Lexical chains as representations of context for the detection and correction of malapropisms

Graeme Hirst and David St-Onge

Department of Computer Science
University of Toronto
Toronto, Ontario
Canada M5S 1A4

416-978-8747; fax 416-978-1455
gh@cs.toronto.edu

26 April 1995

Submitted to:
WordNet, edited by Christiane Fellbaum, Cambridge, MA: The MIT Press, 1995.

1 Introduction

Natural language utterances are, in general, highly ambiguous, and a unique interpretation can usually be determined only by taking into account the constraining influence of the context in which the utterance occurred. Much of the research in natural language understanding in the last twenty years can be thought of as attempts to characterize and represent context and then derive interpretations that fit best with that context. Typically, this research was heavy with AI, taking context to be nothing less than a complete conceptual understanding of the preceding utterances. This was reasonable, as such an understanding of a text was often the main task anyway. However, there are many text-processing tasks that require only a partial understanding of the text, and hence a 'lighter' representation of context is sufficient. In this paper, we examine the idea of **lexical chains** as such a representation. We show how they can be constructed by means of WordNet, and how they can be applied in one particular linguistic task: the detection and correction of **malapropisms**.

A malapropism is the confounding of an intended word with another word of similar sound or similar spelling that has a quite different and malapropos meaning, *e.g.*, *an ingenuous* [for *ingenious*] *machine for peeling oranges*. In this example, there is a one-letter difference between the malapropism and the correct word. Ignorance, or a simple typing mistake, might cause such errors. However, since *ingenuous* is a correctly spelled word, traditional spelling checkers cannot detect this kind of mistake. In section 4, we will propose an algorithm for detecting and correcting malapropisms that is based on the construction of lexical chains.

2 Lexical chains

If a text is cohesive and coherent, successive sentences are likely to refer to concepts that were previously mentioned and to other concepts that are related to them. Halliday and Hasan (1976) suggested that the words of the text that make such references can be thought of as forming **cohesive chains** in the text. Each word in the chain is related to its predecessors by a particular **cohesive relation** such as identity of reference. For example, in (1), the italicized words form a chain with this relation.

(1) The major potential complication of total joint replacement is *infection*. *It* may occur just in the area of the wound or deep around the prosthesis. *It* may occur during the hospital stay or after the patient goes home. … *Infections* in the wound area are generally treated with antibiotics.

But the relationship need not be identity; we also find cohesive chains of words whose meanings (in the text) are related to one another in more-general ways such as hyponymy or meronymy or even just general association of ideas. Example (2) shows chains in which the relationship is hyponymy (an infection is a kind of complication) and (3) shows a chain in which the relation is general association.

(2) The major potential *complication* of total joint replacement is *infection*.

(3) The evening prior to admission, take a *shower* or *bath*, *scrubbing* yourself well. Rinse off all the *soap*.

Morris and Hirst (1991; Morris 1988) suggested that the discourse structure of a text may be determined by finding **lexical chains** in the text, where a lexical chain is, in essence, a cohesive chain in which the criterion for inclusion of a word is that it bear some kind of cohesive relationship (not necessarily one specific relationship) to a word that is already in the chain. To make this idea precise, it is necessary to specify exactly what counts as a cohesive relationship between words—and in particular, what counts as 'general association of ideas'. Morris and Hirst's suggestion was that a thesaurus, such as *Roget's* (*e.g.,* Chapman 1992), could be used to define this. Two words could be considered to be related if they are 'connected' in the thesaurus in one (or more) of five possible ways:

1. Their index entries point to the same thesaurus category, or point to adjacent categories.
2. The index entry of one contains the other.
3. The index entry of one points to a thesaurus category that contains the other.
4. The index entry of one points to a thesaurus category that in turn contains a pointer to a category pointed to by the index entry of the other.

5. The index entries of each point to thesaurus categories that in turn contain a pointer to the same category.

Morris and Hirst showed that the distribution through a text of lexical chains defined in this manner was indicative of the intentional structure of the text, in the sense of Grosz and Sidner (1986). They also suggested that lexical chains often provided enough context to resolve lexical ambiguities, an idea subsequently developed by Okumura and Honda (1994).

Unfortunately, however, Morris and Hirst were never able to implement their algorithm for finding lexical chains with *Roget's* because no on-line copy of the thesaurus was available to them.[1] However, the subsequent development of WordNet raises the possibility that, with a suitable modification of the algorithm, it could be used in place of *Roget's*.

3 WordNet as a knowledge source for a lexical chainer

3.1 Relations between words

Because the structure of WordNet is quite different from that of *Roget's Thesaurus*, if we are to replace *Roget's* with WordNet in Morris and Hirst's algorithm, we must replace their Roget-based definition of semantic relatedness with one based on WordNet, while retaining the algorithm's essential properties.

Our new definition centers upon the synset. In WordNet, a word may have more than one synset, each corresponding to a different sense of the word. When we look for a relation between two different words, we consider the synsets of all the senses of each word that have not been ruled out, looking for a possible connection between some sense of the first word and some sense of the second. Three kinds of relation are defined: extra-strong, strong, and medium-strong.

An **extra-strong** relation holds only between a word and its literal repetition; such relations have the highest **weight** of all relations. There are three kinds of **strong** relations, illustrated in Figure 1. The first occurs when there is a synset common to two words. The second occurs when there is a

[1]Recent editions of *Roget's* could not be licensed. The on-line version of 1911 edition was available, but it does not include the index that is crucial to the algorithm. Moreover, because of its age, it lacks much of the vocabulary that is necessary for processing many contemporary texts, especially newspaper and magazine articles and technical papers. Stairmand (1994) tried to implement a lexical chainer with this edition nonetheless, but concluded that it was not possible.

horizontal link (e.g., antonymy, similarity, see also) between a synset of each word. The third occurs when there is any kind of link at all between a synset of each word if one word is a compound word or a phrase that includes the other. A strong relation has a lower weight than an extra-strong relation and a higher weight than a medium-strong relation.

A **medium-strong** relation between two words occurs when a member of a set of allowable paths connects a synset of each word. A path is a sequence of between two and five links between synsets, and is allowable if it corresponds to one of the patterns shown in Figure 2(a) (where each vector represents a sequence of one or more links in the same direction). Paths whose patterns are not allowed are shown in Figure 2(b). Figure 3 shows an example of a medium-strong relation between two words. Unlike extra-strong and strong relations, medium-strong relations have different weights. The weight of a path is given by

$$\textit{weight} = C - \textit{path length} - k * \textit{number of changes of direction}$$

(where *C* and *k* are constants). Thus, the longer the path and the more changes of direction, the lower the weight.

The rationale for the allowable patterns of Figure 2 is as follows: If a multilink path between two synsets is to be indicative of some reasonable semantic proximity, the semantics of each lexical relation must be taken into consideration. Now, an upward direction corresponds to generalization. For instance, an upward link from {*apple*}to {*fruit*}means that {*fruit*}is a semantically more general synset than {*apple*}. Similarly, a downward link corresponds to specialization. Horizontal links are less frequent than upward and downward links; a synset rarely has more than one. Such links are usually very specific of meaning. (In Figure 1(b), the horizontal link between {*successor*}and {*predecessor, precursor, antecedent*}is a very accurate specification of the meaning of the word *successor*.) So, to ensure that a path corresponds to a reasonable relation between the source and the target word, two rules have been defined as to which patterns are allowable:

(R1) No other direction may precede an upward link.

Once a link that narrows down the context (downward or horizontal) has been used, it is not permitted

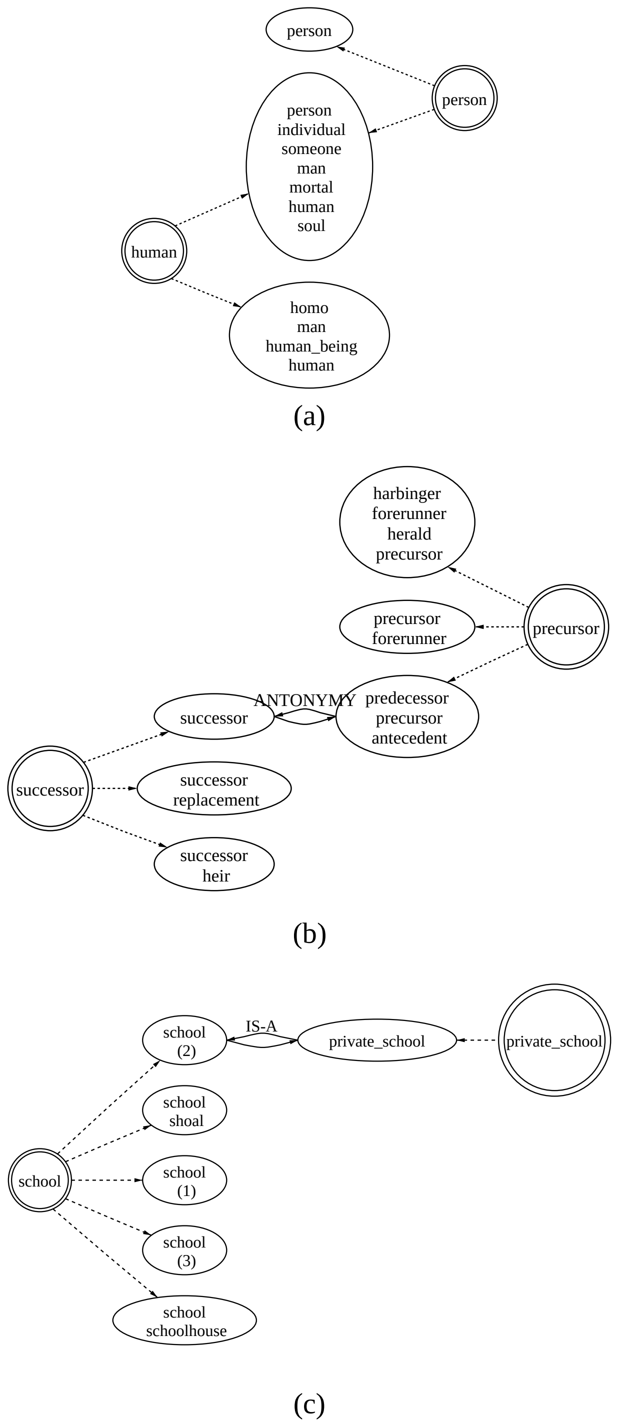
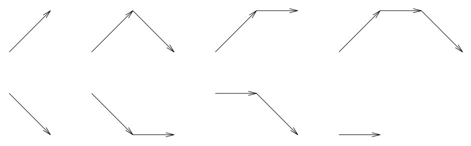


Figure 1: Examples of the three kinds of strong relation between words: (a) a synset in common; (b) a horizontal link between two synsets; (c) any link between two synsets if one word is a compound word or phrase that includes the other word. (Double circles indicate words and ellipses indicate synsets.)


(a)

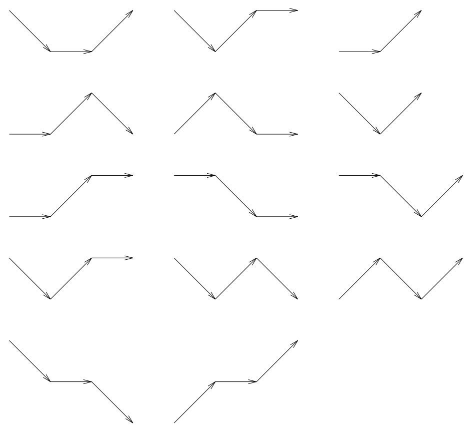

(b)

Figure 2: (a) Patterns of paths allowable in medium-strong relations and (b) patterns of paths not allowable. (Each vector denotes one or more links in the same direction.)

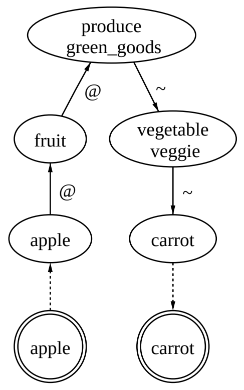


Figure 3: Example of a regular relation between two words. (@ = hypernymy, $\sim$ = hyponymy)

to enlarge the context again by using an upward link.

(R2) At most one change of direction is allowed.

Changes of direction constitute large semantic steps. Therefore, they must be limited. However, this second rule has the following exception:

(R2$'$) It is permitted to use a horizontal link to make a transition from an upward to a downward direction.

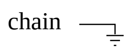
Horizontal links correspond to small semantic distance for words such as *heat* and *hot*, which are linked by an attribute relation. In this case, this exception to the second rule enables connections between subordinates of *heat* and subordinates of *hot*. Thus, it has been assumed that enabling such a connection between two superordinates does not imply too large a semantic step.

3.2 Creating and managing chains

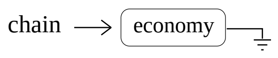
Although a lexical chain may be thought of as a set of words, its internal structure is more complex. Figure 4 gives an example of the construction of a chain. First, an empty chain is created (see Figure 4 (i)). Then, a chain word record is allocated, initialized with the word *economy*, and inserted into the new chain (Figure 4 (ii)). Next, to insert *sectors*, another word record is constructed and inserted into the chain. The kind of relation (extra-strong, strong, or medium-strong) between the new word and its related word in the chain is also stored in the word record. In Figure 4 (iii), *sectors* precedes *economy* in the chain and another form of connection, which will be described soon, illustrates its relation with *economy*. In Figure 4 (iv), *economic_system* is inserted into the chain, not from a relation with *sectors*, its immediate successor in the chain, but from a relation with *economy*. Thus, the word order in a chain does not necessarily correspond to relations between words, but only to the insertion order.

Because a word may have more than one synset, when a new word record is constructed, a list of pointers to every synset of the word is created and attached to it. When a word starts a new chain, all its synsets are kept, since, at this point, no contextual information is available to discriminate among them (see Figure 5). Inserting another word into the chain results in a connection between the words by linking the synsets involved in the relation. When a word is inserted into a chain because

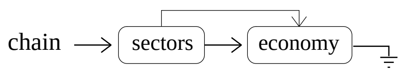
(i) {}



(ii) {economy}



(iii) {sectors, economy}



(iv) {economic_system, sectors, economy}

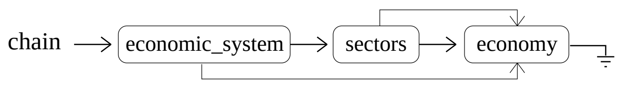


Figure 4: Chain management

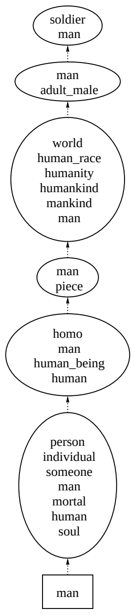


Figure 5: A word starting a new chain. (The word *man* has six synsets.)

of an extra-strong relation, all corresponding synsets are connected (see Figure 6). When the relation involved is strong, all pairs of strongly related synsets are connected (see Figure 7). And when the relation involved is medium-strong, the pair (or pairs) of synsets whose weight is of the greatest (or equal greatest) weight are connected.

After the connection between words is made, any unconnected synsets of the new word are deleted and the chain is scanned to remove other synsets wherever possible. Synsets that are not involved in the current word connection are removed. Removing synsets while inserting words in a chain progressively disambiguates each word of the chain by removing uninvolved interpretations, thereby narrowing down the context. This idea comes from Hirst's Polaroid Words (1987). Figure 8 illustrates the word-sense disambiguation process resulting from the situation illustrated in Figure 7.

When *economic_system* in Figure 4(iv) is added to the chain, not only are the synset lists of *economic_system* and *economy* updated but also the synset list of *sectors*. This is possible because the record for *economy* contains a pointer to the chain word object for *sectors*. Thus, each time a word is pushed onto a chain, the whole chain is traversed by following the word connections to update the synset list of each word of the chain.

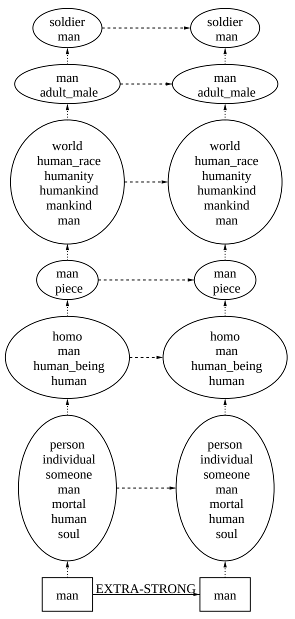


Figure 6: Push the same word

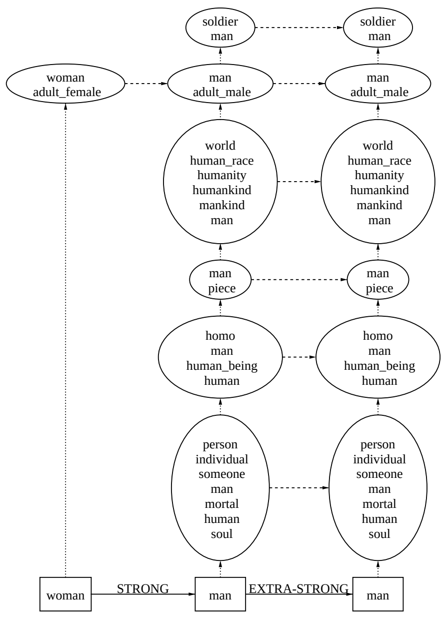


Figure 7: Push an antonym

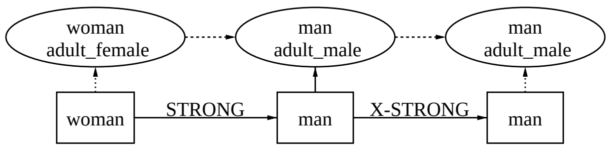


Figure 8: Updated chain after insertion

3.3 Identifying words and relations

Because the verb file of WordNet has no relation with the three other files and the adverb file has only unidirectional relations with the adjective file, we were forced to limit the chaining process to nouns in the present version of our software. (We hope that future versions of WordNet will remove this limitation.) However, we have not used any grammatical parsing in our program, because of the slowdown and the error that would have resulted. Instead, we decided to consider as a noun any word that could be found in the noun index as is or could be morphologically transformed to such a word. This is based on the assumption that most words in other grammatical categories that have a nominal form are semantically close to that form (e.g., *to walk* and *a walk*), and our experimentation has showed that this assumption was true. However, an attempt to morphologically transform unidentified words as if they were verbs before searching for them in the noun index introduced too much inaccuracy.

Wherever possible, we try to identify in the input text any compound words and phrases that are included in WordNet, because they are much better indicators of the meaning of the text than the words that compose them, taken separately. For instance, *private school*, which is listed in the noun index as *private_school*, is more indicative than *private* and *school* taken separately. During this phrase identification process, each word must also pass a validity test to ensure its suitability for lexical chaining: as explained above, it must be a WordNet noun or transformable to a noun, and it must not appear in the stop-word list. The stop-word list contains closed-class words and many vague high-frequency words that tend to weaken chains by having little content (e.g., *one, two, dozen, little, relative, right*).

If a word is potentially chainable, an extra-strong relation is sought throughout all chains, and if one is found, the word is added to the appropriate chain. If not, strong relations are sought, but for these, the search scope is limited to the words in any chain that is no more than seven sentences back in the text; the search ends as soon as a strong relation is found. Finally, if no relationship has yet been found, medium-strong relations are sought, with the search scope limited to words in chains that are no more than three sentences back. Since the weight of medium-strong connections varies, all medium-strong connections within the search scope must be found, in order to retain the one with the highest weight.[2] If no relation can be found at all, a new chain is created for the word.

A formal algorithmic specification of chaining and details of the software implementation of the chainer—the LexC program—are given by St-Onge (1995).

3.4 Testing the lexical chainer

Testing the lexical chainer is difficult, as what counts as a reasonable chain depends upon linguistic intuition, and what counts as a useful chain depends upon the particular application to which it is being put, where one can see whether it does or doesn't serve the task. Consequently, much of our evaluation is postponed to section 4.3.2, in which we describe our results in real-word spelling correction. Here we just briefly outline some of our observations from trying the chainer out on various texts, examining the chains that it builds, and seeing how they accord with intuition. We found that many chains did indeed match our expectations, and many words were correctly disambiguated. More details and many examples are given by St-Onge (1995).

Two kinds of disappointment were possible: words not included in chains in which they clearly belonged, and words included in chains in which they did not belong. These problems arise from several sources:

1. Limitations in the set of relations in WordNet, or a missing connection.
2. Inconsistency in the semantic proximity implicit in links in WordNet.
3. Incorrect or incomplete disambiguation.
4. Non-literal uses (*e.g.,* metaphors).

[2]The searches for both extra-strong and strong relations are very fast processes. However, that for medium-strong relations is the most expensive operation of the whole lexical chaining process in terms of CPU time.

The following sentence gives an example of the first problem.

(4) School administrators say these same taxpayers expect the *schools* to provide *child care* and *school* lunches, to integrate immigrants into the community, to offer special classes for adult students, …

Here, one would want *child care* to be connected to *school*. However, WordNet does not have a sufficient set of relations to relate these two words. In fact, relations such as antonymy, holonymy, or meronymy are not appropriate to link *child care* to *school*. Rather, the relation that exists between these two words is a **situational relation**. Many such relations are hard to classify (*e.g.*, the situational relation between *physician* and *hospital*), and it was a strength of Morris and Hirst's *Roget*-based algorithm that it was able to make such connections nonetheless.

The following sentence illustrates the second problem:

(5) The cost means no holiday trips and more *stew* than *steak*, but she is satisfied that her children, now in grades 3 and 4, are being properly taught.

Here, *stew* and *steak* are obviously somehow related. However, these two words were not linked to each other. Here is their mutual relation in WordNet:

(6) *stew* IS A *dish* IS A *aliment* INCLUDES *meat* INCLUDES *cut / cut of meat* INCLUDES *piece / slice* INCLUDES *steak*

The inter-synset distance between *stew* and *steak* is six synsets, which is greater than the limit that is set in the lexical chainer. In general, the greater the distance limit, the greater the number of weak connections. However, links in WordNet do not all reflect the same **semantic distance**. In other words, there are situations, as with *stew* and *steak*, where words have an obvious semantic proximity but are distant in WordNet.

There are also situations were words are close to each other in WordNet while being quite distant semantically. This introduces the problem of over-chaining. For example, in one chain, we found *public* linked to *professionals* by the following relationship:

(7) *public* IS A *people* HAS MEMBER *person* INCLUDES *adult* INCLUDES *professional*

The problem of incorrect or incomplete disambiguation often follows from under- and over-chaining, as the following example demonstrates:

(8) We suppose a very long *train* traveling along the *rails* with the constant *velocity v* and in the direction indicated …

Here, a chain is created with the word *train*, which has six senses in WordNet. But the word *rails* is not associated with it, the distance between these two words being too great in WordNet. The word *velocity* is then connected to *train* with the undesired effect of wrongly disambiguating it, as the following sense is selected:

(9) *sequence, succession, sequel, <u>train</u>*—events that are ordered in time.

4 Automatically detecting malapropisms

We now propose an algorithm for detecting and correcting malapropisms that is based on the construction of lexical chains.

4.1 Spelling checkers

Traditional spelling checkers can detect only non-word errors. The main two techniques that they use are lexicon lookup and *n*-gram analysis. In the former case, each word of the input text is sought in a lexicon and considered to be an error if not found. The size of the lexicon is an important issue: an insufficient lexicon will result in too many false rejections while a too-large lexicon (with rare or unusual words) will result in too many false acceptances (Peterson 1986). False rejections might also be caused by the use of a lexicon that is not adapted to a specific area of application. The second detection method, called *n*-gram analysis, is based on the probability of a given sequence of letters of length *n*, usually two (digram) or three (trigram). Under a certain threshold, an error is signaled. An attempt may then be made to find candidates for the word that was intended. Techniques for doing this typically generate strings that are similar to the erroneous word, by means of transformations such as adding, deleting, or transposing characters, and then filtering out non-words by checking the lexicon. (See Kukich 1992 for a survey of techniques.)

Real-word errors are much more difficult to detect than non-word errors and even more difficult to correct. Kukich (1992) classifies real-word errors into four categories:

1. syntactic errors (*e.g., The students are doing <u>there</u> homework*);
2. semantic errors or malapropisms (*e.g., He spent his summer travelling around the <u>word</u>*);
3. structural errors (*e.g., I need <u>three</u> ingredients: red wine, sugar, cinnamon, and cloves*);
4. pragmatic errors (*e.g., He studies at the University of Toronto in <u>England</u> and she studies at Cambridge*).

Errors that belong to the first category (which are, strictly speaking, also malapropisms) can be detected by using a natural-language parser or by performing a word-level *n*-gram analysis to detect words that have a low probability of succession. The same tools could be used to suggest replacements. However, errors that belong to the other three categories are much harder to detect and even harder to correct.

Few studies have been made on the frequency of real-word errors. Mitton (1987) studied 925 essays written by high-school students and found that 40 percent of all errors were real-word errors. He noticed that most of these real-word errors belonged to the first category. Atwell and Elliot (1987) with the University of Lancaster Unit for Computer Research on the English Language (UCREL) (Garside, Leech, and Sampson 1987) analyzed three different kinds of text that had not been automatically proofread previously: published texts, 11- and 12-year-old students' essays, and text written by non-native English speakers. They found that the corresponding amount of real-word errors were 48%, 64%, and 96% respectively. Among the real-word errors, 25%, 16%, and 38% respectively belonged to the semantic category.

4.2 An algorithm for detecting probable malapropisms

As discussed above, each discourse unit of a text tends to use related words. Our hypothesis is that the more distant a word is semantically from all the other words of a text, the higher the probability is that it is a malapropism. But lexical chains can be thought of as sets of words that are semantically close. Hence a word in a text that cannot be fitted into a current lexical chain, but which is close in spelling to a word that **could** be fitted, is likely to be a malapropism. More formally, a spelling checker can detect likely malapropisms, and suggest corrections, by the following method:

> Assume that (as in all but the simplest spelling checkers) there is already in place in the program a mechanism that, given a character string w, can produce a set $P(w)$ of all words in the program's lexicon for which w is a plausible mistyping.
>
> 1. The program first looks for non-word errors in the text, and solicits corrections from the user (or chooses a correction automatically).
> 2. The program next constructs lexical chains between the high-content words in the text. (Stop-words are not considered; the erroneous occurrence of these words cannot be detected by this method.)
> 3. The program then hypothesizes that a word w is in error, even though it is a correctly spelled word, if w is not a member of any lexical chain, but there is a word $w' \in P(w)$ that would be in a lexical chain had it appeared in the text instead of w. It is then likely enough that w' was intended where w appears that the user should be alerted to the possibility of the error.

We have implemented this algorithm with the lexical chainer described in section 3 above. (The implementation covers only the detection of malapropisms in steps 2–3 above; it does not check for non-word spelling errors.) Any **atomic chain**—one that contains only one word[3]—is extracted and considered to be a potential malapropism.[4] A set of possible corrections is then sought for each potential malapropism, using the spelling correction procedure of a spelling checker. For each possible correction, an attempt is made to find a relation with a word that is in one of the lexical chains. This chaining process is done with the normal chaining mechanism, except that the word-chain search scope is both backward and forward and is limited to the same word-chain distance in both directions. All

[3]Since a **chain** is, by definition, a series, the term **atomic chain** may seem awkward. However, it is a convenient way to represent a word that, although potentially chainable, has not been related to any other words in the text.

[4]Atomic chains that contain a compound word (*e.g., black-and-white*) or a phrase (*e.g., elementary_school*) in WordNet are not considered to be potential malapropisms. The probability that two words together form a known compound and yet are malapropos is extremely low; indeed, such compounds can be thought of as a special kind of chain in and of themselves.

possible corrections that have a relation with a word in a chain are retained. If a potential malapropism has a chainable possible correction, an alarm is raised, suggesting to the user the possibility that the potential malapropism is an error and that one of the chainable corrections was the word that was intended.

4.3 An experiment

It is difficult to test the algorithm on naturally occurring text, because large, on-line corpora that are available are mostly edited, published texts by professional writers, and hence may be assumed to contain an extremely small number of malapropisms. Ideally, we would test the algorithm on a large supply of the kinds of texts that are submitted to spelling checkers—unedited first drafts written by typical users of word processors, including students and others who are not professional writers—in which all the malapropisms have been identified by a human judge, so that the algorithm's performance may be compared to the human's. Such texts are not available, but we can simulate them by inserting deliberate malapropisms into a published text.

So, to test our algorithm, we took 500 articles on many different topics selected randomly from the *Wall Street Journal* from 1987 to 1989, replacing roughly each 200th word with a malapropism. We then ran our algorithm on the modified text, seeing what proportion of the malapropisms could be identified as such.

4.3.1 Creating the experimental text

To create malapropisms, we used the code that generates error replacement suggestions in *Ispell 1.123*, a spelling checker for non-word errors.[5] This code returns 'near misses'—words found in the lexicon that differ only slightly (usually by a single letter or a transposition) from the input string. By giving it a real word as input instead of a non-word error, it becomes, in effect, a malapropism generator. It tries the following transformations:

1. restore a missing letter, *e.g.*, *girder* $\rightarrow$ *girdler*;

[5]*Ispell* is a program that has evolved in PDP-10, Unix, and Usenet circles for more than 20 years, with contributions from many authors. Principal contributors to the current version include Pace Willisson and Geoff Kuenning.

2. delete an extra letter, *e.g., beast* $\rightarrow$ *best, lumpfish* $\rightarrow$ *lumpish*;
3. transpose a pair of adjacent letters, *e.g., elan* $\rightarrow$ *lean*;
4. replace one letter, *e.g., recuse* $\rightarrow$ *refuse*;
5. restore a missing space, *e.g., weeknight* $\rightarrow$ *wee knight, Superbowl* $\rightarrow$ *Superb owl;*
6. restore a missing hyphen, *e.g., relay* $\rightarrow$ *re-lay*.

A string transformed in this way is accepted as a malapropism if it meets three conditions: it must not be in the stop-word list; it must be in the noun database of WordNet; and it must not be a morphological variation upon the original word.[6]

We replaced one word in every 200 in our sample texts with a malapropism that was generated in this way. If a malapropism could not be found for a target word, subsequent words were considered one by one until a suitable one was found.

Some of the sample texts did not have any malapropisms because they were less than 200 words long. However, whenever a text is too small to provide enough context, the algorithm used to identify lexical chains is not valid. For this experiment, a text was considered too small if it did not get at least one malapropism (though some small articles **did** get one). Eighteen such articles were found and removed.

Here is a sample of the malapropisms inserted in one article. The original words are shown in brackets:

(10) Much of that data, he notes, is available *toady* [*today*] electronically.

(11) Among the largest OTC issues, Farmers Group, which expects B.A.T. Industries to launch a hostile *tenter* [*tender*] offer for it, jumped $2\frac{3}{8}$ to 62 yesterday.

[6]This technique has the disadvantage of sometimes generating a new word that is very close semantically to the original one (*e.g., billion* $\rightarrow$ *million*). These new words are not actual malapropisms in the sense of being malapropos in the text, and hence are not candidates for detection by our algorithm. Nonetheless, they are counted as malapropisms in the computation of the results of the experiment. Fortunately, such situations were rare.

(12) But most of yesterday's popular issues were small out-of-the-limelight technology companies that slipped in price a bit last year after the *crush* [*crash*], although their earnings are on the rise.

4.3.2 Results

We will give examples of successes and failures in the experiment, and then quantify the results.

First, we show examples of the algorithm's performance on genuine malapropisms. The malapropism *toady* shown in (10) is an example of a malapropism that was detected as such and the correct replacement was found; that is, *toady* was placed in a chain by itself, and the spelling variant *today* was found to fit in a chain with other words such as *yesterday* and *month* from the same article.[7] The malapropism *tenter* shown in (11) was not detected, as it did not appear in an atomic chain, having been connected to *stock* by the following chain:

(13) *tenter* IS A *framework / frame* INCLUDES *handbarrow* HAS PART *handle / grip / hold* INCLUDES *stock*

This is because while the article contained many references to *stocks* in the financial sense, there was nothing (except the complete meaning of the article and the context in which it was published) to disambiguate the word—except, ironically, the word that was transformed into a malapropism! The malapropism *crush* shown in (12) was also detected, but the correct replacement, *crash*, was not suggested as it did not fit into any chain; rather, *brush* was suggested, as it too fitted with *stock* (because brushes have handles).

Now we show examples of the algorithm's performance on non-malapropisms. In the following sentence, the word *television* was placed in an atomic chain (despite the presence of *network*, which has been wrongly disambiguated) and hence regarded as suspicious:

(14) QVC Network, a 24-hour home *television* shopping issue, said yesterday it expects fiscal 1989 sales of $170 million to $200 million, …

[7]The success of the algorithm here is surprising, as these surely are all such common words in newspaper articles, that they should really have been stop-words.

However, no spelling variants were found for *television*, so no alarm was raised. In the following sentence, the word *souring* was placed in an atomic chain and hence regarded as suspicious:

(15) It is suffering huge loan losses from *souring* real estate loans, and is the focus of increased monitoring by federal regulators who have braced themselves for a possible rescue.

It has three spelling variants— *pouring, scouring,* and *soaring*—but none of them were chainable, and so no alarm was raised. In the following sentence, the word *fear* was placed in an atomic chain:

(16) And while institutions until the past month or so stayed away from the smallest issues for *fear* they would get stuck in an illiquid stock, …

Moreover, chains were found for three of its many spelling variants, *gear, pear,* or *year,* and so the word was flagged as a possible error and an alarm raised. (*Pear* was chained to *Lotus*, the name of a company mentioned in the article, because both are plants.)

Analysis of results Table 1 displays the results of the experiment quantitatively. The 482 articles retained for the experiment included a total of 322,645 words, of which 1,409 were malapropisms. Of all the words, 33.9% were inserted into lexical chains. Of the malapropisms, 442 (31.4%) were placed in atomic chains. 7,572 (7.01%) of the non-malapropisms were also inserted in atomic chains. Thus, actual malapropisms were 4.47 times more likely to be inserted in an atomic chain.

Alarms resulted from 89.8% of the malapropisms in atomic chains and from 36.6% of the non-malapropisms (the latter being false alarms). Thus malapropisms were 2.46 times more likely to result in alarms than non-malapropisms. The proportion of alarms that were false is 87.5%. The average number of replacement suggestions per alarm was 2.66.

Overall, an alarm was generated for 28.2% of the malapropisms. Furthermore, an alarm in which the original word (the word for which a malapropism was substituted) was one of the replacement suggestions was generated for 24.8% of the malapropisms. Malapropisms were 11 times more likely to result in an alarm than other words. However, this was at the cost of 25.3 false alarms per 1000 words eligible for chaining, or 8.59 false alarms per 1000 words of text.

Table 1: Results

Total number of words in corpus	322,645
Number of words in chains	109,407
Number of non-malapropisms	107,998
Number of malapropisms	1,409
Number of atomic chains	8,014
Number that contained malapropisms	442
Number that did not	7,572
Performance factor	4.47
Number of alarms	3,167
Number of true alarms	397
Number of false alarms	2,770
Performance factor	2.46
Performance factor overall	11.0
Number of perfectly detected and corrected malapropisms	349

5 Conclusion

5.1 Review

In this paper, we have adapted Morris and Hirst's (1991) *Roget*-based algorithm for lexical chains to WordNet, and used the result in an experiment in the detection and correction of malapropisms. Although concessions had to be made to the structure and content of WordNet, the results are nonetheless encouraging. The further development of WordNet will surely permit better lexical chaining, which in turn will lead to more-acceptable performance by the algorithm for malapropism detection and correction.

Two important ways that WordNet is limited compared to *Roget's* are its restriction to formal relations rather than connections by general association, which the *Roget*-based algorithm exploits, and its varying conceptual density. The reasons for the first restriction are understandable: if 'fuzzy' relationships such as *secretary–typewriter* are admitted, it's hard to know where to stop; unlike *Roget's*, WordNet is not intended as a 'memory-jogger'. Nonetheless, the addition of relations based on required or typical role-fillers, such as *bath–soap*, would surely be helpful. The density problem is not just a problem with WordNet; one would naturally expect to find more concepts in some subjects than others and therefore a higher density of synsets. But the formal structure of WordNet exacerbates the problem compared to *Roget's*.

In addition, we were limited by the division of WordNet into separate files by syntactic category, with limited connections. The relations of lexical chaining stand above syntactic category; for our purposes, the relation between *scholar* and *teach* (noun and verb) is no different than between *scholar* and *teacher* (noun and noun); stronger cross-category connections in WordNet would be helpful.

Our method for detecting and correcting malapropisms with lexical chains is, of course, limited by the accuracy of the lexical chainer, which can never be perfect. A more deep-seated limitation of the method is in its assumption that a malapropism will almost always be unrelated semantically to the text in which it occurs. This is probably untrue. Lexical substitution errors in speech show a bias towards concepts that are active in the current discourse (see the papers in Fromkin 1980, especially Hotopf 1980), and it is reasonable to expect analogous errors in typing to follow a similar pattern (as is implicit in, for example, Rumelhart and Norman's (1982) model of typing).[8]

5.2 Similar research

Stairmand (1994) has also developed a lexical chainer based on WordNet. Unlike the one described here, Stairmand's is intended primarily for use in information retrieval, taking into account the idea of the **density** of a chain in different places in the text. Stairmand's chainer works by a somewhat different method from ours. First, he collects all the content words in the text; he then generates the set of all word senses in WordNet that are close to the words of the text, the set of so-called **expanded terms**; and finally he looks for links that expanded terms form between words in the text. Disambiguation, to the extent that it occurs, is apparently an implicit side-effect. It is unclear whether

[8]For example, one of our colleagues observed the following malapropism:

(i) In this model, auxin-enhanced excytotic vesicle transport and insertion of a rapidly turning-over H^+-ATPase into the plasma membrane are envisioned to stimulate hydrogen ion excretion into the apoplast and initiate wall loosening. In this model, fusicoccin stimulates *protein* excretion via a separate independent mechanism.

The word *protein* (which does not occur elsewhere in the paper) should be *proton* (which has many related words in the context); but the previous sentence contains the name of a particular protein, H^+-ATPase, and this is surely the cause of the malapropism. Similarly, our algorithm would connect *protein* back to *H^+-ATPase* (were the latter to find its way into WordNet), and find nothing to worry about. (We are grateful to Nadia Talent for pointing this example out to us. It appeared in: Rayle, David L. and Cleland, Robert E. "The acid growth theory of auxin-induced cell elongation is alive and well." *Plant Physiology*, **99**, 1992, p. 1274.)

or not this batch-oriented approach, which has a more-limited notion of semantic relatedness, leads to chains that differ significantly from ours. However, as it seems to discard the necessary information as to position in the text, the method could not be used for tasks such as discourse segmentation, which was Morris and Hirst's original motivation for lexical chains.

Li, Szpakowicz, and Matwin (1995) have described a WordNet-based algorithm that disambiguates nouns in the context of their verbs by looking for other nouns in similar contexts and taking the sense of each that makes them most closely related in WordNet. This might be thought of as using, in effect, miniature, ad hoc, lexical chains. *[In the final version, we should include a short discussion of the chapters by Voorhees and Leacock & Chodorow on disambiguation, and that of Kazman and Al-Halimi on enhancements to lexical chains.]*

5.3 Lexical chains as context

The use of lexical chains as a context for tasks such as disambiguation, discourse segmentation, and finding malapropisms can be thought of as a lite form of methods based on spreading activation or marker passing in knowledge bases (Hirst 1987). The idea that all methods have in common is that semantic distance is the primary cue that context provides, and that measures of semantic distance are inherent in a network structure. In the case of spreading activation or marker passing, the network in question is assumed to be a fully articulated network of concepts; in lexical chaining, it assumed to be a network of word senses with conceptual relations. The former, of course, is a richer representation, and (at least in principle) can perform the task more accurately; but its use assumes the ability to determine fairly precisely the concepts that are explicit and implicit in the text, and it is easily led completely astray by errors. The latter, on the other hand, is a relatively impoverished representation that could not be the basis for any kind of conceptual 'understanding' of a text; but it is more flexible and forgiving of errors, and hence can be used in tasks that, while semantic, do not require a complete analysis of meaning.

Acknowledgements

We are grateful to Jane Morris, Stephen Green, Daniel Marcus, Philip Edmonds, and Jeffrey Mark Siskind for discussions, help, and feedback. This research was supported by a grant to the first author from the Natural Sciences and Engineering Research Council of Canada, and a scholarship to the second author from Fonds pour la Formation de Chercheurs et l'Aide à la Recherche.

References

Atwell, Eric and Elliott, S. (1987). "Dealing with ill-formed English text." In Garside, Leech, and Sampson, 1987, chapter 10.

Chapman, Robert L. (editor) (1992). *Roget's International Thesaurus*, 5th edition. New York: HarperCollins Publishers.

Fromkin, Victoria A. (1980). *Errors in linguistic performance: Slips of the tongue, ear, pen, and hand.* New York: Academic Press.

Garside, Roger; Leech, Geoffrey; and Sampson, Geoffrey (1987). *The computational analysis of English: A corpus-based approach*. Longman Inc., New York.

Grosz, Barbara J. and Sidner, Candace L. (1986). "Attention, intentions, and the structure of discourse." *Computational Linguistics*, **12**(3), 175–204.

Halliday, M.A.K. and Hasan, Ruqaiya (1976). *Cohesion in English.* London: Longman.

Hirst, Graeme (1987). *Semantic interpretation and the resolution of ambiguity.* Cambridge University Press.

Hotopf, W.H.N. (1980). "Semantic similarity as a factor in whole-word slips of the tongue." In Fromkin 1980, 97–109.

Kukich, Karen (1992). "Techniques for automatically correcting words in text." *ACM Computing Surveys*, **24**(4), 377–439.

Li, Xiaobin; Szpakowicz, Stan; and Matwin, Stan (1995). "A WordNet-based algorithm for word sense disambiguation." *Proceedings, 14th International Joint Conference on Artificial Intelligence*, Montreal, August 1995.

Mitton, R. (1987). "Spelling checkers, spelling correctors, and the misspelling of poor spellers." *Information Processing and Management*, **23**(5), 495–505.

Morris, Jane (1988). *Lexical cohesion, the thesaurus, and the structure of text*, MSc thesis, Department of Computer Science, University of Toronto (published as technical report CSRI-219).

Morris, Jane and Hirst, Graeme (1991). "Lexical cohesion computed by thesaural relations as an indicator of the structure of text." *Computational Linguistics*, **17**(1), 21–48.

Okumura, Manabu and Honda, Takeo (1994). "Word sense disambiguation and text segmentation based on lexical cohesion." In *Proceedings of the Fifteenth International Conference on Computational Linguistics (COLING-94)*, volume 2, 755–761.

Peterson, James L. (1986). "A note on undetected typing errors." *Communications of the ACM*, **29**(7), 633–637.

Rumelhart, David E. and Norman, Donald A. (1982). "Simulating a skilled typist: A study of skilled cognitive-motor performance." *Cognitive Science*, **6**(1), 1–36.

St-Onge, David (1995). *Detecting and correcting malapropisms with lexical chains.*. MSc thesis, Department of Computer Science, University of Toronto, (published as technical report CSRI-319). *ftp://ftp.csri.toronto.edu/csri-technical-reports/319*

Stairmand, Mark (1994). "Lexical chains, WordNet and information retrieval." Unpublished MS, Centre for Computational Linguistics, UMIST, Manchester.