Improving the Quality of Text Understanding by Delaying Ambiguity Resolution

blind

Abstract

Text Understanding systems often commit to a *single best* interpretation of a sentence before analyzing subsequent text. The single best interpretation is chosen by resolving ambiguities to the alternatives for which the system has the highest confidence, given the context available at the time of commitment. Subsequent text, however, may contain information that could change which alternatives have the highest confidence. This may especially be the case when the system is able to read multiple redundant texts on the same topic. Ideally, the system would delay choosing among ambiguous alternatives until more text has been read.

One solution is to maintain multiple candidate interpretations of each sentence until the system acquires more evidence. Unfortunately, the number of alternative interpretations explodes quickly. In this paper, we propose a *packed graphical representation* (PG representation) that can efficiently represent a large number of alternative interpretations along with dependencies among them. We also present an algorithm for combining evidence from multiple PG representations to help resolve ambiguity and prune alternatives once the decision to commit to a single interpretation has been made.

Our controlled experiments show that by delaying ambiguity resolution until multiple texts have been read, our prototype's accuracy is higher than when committing to interpretations sentence-by-sentence.

A typical text understanding system confronts ambiguity at each step of processing, including parsing, mapping words to concepts and formal relations, resolving coreferences, and integrating the knowledge derived from separate sentences or texts. The system is forced to discard many candidate interpretations in order to dampen the combinatorial explosion of possibilities. Commonly, after reading each sentence, a system will commit to its top ranked interpretation of the sentence before reading the next.

If a text understanding system could postpone committing to an interpretation without becoming swamped by a combinatorial explosion of alternatives, its accuracy would almost surely improve. This intuition follows from the observation that text is redundant in at least two ways. First, within a single coherent text (with sentences referencing the same set of entities and events), each sentence informs the interpretation of its neighbors. Second, within a corpus of texts on the same topic, the same information will be expressed in different surface forms, ambiguous in different ways. Each text provides context that may help inform the interpretation of the others. Related fields, such as Information Extraction, exploit textual redundancy to good effect, and perhaps text understanding can as well.

One approach is for the text understanding system to maintain multiple complete candidate interpretations. After reading each sentence, for example, the system would retain a beam of the n-best interpretations of the sentence. While this approach avoids a combinatorial explosion (for reasonable values of n), several problems remain. First, because the beam width is limited, the system may still discard correct interpretations before benefiting from the extra context from related text. Second, enumeration of the candidate interpretations does not represent the dependencies among them. For example, there may be multiple candidate word senses and semantic roles for a given sentence, but sense alternatives might be dependent on role selection (and vice-versa). The set of reasonable interpretations may be a subset of all combinations. Finally, maintaining distinct interpretations does not contribute to addressing the problem of combining evidence to narrow down alternatives and ultimately select a single best interpretation of a text.

This paper addresses these three problems. We propose an approach to text understanding in which the system postpones committing to the interpretation of a text by representing ambiguities and the dependencies among them. With our approach, there may be combinatorial growth in the set of alternative interpretations, but they are represented only intensionally, using a packed representation, which maintains alternatives while avoiding enumerating them. Furthermore, we propose an algorithm for updating and pruning the packed representation as more sentences and texts are read.

We evaluate our approach by comparing the accuracy of two reading systems: a baseline system that commits to its best interpretation after each sentence, and our prototype system that uses a packed representation to maintain all possible interpretations until further reading enables it to prune. For this initial proof of concept, we use a small corpus of redundant texts. The results indicate that our approach improves the quality of text interpretation by preventing aggressive pruning while avoiding combinatorial explosion.

In the following sections, we first describe our target semantic representation of the interpretation of sentences. We then present the details of our *packed graphical representation (PG representation)* and our algorithm to resolve ambiguities in the PG representations as disambiguating evidence from subsequent text accrues. We describe the architecture of a prototype that produces PG representations for text and implements the disambiguating algorithm. Finally, we present the results from controlled experiments designed to compare the accuracy of the prototype to a baseline system that prunes more aggressively.

Target semantic representation

Our target representation is a simple semantic graph in which nodes are words from the sentence and the types in an ontology to which the words map. Edges are formal se-

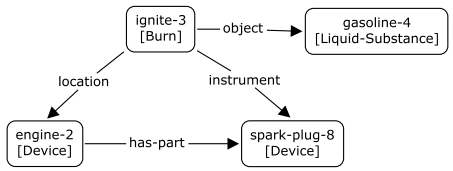

Figure 1: The target semantic graph representation for S1

mantic relations which may correspond to words from the sentence or to syntactic relations in the sentence's parse.

Fig. 1 shows the target semantic representation for the following simple sentence:

S1: *An engine ignites gasoline with its spark plug.*

PG representation

Multiple alternative semantic interpretations for a sentence can be captured with a single PG representation in which ambiguities are represented as local alternatives. Because the candidate semantic representations are often structurally similar, a PG representation can significantly compress the representation of alternative interpretations.

Fig. 2 shows the PG representation of alternate interpretations of S1 (PG1). The different types of ambiguity captured by the PG representation are as follows.

Type ambiguity. Ambiguity in the assignment of a type for a word. In PG1, the node engine-2a corresponds to the word "engine" in S1. Its annotation [LIVING-ENTITY .3 | DEVICE .7] says that the word may map to either LIVING-ENTITY (probability 0.3) or DEVICE (probability 0.7). The PG representation does not presume a particular uncertainty formalism. Any formalism, (Dempster-Shafer theory (Pearl 1988), Markov Logic Networks (Richardson and Domingos 2006), etc.) could be used.

Relational ambiguity. Ambiguity in the assignment of semantic relation between nodes. In PG1, the edge label <agent .6 | location .4> from ignite-3a to engine-2a says that the engine is either *agent* or *location* of the ignition.

Structural ambiguity. The PG representation also captures structural alternatives. In PG1, edges D and E are alternatives corresponding to the different prepositional phrase attachments for "with its spark plug" (to ignite-3a or gasoline-4a). The annotation {D .3 | E .7} says that the choices are mutually exclusive with probabilities of 0.3 and 0.7.

Co-reference ambiguity. Co-reference of nodes in a PG representation is captured using a "co-reference" edge. In PG1, the edge labeled <coref .7> represents the probability that engine-2a and its-7a are co-referent.

In addition to storing ambiguities explicitly, the PG representation also captures dependencies among alternatives.

Simple dependency. The existence of one element in the graph depends on the existence of another element. If subsequent evidence suggests that an element is incorrect, its dependents should be pruned. For example, the dependency A → C, means that if LIVING-ENTITY is ultimately rejected as the type for engine-2a, the agent relation should be pruned.

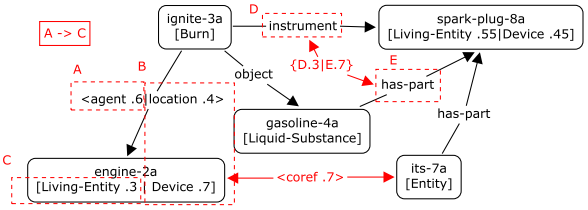

Figure 2: The PG representation for S1 (PG1)

Mutual dependency. Elements of a mutual dependency set are mutually confirming. If enough evidence accrues to confirm or reject an element, other elements in the set should also be confirmed or rejected. In the example, the box labeled B says that the two elements (engine-2a type DEVICE) and (ignite-3a location engine-2a) should both be confirmed or pruned when either of them is confirmed or pruned.

Formally, the PG representation is a structure consisting of (a) *semantic triples* – e.g., (ignite-3a type BURN), (b) *macros* – e.g., the symbol A refers to (ignite-3a agent engine-2a), and (c) *constraints* – e.g., A depends on C.

Combining multiple PG representations

Maintaining ambiguity within the PG representation allows us to delay commitment to an interpretation until enough evidence accrues to disambiguate. For any text fragment that results in a PG representation (PGa) containing ambiguity, there may exist other text fragments somewhere that are partly redundant, but result in a less ambiguous (or differently ambiguous) representation (PGb). The less ambiguous representation (PGb) can be used to adjust confidences in the ambiguous representation (PGa). Enough such evidence would allow us to prune unlikely interpretations, ultimately disambiguating the original representation.

For example, sentence S3 does not have sufficient context to disambiguate between the MOTOR sense of "engine" and the VEHICLE sense (as in *locomotive*).

S3: *General Electric announced plans this week for their much anticipated new engine.*

The PG3 representation for S3 (PG3) would maintain the ambiguous representation (with confidences for each sense based on prior probabilities, for example).

On subsequently encountering sentence S4, a Lesk-based word sense disambiguation module (such as the one we use in our prototype) would produce a PG4 representation with a strong preference for the locomotive sense of "engine", given the more specific context of S4.

S4: *The announcement comes to the relief of many in the railway industry looking to replace the engines in their aging locomotive fleets.*

To use interpretations from PG4 to disambiguate PG3, we need to align PG3 and PG4 semantically and merge their conflict sets. (In the simple example, the conflict sets for the word engine might be something like [MOTOR .5 | VEHICLE .5] in PG3 and [MOTOR .2 | VEHICLE .8] in PG4).

Algorithm 1 Combining PG representations

1. Identify the initial seed mappings between the two input PG representations using coreference detection methods. Currently, we use a naive heuristic which considers nouns derived from the same base form to be candidate co-references.
2. Starting from the current mappings, identify additional mappings by aligning the labels of the edges and the types of the nodes (similar to the greedy algorithm of finding a maximal common subgraph); merge the mapped nodes and edges; mutate the PG representations as follows:
   (a) Update associated constraints:
   - **type/relational ambiguity** Combine the conflict sets of merged nodes or edges according to the uncertainty formalism used in the PG representation.
   - **structural ambiguity** If one alternative in a structural interpretation has been aligned and merged, increase the relative confidence of that alternative.
   - **co-reference ambiguity** Merge two nodes connected by a co-reference edge if merging them produces more mappings (strengthening the alignment).

   (b) Prune interpretations whose confidence falls below threshold; if only one interpretation remains after pruning, that interpretation is confirmed.
   (c) If a pruned interpretation has dependents, prune its dependents; if a dependent interpretation is confirmed, confirm the interpretation it is dependent on; if an interpretation within a mutual dependency set is confirmed or pruned, confirm/prune the other interpretations in the set.
   (d) If an interpretation is confirmed, prune competing alternatives.
3. Repeat Step2 until no more mappings are discovered.

Algorithm 1 describes how two PG representations can be combined to help resolve their ambiguities. The algorithm attempts to identify their isomorphic subgraphs (redundant portions of the interpretations) and then adjusts the confidence scores in alternatives. Finally, the algorithm provides a method for confirming/pruning the interpretations of the PG representations based on confidence scores and the dependencies within the PG representations.

We will now step through Algorithm 1, merging PG1 (fig. 2) with PG2 (fig. 3). The resulting representation, fully pruned, is identical to the target representation for S1 (fig. 1). The uncertainty formalism and confidence thresholds in the example are simple for illustration.

1. The algorithm identifies (engine-2a, Engine-1b), (spark-plug-8a, spark-plug-3b) and (gasoline-4a, gasoline-6b) as candidate co-references. It merges the pairs and their conflict sets according to the uncertainty formalism used in the PG representation. In our current prototype, the confidences of the conflict sets are simply added. Therefore, [LIVING-ENTITY .3 | DEVICE .7] of engine-2a is merged with [DEVICE 1] of Engine-1b to produce [LIVING-ENTITY .3 | DEVICE 1.7]. LIVING-ENTITY is discarded, because its relative score falls below threshold. [1].
2. Deleting LIVING-ENTITY causes deletion of the *agent* edge between ignite-3a and engine-2a due to the dependency constraint A → C, (meaning *agent* (in A) depends on the existence of LIVING-ENTITY (in C)).
3. Co-reference between engine-2a and its-7a is greedily confirmed because merging the two nodes enables the alignment of (its-7a has-part spark-plug-8a) with (Engine-1b has-part spark-plug-3b).
4. The algorithm aligns (ignite-3a instrument spark-plug-8a) with (combust-5b instrument spark-plug-3b), because ignite-3a and combust-5b share the same type, [BURN]. This operation increases the score of D (the structure corresponding to PP attachment of "with its spark plug" to "ignite") over E (the structure corresponding to attachment of "with its spark plug" to "gasoline").

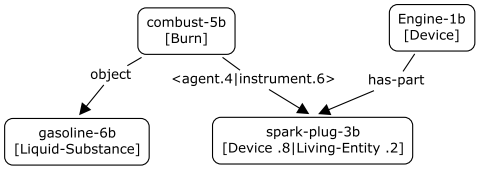

Figure 3: PG representation for S2, *"The engine's spark plug combusts gasoline."*

Any remaining ambiguity could simply be left in the PG representation (to be dealt with by subsequent reasoners). If an unambiguous final representation is appropriate, all lower scoring interpretations could be pruned. In this example, E would be pruned, making the result identical to fig. 1.

Prototype system

To evaluate our approach, we built a prototype system implementing the PG representation and Algorithm 1.

Parser. The system uses the Stanford Parser (Klein and Manning 2003). To capture structural ambiguity for our experiments, we manually converted the parser output to a syntactic PG representation by adding corrections as alternatives wherever the parse tree was incorrect. This gave a syntactic PG representation with both incorrect and correct alternatives. We arbitrarily gave the original, incorrect alternatives high confidence scores and the added, correct alternatives low scores. This approach simulates the situation in which the parser pruned the correct interpretation in favor of an incorrect one with a higher confidence score. The syntactic PG representation for S1 is shown in fig. 4. We have recently designed a modification to the Stanford Parser to make it produce syntactic PG representations natively, based on the complete chart built during parsing.

Semantic Interpreter. The semantic interpreter assigns types to nodes in the syntactic PG representation and se-

[1] In our prototype, we set the pruning threshold at $\frac{1}{3}\times$the score of the top-scored interpretation.

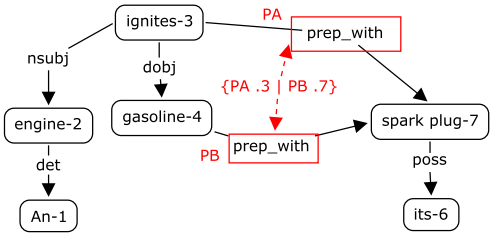

Figure 4: Syntactic PG representation for S1, capturing the PP-attachment ambiguity of "with its spark plug".

mantic relations to the edges. The resulting semantic PG representation has the following semantics and constraints.

- **Type ambiguity.** Types and confidence scores are assigned to words using SenseRelate (Patwardhan and *et al* 2005), WSD software based on the Lesk Algorithm (Lesk 1986). Assigned senses are then mapped to the Component Library ontology using its built-in WordNet mappings.
- **Relational ambiguity.** Semantic relations are assigned to the dependency relations in the syntactic PG representation according to semantic interpretation rules [self-cite]. Most of the rules consider the types of the head and the tail as well as the dependency relation, but do not produce confidence scores. Our experimental prototype simply scores candidates equally. We plan to incorporate a more sophisticated scoring method such as (Punyakanok and *et al* 2005).
- **Structural ambiguity.** Parse ambiguities (such as PA vs. PB in fig. 4) are converted directly to structural ambiguity representations (D vs. E in fig. 2) in the semantic PG representation.
- **Simple Dependency.** A dependency is installed between a type t for word w and a semantic relation r when (1) r is produced by a rule based on t and (2) r is dependent on no other candidate type for w. In fig. 2, a dependency relation is installed from A to C, because (1) LIVING-ENTITY in engine-2a was used in the rule assigning *agent* between ignite-3a and engine-2a and (2) the assignment of *agent* is not dependent on DEVICE, the other candidate type of engine-2a.
- **Mutual dependency.** If multiple interpretations depend on one another, a mutual dependency set is created to include them.

PG Merger. This module implements Algorithm 1 to combine PG representations from multiple sentences. The PG representation for each sentence is merged with the combined PG representation from previous sentences. The global PG representation integrates sentence-level PG representations to the extent that they align semantically. In the worst case (completely unrelated sentences), the global PG representation would simply be the union of individual PG representations. The extent to which the global PG representation is more coherent reflects redundancy and semantic overlap in the sentences.

Original Text Hearts pump blood through the body. Blood carries oxygen to organs throughout the body. Blood leaves the heart, then goes to the lungs where it is oxygenated. The oxygen given to the blood by the lungs is then burned by organs throughout the body. Eventually the blood returns to the heart, depleted of oxygen.

Paraphrase The heart begins to pump blood into the body. The blood first travels to the lungs, where it picks up oxygen. The blood will then be deposited into the organs, which burn the oxygen. The blood will then return to the heart, where it will be lacking oxygen, and start over again.

Figure 5: The original text and a paraphrase

Experiment 1

Our first experiment attempts to evaluate the claim that delaying ambiguity resolution improves accuracy. Redundancy and semantic overlap in subsequent sentences should allow Algorithm 1 to adjust the confidence in ambiguous alternatives given more context. To do this, we needed known redundant texts. In practice, we envision a system whose task is to develop a model of a particular topic by interpreting multiple tutorial texts on the topic. Such a system might be given, a priori, a clustered set of documents on the topic. Alternatively, given a single tutorial text on a topic, a system could perform its own information retrieval to collect a small corpus of texts with some confidence in their semantic overlap. For this experiment, to ensure redundancy, we generated a set of ten texts by having volunteers rewrite a short, tutorial text, using Amazon Turk (http://mturk.com). The volunteers had no knowledge of the purpose of the task, and were asked simply to rewrite the text using "different" language. Fig. 5 shows the original text and one volunteer rewrite. The total number of sentences over the ten texts was 37. Average sentence length was 14.5 words.

Evaluation Procedure

We ran two systems over the ten texts. The baseline system commits to the highest scoring consistent interpretation after each sentence. The prototype system produces an ambiguity-preserving PG representation. As the prototype reads each sentence, it uses Algorithm 1 to merge the PG representation of the sentence with that of the previous sentences. After N sentences (varying N from 1..37), the system is forced to commit to the highest scoring consistent interpretation from the PG representation. For N=1 (the prototype system is forced to commit after reading the first sentence), both the baseline and prototype systems produce the same result. For N=2, the baseline system produces the union of the highest scoring interpretations for each of the first two sentences in isolation. The prototype system produces a merged PG representations for the first two sentences and then prunes to the highest scoring alternatives.

At each value of N, we measured the correctness of the interpretations (the percentage of correct semantic triples) committed by each system by comparing the committed triples against human-generated gold standard triples for the texts.

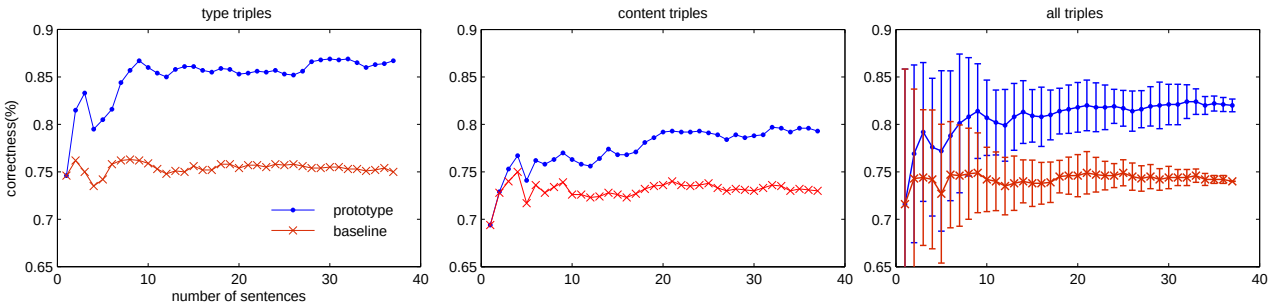

Figure 6: Correctness scores for the prototype vs. baseline system on (a) type triples (word sense assignment), (b) content triples (semantic relations) and (c) all triples (with standard deviation).

We repeated the experiment ten times with different random orderings of the 37 sentences, averaging the results.

Evaluation result

Fig. 6 shows that the quality of both type assignment and semantic relation assignment by the prototype system increase as the system acquires more evidence from other sentences. This result confirms our hypothesis that delaying commitment to an interpretation resolves ambiguities better by avoiding overly aggressive pruning.

To determine an upper bound of correctness for the prototype system, we inspected the PG representations to see how many alternative sets within the PG still contained the correct interpretation even if not the highest scoring alternative. This number is different from the correctness score in fig. 6, which is the percentage of gold standard triples in the PG representation after committing (pruning) to the highest scoring alternatives.

	baseline	prototype
nodes containing the correct type	76	**91**
edges containing the correct relation	74	**88**

Table 1: Percentage of nodes and edges maintaining the correct types and semantic relations in the baseline system and the prototype system for all 37 sentences.

Table. 1 shows that 91% of the nodes in the PG contain the correct type (though not necessarily the highest scoring). 88% of the edges contain the correct semantic relations among the alternatives. In contrast, the baseline system has pruned away 24% of the correct types and 26% of the correct semantic relations.

Experiment 2

Our second experiment aims to evaluate the claim that the prototype system can efficiently manage a large number of alternative interpretations. The top line in Fig. 7 shows the number of triples in the PG representations input to the prototype system. This is the total number of triples (including ambiguity alternatives) in the PG representation for each sentence prior to invoking Algorithm 1. The middle line is the number of triples remaining after merging and pruning by Algorithm 1. The bottom line is the number of triples after pruning all but the highest scoring alternatives (the baseline system). The results show that Algorithm 1 achieves

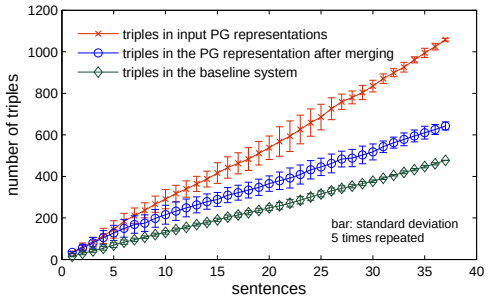

Figure 7: Total number of triples in individual sentence PG representations (top); total number of triples in the PG representation after merging in the prototype system (middle); total number of triples after pruning to the highest scoring alternative (bottom).

significant compression over unmerged PG representations. The resulting size of the merged PG representations more closely tracks the size of the aggressively pruned representations.

Experiment 3

Finally, we wanted to measure the sensitivity of our approach to the quality of the natural language interpretation. In this experiment, we artificially varied the confidence scores for the correct interpretations in the PG representations input to the prototype and baseline systems by a fixed percentage. For example, consider a node heart-1 in a PG representation. Among the candidate types is the correct sense for its context: INTERNAL-ORGAN with confidence 0.8. We reran Experiment 1 varying the confidence in INTERNAL-ORGAN in increments of both +10% and -10%, while scaling the confidences in the incorrect types equally. As the confidence in correct interpretations is increased, all correct interpretations become the highest scoring, so aggressive pruning is justified and the baseline system performance approaches the prototype system performance. As the confidences in correct interpretations are decreased, they are more likely to be pruned by both systems.

Fig. 8 shows that Algorithm 1 is able to recover at least some correct interpretations even when their original scores (relative to incorrect alternatives) is quite low.

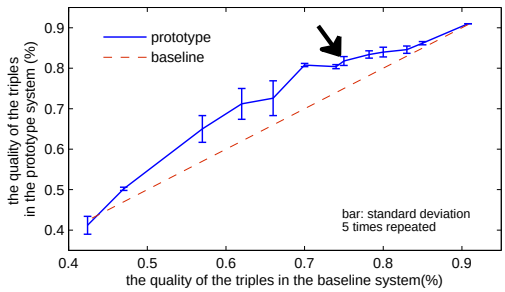

Figure 8: Sensitivity of the prototype and baseline systems to the quality of the NL system output. The quality of input triples is perturbed affecting performance accuracy of the two systems. For example, when the quality of input triples is such that the baseline system performs at 70% accuracy, the prototype system performs at 80%. The arrow indicates unperturbed language interpreter performance.

Discussion and Future Work

The results of our controlled experiments suggest that it is both desirable and feasible to delay commitment to ambiguity resolution beyond the sentence and text boundaries. Improvements in the correctness of the semantic interpretation of sentences is possible without an explosion in size when maintaining multiple interpretations.

Nevertheless, the experiments reported are proofs of concept. The results confirm that it is worthwhile subject our prototype to a more real-world, practical application. To do so, we need to address several issues.

First, we will complete modifications to the Stanford Parser to produce PG representations natively. This change will result in a significant increase in the number of alternatives stored in the PG representation over the current prototype. Our initial investigations suggest that there is still enough structural overlap among the many possible parse trees to allow the PG representation to control explosion, but this is an empirical question that will need to be confirmed.

We are modifying our semantic interpreter to admit induced semantic interpretation rules [self-cite] which will allow us to train the system on corpora from new domains.

The current prototype uses a simple, naive heuristic for identifying co-reference candidates. We plan to plug in a more sophisticated, off-the-shelf co-reference system.

Finally, we will explore the use of more sophisticated mechanisms for managing uncertainty. In particular, the current heuristics for adjusting probabilities when merging PG representations and the thresholds for confirming or pruning interpretations need to be replaced.

Once these updates are complete, we will perform more wide-scale evaluations. We will investigate test corpus construction using text clustering to find redundant/overlapping texts and conduct experiments in multiple domains.

Related Work

The idea of succinctly representing multiple interpretations has been explored by several researchers for different NLP tasks. For example, *the packed representation* (Crouch 2005), represents alternative semantic representations for a sentence succinctly with first-order logic and then runs a SAT solver against several types of constraints to find probable interpretations that are consistent with the constraints. The *underspecified representation* used in (Alshawi and Crouch 1992) (Bos 2004) (Schilder 1998) allow systems to defer interpretation decisions until they acquire sufficient evidence. Unlike the PG representation, this work generally focuses on one particular type of ambiguity such as scope ambiguity (Alshawi and Crouch 1992) (Bos 2004) or discourse representation (Schilder 1998).

Conclusion

In this paper we have begun to address the challenge of efficiently managing multiple alternative interpretations of text in a text understanding system. We have presented (1) a *packed graphical representation* that succinctly represents multiple alternative interpretations as well as the constraints among them, and (2) an algorithm for combining multiple PG representations to reinforce correct interpretations and discount implausible interpretations. Controlled experiments show that it is possible to improve the correctness of semantic interpretations of text by delaying disambiguation, without incurring the cost of an exponentially expanding representation.

References

Alshawi, H., and Crouch, R. S. 1992. Monotonic semantic interpretation. In *ACL*, 32–39.

Bos, J. 2004. Computational semantics in discourse: Underspecification, resolution, and inference. *Journal of Logic, Language, and Information* 13(2):139–157.

Crouch, D. 2005. Packed rewriting for mapping semantics to kr. In *Sixth IWCS*.

Klein, D., and Manning, C. 2003. Accurate unlexicalized parsing. In *Proc. of ACL*.

Lesk, M. 1986. Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In *SIGDOC*.

Patwardhan, S., and *et al*. 2005. Senserelate:: Targetword-A generalized framework for word sense disambiguation. In *ACL*.

Pearl, J. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan-Kaufmann.

Punyakanok, V., and *et al*. 2005. The necessity of syntactic parsing for semantic role labeling. In *IJCAI*.

Richardson, M., and Domingos, P. 2006. *Markov logic networks*. Kluwer Academic Publishers.

Schilder, F. 1998. An underspecified segmented discourse representation theory (USDRT). In *COLING-ACL*.