Building a Tool for Annotating Reference in Discourse

Jonathan DeCristofaro
CIS Department
University of Delaware
103 Smith Hall
Newark, DE 19716, USA
decristo@cis.udel.edu

Michael Strube
IRCS
University of Pennsylvania
3401 Walnut Street, Suite 400A
Philadelphia, PA 19104, USA
strube@linc.cis.upenn.edu

Kathleen F. McCoy
CIS Department
University of Delaware
103 Smith Hall
Newark, DE 19716, USA
mccoy@cis.udel.edu

Abstract

We discuss the development of a system for marking several types of reference to facilitate the analysis of reference in discourse. The tool is designed to be used in three applications: generating training data for machine learning of co-reference relations, evaluating theories of referring expression generation and resolution in texts, and developing theories for understanding reference in dialogs. The need to mark any of a broad set of relations which may span several levels of discourse structure drives the system architecture. The system has the ability to collect statistics over encoded relations and measure inter-coder reliability, and includes tools to increase the accuracy of the user's markings by highlighting the discrepancies between two sets of markings. Using parsed corpora as the input further reduces the human workload and increases reliability.

1 Motivation

To examine the phenomenon of reference in discourse, and to analyze how discourse structure and reference interact, we need a tool which allows several kinds of functionality including mark-up, visualization, and evaluation. Before designing such a tool, we must carefully analyze the kinds of information each application requires.

Three applications have driven the design of the system. These are: 1) the creation of training data for automatic derivation of reference resolution algorithms (*i.e.*, machine learning), 2) the formation of a testbed for evaluating proposed reference generation and anaphora resolution theories, and 3) the development of theories about understanding reference in dialog. The influence that these three areas have upon the functional requirements of an annotation system are discussed below.

In this paper we first describe the requirements that each of these three related applications demand from a discourse annotation tool geared to aid in answering questions concerning reference. We next discuss some of the theoretical implications and decisions concerning the tool development that have arisen from these requirements. Next we describe the tool itself. Finally, we discuss related work, future directions of this work, and some conclusions.

1.1 Machine Learning

Consider a learning task in which we will present the learner with a sequence of triples of the form $\langle E, F, C \rangle$, where:

- E is a pair of text expressions E_A and E_B,
- F is vector of features describing the expressions, and
- C is the classification: **+** if E_A and E_B co-refer, **-** otherwise.

(Two expressions *co-refer* when they denote the same discourse entity (DE).) A successful learner will output a model which, when given only $\langle E, F \rangle$, can predict the value of C, that is, classify the instance as positive or negative. We intend to use the annotation tool to produce a set of such instances and features.

The first requirement for a tool which would help us generate such a body of data is that it must allow us to mark all the potential referring expressions. This simply means that the user will have the ability to delineate any span of text which represents a DE, and treat that span as a single entity. Of course, this is a time-consuming and error-prone process and thus it is helpful to automate as much as possible. In the training phase, the learner must be given all the potential antecedents for an anaphoric reference, so that it will know how to distinguish the proper antecedent from all the other candidates. For the testing phase, the correct antecedent's span must be included as a marked entity in the corpus, or the learner has no chance of getting that instance of co-reference right.

The other crucial function of an annotation tool is to let the user associate attributes, or feature values, with the marked expression. During the training phase, a learning algorithm is trying to find correlations between the features F and the classification C. Choosing the set of features to include in the learning phase is a very difficult task. The set must be sufficiently rich so as to include all of those features which might affect a referring expression's resolution. On the other hand, since the learner will likely find that only certain features predict co-reference, we do not want to burden the learner with many useless features that will bog it down with computational complexity. Also, a less restricted set of features permits more opportunity for inconsistency in a given coder's markings and disagreement among coders (Condon & Cech, 1995).

We cannot know (before training) exactly which features are most predictive of co-reference. So, we will try to mark a set of features which is a superset of the necessary features. Drawing on the feature sets used in Connolly et al. (1997) and Ge et al. (1998), we believe the following factors *might* indicate co-reference:

- Syntactic role (e.g. Subject, Object, Prepositional Object,...),
- Pronominalization (yes or no),
- Distance between E_A and E_B (an integer),
- Definiteness (yes or no),
- Semantic role (e.g. indicating location, manner, time,...),
- Nesting depth of an NP (an integer),
- Information status (as defined by Strube (1998)) of the DE,
- Gender, Number, Animacy.

The tool must allow the coder to assign values for these features to each marked expression, but should not demand that every expression has a value assigned for every feature.

Since we cannot claim that this set of features is exhaustive, the tool must allow further features to be added by the user. Since reliability of feature assignment is important, the tool should have the ability to extract as many features as possible automatically (for example, from a parsed corpus). In addition since some features must be hand-marked, the tool must have the ability to compare feature marking between two coders for the same text.

1.2 Evaluating Anaphora Generation and Resolution Algorithms

Our discourse annotation and visualization tool also fulfills the role of a testbed in which we can examine theories of generating and resolving anaphoric expressions. From the generation perspective, we look for answers to questions concerning when it is appropriate to generate a pronoun versus some other anaphoric expression (e.g., a definite description or name; see McCoy & Strube (1999a), McCoy & Strube (1999b)).

Some researchers have looked at the question of when to generate a pronoun (versus some other description) (e.g., McDonald (1980), McKeown (1983), McKeown (1985), Appelt (1981)). In this work the decision was based on a notion of focus of attention (Sidner, 1979) – if an entity was the focus of the previous sentence and is the focus of the current sentence, then use a pronoun. To evaluate such claims, not only must co-reference relations be marked in a text, but information concerning focusing data structures must be kept.

Dale (1992) discussed the generation of pronouns in the context of work on generating referring expressions (Appelt, 1985; Reiter, 1990). Dale suggests the principles of efficiency and adequacy which favor generating the smallest referring expression that distinguishes the object in question from all others in the context. This notion was somewhat altered in Dale & Reiter (1995) to more adequately reflect human-generated referring expressions and to be more computationally tractable.

Other researchers have suggested that a notion of discourse structure must be taken into account when generating referring expressions. In particular, Grosz & Sidner (1986) and Reichman (1985) both suggest that a full noun phrase might be generated at discourse segment boundaries when a pronoun might have been adequate (in Dale's sense). Passonneau (1996b) argues for the use of the principles of information adequacy and economy. Her algorithm takes discourse segmentation into account through the use of focus spaces which are associated with discourse segments. Passonneau argues that, a fuller description might be used at a boundary because the set of accessible objects changes at discourse segment boundaries.

Passonneau's work suggests additional features which must be marked in a text to evaluate referring expression generation algorithms. These include discourse segment boundaries and sets of "confusable" DE's contained in the focus space. Thus the definition of what constitutes a discourse segment is another item which is open to research; our tool should allow for alternative markings of discourse segments so that various algorithms can be evaluated. For example, in our current work we look at changes in time as segment boundaries. Other definitions are possible. So, the tool must be able to keep information for various alternative algorithms.

While it is intuitively appealing that notions of discourse segmentation affect pronoun generation, the above work fails to identify how a discourse segment should be defined to a generation algorithm – thus it is not clear how this work can be applied to the generation process.

Given this previous work, we need a tool that will allow us to specify (1) alternative definitions of discourse segmentation, and (2) alternative algorithms for pronoun versus definite description generation (and anaphora resolution). The tool must have the ability to then calculate statistics so that the alternative definitions and algorithms can be compared.

Thus, this application requires the ability to specify co-reference relations, associate various features with referring expressions (both syntactic and discourse-relevant), calculate the results of certain well-specified algorithms on the referring expressions, and tabulate the results of such algorithms. In addition to this information on referring expressions themselves, the tool must allow the marking of arbitrary features over arbitrary pieces of text (e.g., for alternative definitions of discourse segments). Because this work is exploratory in nature, the tool should allow a researcher to easily find places where various algorithms fail so that they can be examined and the algorithms updated as needed.

1.3 Understanding Spoken Dialog

The evaluation of algorithms for anaphora resolution in spoken dialog requires annotation of discourse structure on several levels. This is because spoken dialog shows more complex phenomena than written discourse. Problematic issues in spoken dialog include

- the determination of the center of attention in multi-party discourse;
- utterances with no discourse entities;
- abandoned or partial utterances, interruptions, speech repairs;
- the determination of utterance boundaries;
- the high frequency of discourse deictic and vague anaphora (Eckert & Strube, 1999).

In order to capture the complexity of anaphora resolution in spoken dialog, the annotation requires a multitude of steps.

Dialog Acts. To determine the domain of anaphoric antecedents, the dialog must be divided into short pieces. We have chosen to use units based on dialog acts for this task. Therefore, turns have to be segmented into dialog act units. Our study of anaphoric expressions reveals that in a dialog between two participants A and B, the DE's introduced by A are not added to the shared discourse memory model until A's contribution has been acknowledged by B. Thus the segment is important for resolution algorithms.

As in all coding schemes, intercoder reliability (here, of the dialog act units) must be questioned. For the purpose of applying the Kappa (κ) statistic the segmentation task must be turned into a classification task. So, we view boundaries between dialog acts as one class and non-boundaries as the other (see Passonneau & Litman (1997) for a similar practice). The next step is to classify dialog act units as particular dialog acts. For this task the κ statistic is also appropriate.

Individual and Abstract Object Anaphora. Since spoken dialog shows a high number of discourse deictic and vague anaphora, pronouns and demonstratives have to be classified accordingly. Thus an additional feature, anaphor type, must be marked in the corpus.

Co-Indexation of Anaphora and Antecedents. Vague pronouns do not have a particular antecedent in the text. Hence, they cannot be co-indexed with an antecedent. The co-indexation of individual object anaphora in spoken dialog does not differ from written discourse. However, the high number of discourse deictic pronouns requires a second set of markables since discourse deictic pronouns can co-specify with propositions, sentences and even discourse segments. Therefore, the reliability of the annotation depends on (1) the marking of the correct text span and (2) whether the correct antecedent is linked with the pronoun. Determining the reliability of marking spans of text is difficult when any span can be marked, since this means almost any word boundary is a candidate segment boundary. Here, the κ statistic does not seem meaningful because of the huge disparity in the number of non-boundaries and boundaries. This highly skewed distribution seems to overwhelm κ.

Thus we are exploring more appropriate measures of intercoder reliability on this task. At the moment, our approach to this problem is to use κ, but restrict the annotators, so that they are allowed to mark only certain contiguous linguistic objects like verb phrases, sentences, or a well defined segment spanning more than one turn.

2 Annotating a Parsed Corpus

All of the applications discussed in section 1 depend on having a corpus of reliably marked expressions, features, and relations. In order to determine that these dimensions have been "reliably marked", we need to measure agreement between two coders marking the same text.

One way to increase the reliability of the coding (regardless of the method used to measure reliability) is to automate part of the coding process. Our system can extract a number of markings, features and relations from the parsed, part-of-speech-tagged corpora of the type found in in the Penn Treebank 2 (Marcus et al., 1994). Use of the Treebank data means we can find most of the markables and many of the necessary features before giving the task to a human coder. We do not try to extract any of the co-reference information from the parsed corpora.

2.1 Extracting Markables

In this context, a markable is a text span representing a discourse entity which can be anaphorically referred to in a text or dialog. The majority of markables are noun phrases. Because the Treebank is a fully-parsed and well-defined representation of the text, it is trivial to determine the boundaries of all of the NP's in the text. However, the full set of NP's found by the Treebank parse is too inclusive for our purposes (*i.e.*, it is a superset of the NP markables). While the Treebank delineates all NP's at all levels of embedding, it is not the case that each such NP contributes a distinct DE. Consider the following example containing three NP's in the parsed Treebank:

(1) (NP (NP different parts) (PP of (NP Europe)))

We want to mark both "different parts of Europe" and "Europe", since they both contribute distinct DE's. However, notice that "different parts" does not contribute a DE since it is not possible to refer to this subexpression alone in subsequent discourse.

To avoid finding such undesirable NP's, our system has a heuristic (H1) which says: *Pass over any NP which is a leftmost child of a top-level NP*. This heuristic is too drastic, though, eliminating constructions like (2).

(2) (NP (NP the inner brain) and (NP the eyes))

To avoid losing these examples, we include another heursitic (H2) which says: *H1 does not apply when the NP is a sibling of another NP*. A third heuristic must be added to overrule H1 in the case of a possessor in a possessive construction, such as:

(3) (NP (NP Chicago's) South Side)

where we should extract both "Chicago" and "Chicago's South Side". So, the heuristic H3 is introduced: *H1 does not apply when the NP is a possessive form.*

Even with heuristics eliminating the NP's which we do not need to consider, there are some NP's that will be found by the system which cannot be eliminated automatically. Copular constructions such as (4) introduce unnecessary NP's.

(4) John is a doctor.

"John" and "a doctor" are syntactically NP's, but the second does not contribute a unique DE.

Also, idiomatic expressions such as (5) must be eliminated by hand:

(5) Ned kicked the bucket.

The syntactic NP "the bucket" refers to no DE and cannot be the antecedent of any future referring expression, so it should not be marked.

At this time, we do not have a way for the expression extracting system to detect and avoid these examples. As a result, we must introduce a correction phase in which a human corrects the markings, eliminating those that are superfluous, and adjusting those that may have been mismarked. The goal is to have a set of expressions which is as close as possible to the set of expressions necessary and sufficient for the applications. For example, if there are many extraneous expressions in the machine learning task, they will act as distractors – examples which decrease the accuracy of the learned model by diluting the highly correlative data with noise.

2.2 Extracting Features

In addition to extracting many markables themselves, the parsed corpora contain information from which many of the features can be automatically derived. Some features' values are marked explicitly in the corpus while others can be automatically extracted by examining the tree structure. The simplest source of feature values is the Treebank "functional tags". For example, the grammatical functions (syntactic subject, topicalization, logical subject of passives, etc.) of phrases and the semantic role (vocative, location, manner, etc.) are marked in the corpus.

Other features must be found by walking the tree structure provided in the Treebank. The form of the NP (whether the NP is realized as a personal pronoun, demonstrative pronoun, or definite description) is a function of the part-of-speech tags assigned to the words in the NP. Whether the NP is definite, indefinite, or indeterminable depends on whether an article begins the NP. If the article is "a", "an", or "some", we assume the NP is indefinite. "The" indicates definiteness; otherwise, we assign a value of "none", which simply indicates that there is no simple way of classifying this instance. The case of an NP is usually determined by its position in the tree. Any child of a VP is marked as an "object". Children of PP's are marked "prep-adjunct" unless the PP was tagged "PP-put"[1], which indicates that the PP acts as a complement to the verb. In this case we tag the NP as "prep-complement".

2.3 Relations between Expressions

We allow two classes of relations to hold between markable entities: the co-reference relation and an open class of user-definable directional relations. A co-reference relation holds between A and B when A and B are expressions which both refer to the same discourse entity. Since co-reference is a symmetric, reflexive, and transitive relation, it divides the set of markables into equivalence classes. Within a given equivalence class, all mem-

[1] PP's using "in", "on", or "around" are sometimes marked PP-put.

bers refer to the same DE. Intuitively, our co-reference relation is a set of undirected links connecting all co-referring expressions. The symmetric property implies that it is not meaningful to store the direction of a relation. However, we do store each markable's antecedent when the user defines a co-reference link, so that we can later reconstruct the co-reference chain if necessary.

The other kind of link is directional. We allow the user to define any number of relations which are not symmetric, reflexive, or transitive. The only restriction on these relations is that they hold between exactly two entities. Initially, we postulate four such relations which are necessary to handle indirect co-reference relations, also called bridging relations (see also Passonneau (1996a)):

- **Attribute-of**

  (6) [The car]$_i$ won't start because [the engine]$_i$ is missing.

- **Propositional-inference**

  (7) [The man has a gun.]$_j$ [That]$_j$ scares me.

- **Contains**

  (8) [The peaches]$_k$ are in a basket. Give me [the biggest]$_k$.

- **Member-of**

  (9) [Jack]$_m$ and Jill went up the hill. [They]$_m$ were never seen again.

Clearly, these must be directional (*i.e.*, not symmetric) since, for example, if **Member-of(A,B)**, then we should not assume **Member-of(B,A)**. The user is not prevented, however, from defining two such links, one in each direction. In fact, **Contains** and **Member-of** are logical duals; that is, **Contains(a,b)** $\Leftrightarrow$ **Member(b,a)**. However, we are always interested in the relation of a referring expression to its potential antecedents and so require that the referring expression be the first argument and the antecedent the second. In (8), **Member-of(the biggest, the peaches)**, but in (9), **Contains(They, Jack)** and **Contains(They, Jill)**.

2.4 Measuring Agreement

All of the annotation discussed in the above sections is prone to error when a human is involved. The best way to combat these errors to is have several coders annotate the same corpus according to a coding manual.[2] A high measure of agreement between these coders gives us more confidence in the reliability of the data. Therefore, we must be able to measure agreement between two[3] codings of the same text.

The first kind of agreement that we need to measure is agreement of two sets of markables. Since we expect a few of the markables found by the system to need human editing, we may not assume that two coders working on the same text will have the same set of markables after the correction phase. We define agreement of two sets of markables S_1 and S_2 as

$$\text{Agreement}(S_1, S_2) = \frac{2 * c}{a + b}$$

where $a = |S_1|$, $b = |S_2|$, and $c =$ the number of expressions marked in S_1 that were marked with exactly the same boundaries in S_2. When agreement of markables is found to be less than 1, the coders are shown the expressions on which they disagree and can come to agreement (by referring to the coding manual and remarking those passages). We are developing a function of the tool which will simultaneously display the two versions of the text and highlight the expressions which are not common to the two codings. This will make it easier to visualize the differences between the codings and reach perfect agreement of markables.

The second kind of agreement measures agreement between two coders' co-reference codings. We require that the two coders have the same set of markables before comparing their co-reference annotations, so achieving markable agreement of 1 is a prerequisite for this calculation. As discussed in section 2.3, the co-reference relation divides the set of markables into equivalence classes. A model-theoretic algorithm proposed by Vilain et al. (1995) uses these co-reference classes to define a precision and recall metric which yields intuitively plausible results and is easy to calculate. The method depends on counting how many co-reference links must be added to one coder's equivalence classes to transform the set into that found by the other coder. We adopt this method and enable the tool to perform this computation between any two codings which fully agree on the underlying set of markables.

Finally, we can measure feature-value agreement by viewing the feature assignment task as a kind of classification task, and then computing Kappa (κ), which measures how well the coders agreed compared to their random expected agreement[4](Carletta, 1996). We conform to the method proposed in Poesio & Vieira (1998) for computing actual and expected agreement. (Again we assume the coders have already agreed on the set of markables.) Suppose we are considering a given feature

[2]The intent is that the coders will achieve a high degree of consistency if the manual is clear, and then if the manual accurately represents the desired coding style, consistency among coders implies accuracy of all the codings.

[3]Agreement among a set of $n > 2$ coders is usually calculated as a function of the $\frac{n*(n+1)}{2}$ pairwise agreements, so we will discuss only the pairwise case here, realizing that the full computation is straightforward.

f, which was marked by two coders on each of N expressions in a corpus. Percent agreement is simply the fraction of expressions out of N for which the two coders assigned the same value to f. Expected agreement is not computed by assuming that each value is equally likely, though. We compute the expected agreement based on the actual distribution of values, as follows. For two coders, if f takes on values from V,

$$P(E) = \sum_{v \in V} \left(\frac{c_1(v,f) + c_2(v,f)}{2 * N} \right)^2$$

where $c_i(v,f)$ is the number of times coder i assigned value v to feature f. Thus, if the coders have used the values in a perfectly even distribution among the $|V|$ values, $P(E) = \frac{1}{|V|}$. Any distribution which is not perfectly even will have an expected agreement higher than this.

As with measuring markable agreement, we measure feature-value agreement to ensure that we have reliable features before using the data for one of the applications discussed in section 1. Therefore, coders can ask the system to show the examples for which they disagree on a specified feature. Again, the coders have the opportunity to recode those examples to achieve perfect agreement before passing the data to the application.

3 REFEREE: The Discourse Annotation Tool

We have built a discourse annotation and visualization tool which is designed according to the issues discussed in section 1 and which has all the capabilities described in section 2. REFEREE[5] is a graphical interface tool written in Tcl/Tk. This makes it highly portable and easily extensible.

3.1 Annotation Modes

The tool has three "modes" – reference mode, segment mode, and dialog mode. In reference mode, the user can mark expressions, associate features with any expression, and assign co-reference (or other kinds of reference) links. Clicking on an expression with the mouse displays the features of that expression and highlights all other expressions in the text which co-refer with it. At this point, the user can update the co-reference or feature information or type some notes to be stored with the expression. (These notes are shown with the features whenever this expression is clicked on in the future.) Easy visualization of the co-reference equivalence classes could aid the user as he clicks through the text and sees how the co-reference chains thread through discourse.

A byproduct of the built-in flexibility of REFEREE is the ability to use different feature "masks" in case the user only wants to consider some subset of the complete set of marked features. For example, the user can configure the tool to display and allow changes to only the pronominalization feature. Then, the irrelevant features are not displayed and cannot be changed until the tool is reconfigured. This is also useful for associating different feature sets with different kinds of expressions.

Segment mode allows the user to break the text into arbitrarily nesting and overlapping segments. (These do not have to correspond to any certain definition of discourse segment or text segment.) This allows the user the freedom to choose any degree of constraints upon the structure. When the user selects a region and clicks on the "mark" button, a new segment is created spanning that region. Thus, we can build up a list of start and end points of segments, and automatically determine which segments are contained in or overlap with which other segments. A separate window displays graphically the start and end point of each segment.

At first glance, this seems to replicate the functionality of the reference mode, since both modes allow unconstrained marking (of contiguous text spans). The important difference is that in reference mode, the user delineates *referrable entities*, while in segment mode, the user is marking spans which represent the *structure* of discourse. So, a user could have many spans marked as segments which exactly coincide with markables in reference mode; this simply represents the fact that the user believes it is possible for the text to refer to the segments or the propositions they express. Still, the segment markings are not superfluous. They impose a structure on top of the reference mode markables, even if some of them coincide. (While this could be simulated in the reference mode by adding a binary feature for segmenthood, the visualization of segments would be lost, as would the decoupling of the two kinds of spans we mark.)

The last mode of interaction with REFEREE is dialog mode. This allows the user to code a dialog by breaking it into turns. Each dialog participant's turn can be broken into utterances which may be labeled as initiation or response units. (In some cases, there is overlap between these two dialog acts.) The most important function of dialog mode, as it relates to understanding reference in spoken language is to allow segmentation of the dialog into turns assigned to one speaker or the other. Recall the proper closure of a turn is crucial for determining which DE's are in the shared discourse model.

---

[4]

$$\kappa = \frac{P(A) - P(E)}{1 - P(E)}$$

where $P(A)$ is the proportion of times the annotators agree and $P(E)$ is the proportion of times the annotators are expected to agree by chance.

[5] for Referring Expression Reader/Editor

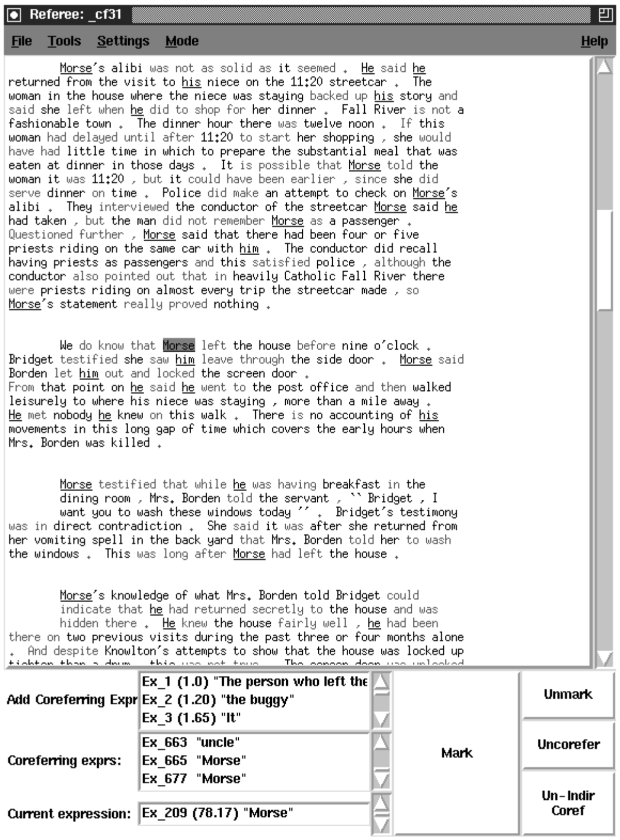

Figure 1: REFEREE in Reference Mode

3.2 Interface and Implementation Notes

Each of the three modes (reference, segment, and dialog) has one main window in which a page of text is displayed. For example, Figure 1 shows the main screen for reference mode. In this figure, dark text represents the NP's that have been marked or extracted from the Treebank. The "current expression" is highlighted and coreferring expressions are underlined. Though this scheme is perhaps visually unpleasing on paper, note that on the computer, the application uses vivid colors and easily differentiable typefaces. Furthermore, elements of the color scheme are customizable by the user.

The tool saves the user's annotations in several data files while leaving the original text file unchanged. Other annotation programs have embedded the annotations into the text using a sublanguage of XML. Files generated under either method are equally capable of representing the desired levels of annotation; we separate the text from the annotations in order to simplify the parsing of the data. In case a REFEREE user should want to port some marked text to a new annotation system, it is straightforward to automatically generate a text-and-annotation file to conform to any XML-style definition.

4 Previous Work

Previous systems were designed for different purposes, and therefore do not provide all of the functionality that our applications require. For example, MITRE's Alembic Workbench (Day et al., 1997) builds up an annotated corpus from scratch, under a mixed-initiative paradigm (in which some markings are given by the user, and some are automatically inserted by the computer). Learning an information extraction system was a primary function of this system. While the associated Alembic NLP system does incorporate some discourse level information into the system, the user may not impose an arbitrarily complex discourse structure whose structure the system can represent.

The Discourse Tagging Tool (Aone & Bennett, 1994) was designed for tagging multilingual corpora, and also does not allow complex marking of discourse structure. Furthermore, the tag sets and relations are fixed and may not be elaborated by the user. Also, this work was not concerned with dialogs.

5 Future Work and Conclusions

We are beginning annotation of a parsed corpus using Referee. We have found that it is much easier to code a corpus and get reliable results when the system has already found the majority of the markables. We intend to improve the tool by providing more functionality and better visualization of patterns in the data. We hope to add more complex feature-extraction rules that search the parse tree more extensively for syntactic features that are evident from the tree structure. We are also interested in using a lexicalized knowledge base to find semantic relationships between the marked expressions.

We believe that the requirements of the intended applications dictate the design of a novel and unique tool for the analysis of the relationship between discourse structure and reference. Referee fills this niche, and greatly reduces the workload placed on the human users. Furthermore, the open design of Referee makes it flexible, extensible, and applicable to any number of other applications.

6 Acknowledgments

This work has been supported by NSF Graduate Traineeship Grant GER-9354869 to the University of Delaware, and a post-doctoral fellowship award from the Institute for Research in Cognitive Science (IRCS) at the University of Pennsylvania (NSF SBR 8920230). Much of this work was completed while the third author was a visiting scholar at IRCS and was supported by a grant from NSF (NSF SBR 8920230). We would like to thank Miriam Eckert for the many valuable discussions of this work.

References

Aone, C. & S. W. Bennett (1994). Discourse tagging tool and discourse-tagged multilingual corpora. In *Proceedings of the International Workshop on Sharable Natural Language Resources (SNLR)*.

Appelt, D. E. (1981). *Planning Natural-Language Utterances to Satisfy Multiple Goals*, (Ph.D. thesis). Stanford University. Also appeared as: SRI International Technical Note 259, March 1982.

Appelt, D. E. (1985). Planning English referring expressions. *Artificial Intelligence*, 26(1):1–33.

Carletta, J. (1996). Assessing agreement on classification tasks: The kappa statistic. *Computational Linguistics*, 22(2):249–254.

Condon, S. & C. Cech (1995). Problems for reliable discourse coding systems. In *Working Notes for AAAI Spring Symposium on Empirical Methods in Discourse Interpretation and Generation*, pp. 27–33. Stanford University.

Connolly, D., J. D. Burger & D. S. Day (1997). A machine learning approach to anaphoric reference. In D. Jones & H. Somers (Eds.), *New Methods in Language Processing*, pp. 133–143. Oxford University Press.

Dale, R. (1992). *Generating Referring Expressions: Constructing Descriptions in a Domain of Objects and Processes*. Cambridge, Mass.: MIT Press.

Dale, R. & E. Reiter (1995). Computational interpretations of the Gricean maxims in the generation of referring expressions. *Cognitive Science*, 18:233–263.

Day, D. S., J. Aberdeen, L. Hirschman, R. Kozierok, P. Robinson & M. Vilain (1997). Mixed-initiative development of language processing systems. In *Proceedings of the Fifth Conference on Applied Natural Language Processing*, pp. 348–355. Washington, D.C.

Eckert, M. & M. Strube (1999). Resolving discourse deictic anaphora in dialogues. In *Proceedings of the 9th Conference of the European Chapter of the Association for Computational Linguistics,* Bergen, Norway, 8–12 June 1999. To appear.

Ge, N., J. Hale & E. Charniak (1998). A statistical approach to anaphora resolution. In *Proceedings of the Sixth Workshop on Very Large Corpora*.

Grosz, B. J. & C. L. Sidner (1986). Attention, intentions, and the structure of discourse. *Computational Linguistics*, 12(3):175–204.

Marcus, M., G. Kim, M. A. Marcinkiewicz, R. MacIntyre, A. Bies, M. Ferguson, K. Katz & B. Schasberger (1994). The Penn treebank: Annotating predicate argument structure. In *Proceedings of ARPA Speech and Natural Language Workshop*.

McCoy, K. F. & M. Strube (1999a). Generating anaphoric expressions: Pronoun or definite description? In *ACL '99 Workshop on the Relationship between Discourse/Dialogue Structure and Reference, University of Maryland, Maryland, 21 June, 1999*. This volume.

McCoy, K. F. & M. Strube (1999b). Taking time to structure discourse: Pronoun generation beyond accessibility. In *Proceedings of the 21st Annual Conference of the Cognitive Science Society,* Vancouver, British Columbia, Canada, 19–21 August 1999. To appear.

McDonald, D. D. (1980). *Natural Language Production as a Process of Decision Making Under Constraint*, (Ph.D. thesis). MIT.

McKeown, K. R. (1983). Focus constraints on language generation. In *Proceedings of the 8th International Joint Conference on Artificial Intelligence,* Karlsruhe, Germany, August 1983, pp. 582–587.

McKeown, K. R. (1985). *Text Generation: Using Discourse Strategies and Focus Constraints to Generate Natural Language Text*. Cambridge, U.K.: Cambridge University Press.

Passonneau, R. (1996a). *Instructions for applying discourse reference annotation for multiple applications (DRAMA)*. Columbia University, New York, Dept. of Computer Science.

Passonneau, R. (1996b). Using centering to relax Gricean constraints on disocurse anaphoric noun ophrases. *Language and Speech*, 39(2):229–264.

Passonneau, R. & D. Litman (1997). Discourse segmentation by human and automated means. *Computational Linguistics*, 23(1):103–139.

Poesio, M. & R. Vieira (1998). A corpus-based investigation of definite description use. *Computational Linguistics*, 24(2):183–216.

Reichman, R. (1985). *Getting Computers to Talk like You and Me*. Cambridge, Mass.: MIT Press.

Reiter, E. (1990). Generating descriptions that exploit a user's domain knowledge. In R. Dale, C. Mellish & M. Zock (Eds.), *Current Research in Natural Language Generation*. London: Academic Press.

Sidner, C. L. (1979). *Towards a Computational Theory of Definite Anaphora Comprehension in English*. Technical Report AI-Memo 537, Cambridge, Mass.: Massachusetts Institute of Technology, AI Lab.

Strube, M. (1998). Never look back: An alternative to centering. In *Proceedings of the 17th International Conference on Computational Linguistics and 36th Annual Meeting of the Association for Computational Linguistics,* Montréal, Québec, Canada, 10–14 August 1998, Vol. 2, pp. 1251–1257.

Vilain, M., J. Burger, J. Aberdeen, D. Connolly & L. Hirschman (1995). A model-theoretic coreference scoring scheme. In *Proceedings fo the 6th Message Understanding Conference (MUC-6)*, pp. 45–52. San Mateo, Cal.: Morgan Kaufmann.