, and in the case both did, a domain priority configuration option (eventually, classification-based steering will also play a great role). Concepts were "bridged" from MathWorld using that site's OAI repository.

impact on the web, and are being embraced by large-scale efforts such as the NSDL.[8]

The modular design of NNexus will also allow developers to use NNexus as a web plugin for on-demand text linking and for various document authoring applications. NNexus could be deployed as a web service to allow third parties to link arbitrary documents to particular corpora.

V. Related Work

The semantic linking problem we studied in the paper bears similarities to the search problem on the web. However in our problem not only the link destination but also the link anchor need to be identified and linked automatically. There are many standard methods for improving searching quality in information retrieval literature that have been applied to the current generation of search engines [3], yet for the most part most of the work in IR has not been explored in the collaborative semantic linking context [6]. Little if anything has been done to examine the overlap of the problem spaces, which is unsurprising given the novelty of collaboratively-built knowledge-bases.

There are several efforts [9], [8], [10] towards using a wiki for collaboratively editing semantic knowledge bases where users can specify semantic information including links in addition to standard wiki text. Most of them focus on improving usability and integrating machine readable data and human-readable editable text. We are not aware of any approach that supports automatic linking to the extent of our present work.

Among the semantic information, links are arguably the most basic and also most relevant markup within a wiki and are interpreted as semantic relations between two concepts described within articles. [10] provides an extension to be integrated in Wikipedia, that allows users to specify typed links in addition to regular links between articles and typed data inside the articles. It would be interesting to see how our framework can be extended to include such semantic enhancements on linking.

There is currently a surge of interest in utilizing Semantic Web technologies for e-learning. [12] discusses the differences between classical training and e-learning and presents different Semantic Web layers and how they can be applied to e-learning. [13] defines "dynamic assembly," which is the process of connecting relevant search results into a learning path for users and linking the learning objects into an organized structure.

Ontologies and metadata and their application to eLearning are discussed in [16]. The standards discussed and used in the paper are the Dublin Core Schema[9] and LOM[10]. The Dublin Core Initiative provides simple standards to facilitate the finding, sharing and management of information on the web and is gaining popularity on the web and is used by many OAI repositories. The LOM data model specifies which aspects of a learning object should be described and how to access and modify these objects

There is also significant research on automatic metadata generation. For example, [4] presents a framework that automatically generates learning object metadata. This can be

[8]For their "Expert Voices" service. See http://www.nsdl.org/.

[9]http://dublincore.org/

[10]http://www.imsglobal.org/metadata/

compared with search engines on the web that index web pages in the background without any intervention of the creator or the host of the site. Although dealing with a different aspect of metadata generation, the work supports a similar viewpoint as ours: users should not have to bother with a laborious process of *ab initio* metadata creation when machine learning can help. If the user wants to correct, add or delete metadata, they will still be able to do so—but most users, most of the time, should be insulated from the task (left to specify the most simple, intuitive, classification meta-information).

VI. Future Directions & Conclusion

Our work in NNexus continues along several threads. A major near-term research and development item is the expansion of ontology mapping capabilities for link steering. We are also continually improving the policy-based link steering and filtering capabilities. Similarly, we continue to optimize the system, and are working to expand its generalization (for instance, abstracting input parsing and output generation to different markup languages). In addition, we are also exploring reputation systems and collaborative filtering techniques [1] to address issues of "competing" entries and different needs and preferences of authors. This especially becomes an issue when one goes beyond a single collaborative corpus, as would typically be the case in linking to them by third parties.

We have presented the challenges of automatically interlinking a dynamic corpus and introduced NNexus, a modular system for performing this task. The achievements of the precursor to the NNexus system, the Noosphere automatic linker, can be seen at PlanetMath.[11] NNexus is now available for general use as open source software, [12] and we look forward to working with others to improve it and apply it more widely to enhance the semantic quality of the web in general.

Acknowledgements

This work has been partially-supported by the Google Summer of Code Program and the Institute of Mathematical Statistics (IMS).

References

[1] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. *IEEE transactions on knowledge and data engineering*, 17(6), 2005.

[2] Troels Andreasen, Jorgen Fischer Nilsson, and Hanne Erdman Thomsen. Ontology-based querying. In *Flexible Query-Answering Systems*, pages 15–26, 2000.

[3] Ricardo A. Baeza-Yates and Berthier A. Ribeiro-Neto. *Modern Information Retrieval*. ACM Press / Addison-Wesley, 1999.

[4] Kris Cardinaels, Michael Meire, and Erik Duval. Automating metadata generation: the simple indexing interface. In *WWW '05: Proceedings of the 14th international conference on World Wide Web*, pages 548–556, New York, NY, USA, 2005. ACM Press.

[5] Yannis Kalfoglou and Marco Schorlemmer. Ontology mapping: The state of the art. In Y. Kalfoglou, M. Schorlemmer, A. Sheth, S. Staab, and M. Uschold, editors, *Semantic Interoperability and Integration*, number 04391 in Dagstuhl Seminar Proceedings. Internationales Begegnungs- und Forschungszentrum fuer Informatik (IBFI), Schloss Dagstuhl, Germany, 2005.

[6] J. Kolbitsch and H. Maurer. Community building around encyclopeadic knowledge. *Journal of Computing and Information Technology*, 14, 2006.

[7] Aaron Krowne. An architecture for collaborative math and science digital libraries. Master's thesis, Virginia Polytechnic Institure and State University, Blacksburg, VA, 2003.

[8] Adam Souzis. Building a semantic wiki. *IEEE Intelligent Systems*, 20(5):87–91, 2005.

[9] S. E. Roberto Tazzoli and Paolo Castagna. Towards a semantic wiki wiki web. In *In Demo Session at ISWC2004*, 2004.

[10] Max Völkel, Markus Krötzsch, Denny Vrandecic, Heiko Haller, and Rudi Studer. Semantic wikipedia. In *WWW '06: Proceedings of the 15th international conference on World Wide Web*, pages 585–594, New York, NY, USA, 2006. ACM Press.

[11] G. Weaver, B. Strickland, and G. Crane. Quantifying the accuracy of relational statements in wikipedia: a methodology. In *JCDL '06: Proceedings of the 6th ACM/IEEE-CS joint conference on Digital libraries*, 2006.

[12] L. Stojanovic, S. Staab, and R. Studer. eLearning based on the Semantic Web. In *WebNet2001: World Conference on the WWW and Internet, Orlando, Florida, USA*, 2001.

[13] R. Farrel, S. Liburd, and J. Thomas. Dynamic Assembly of Learning Objects. In *WWW '04: Proceedings of the 13th international conference on World Wide Web*, 2004.

[14] Natalya Fridman Noy and Mark A. Musen. PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*, 2000.

[15] Zharko Aleksovski and Michel Klein. Ontology mapping using background knowledge. In *K-CAP '05: Proceedings of the 3rd international conference on Knowledge capture*, 2005

[16] J. Brase and W. Nejdl. *Ontologies and Metadata for eLearning.* Springer Verlab, 2003.

[17] Keizo Oyama and Hironobu Gotoda. Dublin Core Conference. In *DC-2001, Proceedings of the International Conference on Dublin Core and Metadata Applications*, 2001.

[18] M. Nilsson, M. Palmer, and J. Brase. The LOM RDF Binding - Principles and Implementation. 2003.

[11] http://planetmath.org/.

[12] http://aux.planetmath.org/nnexus/.