Sentence Fusion for Multidocument News Summarization

Regina Barzilay*
Massachusetts Institute of Technology

Kathleen R. McKeown†
Columbia University

A system that can produce informative summaries, highlighting common information found in many online documents, will help Web users to pinpoint information that they need without extensive reading. In this article, we introduce sentence fusion, a novel text-to-text generation technique for synthesizing common information across documents. Sentence fusion involves bottom-up local multisequence alignment to identify phrases conveying similar information and statistical generation to combine common phrases into a sentence. Sentence fusion moves the summarization field from the use of purely extractive methods to the generation of abstracts that contain sentences not found in any of the input documents and can synthesize information across sources.

1. Introduction

Redundancy in large text collections, such as the Web, creates both problems and opportunities for natural language systems. On the one hand, the presence of numerous sources conveying the same information causes difficulties for end users of search engines and news providers; they must read the same information over and over again. On the other hand, redundancy can be exploited to identify important and accurate information for applications such as summarization and question answering (Mani and Bloedorn 1997; Radev and McKeown 1998; Radev, Prager, and Samn 2000; Clarke, Cormack, and Lynam 2001; Dumais et al. 2002; Chu-Carroll et al. 2003). Clearly, it would be highly desirable to have a mechanism that could identify common information among multiple related documents and fuse it into a coherent text. In this article, we present a method for sentence fusion that exploits redundancy to achieve this task in the context of multidocument summarization.

A straightforward approach for approximating sentence fusion can be found in the use of sentence extraction for multidocument summarization (Carbonell and Goldstein 1998; Radev, Jing, and Budzikowska 2000; Marcu and Gerber 2001; Lin and Hovy 2002). Once a system finds a set of sentences that convey similar information (e.g., by clustering), one of these sentences is selected to represent the set. This is a robust approach that is always guaranteed to output a grammatical sentence. However, extraction is only a coarse approximation of fusion. An extracted sentence may include not only common information, but additional information specific to the article from

* Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA 02458. E-mail: regina@csail.mit.edu.
† Department of Computer Science, Columbia University, New York, NY 10027. E-mail: kathy@cs.columbia.edu.

Submission received: 14 September 2003; revised submission received: 23 February 2005; accepted for publication: 19 March 2005.

which it came, leading to source bias and aggravating fluency problems in the extracted summary. Attempting to solve this problem by including more sentences to restore the original context might lead to a verbose and repetitive summary.

Instead, we want a fine-grained approach that can identify only those pieces of sentences that are common. Language generation offers an appealing approach to the problem, but the use of generation in this context raises significant research challenges. In particular, generation for sentence fusion must be able to operate in a domain-independent fashion, scalable to handle a large variety of input documents with various degrees of overlap. In the past, generation systems were developed for limited domains and required a rich semantic representation as input. In contrast, for this task we require **text-to-text generation,** the ability to produce a new text given a set of related texts as input. If language generation can be scaled to take fully formed text as input without semantic interpretation, selecting content and producing well-formed English sentences as output, then generation has a large potential payoff.

In this article, we present the concept of sentence fusion, a novel text-to-text generation technique which, given a set of similar sentences, produces a new sentence containing the information common to most sentences in the set. The research challenges in developing such an algorithm lie in two areas: identification of the fragments conveying common information and combination of the fragments into a sentence. To identify common information, we have developed a method for aligning syntactic trees of input sentences, incorporating paraphrasing information. Our alignment problem poses unique challenges: We only want to match a subset of the subtrees in each sentence and are given few constraints on permissible alignments (e.g., arising from constituent ordering, start or end points). Our algorithm meets these challenges through bottom-up local multisequence alignment, using words and paraphrases as anchors. Combination of fragments is addressed through construction of a fusion lattice encompassing the resulting alignment and linearization of the lattice into a sentence using a language model. Our approach to sentence fusion thus features the integration of robust statistical techniques, such as local, multisequence alignment and language modeling, with linguistic representations automatically derived from input documents.

Sentence fusion is a significant first step toward the generation of abstracts, as opposed to extracts (Borko and Bernier 1975), for multidocument summarization. Unlike extraction methods (used by the vast majority of summarization researchers), sentence fusion allows for the true synthesis of information from a set of input documents. It has been shown that combining information from several sources is a natural strategy for multidocument summarization. Analysis of human-written summaries reveals that most sentences combine information drawn from multiple documents (Banko and Vanderwende 2004). Sentence fusion achieves this goal automatically. Our evaluation shows that our approach is promising, with sentence fusion outperforming sentence extraction for the task of content selection.

This article focuses on the implementation and evaluation of the sentence fusion method within the multidocument summarization system MultiGen, which daily summarizes multiple news articles on the same event as part[1] of Columbia's news browsing system Newsblaster (http://newsblaster.cs.columbia.edu/). In the next section, we provide an overview of MultiGen, focusing on components that produce input or operate over output of sentence fusion. In Section 3, we provide an overview of

1 In addition to MultiGen, Newsblaster utilizes another summarizer, DEMS (Schiffman, Nenkova, and McKeown 2002), to summarize heterogeneous sets of articles.

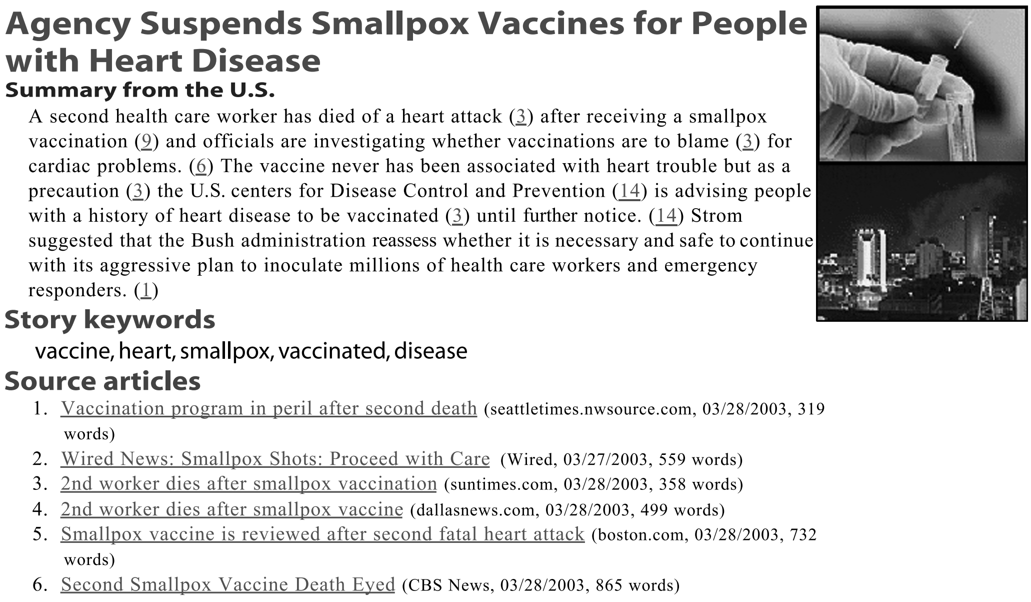
our fusion algorithm and detail on its main steps: identification of common information (Section 3.1), fusion lattice computation (Section 3.2), and lattice linearization (Section 3.3). Evaluation results and their analysis are presented in Section 4. Analysis of the system's output reveals the capabilities and the weaknesses of our text-to-text generation method and identifies interesting challenges that will require new insights. An overview of related work and a discussion of future directions conclude the article.

2. Framework for Sentence Fusion: MultiGen

Sentence fusion is the central technique used within the MultiGen summarization system. MultiGen takes as input a cluster of news stories on the same event and produces a summary which synthesizes common information across input stories. An example of a MultiGen summary is shown in Figure 1. The input clusters are automatically produced from a large quantity of news articles that are retrieved by Newsblaster from 30 news sites each day.

In order to understand the role of sentence fusion within summarization, we overview the MultiGen architecture, providing details on the processes that precede sentence fusion and thus, the input that the fusion component requires. Fusion itself is discussed in the subsequent sections of the article.

MultiGen follows a pipeline architecture, shown in Figure 2. The analysis component of the system, Simfinder (Hatzivassiloglou, Klavans, and Eskin 1999) clusters sentences of input documents into **themes,** groups of sentences that convey similar information (Section 2.1). Once themes are constructed, the system selects a subset of the groups to be included in the summary, depending on the desired compression


Agency Suspends Smallpox Vaccines for People with Heart Disease

Summary from the U.S.

A second health care worker has died of a heart attack (3) after receiving a smallpox vaccination (9) and officials are investigating whether vaccinations are to blame (3) for cardiac problems. (6) The vaccine never has been associated with heart trouble but as a precaution (3) the U.S. centers for Disease Control and Prevention (14) is advising people with a history of heart disease to be vaccinated (3) until further notice. (14) Strom suggested that the Bush administration reassess whether it is necessary and safe to continue with its aggressive plan to inoculate millions of health care workers and emergency responders. (1)

Story keywords

vaccine, heart, smallpox, vaccinated, disease

Source articles

1. Vaccination program in peril after second death (seattletimes.nwsource.com, 03/28/2003, 319 words)
2. Wired News: Smallpox Shots: Proceed with Care (Wired, 03/27/2003, 559 words)
3. 2nd worker dies after smallpox vaccination (suntimes.com, 03/28/2003, 358 words)
4. 2nd worker dies after smallpox vaccine (dallasnews.com, 03/28/2003, 499 words)
5. Smallpox vaccine is reviewed after second fatal heart attack (boston.com, 03/28/2003, 732 words)
6. Second Smallpox Vaccine Death Eyed (CBS News, 03/28/2003, 865 words)

Figure 1
An example of MultiGen summary as shown in the Columbia Newsblaster Interface. Summary phrases are followed by parenthetical numbers indicating their source articles. The last sentence is extracted because it was repeated verbatim in several input articles.

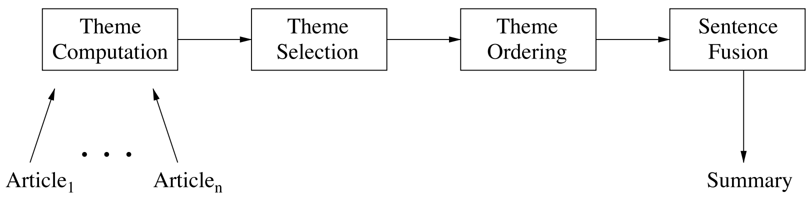


Figure 2
MultiGen architecture.

length (Section 2.2). The selected groups are passed to the ordering component, which selects a complete order among themes (Section 2.3).

2.1 Theme Construction

The analysis component of MultiGen, Simfinder, identifies themes, groups of sentences from different documents that each say roughly the same thing. Each theme will ultimately correspond to at most one sentence in the output summary, generated by the fusion component, and there may be many themes for a set of articles. An example of a theme is shown in Table 1. As the set of sentences in the table illustrates, sentences within a theme are not exact repetitions of each other; they usually include phrases expressing information that is *not* common to all sentences in the theme. Information that is common across sentences is shown in the table in boldface; other portions of the sentence are specific to individual articles. If one of these sentences were used as is to represent the theme, the summary would contain extraneous information. Also, errors in clustering might result in the inclusion of some unrelated sentences. Evaluation involving human judges revealed that Simfinder identifies similar sentences with 49.3% precision at 52.9% recall (Hatzivassiloglou, Klavans, and Eskin 1999). We will discuss later how this error rate influences sentence fusion.

To identify themes, Simfinder extracts linguistically motivated features for each sentence, including WordNet synsets (Miller et al. 1990) and syntactic dependencies, such as subject–verb and verb–object relations. A log-linear regression model is used to combine the evidence from the various features into a single similarity value. The model was trained on a large set of sentences which were manually marked for similarity. The output of the model is a listing of real-valued similarity values on sentence pairs. These similarity values are fed into a clustering algorithm that partitions the sentences into closely related groups.

Table 1
Theme with corresponding fusion sentence.

1. IDF Spokeswoman did not confirm this, but said **the Palestinians fired an antitank missile at a bulldozer.**
2. The clash erupted when **Palestinian militants fired machine guns and antitank missiles at a bulldozer** that was building an embankment in the area to better protect Israeli forces.
3. The army expressed "regret at the loss of innocent lives" but a senior commander said troops had shot in self-defense **after being fired at while using bulldozers** to build a new embankment at an army base in the area.
Fusion sentence: Palestinians fired an antitank missile at a bulldozer.

2.2 Theme Selection

To generate a summary of predetermined length, we induce a ranking on the themes and select the *n* highest.[2] This ranking is based on three features of the theme: size measured as the number of sentences, similarity of sentences in a theme, and salience score. The first two of these scores are produced by Simfinder, and the salience score is computed using **lexical chains** (Morris and Hirst 1991; Barzilay and Elhadad 1997) as described below. Combining different rankings further filters common information in terms of salience. Since each of these scores has a different range of values, we perform ranking based on each score separately, then induce total ranking by summing ranks from individual categories:

$$\textit{Rank}\text{ (theme)} = \textit{Rank}\text{ (Number of sentences in theme)} + \textit{Rank}\text{ (Similarity of sentences in theme)} + \textit{Rank}\text{ (Sum of lexical chain scores in theme)}$$

Lexical chains—sequences of semantically related words—are tightly connected to the lexical cohesive structure of the text and have been shown to be useful for determining which sentences are important for single-document summarization (Barzilay and Elhadad 1997; Silber and McCoy 2002). In the multidocument scenario, lexical chains can be adapted for theme ranking based on the salience of theme sentences within their original documents. Specifically, a theme that has many sentences ranked high by lexical chains as important for a single-document summary is, in turn, given a higher salience score for the multidocument summary. In our implementation, a salience score for a theme is computed as the sum of lexical chain scores of each sentence in a theme.

2.3 Theme Ordering

Once we filter out the themes that have a low rank, the next task is to order the selected themes into coherent text. Our ordering strategy aims to capture chronological order of the main events and ensure coherence. To implement this strategy in MultiGen, we select for each theme the sentence which has the earliest publication time (**theme time stamp**). To increase the coherence of the output text, we identify blocks of topically related themes and then apply chronological ordering on blocks of themes using theme time stamps (Barzilay, Elhadad, and McKeown 2002). These stages produce a sorted set of themes which are passed as input to the sentence fusion component, described in the next section.

3. Sentence Fusion

Given a group of similar sentences—a theme—the problem is to create a concise and fluent fusion of information, reflecting facts common to all sentences. (An example of a fusion sentence is shown in Table 1.) To achieve this goal we need to identify phrases common to most theme sentences, then combine them into a new sentence.

At one extreme, we might consider a shallow approach to the fusion problem, adapting the "bag of words" approach. However, sentence intersection in a set-theoretic sense produces poor results. For example, the intersection of the first two sentences

2 Typically, Simfinder produces at least 20 themes given an average Newsblaster cluster of nine articles. The length of a generated summary typically does not exceed seven sentences.

from the theme shown in Table 1 is (*the, fired, antitank, at, a, bulldozer*). Besides its being ungrammatical, it is impossible to understand what event this intersection describes. The inadequacy of the bag-of-words method to the fusion task demonstrates the need for a more linguistically motivated approach. At the other extreme, previous approaches (Radev and McKeown 1998) have demonstrated that this task is feasible when a detailed semantic representation of the input sentences is available. However, these approaches operate in a limited domain (e.g., terrorist events), where information extraction systems can be used to interpret the source text. The task of mapping input text into a semantic representation in a domain-independent setting extends well beyond the ability of current analysis methods. These considerations suggest that we need a new method for the sentence fusion task. Ideally, such a method would not require a full semantic representation. Rather, it would rely on input texts and shallow linguistic knowledge (such as parse trees) that can be automatically derived from a corpus to generate a fusion sentence.

In our approach, sentence fusion is modeled after the typical generation pipeline: content selection (what to say) and surface realization (how to say it). In contrast to that involved in traditional generation systems in which a content selection component chooses content from semantic units, our task is complicated by the lack of semantics in the textual input. At the same time, we can benefit from the textual information given in the input sentences for the tasks of syntactic realization, phrasing, and ordering; in many cases, constraints on text realization are already present in the input.

The algorithm operates in three phases:

- **Identification of common information** (Section 3.1)
- **Fusion lattice computation** (Section 3.2)
- **Lattice linearization** (Section 3.3)

Content selection occurs primarily in the first phase, in which our algorithm uses local alignment across pairs of parsed sentences, from which we select fragments to be included in the fusion sentence. Instead of examining all possible ways to combine these fragments, we select a sentence in the input which contains most of the fragments and transform its parsed tree into the fusion lattice by eliminating nonessential information and augmenting it with information from other input sentences. This construction of the fusion lattice targets content selection, but in the process, alternative verbalizations are selected, and thus some aspects of realization are also carried out in this phase. Finally, we generate a sentence from this representation based on a language model derived from a large body of texts.

3.1 Identification of Common Information

Our task is to identify information shared between sentences. We do this by aligning constituents in the syntactic parse trees for the input sentences. Our alignment process differs considerably from alignment for other NL tasks, such as machine translation, because we cannot expect a complete alignment. Rather, a subset of the subtrees in one sentence will match different subsets of the subtrees in the others. Furthermore, order across trees is not preserved, there is no natural starting point for alignment, and there are no constraints on crosses. For these reasons we have developed a bottom-up local multisequence alignment algorithm that uses words and phrases as anchors for matching. This algorithm operates on the dependency trees for pairs of input sentences. We use a dependency-based representation because it abstracts over features irrelevant for comparison such as constituent ordering. In the subsections that follow, we describe first how this representation is computed, then how dependency subtrees are aligned, and finally how we choose between constituents conveying overlapping information.

In this section we first describe an algorithm which, given a pair of sentences, determines which sentence constituents convey information appearing in both sentences. This algorithm will be applied to pairwise combinations of sentences in the input set of related sentences.

The intuition behind the algorithm is to compare all constituents of one sentence to those of another and select the most similar ones. Of course, how this comparison is performed depends on the particular sentence representation used. A good sentence representation will emphasize sentence features that are relevant for comparison, such as dependencies between sentence constituents, while ignoring irrelevant features, such as constituent ordering. A representation which fits these requirements is a dependency-based representation (Melcuk 1988). We first detail how this representation is computed, then describe a method for aligning dependency subtrees.

3.1.1 Sentence Representation. Our sentence representation is based on a **dependency tree,** which describes the sentence structure in terms of dependencies between words. The similarity of the dependency tree to a predicate–argument structure makes it a natural representation for our comparison.[3] This representation can be constructed from the output of a traditional parser. In fact, we have developed a rule-based component that transforms the phrase structure output of Collins's (2003) parser into a representation in which a node has a direct link to its dependents. We also mark verb–subject and verb–node dependencies in the tree.

The process of comparing trees can be further facilitated if the dependency tree is abstracted to a canonical form which eliminates features irrelevant to the comparison. We hypothesize that the difference in grammatical features such as auxiliaries, number, and tense has a secondary effect when the meaning of sentences is being compared. Therefore, we represent in the dependency tree only nonauxiliary words with their associated grammatical features. For nouns, we record their number, articles, and class (common or proper). For verbs, we record tense, mood (indicative, conditional, or infinitive), voice, polarity, aspect (simple or continuous), and taxis (perfect or none). The eliminated auxiliary words can be re-created using these recorded features. We also transform all passive-voice sentences to the active voice, changing the order of affected children.

While the alignment algorithm described in Section 3.1.2 produces one-to-one mappings, in practice some paraphrases are not decomposable to words, forming one-to-many or many-to-many paraphrases. Our manual analysis of paraphrased sentences (Barzilay 2003) revealed that such alignments most frequently occur in pairs of noun phrases (e.g., *faculty member* and *professor*) and pairs including verbs with particles (e.g., *stand up*, *rise*). To correctly align such phrases, we flatten subtrees containing noun phrases and verbs with particles into one node. We subsequently determine matches between flattened sentences using statistical metrics.

3 Two paraphrasing sentences which differ in word order may have significantly different trees in phrase-based format. For instance, this phenomenon occurs when an adverbial is moved from a position in the middle of a sentence to the beginning of a sentence. In contrast, dependency representations of these sentences are very similar.

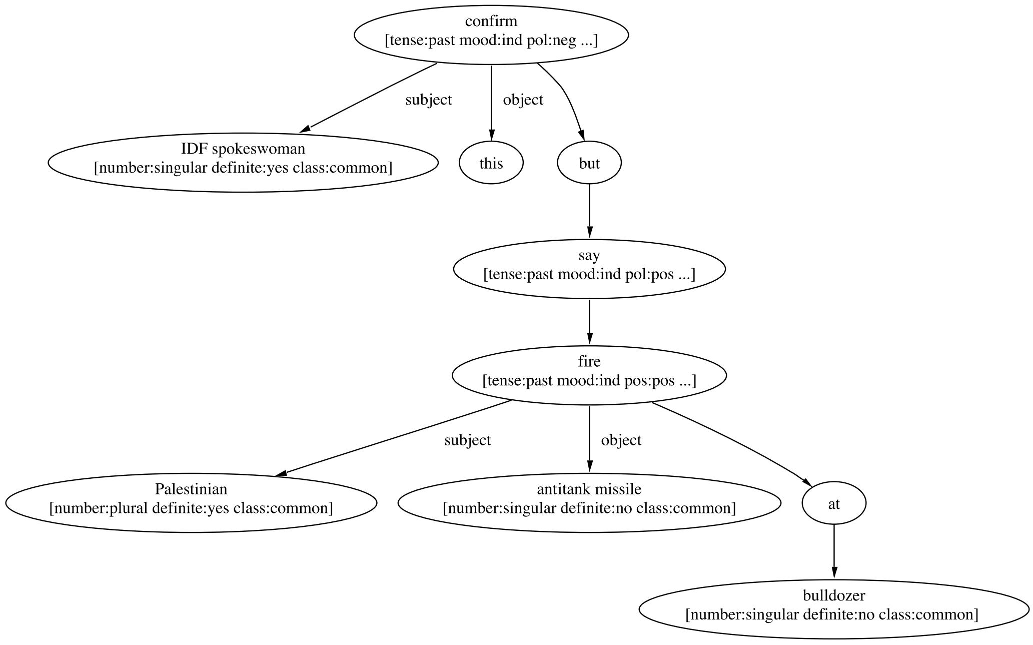

confirm
[tense:past mood:ind pol:neg ...]
subject
object
IDF spokeswoman
[number:singular definite:yes class:common]
this
but
say
[tense:past mood:ind pol:pos ...]
fire
[tense:past mood:ind pos:pos ...]
subject
object
Palestinian
[number:plural definite:yes class:common]
antitank missile
[number:singular definite:no class:common]
at
bulldozer
[number:singular definite:no class:common]

Figure 3
Dependency tree of the sentence *The IDF spokeswoman did not confirm this, but said the Palestinians fired an antitank missile at a bulldozer on the site.* The features of the node *confirm* are explicitly marked.

An example of a sentence and its dependency tree with associated features is shown in Figure 3. (In figures of dependency trees hereafter, node features are omitted for clarity.)

3.1.2 Alignment. Our alignment of dependency trees is driven by two sources of information: the similarity between the structure of the dependency trees and the similarity between lexical items. In determining the structural similarity between two trees, we take into account the types of edges (which indicate the relationships between nodes). An edge is labeled by the syntactic function of the two nodes it connects (e.g., subject–verb). It is unlikely that an edge connecting a subject and verb in one sentence, for example, corresponds to an edge connecting a verb and an adjective in another sentence.

The word similarity measures take into account more than word identity: They also identify pairs of paraphrases, using WordNet and a paraphrasing dictionary. We automatically constructed the paraphrasing dictionary from a large comparable news corpus using the co-training method described in Barzilay and McKeown (2001). The dictionary contains pairs of word-level paraphrases as well as phrase-level paraphrases.[4] Several examples of automatically extracted paraphrases are given in Table 2. During alignment, each pair of nonidentical words that do not comprise a synset in

4 The comparable corpus and the derived dictionary are available at http://www.cs.cornell.edu/~regina/thesis-data/comp/input/processed.tbz2 and http://www.cs.cornell.edu/~regina/thesis-data/comp/output/comp2-ALL.txt. For details on the corpus collection and evaluation of the paraphrase quality, see Barzilay (2003).

Table 2
Lexical paraphrases extracted by the algorithm from the comparable news corpus.

(auto, automobile), (closing, settling), (rejected, does not accept), (military, army), (IWC, International Whaling Commission), (Japan, country), (researching, examining), (harvesting, killing), (mission-control office, control centers), (father, pastor), (past 50 years, four decades), (Wangler, Wanger), (teacher, pastor), (fondling, groping), (Kalkilya, Qalqilya), (accused, suspected), (language, terms), (head, president), (U.N., United Nations), (Islamabad, Kabul), (goes, travels), (said, testified), (article, report), (chaos, upheaval), (Gore, Lieberman), (revolt, uprising), (more restrictive local measures, stronger local regulations) (countries, nations), (barred, suspended), (alert, warning), (declined, refused), (anthrax, infection), (expelled, removed), (White House, White House spokesman Ari Fleischer), (gunmen, militants)

WordNet is looked up in the paraphrasing dictionary; in the case of a match, the pair is considered to be a paraphrase.

We now give an intuitive explanation of how our tree similarity function, denoted by *Sim*, is computed. If the optimal alignment of two trees is known, then the value of the similarity function is the sum of the similarity scores of aligned nodes and aligned edges. Since the best alignment of given trees is not known a priori, we select the maximal score among plausible alignments of the trees. Instead of exhaustively traversing the space of all possible alignments, we recursively construct the best alignment for trees of given depths, assuming that we know how to find an optimal alignment for trees of shorter depths. More specifically, at each point of the traversal we consider two cases, shown in Figure 4. In the first case, two top nodes are aligned with each other, and their children are aligned in an optimal way by applying the algorithm to shorter trees. In the second case, one tree is aligned with one of the children of the top node of the other tree; again we can apply our algorithm for this computation, since we decrease the height of one of the trees.

Before giving the precise definition of *Sim*, we introduce some notation. When T is a tree with root node v, we let $c(T)$ denote the set containing all children of v. For a tree T containing a node s, the subtree of T which has s as its root node is denoted by T_s.

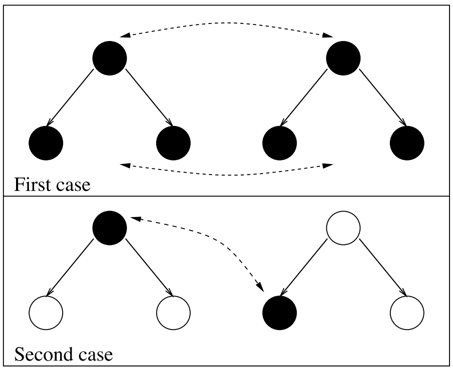


Figure 4
Tree alignment computation. In the first case two tops are aligned, while in the second case the top of one tree is aligned with a child of another tree.

Given two trees T and T' with root nodes v and v', respectively, the similarity $Sim(T, T')$ between the trees is defined to be the maximum of the three expressions $NodeCompare(T, T')$, $\max_{s \in c(T)} Sim(T_s, T')$, and $\max_{s' \in c(T')} Sim(T, T'_{s'})$. The upper part of Figure 4 depicts the computation of $NodeCompare(T, T')$, in which two top nodes are aligned with each other. The remaining expressions, $\max_{s \in c(T)} Sim(T_s, T')$, and $\max_{s' \in c(T')} Sim(T, T'_{s'})$, capture mappings in which the top of one tree is aligned with one of the children of the top node of the other tree (the bottom of Figure 4).

The maximization in the *NodeCompare* formula searches for the best possible alignment for the child nodes of the given pair of nodes and is defined by

$$\begin{aligned} NodeCompare(T, T') = {} & NodeSimilarity(v, v') \\ & + \max_{m \in M(c(T), c(T'))} \left[\sum_{(s,s') \in m} \left(EdgeSimilarity((v, s), (v', s')) + Sim(T_s, T'_{s'}) \right) \right] \end{aligned}$$

where $M(A, A')$ is the set of all possible matchings between A and A', and a matching (between A and A') is a subset m of $A \times A'$ such that for any two distinct elements $(a, a'), (b, b') \in m$, both $a \neq b$ and $a' \neq b'$. In the base case, when one of the trees has depth one, $NodeCompare(T, T')$ is defined to be $NodeSimilarity(v, v')$.

The similarity score $NodeSimilarity(v, v')$ of atomic nodes depends on whether the corresponding words are identical, paraphrases, or unrelated. The similarity scores for pairs of identical words, pairs of synonyms, pairs of paraphrases, and edges (given in Table 3) are manually derived using a small development corpus. While learning of the similarity scores automatically is an appealing alternative, its application in the fusion context is challenging because of the absence of a large training corpus and the lack of an automatic evaluation function.[5] The similarity of nodes containing flattened subtrees,[6] such as noun phrases, is computed as the score of their intersection normalized by the length of the longest phrase. For instance, the similarity score of the noun phrases *antitank missile* and *machine gun and antitank missile* is computed as a ratio between the score of their intersection *antitank missile* (2), divided by the length of the latter phrase (5).

The similarity function *Sim* is computed using bottom-up dynamic programming, in which the shortest subtrees are processed first. The alignment algorithm returns the similarity score of the trees as well as the optimal mapping between the subtrees of input trees. The pseudocode of this function is presented in the Appendix. In the resulting tree mapping, the pairs of nodes whose *NodeSimilarity* positively contributed to the alignment are considered parallel. Figure 5 shows two dependency trees and their alignment.

As is evident from the *Sim* definition, we are considering only one-to-one node "matchings": Every node in one tree is mapped to at most one node in another tree. This restriction is necessary because the problem of optimizing many-to-many alignments

5 Our preliminary experiments with n-gram-based overlap measures, such as BLEU (Papineni et al. 2002) and ROUGE (Lin and Hovy 2003), show that these metrics do not correlate with human judgments on the fusion task, when tested against two reference outputs. This is to be expected: As lexical variability across input sentences grows, the number of possible ways to fuse them by machine as well by human also grows. The accuracy of match between the system output and the reference sentences largely depends on the features of the input sentences, rather than on the underlying fusion method.

6 Pairs of phrases that form an entry in the paraphrasing dictionary are compared as pairs of atomic entries.

Table 3
Node and edge similarity scores used by the alignment algorithm.

Category	Node Similarity	Category	Node Similarity
Identical words	1	Edges are subject-verb	0.03
Synonyms	1	Edges are verb-object	0.03
Paraphrases	0.5	Edges are same type	0.02
Other	−0.1	Other	0

is NP-hard.[7] The subtree flattening performed during the preprocessing stage aims to minimize the negative effect of the restriction on alignment granularity.

Another important property of our algorithm is that it produces a local alignment. Local alignment maps local regions with high similarity to each other rather than creating an overall optimal global alignment of the entire tree. This strategy is more meaningful when only partial meaning overlap is expected between input sentences, as in typical sentence fusion input. Only these high-similarity regions, which we call **intersection subtrees,** are included in the fusion sentence.

3.2 Fusion Lattice Computation

Fusion lattice computation is concerned with combining intersection subtrees. During this process, the system will remove phrases from a selected sentence, add phrases from other sentences, and replace words with the paraphrases that annotate each node. Among the many possible combinations of subtrees, we are interested only in those combinations which yield semantically sound sentences and do not distort the information presented in the input sentences. We cannot explore every possible combination, since the lack of semantic information in the trees prohibits us from assessing the quality of the resulting sentences. In fact, our early experimentation with generation from constituent phrases (e.g., NPs, VPs) demonstrated that it was difficult to ensure that semantically anomalous or ungrammatical sentences would not be generated. Instead, we select a combination already present in the input sentences as a basis and transform it into a fusion sentence by removing extraneous information and augmenting the fusion sentence with information from other sentences. The advantage of this strategy is that, when the initial sentence is semantically correct and the applied transformations aim to preserve semantic correctness, the resulting sentence is a semantically correct one. Our generation strategy is reminiscent of Robin and McKeown's (1996) earlier work on revision for summarization, although Robin and McKeown used a three-tiered representation of each sentence, including its semantics and its deep and surface syntax, all of which were used as triggers for revision.

The three steps of the fusion lattice computation are as follows: selection of the **basis tree,** augmentation of the tree with alternative verbalizations, and pruning of

7 The complexity of our algorithm is polynomial in the number of nodes. Let n_1 denote the number of nodes in the first tree, and n_2 denote the number of nodes in the second tree. We assume that the branching factor of a parse tree is bounded above by a constant. The function *NodeCompare* is evaluated only once on each node pair. Therefore, it is evaluated $n_1 \times n_2$ times totally. Each evaluation is computed in constant time, assuming that values of the function for node children are known. Since we use memoization, the total time of the procedure is $O(n_1 \times n_2)$.

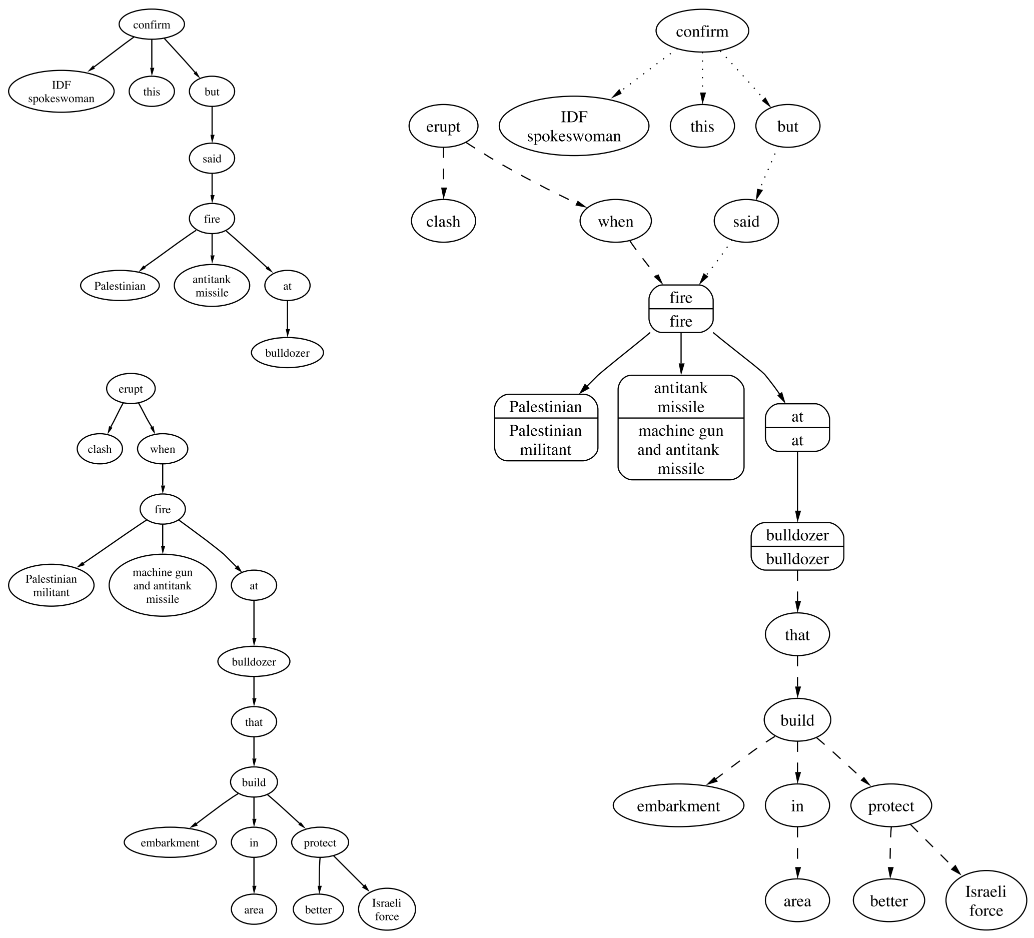


Figure 5
Two dependency trees and their alignment tree. Solid lines represent aligned edges. Dotted and dashed lines represent unaligned edges of the theme sentences.

the extraneous subtrees. Alignment is essential for all the steps. The selection of the basis tree is guided by the number of intersection subtrees it includes; in the best case, it contains all such subtrees. The basis tree is the centroid of the input sentences—the sentence which is the most similar to the other sentences in the input. Using the alignment-based similarity score described in Section 3.1.2, we identify the centroid by computing for each sentence the average similarity score between the sentence and the rest of the input sentences, then selecting the sentence with the highest score.

Next, we augment the basis tree with information present in the other input sentences. More specifically, we add alternative verbalizations for the nodes in the basis tree and the intersection subtrees which are not part of the basis tree. The alternative verbalizations are readily available from the pairwise alignments of the basis tree with other trees in the input computed in the previous section. For each node of the basis tree, we record all verbalizations from the nodes of the other input trees aligned with a given node. A verbalization can be a single word, or it can be a phrase, if a node represents a noun compound or a verb with a particle. An example of a fusion lattice, augmented

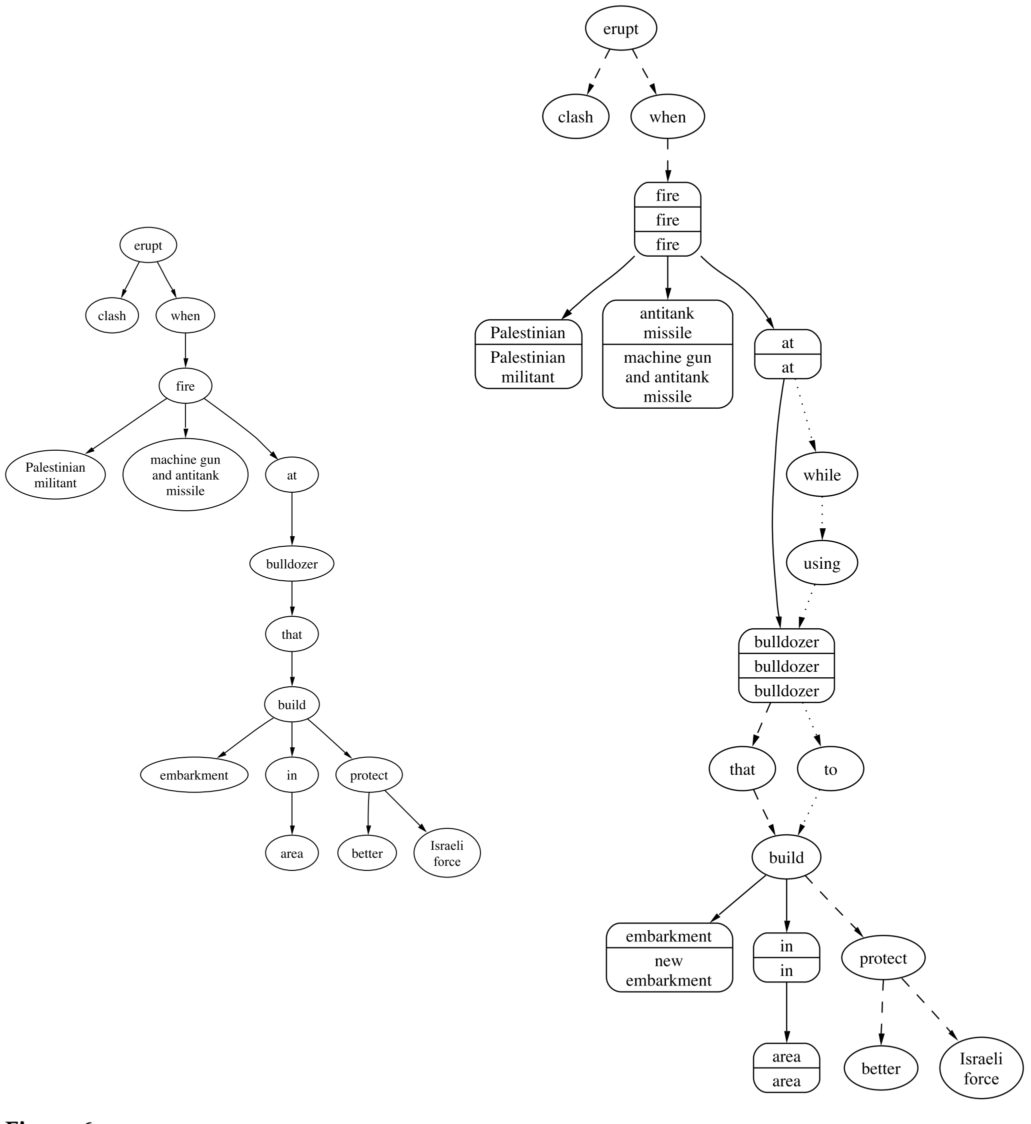


Figure 6
A basis lattice before and after augmentation. Solid lines represent aligned edges of the basis tree. Dashed lines represent unaligned edges of the basis tree, and dotted lines represent insertions from other theme sentences. Added subtrees correspond to sentences from Table 1.

with alternative verbalizations, is given in Figure 6. Even after this augmentation, the fusion lattice may not include all of the intersection subtrees. The main difficulty in subtree insertion is finding an acceptable placement; this is often determined by syntactic, semantic, and idiosyncratic knowledge. Therefore, we follow a conservative insertion policy. Among all the possible aligned sentences, we insert only subtrees whose top node aligns with one of the nodes in a basis tree.[8] We further constrain the insertion procedure by inserting only trees that appear in at least half of the sentences of a theme. These two

8 Our experimental results show that the algorithm inserts a sufficient amount of new subtrees despite this limitation.

constituent-level restrictions prevent the algorithm from generating overly long, unreadable sentences.[9]

Finally, subtrees which are not part of the intersection are pruned off the basis tree. However, removing all such subtrees may result in an ungrammatical or semantically flawed sentence; for example, we might create a sentence without a subject. This overpruning may happen if either the input to the fusion algorithm is noisy or the alignment has failed to recognize similar subtrees. Therefore, we perform a more conservative pruning, deleting only the self-contained components which can be removed without leaving ungrammatical sentences. As previously observed in the literature (Mani, Gates, and Bloedorn 1999; Jing and McKeown 2000), such components include a clause in the clause conjunction, relative clauses, and some elements within a clause (such as adverbs and prepositions). For example, this procedure transforms the lattice in Figure 6 into the pruned basis lattice shown in Figure 7 by deleting the clause *the clash erupted* and the verb phrase *to better protect Israeli forces.* These phrases are eliminated because they do not appear in the other sentences of the theme and at the same time their removal does not interfere with the well-formedness of the fusion sentence. Once these subtrees are removed, the fusion lattice construction is completed.

3.3 Generation

The final stage in sentence fusion is linearization of the fusion lattice. Sentence generation includes selection of a tree traversal order, lexical choice among available alternatives, and placement of auxiliaries, such as determiners. Our generation method utilizes information given in the input sentences to restrict the search space and then chooses among remaining alternatives using a language model derived from a large text collection. We first motivate the need for reordering and rephrasing, then discuss our implementation.

For the word-ordering task, we do not have to consider all the possible traversals, since the number of valid traversals is limited by ordering constraints encoded in the fusion lattice. However, the basis lattice does not uniquely determine the ordering: The placement of trees inserted in the basis lattice from other theme sentences is not restricted by the original basis tree. While the ordering of many sentence constituents is determined by their syntactic roles, some constituents, such as time, location and manner circumstantials, are free to move (Elhadad et al. 2001). Therefore, the algorithm still has to select an appropriate order from among different orders of the inserted trees.

The process so far produces a sentence that can be quite different from the extracted sentence; although the basis sentences provides guidance for the generation process, constituents may be removed, added in, or reordered. Wording can also be modified during this process. Although the selection of words and phrases which appear in the basis tree is a safe choice, enriching the fusion sentence with alternative verbalizations has several benefits. In applications such as summarization, in which the length of the produced sentence is a factor, a shorter alternative is desirable. This goal can be achieved by selecting the shortest paraphrase among available alternatives. Alternate verbalizations can also be used to replace anaphoric expressions, for instance,

9 Furthermore, the preference for shorter fusion sentences is further enforced during the linearization stage because our scoring function monotonically decreases with the length of a sentence.

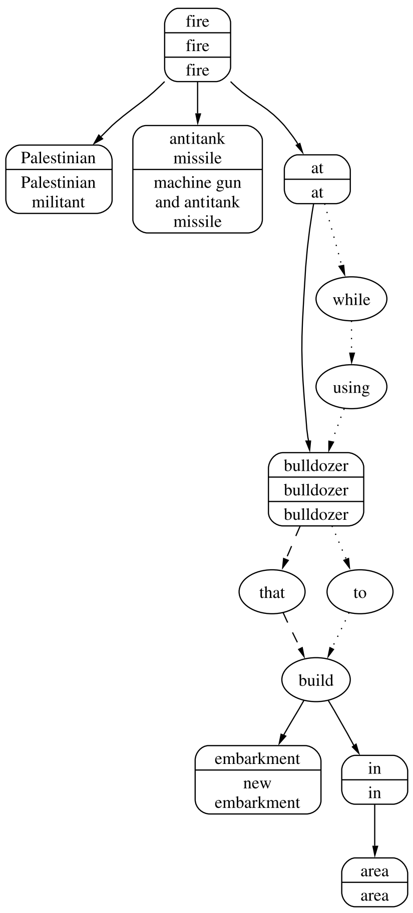


Figure 7
A pruned basis lattice.

when the basis tree contains a noun phrase with anaphoric expressions (e.g., *his visit*) and one of the other verbalizations is anaphora-free. Substitution of the latter for the anaphoric expression may increase the clarity of the produced sentence, since frequently the antecedent of the anaphoric expression is not present in a summary. Moreover, in some cases substitution is mandatory. As a result of subtree insertions and deletions, the words used in the basis tree may not be a good choice after the transformations, and the best verbalization might be achieved by using a paraphrase of them from another theme sentence. As an example, consider the case of two paraphrasing verbs with different subcategorization frames, such as *tell* and *say*. If the phrase *our correspondent* is removed from the sentence *Sharon told our correspondent that the elections were delayed . . .*, a replacement of the verb *told* with *said* yields a more readable sentence.

The task of auxiliary placement is alleviated by the presence of features stored in the input nodes. In most cases, aligned words stored in the same node have the same feature values, which uniquely determine an auxiliary selection and conjugation. However, in some cases, aligned words have different grammatical features, in which case the linearization algorithm needs to select among available alternatives.

Linearization of the fusion sentence involves the selection of the best phrasing and placement of auxiliaries as well as the determination of optimal ordering. Since we do not have sufficient semantic information to perform such selection, our algorithm is driven by corpus-derived knowledge. We generate all possible sentences[10] from the valid traversals of the fusion lattice and score their likelihood according to statistics derived from a corpus. This approach, originally proposed by Knight and Hatzivassiloglou (1995) and Langkilde and Knight (1998), is a standard method used in statistical generation. We trained a trigram model with Good–Turing smoothing over 60 megabytes of news articles collected by Newsblaster using the second version CMU–Cambridge Statistical Language Modeling toolkit (Clarkson and Rosenfeld 1997). The sentence with the lowest length-normalized entropy (the best score) is selected as the verbalization of the fusion lattice. Table 4 shows several verbalizations produced by our algorithm from the central tree in Figure 7. Here, we can see that the lowest-scoring sentence is both grammatical and concise.

Table 4 also illustrates that entropy-based scoring does not always correlate with the quality of the generated sentence. For example, the fifth sentence in Table 4—*Palestinians fired antitank missile at a bulldozer to build a new embankment in the area*—is not a well-formed sentence; however, our language model gave it a better score than its well-formed alternatives, the second and the third sentences (see Section 4 for further discussion). Despite these shortcomings, we preferred entropy-based scoring to symbolic linearization. In the next section, we motivate our choice.

3.3.1 Statistical versus Symbolic Linearization. In the previous version of the system (Barzilay, McKeown, and Elhadad 1999), we performed linearization of a fusion dependency structure using the language generator FUF/SURGE (Elhadad and Robin 1996). As a large-scale linearizer used in many traditional semantic-to-text generation systems, FUF/SURGE could be an appealing solution to the task of surface realization. Because the input structure and the requirements on the linearizer are quite different in text-to-text generation, we had to design rules for mapping between dependency structures produced by the fusion component and FUF/SURGE input. For instance, FUF/SURGE requires that the input contain a semantic role for prepositional phrases, such as *manner*, *purpose*, or *location*, which is not present in our dependency representation; thus we had to augment the dependency representation with this information. In the case of inaccurate prediction or the lack of relevant semantic information, the linearizer scrambles the order of sentence constituents, selects wrong prepositions, or even fails to generate an output. Another feature of the FUF/SURGE system that negatively influences system performance is its limited ability to reuse phrases readily available in the input, instead of generating every phrase from scratch. This makes the generation process more complex and thus prone to error.

While the initial experiments conducted on a set of manually constructed themes seemed promising, the system performance deteriorated significantly when it was applied to automatically constructed themes. Our experience led us to believe that transformation of an arbitrary sentence into a FUF/SURGE input representation is similar in its complexity to semantic parsing, a challenging problem in its own right. Rather than refining the mapping mechanism, we modified MultiGen to use a statis-

10 Because of the efficiency constraints imposed by Newsblaster, we sample only a subset of 20,000 paths. The sample is selected randomly.

Table 4
Alternative linearizations of the fusion lattice with corresponding entropy values.

Sentence	Entropy
Palestinians fired an antitank missile at a bulldozer.	4.25
Palestinian militants fired machine guns and antitank missiles at a bulldozer.	5.86
Palestinian militants fired machine guns and antitank missiles at a bulldozer that was building an embankment in the area.	6.22
Palestinians fired antitank missiles at while using a bulldozer.	7.04
Palestinians fired antitank missile at a bulldozer to build a new embankment in the area.	5.46

tical linearization component, which handles uncertainty and noise in the input in a more robust way.

4. Sentence Fusion Evaluation

In our previous work, we evaluated the overall summarization strategy of MultiGen in multiple experiments, including comparisons with human-written summaries in the Document Understanding Conference (DUC)[11] evaluation (McKeown et al. 2001; McKeown et al. 2002) and quality assessment in the context of a particular information access task in the Newsblaster framework (McKeown et al. 2002).

In this article, we aim to evaluate the sentence fusion algorithm in isolation from other system components; we analyze the algorithm performance in terms of content selection and the grammaticality of the produced sentences. We first present our evaluation methodology (Section 4.1), then we describe our data (Section 4.2), the results (Section 4.3), and our analysis of them (Section 4.4).

4.1 Methods

4.1.1 Construction of a Reference Sentence. We evaluated content selection by comparing an automatically generated sentence with a reference sentence. The reference sentence was produced by a human (hereafter the RFA), who was instructed to generate a sentence conveying information common to many sentences in a theme. The RFA was not familiar with the fusion algorithm. The RFA was provided with the list of theme sentences; the original documents were not included. The instructions given to the RFA included several examples of themes with fusion sentences generated by the authors. Even though the RFA was not instructed to use phrases from input sentences, the sentences presented as examples reused many phrases from the input sentences. We believe that phrase reuse elucidates the connection between input sentences and a resulting fusion sentence. Two examples of themes, reference sentences, and system outputs are shown in Table 5.

4.1.2 Data Selection. We wanted to test the performance of the fusion component on automatically computed inputs which reflect the accuracy of the existing preprocessing tools. For this reason, the test data were selected randomly from material collected by Newsblaster. To remove themes irrelevant for fusion evaluation, we introduced two

11 DUC is a community-based evaluation of summarization systems organized by DARPA.

Table 5
Examples from the test set. Each example contains a theme, a reference sentence generated by the RFA, and a sentence generated by the system. Subscripts in the system-generated sentence represent a theme sentence from which a word was extracted.

#1	The forest is about 70 miles west of Portland.
#2	Their bodies were found Saturday in a remote part of Tillamook State Forest, about 40 miles west of Portland.
#3	Elk hunters found their bodies Saturday in the Tillamook State Forest, about 60 miles west of the family's hometown of Portland.
#4	The area where the bodies were found is in a mountainous forest about 70 miles west of Portland.
Reference	The bodies were found Saturday in the forest area west of Portland.
System	The bodies_4 were found_2 Saturday_2 in_3 the Tillamook_3 State_3 Forest_3 west_2 of_2 Portland_2.
#1	Four people including an Islamic cleric have been detained in Pakistan after a fatal attack on a church on Christmas Day.
#2	Police detained six people on Thursday following a grenade attack on a church that killed three girls and wounded 13 people on Christmas Day.
#3	A grenade attack on a Protestant church in Islamabad killed five people, including a U.S. Embassy employee and her 17-year-old daughter.
Reference	A grenade attack on a church killed several people.
System	A_3 grenade_3 attack_3 on_3 a Protestant_3 church_3 in_3 Islamabad_3 killed_3 six_2 people_2.

additional filters. First, we excluded themes that contained identical or nearly identical sentences (with cosine similarity higher than 0.8). When processing such sentences, our algorithm reduces to sentence extraction, which does not allow us to evaluate the generation abilities of our algorithm. Second, themes for which the RFA was unable to create a reference sentence were also removed from the test set. As mentioned above, Simfinder does not always produce accurate themes,[12] and therefore, the RFA could choose not to generate a reference sentence if the theme sentences had too little in common. An example of a theme for which no sentence was generated is shown in Table 6. As a result of this filtering, 34% of the sentences were removed.

4.1.3 Baselines. In addition to the system-generated sentence, we also included in the evaluation a fusion sentence generated by another human (hereafter, RFA2) and three baselines. (Following the DUC terminology, we refer to the baselines, our system, and the RFA2 as **peers.**) The first baseline is the shortest sentence among the theme sentences, which is obviously grammatical, and it also has a good chance of being representative of common topics conveyed in the input. The second baseline is produced by a simplification of our algorithm, where paraphrase information is omitted during the alignment process. This baseline is included to capture the contribution of paraphrase information to the performance of the fusion algorithm. The third baseline consists of the basis sentence. The comparison with this baseline reveals the contribution of the insertion and deletion stages in the fusion algorithm. The comparison against an RFA2 sentence provides an upper bound on the performance of the system and baselines. In addition, this comparison sheds light on the human agreement on this task.

12 To mitigate the effects of Simfinder noise in MultiGen, we induced a similarity threshold on input trees—trees which are not similar to the basis tree are not used in the fusion process.

Table 6
An example of noisy Simfinder output.

The shares have fallen 60% this year.
They said Qwest was forcing them to exchange their bonds at a fraction of face value—between 52.5% and 82.5%, depending on the bond—or else fall lower in the pecking order for repayment in case Qwest went broke.
Qwest had offered to exchange up to $12.9 billion of the old bonds, which carried interest rates between 5.875% and 7.9%.
The new debt carries rates between 13% and 14%.
Their yield fell to about 15.22% from 15.98%.

4.1.4 Comparison against the Reference Sentence. One judge was given a peer sentence along with the corresponding reference sentence. The judge also had access to the original theme from which these sentences were generated. The order of the presentation was randomized across themes and peer systems. Reference and peer sentences were divided into clauses by the authors. The judges assessed overlap on the clause level between reference and peer sentences. The wording of the instructions was inspired by the DUC instructions for clause comparison. For each clause in the reference sentence, the judge decided whether the meaning of a corresponding clause was conveyed in a peer sentence. In addition to 0 score for *no overlap* and 1 for *full overlap,* this framework allows for *partial overlap* with a score of 0.5. From the overlap data, we computed weighted recall and precision based on fractional count (Hatzivassiloglou and McKeown 1993). Recall is a ratio of weighted clause overlap between a peer and a reference sentence, and the number of clauses in a reference sentence. Precision is a ratio of weighted clause overlap between a peer and a reference sentence, and the number of clauses in a peer sentence.

4.1.5 Grammaticality Assessment. Grammaticality was rated in three categories: *grammatical* (3), *partially grammatical* (2), and *not grammatical* (1). The judge was instructed to rate a sentence in the *grammatical* category if it contained no grammatical mistakes. *Partially grammatical* included sentences that contained at most one mistake in agreement, articles, and tense realization. The *not grammatical* category included sentences that were corrupted by multiple mistakes of the former type, by erroneous component order or by the omission of important components (e.g., subject).

Punctuation is one issue in assessing grammaticality. Improper placement of punctuation is a limitation of our implementation of the sentence fusion algorithm that we are well aware of.[13] Therefore, in our grammaticality evaluation (following the DUC procedure), the judge was asked to ignore punctuation.

4.2 Data

To evaluate our sentence fusion algorithm, we selected 100 themes following the procedure described in the previous section. Each set varied from three to seven sentences,

13 We were unable to develop a set of rules which works in most cases. Punctuation placement is determined by a variety of features; considering all possible interactions of these features is hard. We believe that corpus-based algorithms for automatic restoration of punctuation developed for speech recognition applications (Beeferman, Berger, and Lafferty 1998; Shieber and Tao 2003) could help in our task, and we plan to experiment with them in the future.

with 4.22 sentences on average. The generated fusion sentences consisted of 1.91 clauses on average. None of the sentences in the test set were fully extracted; on average, each sentence fused fragments from 2.14 theme sentences. Out of 100 sentence, 57 sentences produced by the algorithm combined phrases from several sentences, while the rest of the sentences comprised subsequences of one of the theme sentences. (Note that compression is different from sentence extraction.) We included these sentences in the evaluation, because they reflect both content selection and realization capacities of the algorithm.

Table 5 shows two sentences from the test corpus, along with input sentences. The examples are chosen so as to reflect good- and bad-performance cases. Note that the first example results in inclusion of the essential information (the fact that bodies were found, along with time and place) and leaves out details (that it was a remote location or how many miles west it was, a piece of information that is in dispute in any case). The problematic example incorrectly selects the number of people killed as six, even though this number is not repeated and different numbers are referred to in the text. This mistake is caused by a noisy entry in our paraphrasing dictionary which erroneously identifies "five" and "six" as paraphrases of each other.

4.3 Results

Table 7 shows the length ratio, precision, recall, *F*-measure, and grammaticality score for each algorithm. The length ratio of a sentence was computed as the ratio of its output length to the average length of the theme input sentences.

4.4 Discussion

The results in Table 7 demonstrate that sentences manually generated by the second human participant (RFA2) not only are the shortest, but are also closest to the reference sentence in terms of selected information. The tight connection[14] between sentences generated by the RFAs establishes a high upper bound for the fusion task. While neither our system nor the baselines were able to reach this level of performance, the fusion algorithm clearly outperforms all the baselines in terms of content selection, at a reasonable level of compression. The performance of baseline 1 and baseline 2 demonstrates that neither the shortest sentence nor the basis sentence is an adequate substitution for fusion in terms of content selection. The gap in recall between our system and baseline 3 confirms our hypothesis about the importance of paraphrasing information for the fusion process. Omission of paraphrases causes an 8% drop in recall due to the inability to match equivalent phrases with different wording.

Table 7 also reveals a downside of the fusion algorithm: Automatically generated sentences contain grammatical errors, unlike fully extracted, human-written sentences. Given the high sensitivity of humans to processing ungrammatical sentences, one has to consider the benefits of flexible information selection against the decrease in readability of the generated sentences. Sentence fusion may not be a worthy direction to pursue if low grammaticality is intrinsic to the algorithm and its correction requires

14 We cannot apply kappa statistics (Siegel and Castellan 1988) for measuring agreement in the content selection task since the event space is not well-defined. This prevents us from computing the probability of random agreement.

Table 7
Evaluation results for a human-crafted fusion sentence (RFA2), our system output, the shortest sentence in the theme (baseline 1), the basis sentence (baseline 2), and a simplified version of our algorithm without paraphrasing information (baseline 3).

Peer	Length Ratio	Precision	Recall	*F*-measure	Grammaticality
RFA2	54%	98%	94%	96%	2.9
Fusion	78%	65%	72%	68%	2.3
Baseline 1	69%	52%	38%	44%	3.0
Baseline 2	111%	41%	67%	51%	3.0
Baseline 3	73%	63%	64%	63%	2.4

knowledge which cannot be automatically acquired. In the remainder of the section, we show that this is not the case. Our manual analysis of generated sentences revealed that most of the grammatical mistakes are caused by the linearization component, or more specifically, by suboptimal scoring of the language model. Language modeling is an active area of research, and we believe that advances in this direction will be able to dramatically boost the linearization capacity of our algorithm.

4.4.1 Error Analysis. In this section, we discuss the results of our manual analysis of mistakes in content selection and surface realization. Note that in some cases multiple errors are entwined in one sentence, which makes it hard to distinguish between a sequence of independent mistakes and a cause-and-effect chain. Therefore, the presented counts should be viewed as approximations, rather than precise numbers.

We start with the analysis of the test set and continue with the description of some interesting mistakes that we encountered during system development.

Mistakes in Content Selection. Most of the mistakes in content selection can be attributed to problems with alignment. In most cases (17), erroneous alignments missed relevant word mappings as a result of the lack of a corresponding entry in our paraphrasing resources. At the same time, mapping of unrelated words (as shown in Table 5) was quite rare (two cases). This performance level is quite predictable given the accuracy of an automatically constructed dictionary and limited coverage of WordNet. Even in the presence of accurate lexical information, the algorithm occasionally produced suboptimal alignments (four cases) because of the simplicity of our weighting scheme, which supports limited forms of mapping typology and also uses manually assigned weights.

Another source of errors (two cases) was the algorithm's inability to handle many-to-many alignments. Namely, two trees conveying the same meaning may not be decomposable into the node-level mappings which our algorithm aims to compute. For example, the mapping between the sentences in Table 8 expressed by the rule X *denied claims by* Y $\leftrightarrow$ X *said that* Y*'s claim was untrue* cannot be decomposed into smaller matching units. At least two mistakes resulted from noisy preprocessing (tokenization and parsing).

In addition to alignment, overcutting during lattice pruning caused the omission of three clauses that were present in the corresponding reference sentences. The sentence *Conservatives were cheering language* is an example of an incomplete sentence derived from the following input sentence: *Conservatives were cheering language in the final version*

Table 8
A pair of sentences which cannot be fully decomposed.

Syria denied claims by Israeli Prime Minister Ariel Sharon ...
The Syrian spokesman said that Sharon's claim was untrue ...

that ensures that one-third of all funds for prevention programs be used to promote abstinence. The omission of a relative clause was possible because some sentences in the input theme contained the noun *language* without any relative clauses.

Mistakes in Surface Realization. Grammatical mistakes included incorrect selection of determiners, erroneous word ordering, omission of essential sentence constituents, and incorrect realization of negation constructions and tense. These mistakes (42) originated during linearization of the lattice and were caused either by incompleteness of the linearizer or by suboptimal scoring of the language model. Mistakes of the first type are caused by missing rules for generating auxiliaries given node features. An example of this phenomenon is the sentence *The coalition to have play a central role,* which verbalizes the verb construction *will have to play* incorrectly. Our linearizer lacks the completeness of existing application-independent linearizers, such as the unification-based FUF/SURGE (Elhadad and Robin 1996) and the probabilistic Fergus (Bangalore and Rambow 2000). Unfortunately, we were unable to reuse any of the existing large-scale linearizers because of significant structural differences between input expected by these linearizers and the format of a fusion lattice. We are currently working on adapting Fergus for the sentence fusion task.

Mistakes related to suboptimal scoring were the most common (33 out of 42); in these cases, a language model selected ill-formed sentences, assigning a worse score to a better sentence. The sentence *The diplomats were given to leave the country in 10 days* illustrates a suboptimal linearization of the fusion lattice. The correct linearizations—*The diplomats were given 10 days to leave the country* and *The diplomats were ordered to leave the country in 10 days*—were present in the fusion lattice, but the language model picked the incorrect verbalization. We found that in 27 cases the optimal verbalizations (in the authors' view) were ranked below the top-10 sentences ranked by the language model. We believe that more powerful language models that incorporate linguistic knowledge (such as syntax-based models) can improve the quality of generated sentences.

4.4.2 Further Analysis. In addition to analyzing errors found in this particular study, we also regularly track the quality of generated summaries on Newsblaster's Web page. We have noted a number of interesting errors that crop up from time to time that seem to require information about the full syntactic parse, semantics, or even discourse. Consider, for example, the last sentence from a summary entitled *Estrogen-Progestin Supplements Now Linked to Dementia,* which is shown in Table 9. This sentence was created by sentence fusion and clearly, there is a problem. Certainly, there was a study *finding the risk of dementia in women who took one type of combined hormone pill,* but it was not the government study which was abruptly halted last summer. In looking at the two sentences from which this summary sentence was drawn, we can see that there is a good amount of overlap between the two, but the component does not have enough information about the referents of the different terms to know that two different

Table 9
An example of wrong reference selection. Subscripts in the generated sentence indicate the theme sentence from which the words were extracted.

#1	Last summer, a government study was abruptly halted after finding an increased risk of breast cancer, heart attacks, and strokes in women who took one type of combined hormone pill.
#2	The most common form of hormone replacement therapy, already linked to breast cancer, stroke, and heart disease, does not improve mental functioning as some earlier studies suggested and may increase the risk of dementia, researchers said on Tuesday.
System	Last_1 summer_1 a_1 government_1 study_1 abruptly_1 was_1 halted_1 after_1 finding_1 the_2 risk_2 of_2 dementia_2 in_1 women_1 who_1 took_1 one_1 type_1 of_1 combined_1 hormone_1 pill_1.

studies are involved and that fusion should not take place. One topic of our future work (Section 6) is the problem of reference and summarization.

Another example is shown in Table 10. Here again, the problem is reference. The first error is in the references to *the segments*. The two uses of *segments* in the first source document sentence do not refer to the same entity and thus, when the modifier is dropped, we get an anomaly. The second, more unusual problem is in the equation of *Clinton/Dole, Dole/Clinton*, and *Clinton and Dole*.

5. Related Work

5.1 Text-to-Text Generation

Unlike traditional concept-to-text generation approaches, text-to-text generation methods take text as input and transform it into a new text satisfying some constraints (e.g., length or level of sophistication). In addition to sentence fusion, compression algorithms (Chandrasekar, Doran, and Bangalore 1996; Grefenstette 1998; Mani, Gates, and Bloedorn 1999; Knight and Marcu 2002; Jing and McKeown 2000; Reizler et al. 2003) and methods for expansion of a multiparallel corpus (Pang, Knight, and Marcu 2003) are other instances of such methods.

Compression methods have been developed for single-document summarization, and they aim to reduce a sentence by eliminating constituents which are not crucial for understanding the sentence and not salient enough to include in the summary. These approaches are based on the observation that the "importance" of a sentence constituent can often be determined based on shallow features, such as its syntactic role and the words it contains. For example, in many cases a relative clause that is

Table 10
An example of incorrect reference selection. Subscripts in the generated sentence indicate the theme sentence from which the words were extracted.

#1	The segments will revive the "Point-Counterpoint" segments popular until they stopped airing in 1979, but will instead be called "Clinton/Dole" one week and "Dole/Clinton" the next week.
#2	Clinton and Dole have signed up to do the segment for the next 10 weeks, Hewitt said.
#3	The segments will be called "Clinton Dole" one week and "Dole Clinton" the next.
System	The_1 segments_1 will_1 revive_1 the_3 segments_3 until_1 they_1 stopped_1 airing_1 in_1 1979_1 but_1 instead_1 will_1 be_1 called_1 Clinton_2 and_2 Dole_2.

peripheral to the central point of the document can be removed from a sentence without significantly distorting its meaning. While earlier approaches for text compression were based on symbolic reduction rules (Grefenstette 1998; Mani, Gates, and Bloedorn 1999), more recent approaches use an aligned corpus of documents and their human written summaries to determine which constituents can be reduced (Knight and Marcu 2002; Jing and McKeown 2000; Reizler et al. 2003). The summary sentences, which have been manually compressed, are aligned with the original sentences from which they were drawn.

Knight and Marcu (2000) treat reduction as a translation process using a noisy-channel model (Brown et al. 1993). In this model, a short (compressed) string is treated as a source, and additions to this string are considered to be noise. The probability of a source string s is computed by combining a standard probabilistic context-free grammar score, which is derived from the grammar rules that yielded tree s, and a word-bigram score, computed over the leaves of the tree. The stochastic channel model creates a large tree t from a smaller tree s by choosing an extension template for each node based on the labels of the node and its children. In the decoding stage, the system searches for the short string s that maximizes $P(s|t)$, which (for fixed t) is equivalent to maximizing $P(s) \times P(t|s)$.

While this approach exploits only syntactic and lexical information, Jing and McKeown (2000) also rely on cohesion information, derived from word distribution in a text: Phrases that are linked to a local context are retained, while phrases that have no such links are dropped. Another difference between these two methods is the extensive use of knowledge resources in the latter. For example, a lexicon is used to identify which components of the sentence are obligatory to keep it grammatically correct. The corpus in this approach is used to estimate the degree to which a fragment is extraneous and can be omitted from a summary. A phrase is removed only if it is not grammatically obligatory, is not linked to a local context, and has a reasonable probability of being removed by humans. In addition to reducing the original sentences, Jing and McKeown (2000) use a number of manually compiled rules to aggregate reduced sentences; for example, reduced clauses might be conjoined with *and*.

Sentence fusion exhibits similarities with compression algorithms in the ways in which it copes with the lack of semantic data in the generation process, relying on shallow analysis of the input and statistics derived from a corpus. Clearly, the difference in the nature of both tasks and in the type of input they expect (single sentence versus multiple sentences) dictates the use of different methods. Having multiple sentences in the input poses new challenges—such as a need for sentence comparison—but at the same time it opens up new possibilities for generation. While the output of existing compression algorithms is always a substring of the original sentence, sentence fusion may generate a new sentence which is not a substring of any of the input sentences. This is achieved by arranging fragments of several input sentences into one sentence.

The only other text-to-text generation approach able to produce new utterances is that of Pang, Knight, and Marcu (2003). Their method operates over multiple English translations of the same foreign sentence and is intended to generate novel paraphrases of the input sentences. Like sentence fusion, their method aligns parse trees of the input sentences and then uses a language model to linearize the derived lattice. The main difference between the two methods is in the type of the alignment: Our algorithm performs local alignment, while the algorithm of Pang, Knight, and Marcu (2003) performs global alignment. The differences in alignment are caused by differences in input: Pang, Knight, and Marcu's method expects semantically equivalent sentences, while our algorithm operates over sentences with only partial meaning overlap. The presence of deletions and insertions in input sentences makes alignment of comparable trees a new and particularly significant challenge.

5.2 Computation of an Agreement Tree

The alignment method described in Section 3 falls into a class of tree comparison algorithms extensively studied in theoretical computer science (Sankoff 1975; Finden and Gordon 1985; Amir and Keselman 1994; Farach, Przytycka, and Thorup 1995) and widely applied in many areas of computer science, primarily computational biology (Gusfield 1997). These algorithms aim to find an overlap subtree that captures structural commonality across a set of related trees. A typical tree similarity measure considers the proximity, at both the node and the edge levels, between input trees. In addition, some algorithms constrain the topology of the resulting alignment based on the domain-specific knowledge. These constraints not only narrow the search space but also increase the robustness of the algorithm in the presence of a weak similarity function.

In the NLP context, this class of algorithms has been used previously in example-based machine translation, in which the goal is to find an optimal alignment between the source and the target sentences (Meyers, Yangarber, and Grishman 1996). The algorithm operates over pairs of parallel sentences, where each sentence is represented by a structure-sharing forest of plausible syntactic trees. The similarity function is driven by lexical mapping between tree nodes and is derived from a bilingual dictionary. The search procedure is greedy and is subject to a number of constraints needed for alignment of parallel sentences.

This algorithm has several features in common with our method: It operates over syntactic dependency representations and employs recursive computation to find an optimal solution. However, our method is different in two key aspects. First, our algorithm looks for **local regions** with high similarity in nonparallel data, rather than for full alignment, expected in the case of parallel trees. The change in optimization criteria introduces differences in the similarity measure—specifically, the relaxation of certain constraints—and the search procedure, which in our work uses dynamic programming. Second, our method is an instance of a **multisequence** alignment,[15] in contrast to the pairwise alignment described in Meyers, Yangarber, and Grishman (1996). Combining evidence from multiple trees is an essential step of our algorithm—pairwise comparison of nonparallel trees may not provide enough information regarding their underlying correspondences. In fact, previous applications of multisequence alignment have been shown to increase the accuracy of the comparison in other NLP tasks (Barzilay and Lee 2002; Bangalore, Murdock, and Riccardi 2002; Lacatusu, Maiorano, and Harabagiu 2004); unlike our work these approaches operate on strings, not trees, and with the exception of (Lacatusu, Maiorano, and Harabagiu 2004), they apply alignment to parallel data, not comparable texts.

6. Conclusions and Future Work

In this article, we have presented sentence fusion, a novel method for text-to-text generation which, given a set of similar sentences, produces a new sentence containing the information common to most sentences. Unlike traditional generation methods,

15 See Gusfield (1997) and Durbin et al. (1998) for an overview of multisequence alignment.

sentence fusion does not require an elaborate semantic representation of the input but instead relies on the shallow linguistic representation automatically derived from the input documents and knowledge acquired from a large text corpus. Generation is performed by reusing and altering phrases from input sentences.

As the evaluation described in Section 4 shows, our method accurately identifies common information and in most cases generates a well-formed fusion sentence. Our algorithm outperforms the shortest-sentence baseline in terms of content selection, without a significant drop in grammaticality. We also show that augmenting the fusion process with paraphrasing knowledge improves the output by both measures. However, there is still a gap between the performance of our system and human performance.

An important goal for future work on sentence fusion is to increase the flexibility of content selection and realization. We believe that the process of aligning theme sentences can be greatly improved by having the system learn the similarity function, instead of using manually assigned weights. An interesting question is how such a similarity function can be induced in an unsupervised fashion. In addition, we can improve the flexibility of the fusion algorithm by using a more powerful language model. Recent research (Daume et al. 2002) has show that syntax-based language models are more suitable for language generation tasks; the study of such models is a promising direction to explore.

An important feature of the sentence fusion algorithm is its ability to generate multiple verbalizations of a given fusion lattice. In our implementation, this property is utilized only to produce grammatical texts in the changed syntactic context, but it can also be used to increase coherence of the text at the discourse level by taking context into account. In our current system, each sentence is generated in isolation, independently from what is said before and what will be said after. Clear evidence of the limitation of this approach is found in the selection of referring expressions. For example, all summary sentences may contain the full description of a named entity (e.g., *President of Columbia University Lee Bollinger*), while the use of shorter descriptions such as *Bollinger* or anaphoric expressions in some summary sentences would increase the summary's readability (Schiffman, Nenkova, and McKeown 2002; Nenkova and McKeown 2003). These constraints can be incorporated into the sentence fusion algorithm, since our alignment-based representation of themes often contains several alternative descriptions of the same object.

Beyond the problem of referring-expression generation, we found that by selecting appropriate paraphrases of each summary sentence, we can significantly improve the coherence of an output summary. An important research direction for future work is to develop a probabilistic text model that can capture properties of well-formed texts, just as a language model captures properties of sentence grammaticality. Ideally, such a model would be able to discriminate between cohesive fluent texts and ill-formed texts, guiding the selection of sentence paraphrases to achieve an optimal sentence sequence.

Appendix. Alignment Pseudocode

```
Function: EdgeSim($edge_1$, $edge_2$)
Returns: The similarity score of two input edges based on their type
begin
    if type_of(edge$_1$) = type_of(edge$_2$) = 'subject-verb' then
        return SUBJECT_VERB_SCORE ;
    else if $type\_of(\text{edge}_1) = type\_of(\text{edge}_2) =$ 'object-verb' then
        return OBJECT_VERB_SCORE ;
    else
        return EDGE_DEFAULT ;
    end
end
```

Function: `NodeSim`(*node*$_1$, *node*$_2$)
Returns: The similarity score of two words or flattened noun phrases based on their semantic relation

```
begin
    if is_phrase ($node_1$) or is_phrase ($node_2$) then
        return IDENTITY_SCORE * $\frac{|intersection(node_1, node_2)|}{max(|node_1|, |node_2|)}$ ;
    else if $node_1 = node_2$ then
        return IDENTITY_SCORE ;
    else if is_synonym ($node_1$, $node_2$) then
        return SYNONYMY_SCORE ;
    else
        return NODE_DEFAULT ;
    end
end
```

All the comparison functions employ memoization, implemented by hash table wrappers.

Function: `MapChildren`(*tree*$_1$, *tree*$_2$) ***memoized***
Returns: Given two dependency trees, `MapChildren` finds the optimal alignment of tree children. The function returns the score of the alignment and the mapping itself.

```
begin
    /*Generate all legitimate mappings between the children on tree$_1$ and tree$_2$ */
    all-maps ← GenerateAllPermutations ($tree_1$, $tree_2$) ;
    best ← $\langle -1, \text{void} \rangle$ ;
    /*Compute the score of each mapping, and select the one with the highest score */
    foreach map in all-maps do
        res ← 0 ;
        foreach $\langle s_1, s_2 \rangle$ in map do
            res ← res + EdgeSim (edge (tree$_1$.top, $s_1$), edge (tree$_2$.top, $s_2$))
                      + Sim (subtree (tree$_1$, $s_1$), (subtree (tree$_2$, $s_2$));
        end
        if res > best.score then
            best.score ← res ;
            best.map ← map ;
        end
    end
    return best
end
```

Function: `NodeCompare`(*tree*$_1$, *tree*$_2$) ***memoized***
Returns: Given two dependency trees, `NodeCompare` finds their optimal alignment that

```
        maps two top nodes of the tree one to another. The function returns the score
        of the alignment and the mapping itself.
begin
    node-sim ← NodeSim(tree_1.top, tree_2.top) ;
    /*If one of the trees is of height one, return the NodeSim score between two tops   */
    if is_leaf(tree_1) or is_leaf(tree_2) then
        return ⟨node-sim, ⟨tree_1, tree_2⟩⟩ ;
    else
        /*Find an optimal alignment of the children nodes                               */
        res ← MapChildren(tree_1, tree_2) ;
        /*The alignment score is computed as a sum of the similarity of top nodes and
          the score of the optimal alignment of node. The tree alignment is assembled
          by adding a pair of top nodes to the optimal alignment of their children.     */
        return ⟨node-sim + res.score, ⟨tree_1.top, tree_2.top⟩ ∪ res.map⟩ ;
    end
end
```

```
Function: NodeCompare(tree_1, tree_2) memoized
Returns: Given two dependency trees, Sim finds their optimal alignment. The function
        returns the score of the alignment and the mapping itself.
begin
    best ← ⟨−1, void⟩ ;
    /*find an optimal alignment between one of the children of tree_1 and tree_2        */
    foreach s in tree_1.children do
        res ← Sim(s, tree_2) ;
        if res.score > best.score then best ← res ;
    end
    /*find an optimal alignment between one of the children of tree_1 and tree_2        */
    foreach s in tree_2.children do
        res ← Sim(tree_1, s) ;
        if res.score > best.score then best ← res ;
    end
    /*find an optimal alignment that include the two top nodes                          */
    res ← NodeCompare(tree_1, tree_2) ;
    if res.score > best.score then best ← res ;
    return best
end
```

Acknowledgments
We are grateful to Eli Barzilay, Michael Collins, Noèmie Elhadad, Julia Hirschberg, Mirella Lapata, Lillian Lee, Smaranda Muresan, and the anonymous reviewers for helpful comments and conversations. Portions of this work were completed while the first author was a graduate student at Columbia University. This article is based upon work supported in part by the National Science Foundation under grant IIS-0448168, DARPA grant N66001-00-1-8919 and a Louis Morin scholarship. Any opinions, findings, and conclusions or recommendations expressed above are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

Amir, Amihood and Dmitry Keselman. 1994. Maximum agreement subtree in a set of evolutionary trees—Metrics and efficient algorithms. In *Proceedings of FOCS*, pages 758–769, Santa Fe, NM.

Bangalore, Srinivas, Vanessa Murdock, and Giuseppe Riccardi. 2002. Bootstrapping bilingual data using consensus translation for a multilingual instant messaging system. In *International Conference on Computational Linguistics (COLING 2002)*, Tapei, Taiwan.

Bangalore, Srinivas and Owen Rambow. 2000. Exploiting a probabilistic hierarchical model for generation. In *Proceedings of COLING*, pages 42–48, Saarbruken, Germany.

Banko, Michele and Lucy Vanderwende. 2004. Using *n*-grams to understand the nature of summaries. In *Proceedings of HLT-NAACL*, pages 1– 4, Boston, MA.

Barzilay, Regina. 2003. *Information Fusion for Multi-document Summarization: Paraphrasing and Generation*. Ph.D. thesis, Columbia University.

Barzilay, Regina and Michael Elhadad. 1997. Using lexical chains for text summarization. In *Proceedings of the ACL Workshop on Intelligent Scalable Text Summarization*, pages 10–17, Madrid.

Barzilay, Regina, Noemie Elhadad, and Kathleen McKeown. 2002. Inferring strategies for sentence ordering in multi-document news summarization. *Journal of Artificial Intelligence Research*, 17:35–55.

Barzilay, Regina and Lillian Lee. 2002. Bootstrapping lexical choice via multiple-sequence alignment. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 164–171, Philadelphia, PA.

Barzilay, Regina and Kathleen McKeown. 2001. Extracting paraphrases from a parallel corpus. In *Proceedings of the ACL/EACL*, pages 50–57, Toulouse, France.

Barzilay, Regina, Kathleen McKeown, and Michael Elhadad. 1999. Information fusion in the context of multi-document summarization. In *Proceedings of the ACL*, pages 550–557, College Park, MD.

Beeferman, Doug, Adam Berger, and John Lafferty. 1998. Cyberpunc: A lightweight punctuation annotation system for speech. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 689–692, Seattle, WA.

Borko, Harold and Charles Bernier. 1975. *Abstracting Concepts and Methods.* Academic Press, New York.

Brown, Peter F., Stephen Della Pietra, Vincent Della Pietra, and Robert Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

Carbonell, Jaime and Jade Goldstein. 1998. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of SIGIR*, pages 335–336, Melbourne, Australia.

Chandrasekar, Raman, Christine Doran, and Srinivas Bangalore. 1996. Motivations and methods for text simplification. In *International Conference on Computational Linguistics (COLING 1996)*, pages 1041–1044, Copenhagen, Denmark.

Chu-Carroll, Jennifer, Krzysztof Czuba, John Prager, and Abraham Ittycheriah. 2003. In question answering: Two heads are better than one. In *Proceedings of HLT-NAACL*, pages 24–31, Edmonton, Alberta.

Clarke, Charles, Gordon Cormack, and Thomas Lynam. 2001. Exploiting redundancy in question answering. In *Proceedings of SIGIR*, pages 358–365, New Orleans, LA.

Clarkson, Philip and R. Rosenfeld. 1997. Statistical language modeling using the CMU-Cambridge toolkit. In *Proceedings ESCA Eurospeech*, volume 5, pages 2707–2710, Rhodes, Greece.

Collins, Michael. 2003. Head-driven statistical models for natural language parsing. *Computational Linguistics*, 29(4):589–637.

Daume, Hal, Kevin Knight, Irene Langkilde-Geary, Daniel Marcu, and Kenji Yamada. 2002. The importance of lexicalized syntax models for natural language generation tasks. In *Proceedings of INLG*, pages 9–16, Arden House, Harriman, NJ.

Dumais, Susan, Michele Banko, Eric Brill, Jimmy Lin, and Andrew Ng. 2002. Web question answering: Is more always better? In *Proceedings of SIGIR*, pages 291–298, Tampere, Finland.

Durbin, Richard, Sean Eddy, Anders Krogh, and Graeme Mitchison. 1998. *Biological Sequence Analysis.* Cambridge University Press.

Elhadad, Michael, Yael Netzer, Regina Barzilay, and Kathleen McKeown. 2001. Ordering circumstantials for multi-document summarization. In *Proceedings of BISFAI*, Ramat Gan, Israel.

Elhadad, Michael and Jacques Robin. 1996. An overview of surge: A reusable comprehensive syntactic realization component. Technical Report 96–03, Department of Mathematics and Computer Science, Ben Gurion University, Beer Sheva, Israel.

Farach, Martin, Teresa Przytycka, and Mikkel Thorup. 1995. On the agreement of many trees. *Information Processing Letters*, 55(6):297–301.

Finden, C. R. and A. D. Gordon. 1985. Obtaining common pruned trees. *Journal of Classification*, 2:255–276.

Grefenstette, Gregory. 1998. Producing intelligent telegraphic text reduction to provide an audio scanning service for the blind. In *Proceedings of the AAAI Spring Workshop on Intelligent Text Summarization*, pages 111–115, Palo Alto, CA.

Gusfield, Dan. 1997. *Algorithms on strings, trees and sequences*. Cambridge University Press.

Hatzivassiloglou, Vasileios, Judith Klavans, and Eleazar Eskin. 1999. Detecting text similarity over short passages: Exploring linguistic feature combinations via machine learning. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, College Park, MD.

Hatzivassiloglou, Vasileios and Kathleen McKeown. 1993. Towards the automatic identification of adjectival scales: Clustering adjectives according to meaning. In *Proceedings of the ACL*, pages 172–182, Columbus, OH.

Jing, Hongyang and Kathleen McKeown. 2000. Cut and paste based summarization. In *Proceedings of the First Conference of the North American Chapter of the Association of Computational Linguistics*, pages 178–185, Seattle.

Knight, Kevin and Vasileios Hatzivassiloglou. 1995. Two-level, many-path generation. In *Proceedings of the ACL*, pages 252–260, Cambridge, MA.

Knight, Kevin and Daniel Marcu. 2002. Summarization beyond sentence extraction: A probabilistic approach to sentence compression. *Artificial Intelligence Journal*, 139(1):91–107.

Lacatusu, V. Finley, Steven J. Maiorano, and Sanda M. Harabagiu. 2004. Multi-document summarization using multi-sequence alignment. In *International Conference on Language Resources and Evaluation*, Lisbon, Portugal.

Langkilde, Irene and Kevin Knight. 1998. Generation that exploits corpus-based statistical knowledge. In *Proceedings of the ACL/COLING*, pages 704–710, Montreal, Quebec.

Lin, Chin-Yew and Eduard H. Hovy. 2002. From single to multi-document summarization: A prototype system and its evaluation. In *Proceedings of the ACL*, pages 457–464, Philadelphia, PA.

Lin, Chin-Yew and Eduard H. Hovy. 2003. Automatic evaluation of summaries using *n*-gram co-occurrence statistics. In *Proceedings of HLT-NAACL*, pages 150–157, Edmonton, Alberta.

Mani, Inderjeet and Eric Bloedorn. 1997. Multi-document summarization by graph search and matching. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI-97)*, pages 622–628, Providence, RI. AAAI.

Mani, Inderjeet, Barbara Gates, and Eric Bloedorn. 1999. Improving summaries by revising them. In *Proceedings of the ACL*, pages 558–565, College Park, MD.

Marcu, Daniel and Laurie Gerber. 2001. An inquiry into the nature of multidocument abstracts, extracts, and their evaluation. In *Proceedings of the NAACL Workshop on Automatic Summarization*, pages 2–11, Pittsburgh, PA.

McKeown, Kathleen R., Regina Barzilay, David Evans, Vasileios Hatzivassiloglou, Judith Klavans, Ani Nenkova, Carl Sable, Barry Schiffman, and Sergey Sigelman. 2002. Tracking and summarizing news on a daily basis with Columbia's Newsblaster. In *Proceedings of the Human Language Technology Conference (HLT-02)*, pages 280–285, San Diego, CA.

McKeown, Kathleen R., Regina Barzilay, David Evans, Vasileios Hatzivassiloglou, Min Yen Kan, Barry Schiffman, and Simone Teufel. 2001. Columbia multi-document summarization: Approach and evaluation. In *Proceedings of the Document Understanding Conference (DUC01)*, New Orleans, LA.

Melcuk, Igor. 1988. *Dependency Syntax: Theory and Practice*. Albany: State University of New York Press.

Meyers, Adam, Roman Yangarber, and Ralph Grishman. 1996. Alignment of shared forests for bilingual corpora. In *International Conference on Computational Linguistics (COLING 1996)*, pages 460–465, Copenhagen, Denmark.

Miller, George A., Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine J. Miller. 1990. Introduction to WordNet: An on-line lexical database. *International Journal of Lexicography*, 3(4):235–245.

Morris, Jane and Graeme Hirst. 1991. Lexical cohesion, the thesaurus, and the structure of text. *Computational Linguistics*, 17(1):21–48.

Nenkova, Ani and Kathleen R. McKeown. 2003. References to named entities: A corpus study. In *Proceedings of the Human Language Technology Conference, Companion Volume*, pages 70–73, Edmonton, Alberta.

Pang, Bo, Kevin Knight, and Daniel Marcu. 2003. Syntax-based alignment of multiple translations: Extracting paraphrases and generating new sentences. In *Proceedings of HLT-NAACL*, pages 180–187, Edmonton, Alberta.

Papineni, Kishore A., Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the ACL*, pages 311–318, Philadelphia, PA.

Radev, Dragomir, Hongyan Jing, and Malgorzata Budzikowska. 2000. Centroid-based summarization of multiple documents: Sentence extraction, utility-based evaluation, and user studies. In *Proceedings of the ANLP/NAACL 2000 Workshop on Automatic Summarization*, pages 165–172, Seattle, WA.

Radev, Dragomir and Kathleen R. McKeown. 1998. Generating natural language summaries from multiple on-line sources. *Computational Linguistics*, 24(3):469–500.

Radev, Dragomir, John Prager, and Valerie Samn. 2000. Ranking suspected answers to natural language questions using predictive annotation. In *Proceedings of Sixth Conference on Applied Natural Language Processing (ANLP)*, pages 150–157, Philadelphia, PA.

Reizler, Stefan, Tracy H. King, Richard Crouch, and Annie Zaenen. 2003. Statistical sentence condensation using ambiguity packing and stochastic disambiguation methods for lexical-functional grammar. In *Proceedings of HLT-NAACL*, pages 197–204, Edmonton, Alberta.

Robin, Jacques and Kathleen McKeown. 1996. Empirically designing and evaluating a new revision-based model for summary generation. *Artificial Intelligence*, 85(1–2):135–179.

Sankoff, David. 1975. Minimal mutation trees of sequences. *SIAM Journal of Applied Mathematics*, 28(1):35–42.

Schiffman, Barry, Ani Nenkova, and Kathleen R. McKeown. 2002. Experiments in multidocument summarization. In *Proceedings of HLT*, pages 52–58, San Diego, CA.

Shieber, Stuart and Xiapong Tao. 2003. Comma restoration using constituency information. In *Proceedings of HLT-NAACL*, pages 142–148, Edmonton, Alberta.

Siegel, Sidney and N. John Castellan. 1988. *Nonparametric Statistics for Behavioral Sciences*. McGraw-Hill.

Silber, Gregory and Kathleen McCoy. 2002. Computed lexical chains as an intermediate representation for automatic text summarization. *Computational Linguistics*, 28(4):487–496.