NNexus: An Automatic Linker for Collaborative Web-Based Corpora

James Gardner, Aaron Krowne, and Li Xiong

Abstract—In this paper, we introduce Noosphere Networked Entry eXtension and Unification System (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. The main challenges of this problem space include: 1) linking quality (correctly identifying which terms to link and which entry to link to with minimal effort on the part of users), 2) efficiency and scalability, and 3) generalization to multiple knowledge bases and web-based information environment. We present the NNexus approach that utilizes subject classification and other metadata to address these challenges. We also present evaluation results demonstrating the effectiveness and efficiency of the approach and discuss ongoing and future directions of research.

Index Terms—E-learning, automatic linking, wiki, Semantic Web.

1 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.

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.

1. http://www.wikipedia.org.
2. http://www.planetmath.org.

- *J. Gardner and L. Xiong are with the Department of Mathematics and Computer Science, Emory University, 400 Dowman Dr., Atlanta, GA 30322. E-mail: jgardn3@emory.edu, lxiong@mathcs.emory.edu.*
- *A. Krowne is with the Woodruff Library, Emory University, 400 Dowman Dr., Atlanta, GA 30322, and Planetmath.org, Ltd., 4336 Birchlake Ct, Alexandria, VA 22309. E-mail: akrowne@gmail.com.*

Manuscript received 12 July 2007; revised 16 Jan. 2008; accepted 16 June 2008; published online 25 June 2008.
Recommended for acceptance by Q. Li, R.W.H. Lau, D. McLeod, and J. Liu. For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number TKDE-2007-07-0362. Digital Object Identifier no. 10.1109/TKDE.2008.136.

1.1 Problem Definition

We study the problem of invocation linking in this paper to build a semantic network for collaborative online encyclopedia. We first define a number of terminologies and define our problem to facilitate our discussion. 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* or an *object*. We say *invocation* referring to a specific kind of semantic link: that of *concept invocation*. Concept invocation refers to the use of a concept in the text. 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 tuples of words *concept labels*. An *invocation link* is a hyperlink from these tuples of words in an entry that represent a concept to an entry that defines the concept. We refer to the tuples of words being linked from as *link source* and the entry being linked to as *link target*. The problem of *invocation linking* is how to add these invocation links in a collaborative online encyclopedia.

While it is possible to extend the problem definition and the techniques developed in this paper 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 *concept* or *definitional* linking.

The table in Fig. 1 shows a list of entries (objects) in an example online encyclopedia[3] corpus with their object ID and metadata including what concepts each entry defines and the Mathematical Subject Classification (MSC) for each entry. It also shows an example entry[4] with links to

3. http://planetmath.org.
4. Extracted from http://planetmath.org/encyclopedia/PlaneGraph.html.

ObjectId	Concepts Defined	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 ...

Fig. 1. Example document corpora with metadata and example entry.

concepts that are defined in the same corpus. The terms underlined indicate terms that need to be linked based on the metadata in the table. For example, *planar graph* in the example entry needs to be linked to object (entry) 2 that defines the concept *planar graph*. We will use this example to explain the concepts discussed in this paper.

1.2 Existing and Potential Solutions

We briefly survey the existing and potential solutions for invocation linking and motivate our automatic linking approach. The existing and potential linking approaches can be mainly classified into the following three categories, namely, *manual linking*, *semiautomatic linking*, and *automatic linking*:

- *Manual linking* refers to the linking technique where both the link source and link target are explicitly defined, e.g., anchor tags in html documents. Most current online encyclopedias or wikis use the manual approach. Blog software (such as Wordpress) generally requires writers create links manually.
- *Semiautomatic linking* refers to the technique where the terms at the source are explicitly marked for linking, but the link target is determined by the collaborative online encyclopedia system. Some current online encyclopedias (including Wikipedia) use the semiautomatic approach.
- *Automatic linking* (or more specifically automatic invocation linking) refers to the technique where the terms at the source and link target are both automatically determined by the system. This is the approach that we advocate in order to build the semantic network with minimal manual effort [7].

Wikipedia (which is powered by the Mediawiki software) uses a semiautomatic approach. That is, the links are manually delimited by authors when the author invokes a concept that they believe should be defined in the collection, but the system figures out the destination. If an entry for a concept is present only by an alternate name, the link might fail to be connected. Links to nonexistent entries are rendered specially as "broken" links, and the Mediawiki system makes it easy to start a new entry for that term. However, this is inherently somewhat distracting to those uninterested in creating a new entry. Mediawiki and other systems that take a similar approach also fail to provide systemic treatment of homonymy. The Wikipedia convention is to manually create "disambiguation nodes," which contain links to all homonymous concepts with a particular label. Such nodes add an extra step to navigation, require ongoing maintenance, and can contain an extremely random and distractive jumble of topics.

The perspective taken in our work is that the manual and semiautomatic approaches are an unnecessary burden on contributors, since the knowledge management environment (or wiki) should "know" which concepts are present and how they should be cited. By contrast, authors will usually not be aware of all concepts that are already present within the system—especially for large or distributed corpora.

A more challenging problem with the manual and semiautomatic 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 reinspection of the entire corpus by writers or other maintainers, which is a $\mathcal{O}(n^2)$-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 interlinkage 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.

There are a number of potential technologies that one might apply to the *automatic linking* approach. We briefly review them below and discuss their limitations and implications on the automatic linking problem.

One technology is the information retrieval (IR) approach [3] for web search. One part of our problem in identifying the best linking target for a concept label bears similarity to the search problem in finding the most relevant documents based on a keyword. Yet, for the most part, the work in IR has not been explored in the collaborative semantic linking context [6]. While typical IR issues such as plurality, homonyms, and polysemy are all relevant for the linking process, some of the techniques are not directly applicable. For example, the traditional IR approach relies on term-frequency and inverse document-frequency (TFIDF) to rank the retrieved documents. In our problem, the entries that define a particular concept may not contain the actual concept label (terms). Thus, the term-frequency-based approaches are not directly applicable in identifying the best link target (entry). In addition, in our problem, not only the link target but also the link source need to be identified and linked automatically.

Another technology is the recommender systems [1] that aim to predict ratings of a particular item for a particular user using a set of similar users based on a user-item rating matrix. At an initial glance, we can model our problem as an entry-entry link matrix where each cell represent a link or nonlink from a certain entry to another entry and use entry similarities to help determine the best entry to link to for a term that belongs to a certain entry. While this approach is more appropriate for relevance linking and may help to narrow down the potential link targets, it alone is not sufficient for the invocation or concept linking

problem. Nevertheless, it is on our research agenda to enhance our current link ranking strategy by adapting the collaborative filtering technologies to enhance the linking precision by incorporating entry similarities and user feedback into the linking process.

1.3 Design Goals

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.

In this paper, we sought to build an automatic linking system using the metadata of the entries. Building such an automatic invocation linking system for a collaborative online encyclopedia presents a number of computing challenges. We discuss the challenges below and outline our design goals to address them.

Linking quality. The main analytic challenges lie in how to determine which terms or phrases to link and which entries to link to. Typical IR 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 light of all these challenges, the linking process is necessarily imperfect and so *linking errors* may be present. We characterize many such forms of errors as follows:

- *Mislinking* refers to the error that a term or phrase is linked to an incorrect link target, e.g., an incorrect homonym from a group of homonyms. For example, in our sample entry shown in Fig. 1, if "graph" is linked to object 6 instead of 5, then we have a mislink.
- *Overlinking* refers to the error that a term or phrase is linked when there should be no link at all. Note that overlinking also contributes to mislinking because the term is mislinked. For example, if the term "even" is used as a common term (not in a mathematics sense) but was linked to an entry that defines "even number," we have an overlink.
- *Underlinking* refers to the error that a term or phrase is not linked when there should be a link because it invokes a concept that is defined in the corpus. For example, consider our sample entry shown in Fig. 1 again, if "planar graph" is not linked, then we have an underlink.

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

Efficiency and scalability. Another important design goal of an automatic linking system is its efficiency so the links can be created near-real time during rendering of the entries and its scalability so it can handle the large size of an online encyclopedia corpus. In addition, 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. A continually changing corpus must be dealt in such a way that the analysis and processing of automatic links is tractable and scalable.

Generalization to multiple corpus. 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 set up for the widespread adoption for linking various materials across multiple sites.

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 the Web Ontology Language (OWL). Our design goal is to leverage these standards so that our automatic linking system would not only enable intralinking collaborative encyclopedias, such as PlanetMath.org, but also allow for linking educational materials such as lecture notes, blogs, and 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.

1.4 Contributions

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

NNexus has a number of key features addressing the challenges we outlined above. We summarize the contributions of this paper below:

- First, it provides an effective indexing and linking scheme that utilizes metadata to automatically identify link sources and link targets. It achieves perfect link recall without *underlinking* error. It uses a classification-based link steering approach to address the mislinking problem and enhance the link precision. It also provides an interactive entry filtering component to address overlinking problem and further enhance the link precision for a minority of "tough cases."
- Second, NNexus achieves good efficiency and scalability by its efficient data structures and algorithm design. It has mechanisms for efficiently updating the links between entries that are related to newly defined or modified concepts in the corpus.

- Finally, NNexus is fully implemented and we evaluate it through an extensive set of experiments using real online corpus and demonstrate its feasibility, effectiveness, and efficiency. NNexus utilizes OWL and has a simple interface, which allows for an almost unlimited number of online corpora to interconnect for automatic linking.

In the rest of this paper, we first explain the model behind NNexus and present its key components and techniques (Section 2). Then, we briefly discuss the implementation and the interface to NNexus as a general, open source tool, and present evaluation and deployment results of NNexus (Section 3). Finally, we briefly review the related work (Section 4) and provide a summary and discussion of future works (Section 5).

2 NNexus Framework

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

2.1 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 or concept labels (*link source*) and they are ultimately turned into hyperlinks to the corresponding entries (*link target*) in the output rendering.

There are two basic steps in performing the invocation linking. When an article is submitted, NNexus starts *link source identification* 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) that invoke concepts defined in other entries are then used for *link target identification* to determine the entries to link to.

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. The tokens and token tuples (phrases) that are identified as link sources are searched to retrieve the candidate links using the concept map (see Section 2.2). After the candidate links are determined, they are filtered based on linking policies (see Section 2.4). The candidates are then compared by "classification proximity" and the object with the closest classification is then selected as the link target (see Section 2.3). The "winning" candidate for each position is then substituted into the original text and the linked document is then returned. Fig. 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 rescanned 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 2.5).

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 an 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 specify link policies for steering the automatic linker (see Section 2.4). While complemented and enhanced by the interactive learning components, NNexus is a completely automatic system, and we show in the experiment section later that NNexus performs well even without any human efforts.

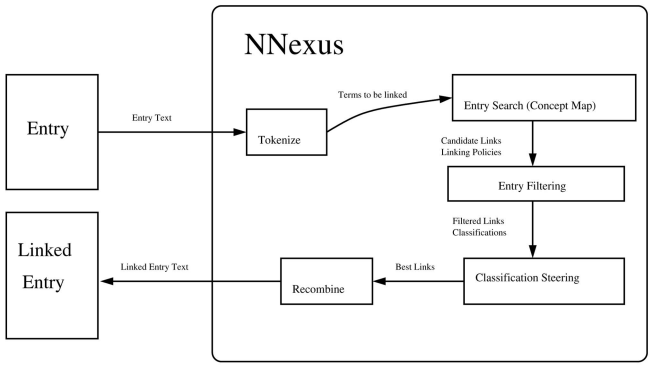


Fig. 2. Linking diagram: When an entry is linked through NNexus, the candidate links are found in the concept map. These candidates are then compared against the linking policies and sent through the classification module. The top candidate links are then recombined into the original text and returned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

2.2 Entry Search

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. Below, we present the details of how to build the concept map and how it is used for entry search.

When adding a new object (entry) to NNexus, a list of terms the object defines, synonyms, and a title are provided (the concept labels) by the author as metadata. 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 Fig. 3.

NNexus also performs *morphological transformations* on concept labels when building concept map in order to handle morphological invariances and 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 nonpossessive, with a canonicalized encoding.

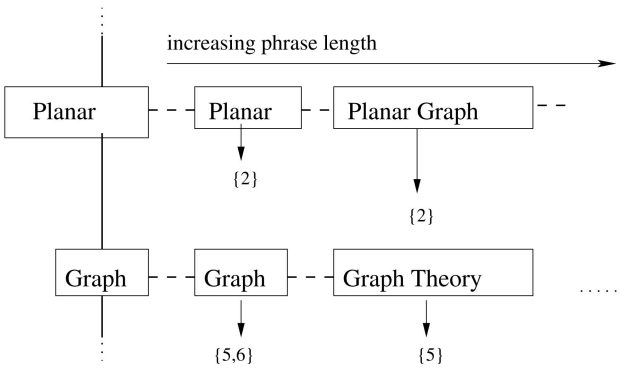


Fig. 3. Concept map: A fast-access (chained-hash-based) structure filled with all the concept labels for all included corpora, used for determining available linking targets as the text is being scanned. This figure contains a subset that would be generated based on our example corpus.

We now discuss how the concept map is used for entry search. When searching for candidate links for a given entry, the entry is represented as an array of word tokens (concept labels). The tokenized text of the entry is iterated over and searched in the concept map. If a word matches the first word of an indexed concept label in the concept map, 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. NNexus always performs the *longest phrase match*. For example, if an author 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. NNexus only links the first occurrence of a term or phrase to reduce visual clutter.

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. All possible link targets of the terms or phrases are added to the match array. The match array is then iterated over and the possible link targets are then disambiguated to determine the best link target for each term or phrase. Classification-based link steering is the main technique used in disambiguation and is discussed in the next section.

2.3 Classification Steering

As we discussed in Section 1.3, one of the main challenges of building an automatic linking system is to cope with possible mislinking errors. Online encyclopedias are typically organized into a classification hierarchy, and this ontological knowledge can be utilized to increase the precision of automatic linking by helping identifying the best link targets that are closely related to the link source in the ontological hierarchy. Below, we present our classification steering approach that is designed to reduce mislinking errors and to enhance link precision.

Each object in the NNexus corpus may contain one or more classifications. The classification table maps entries (by object ID) to lists of classifications that have been assigned to them by users. The classification hierarchy is represented as a tree. A subtree of the MSC hierarchy is shown as an example in Fig. 4. Each class is represented as a node in the tree. Edges represent parent/child relationships between the classes. In order to select the most relevant link target for a link source, NNexus compares the classes of the candidate link targets to the classes of the link source and selects the closest object with the shortest *distance* in the classification tree. Algorithm 1 presents a sketch of the classification steering algorithm.

Algorithm 1. Algorithm of classification steering: it returns the target objects that are closest in classification to the link source in the NNexus classification graph.

1: $sourceclasses \leftarrow$ list of classes of source object
2: $targetobjects \leftarrow$ list of candidate target objects
3: **for all** $object_i \in targetobjects$ **do**
4: $\quad targetclasses_i \leftarrow$ list of classes of $object_i$
5: $\quad distance_i \leftarrow$ minimum distance between all $sourceclasses$ and $targetclasses_i$ pairs
6: **end for**
7: **return** $\{object_i | distance_i = \min_i distance_i\}$

The key to the algorithm is how to compute the distance between two classes (nodes) in the classification tree. Note

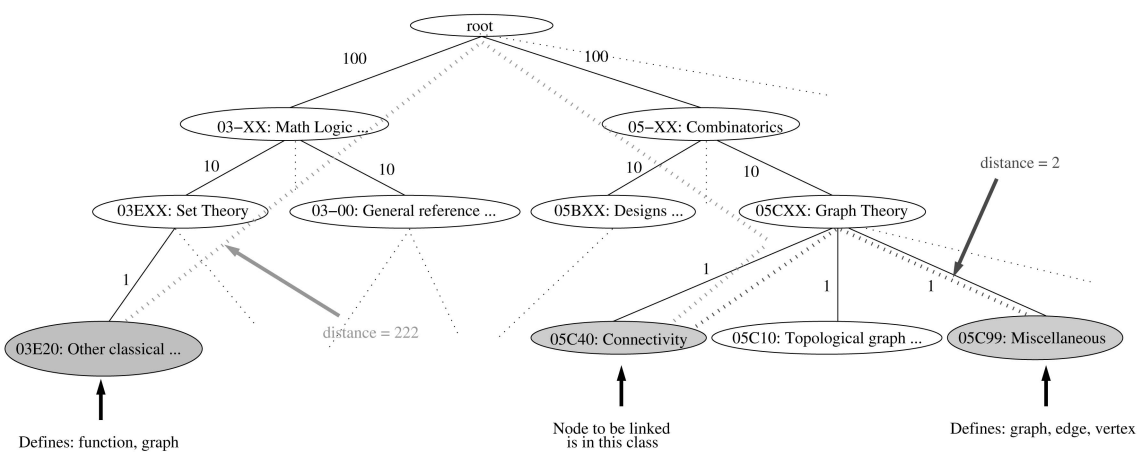


Fig. 4. Example classification tree: This is the MSC represented as a weighted graph. The shaded nodes indicate the classification of the source node (where "graph" is to be linked") and the classifications of the two target nodes. The weights are assigned with base 10.

that when there are multiple classes associated with the link source or link target, the minimum distance of all possible pairs of classes is used. We adopt two approaches, namely *nonweighted approach* and *weighted approach*, for computing the distance between two classes and discuss each them below.

In the *nonweighted approach*, the distance between two classes are simply the length of the shortest path between two classes. Intuitively, a node further away is less related to a given node in the tree. NNexus uses Johnson's All Pairs Shortest Path algorithm to compute the distances between all classes at startup.

In the *weighted approach*, each edge is assigned a weight. This is motivated by the observation that classes at the same level and in the same subtree should be considered closer than classes at a higher level in the same subtree and classes deeper in a subtree are more closely related than classes higher in the same subtree. For example, in Fig. 4, 05C10 (Connectivity) and 05C40 (Topological graph ...) are more closely related than the node 05CXX (Graph theory) and 05BXX (Designs ...). Based on this observation, we assign a weight to each edge that is inversely proportional to their depth in the tree. We define a 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. The distance is computed as the weighted shortest path between two nodes. Please note that when the base weight is 1, it becomes the nonweighted approach.

Fig. 4 also illustrates a scenario of the classification steering algorithm and the distance computation using our example in Fig. 1, the MSC of our source entry 05C40. The term to be linked, "graph," has two possible link targets: object 5 with classification 05C99 and object 6 with classification 03E20. We examine the weighted distance (with weight base 10) between the source class and the two target classes to determine which is a better link target. As the weighted distance from 05C99 to 05C40 is shorter in the weighted classification graph than 03E20, "graph" is linked to object 5.

It is worth mentioning that 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. To address the general problem of interlinking 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]

2.4 Entry Filtering

NNexus achieves perfect link recall without underlinking errors as every linkable terms will be linked in an entry. However, it is possible to have overlinking errors when a term that should not be linked (at all) is linked to an entry in the corpus (recall Section 1.3). For example, many articles will contain the word "even." In many cases, this is not used in mathematical context and should be forbidden from linking to the entry defining "even number."

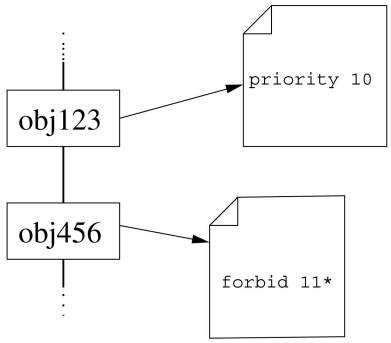


Fig. 5. The linking policy table stores a text chunk for each entry, containing optional user-supplied link-steering directives.

In order to combat this overlinking problem and those rare cases where the classification of target articles does not completely disambiguate the link targets, NNexus includes an interactive learning component, entry filtering by linking policies, that is designed to complement and further enhance the link precision by allowing users to specify linking policies. *Linking policies* are a set of directives controlling linking based on the subject classification system within the encyclopedia. The linking policy of an article describes, in terms of subject classes, where links may be made or prohibited. Thus, the entry for "even number" would forbid all articles from linking to the concept "even" unless they were in the number theory category. The author need only supply a linking policy for those terms that the article defines that are used commonly in language and are not meant in a mathematical sense.

For each object, there is stored text chunk representing the user-supplied linking policy. The linking policy table is keyed by object ID. A diagram of the table is shown in Fig. 5. The linking policies can be specified by the author but administrators also have the ability to modify the linking policies.

We note that the linking policy component requires minimal work from the users, and we will show in the experiment section later that by adding linking policies for a very small number (percentage) of entries, the precision for the overall corpus is enhanced significantly.

We are also exploring automatic keyword extraction techniques in order to extract those terms that should be or should not be linked in an automatic way. In addition, 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 to handle the overlinking problem in a more automatic way.

2.5 Invalidation

As an optimization technique to further enhance the efficiency and performance of the system, NNexus also includes an invalidation component. When a new object is added, NNexus utilizes an *invalidation index* to determine which articles may possibly link to the new object and need to be "invalidated" (marked for reprocessing before being

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

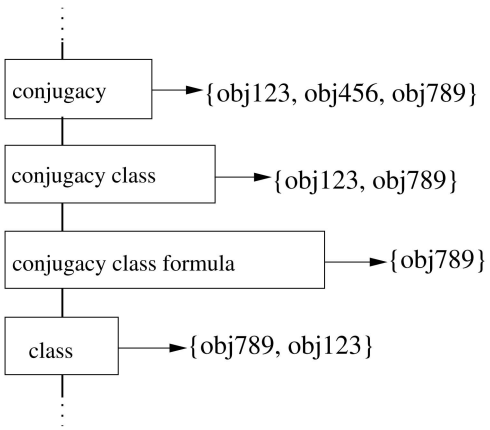


Fig. 6. Invalidation index: An adaptive inverted index containing both words and phrases, used for determining which text objects are likely to need to be reanalyzed for linking after concept definition updates have occurred to the corpus. The structure is a chained hash, with words and more common phrases 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. If a word-based inverted index were used, objects 123 and 456 would also need to be invalidated and checked for possible linking to conjugacy class formula.

displayed again). 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 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 Fig. 6. Since the falloff in occurrence count by phrase length in a typical collection follows a Zipf distribution, the invalidation index tends to be around twice the size of a simple word-based inverted index.

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 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 a minimal superset of entries 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 reexamined, 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 reanalyzed by the linker before being viewed.

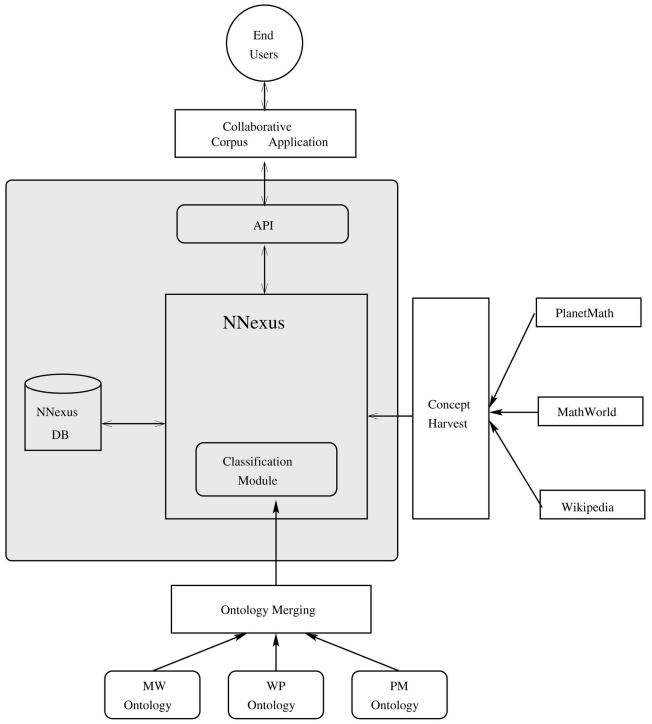


Fig. 7. NNexus system architecture (in an example deployment): The shaded region denotes NNexus proper. The classification module provides classification-invariant link steering between multiple ontologies.

3 Implementation and Experiments

The core methods of NNexus have essentially proven their large-scale applicability in the PlanetMath[6] site, a collaborative and dynamic mathematics encyclopedia in existence on the web for about seven years now. As of this writing, PlanetMath had more than 7,145 entries defining more than 12,171 concepts.

In this section, we briefly present some implementation features, and present a set of experimental results examining NNexus in terms of the linking quality, efficiency, scalability, and its deployability to a variety of settings.

3.1 Implementation

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.[7] Fig. 7 shows a diagram of the overall NNexus system architecture.

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. NNexus has XML configuration files that provide NNexus with information about supported domains, how to link to an entry in a specific domain, and

6. Visit PlanetMath on the web at http://www.planetmath.org. In fact, the central innovations of NNexus evolved from practical scalability challenges on PlanetMath; they did not occur in a vacuum!

7. NNexus is released under an MIT/X11 style license.

TABLE 1
Overlinking Statistics: Before and after Updating the Linking Policies for the Offending Entries of the Five Random Entries in a Random Subset of 20

Statistic	w/o Linking Policies	w/ Linking Policies
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 2
Automatic Linking Statistics for the Entire PlanetMath Corpus: Without Classification-Based Link Steering or Link Polices versus with Classification-Based Link Steering versus with Classification-Based Link Steering and 67 User-Supplied Linking Policies

Statistic	No Steering	Steering	Linking Policies
number of links	761	761	701
good links	630	672	646
mislinks	131	89	55
overlinks	69	69	36
% mislink	17.2	11.7	7.8
% overlink	9.1	9.1	5.1
Precision	82.8	88.3	92.2

The experiment was performed over the entire PlanetMath corpus but the statistics were estimated from a sample of 50 random entries in the corpus.

classification scheme information. All communications with NNexus are over socket connections, and all requests and responses with the NNexus server are in XML format.

3.2 Linking Quality

We define *recall* as the number of created (retrieved) links divided by the number of concepts invoked the entry that are actually defined in the corpus. We define *precision* as the number of *correct* links (those to the appropriate destination) divided by 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 polysemy and various other problems which will be discussed in more detail. This is why we introduced linking policies that utilize classification-based filtering (see Sections 2.3 and 2.4).

We performed a mislinking and overlinking study in June 2006 on the entire PlanetMath collection with and without linking policies. About 12 percent of links were mislinks and 7.9 percent of links were overlinks (thus, 61.1 percent 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 percent to 15 percent mislinks can be expected in a real-world corpus with only lexical matching and classification steering.

However, based on the overlinking aspect of these results, we hypothesized that using linking policies to take care of most of the overlinking "culprits" would yield an improvement of precision to 95 percent or more in most cases. This is because restricting linking to terms like "even" need only be done in one place a small number of times in order to prevent overlinking throughout the whole collection.

To begin to explore this assumption, we performed another small investigation to isolate the effect of linking policies on precision. We 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 percent mislinks and 11.5 percent overlinks (that is, about 86 percent of mislinks were due to overlinks) (see Table 1 for details). We then randomly selected five of these objects and fixed all of their overlinks by creating new link policies (added to eight problematic target objects). After eliminating all overlinks for these five objects, we resurveyed the initial 20 objects for linking quality. We found that the mislinking went down to 6.9 percent and the overlinking was reduced to 4.8 percent.

In order to verify our hypothesis in a larger scale, we performed another variation of this study on the live PlanetMath collection examining the precision with and without linking policies (as well as with and without classification steering). Note that these policies were supplied by real-world users with no prompting, and no effort was made to tackle the remaining problematic cases of overlinking. Nevertheless, the linking policies drove precision up to more than 92 percent[8] (see Table 2 for details).

We believe these results provide compelling support for our hypothesis that NNexus with classification-based link steering achieves good linking quality. Further, 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.

Comparison to related approaches. A survey in [11] shows that about 97 percent to 99 percent of Wikipedia links are accurate. However, this study is not directly comparable to our survey for a number of reasons. First, because it relies on the convention of "disambiguation nodes" (which NNexus allows one to avoid) and second, it does not take into account link recall (underlinking). In other words, links in Wikipedia tend to be accurate, but some of this "accuracy" is due to the presence of disambiguation nodes, and some is likely due to the fact that many links simply are not being made.

Most significantly, from a usability and productivity standpoint, no formal comparison of the *effort* required for link maintenance in the manual/semiautomatic versus automatic paradigms has been made. However, anecdotal evidence suggests our approach to linking is less work for authors and more appreciated by them; and with classification and linking steering, precision approaches that achieved on Wikipedia with manual effort and disambiguation nodes.

3.3 Scalability and Efficiency

To study the scalability and efficiency of our approach, we ran experiments on a modest Mac machine running OS X with a 1.83-GHz Intel Core Duo and 512-Mbyte DDR2

8. Likely, this number could exceed 95 percent with a little bit of targeted effort, and given that these policies have been available on PlanetMath for less than two years, the numbers will likely continue to improve on their own.

TABLE 3
Scalability Study: Running Linking on Random Subsets of Our Test Corpus of Gradually Increasing Size

Corpus Size	# of Links	Total Time (sec)	Time/Link
200	640	126	0.197
500	2067	290	0.140
1000	5837	617	0.106
2000	17757	1218	0.069
3000	35682	1972	0.055
4000	52030	2881	0.055
5000	79139	3737	0.047
6000	101787	4487	0.044
7132	127430	5599	0.044

SDRAM. We selected random subsets of sizes 200 to 7,132 from the PlanetMath corpus and kept track of the number of seconds to link every object in the subset corpora.

Table 3 and Fig. 8 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. All overhead quickly amortizes and diminishes relative to productive linking work done by the system, meaning that NNexus automatic linking is a legitimate feature to build into expanding collections and growing ensembles of interlinked collections on the web.

3.4 Deployability and Other Applications

In addition to enabling intralinking in a single encyclopedic knowledge base such as PlanetMath, NNexus also provides a 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 PlanetMath and MathWorld, but potentially extending to others, such as Wikipedia, the Digital Library of Mathematical Functions (DLMF), and more). Fig. 9 demonstrates this sort of use, showing screenshots of automatically linked notes from a probabilities course taught by Professor Jim Pitman at UC Berkeley—before and after automatic linking with NNexus (the links in this example are to both PlanetMath and MathWorld).

Because of the ease-of-use and success of linking lecture notes, we are confident that we can extend NNexus to other applications with diminishing additional effort. We are especially 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 to 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 impact on the web, and are being embraced by large-scale efforts such as the NSDL.[9]

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.

9. For their "Expert Voices" service, see http://www.nsdl.org/.

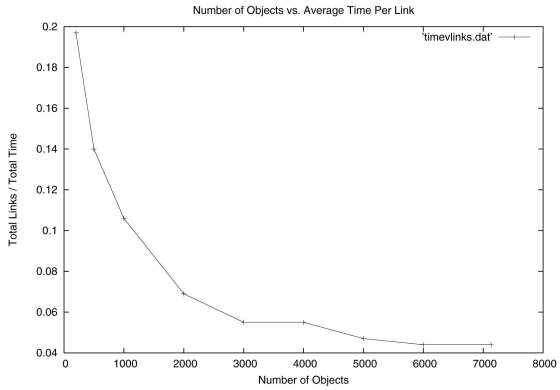


Fig. 8. Scalability study: Time-per-link for progressively larger corpora, showing clearly that the automatic linking process is sublinear in time complexity.

4 Related Work

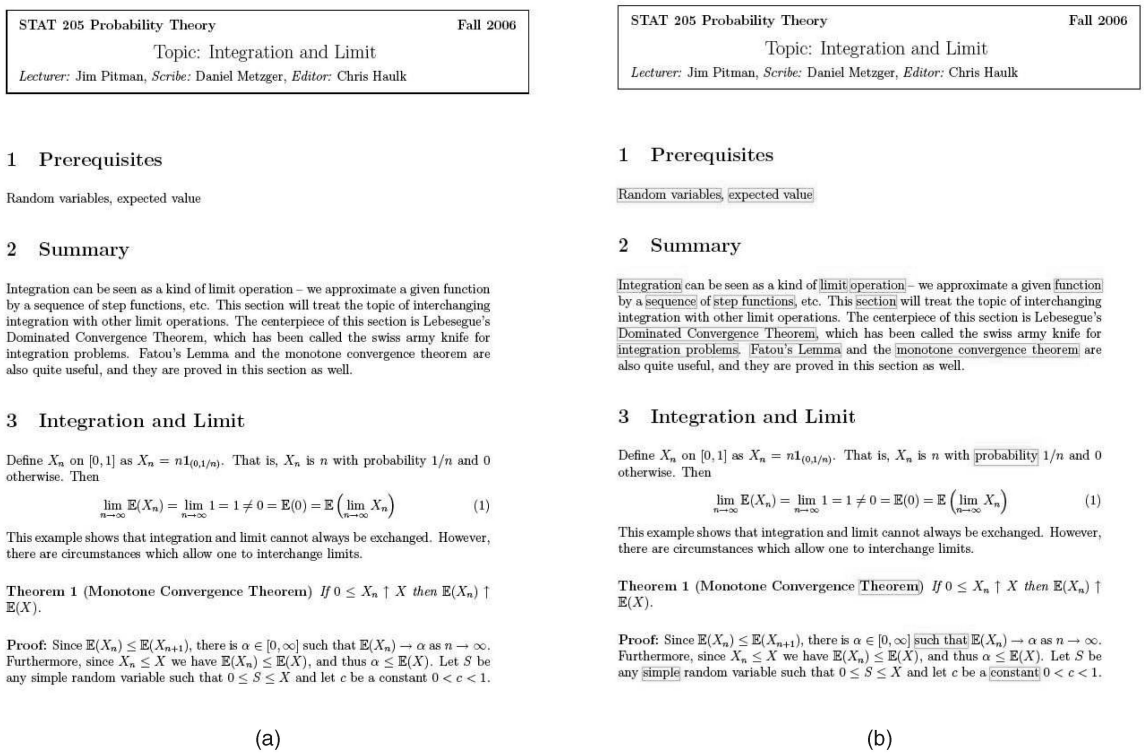
Our work is informed and inspired by a number of areas. We briefly discuss them below.

The semantic linking problem we studied in this paper bears similarities to the search problem on the web. However, as we discussed earlier in the paper, in our problem not only the link target but also the link source need to be identified and linked automatically. There are many standard methods for improving searching quality in IR 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.

Another related area and technology that might be applicable is the recommender systems [1]. We can model our problem as an entry-entry link matrix where each cell represents a link or nonlink from a certain entry to another entry and use entry similarities to help determine the best entry to link to for a term that belongs to a certain entry. While this approach is more appropriate for relevance linking and may help to narrow down the potential link targets, it alone is not sufficient for the invocation or concept linking problem. It is on our research agenda to enhance our current link ranking strategy by adapting the collaborative filtering technologies to enhance the linking precision by incorporating entry similarities and user feedback into the linking process.

There are several efforts [9], [8], [10] toward 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 the most relevant markup within a wiki, and are interpreted as semantic relations between two concepts described within articles. Völkel et al. [10] provide an extension to be integrated in Wikipedia, which allows users to specify typed links in addition to regular links between articles and typed data inside the articles. It would



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 $0 < c < 1$.

(a)

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 $0 < c < 1$.

(b)

Fig. 9. Screenshot of (a) original and (b) automatically linked 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 collection priority configuration option determined the outcome. Concepts were "imported" from MathWorld using that site's OAI repository.

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. Stojanovic et al. [12] discuss the differences between classical training and e-learning, and presents different Semantic Web layers and how they can be applied to e-learning. Farrel et al. [13] define "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 this paper are the Dublin Core Schema[10] and LOM.[11] 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, Cardinaels et al. [4] present a framework that automatically generates learning object metadata. This can be 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 metainformation).

5 Conclusion

We have presented NNexus, an automatic linking system for providing invocation linking capabilities for online collaborative encyclopedia. We outlined the design goals for any automatic linking system should strive to achieve. We presented the key components of NNexus including the concept map and classification-based link steering as well as entry filtering by link policies. We presented a set of experiments demonstrating the effectiveness and efficiency of our approach. The achievements of the precursor to the NNexus system, the Noosphere automatic linker, can be seen at PlanetMath.[12] NNexus is now available for general use as open source software[13] 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.

Our work in NNexus continues along several threads. First, we are exploring automatic keyword extraction techniques to better extract concept labels to be linked. Second, we are exploring reputation systems and collaborative filtering techniques [1] to further enhance the link steering by addressing issues of "competing" entries and different needs and preferences of authors. This especially becomes an issue when one goes beyond a single

10. http://dublincore.org/.
11. http://www.imsglobal.org/metadata/.
12. http://planetmath.org/.
13. http://aux.planetmath.org/nnexus/.

collaborative corpus, as would typically be the case in linking to them by third parties. Finally, a major research and development item is the generalization of NNexus for interlinking of multiple corpus across domains with expansion of ontology mapping capabilities.

ACKNOWLEDGMENTS

The authors would like to thank the editors of the special issue and the anonymous reviewers for their valuable comments that improved this paper. 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 Trans. Knowledge and Data Eng.*, vol. 17, no. 6, June 2005.
[2] T. Andreasen, J.F. Nilsson, and H.E. Thomsen, "Ontology-Based Querying," *Flexible Query-Answering Systems*, pp. 15-26, 2000.
[3] R.A. Baeza-Yates and B.A. Ribeiro-Neto, *Modern Information Retrieval*. ACM Press/Addison-Wesley, 1999.
[4] K. Cardinaels, M. Meire, and E. Duval, "Automating Metadata Generation: The Simple Indexing Interface," *Proc. 14th Int'l Conf. World Wide Web (WWW '05)*, pp. 548-556, 2005.
[5] Y. Kalfoglou and M. Schorlemmer, "Ontology Mapping: The State of the Art," *Semantic Interoperability and Integration*, Y. Kalfoglou, M. Schorlemmer, A. Sheth, S. Staab, and M. Uschold, eds., number 04391 in Dagstuhl Seminar Proc. Internationales Begegnungsund Forschungszentrum fuer Informatik (IBFI), Schloss Dagstuhl, 2005.
[6] J. Kolbitsch and H. Maurer, "Community Building around Encyclopeadic Knowledge," *J. Computing and Information Technology*, vol. 14, 2006.
[7] A. Krowne, "An Architecture for Collaborative Math and Science Digital Libraries," master's thesis, Virginia Polytechnic Inst. and State Univ., 2003.
[8] A. Souzis, "Building a Semantic Wiki," *IEEE Intelligent Systems*, vol. 20, no. 5, pp. 87-91, Sept./Oct. 2005.
[9] S.E. Roberto Tazzoli and P. Castagna, "Towards a Semantic Wiki Wiki Web," *Proc. Third Int'l Semantic Web Conf. (ISWC)*, demo session, 2004.
[10] M. Völkel, M. Krötzsch, D. Vrandecic, H. Haller, and R. Studer, "Semantic Wikipedia," *Proc. 15th Int'l Conf. World Wide Web (WWW '06)*, pp. 585-594, 2006.
[11] G. Weaver, B. Strickland, and G. Crane, "Quantifying the Accuracy of Relational Statements in Wikipedia: A Methodology," *Proc. Sixth ACM/IEEE-CS Joint Conf. Digital Libraries (JCDL)*, 2006.
[12] L. Stojanovic, S. Staab, and R. Studer, "eLearning Based on the Semantic Web," *Proc. World Conf. WWW and Internet (WebNet)*, 2001.
[13] R. Farrel, S. Liburd, and J. Thomas, "Dynamic Assembly of Learning Objects," *Proc. 13th Int'l Conf. World Wide Web (WWW)*, 2004.
[14] N. Fridman Noy and M.A. Musen, "PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment," *Proc. 17th Nat'l Conf. Artificial Intelligence and 12th Conf. Innovative Applications of Artificial Intelligence*, 2000.
[15] Z. Aleksovski and M. Klein, "Ontology Mapping Using Background Knowledge," *Proc. Third Int'l Conf. Knowledge Capture (K-CAP)*, 2005.
[16] J. Brase and W. Nejdl, *Ontologies and Metadata for eLearning*. Springer Verlag, 2003.
[17] K. Oyama and H. Gotoda, "Dublin Core Conference," *Proc. Int'l Conf. Dublin Core and Metadata Applications (DC)*, 2001.
[18] M. Nilsson, M. Palmer, and J. Brase, *The LOM RDF Binding—Principles and Implementation*, 2003.

James Gardner received the BS degree in mathematics and computer science from East Tennessee State University and the MS degree in computer science from Emory University, Atlanta, where he is currently working toward the PhD degree in the Department of Mathematics and Computer Science. His current research interests include machine learning, natural language processing, and Semantic Web technologies.

Aaron Krowne received the BS degree in math and the MS degree in computer science from Virginia Tech. He is the president of PlanetMath.org, Ltd., and the CEO of IEHI, Inc. and Krowne Concepts, Inc. His research interests include the intersection of technology, community, and media. His best-known web site projects are http://planetmath.org (math) and http://ml-implode.com (housing economics). From 2001 to mid-2003, he was a research assistant at Virginia Tech's Digital Library Research Lab with advisor Edward A. Fox. There he worked chiefly on the CITIDEL project (part of the National Science Digital Library) and PlanetMath. From late 2003 to 2007, he was the head of the Digital Library Research at the Woodruff Library, Emory University, Atlanta, where he led the research on a number of digital library grant projects. Notably, these included the MetaCombine project (Mellon), the OCKHAM project (NSD), the Quality Metrics project (IMLS), and the Cyberinfrastructure for Scholars project (Mellon).

Li Xiong received the MS degree in computer science from Johns Hopkins University and the PhD degree in computer science from Georgia Institute of Technology. She also worked as a software engineer in IT industry for several years prior to pursuing her doctorate. She is an assistant professor of mathematics and computer science in the Department of Mathematics and Computer Science, Emory University, Atlanta. Her general interests include data and information management, distributed systems, trust, and information privacy. Her current research is focused on developing algorithms and techniques that facilitate large-scale and secure information sharing.