COMPUTATIONAL ASPECTS OF DISCOURSE IN THE CONTEXT OF MUC-3

Lucja Iwańska (GE), Douglas Appelt (SRI), Damaris Ayuso (BBN), Kathy Dahlgren (ITP), Bonnie Glover Stalls (LSI), Ralph Grishman (NYU), George Krupka (GE), Christine Montgomery (LSI) and Ellen Riloff (UMass)

INTRODUCTION

Discourse comprises those phenomena that usually do not arise when processing a single sentence. It appears to be the most difficult and probably the least understood aspect of automated message understanding. Five out of fifteen sites on a MUC-3 survey listed discourse as their main weakness and an area in which to concentrate future research. Virtually all systems presented here take a sentence-by-sentence approach to text understanding. Parsing and domain-dependent interpretation of sentences or sentence fragments (usually the latter) are followed by modules that attempt to connect these interpretations into a coherent whole. This paper gives an overview of the modules that make the transition from the interpretation of sentences to the interpretation of the text that contains these sentences. Systems presented in this paper exhibit various degrees of the following discourse understanding capabilities:

- identifying portions of text that describe different domain events; this includes the capability of recognizing a single event and the capability of distinguishing multiple events;
- resolving references:
  - pronoun references, e.g., finding the referent of *It* in the sentence *It took place this morning*,
  - proper name references, e.g., understanding that *Luis Galan* may be referred to as *Senator Galan*;
  - definite references, e.g., deciding what is the referent for *The attack* in the sentence *The attack took us by surprise.*
- discourse representation: representation at the message level.

Though these capabilities emerged[1] as critical for performing MUC-3 tasks they are important for any text understanding. MUC-3 specificity is reflected in the fact that one deals with terrorist events only, and that one tends to ignore information irrelevant for the MUC-3 slot fillers.

The systems exhibited much commonality in their approaches to accomplishing these tasks. For example, distinguishing different events is typically accomplished by some form of merging compatible events and resolving definite references. This commonality illustrates what was practically achievable [2] within limited financial resources and limited time for porting the existing modules or developing new ones.

Many more discourse understanding capabilities exist in the literature than were exhibited in the systems that participated in MUC-3; for example, explicit script (plan) knowledge, speakers' beliefs, reasoning about

[1]The sites represented by the authors of this paper were asked to describe their discourse-related capabilities. We did not specify which problems should or should not be included.

[2]We consider the fifteen systems that participated in MUC-3 as a representative sample of implemented natural language text processing systems in the USA.

actions, discourse structure etc [15] [6] . Moreover, other versions of some of the text understanding systems represented in MUC-3 have incorporated such capabilities for discourse analysis in narrow domains (see references in the following sections). While there have been implementations of much more detailed discourse analysis for narrow domains [26] [11] no implementation of discourse analyses for broad domains exist which could guide efforts on the MUC-3 domain.

If implemented, those missing capabilities would most certainly improve system performance. There are several reasons why the MUC-3 versions of the participating systems lack them at this point. First, given time constraints (it has been proposed to hold MUC annually) and the fact that the existing capabilities directly contribute to the performance in the MUC-3 domain, their improvement takes priority over implementing new ones. Second, the significant effort required to research and implement any of these capabilities may not be proportional to the increase in performance.

Many aspects of discourse, e.g., constructing a theoretical and computational model of the relation that holds between different sentences of a text, are still open research problems [18] [22]. A typical MUC-3 message contains fairly unrestricted text with long and complex sentences. A fair amount of knowledge is required in order to correctly interpret it. Both solving theoretical aspects of these problems and implementing them on such a broad and complex domain as MUC-3 [3] is an extremely challenging goal, practically unachievable within the limited resources most systems have at their disposal.

The purpose of this paper is to:

1. present and compare the approaches used in the systems represented here in addressing the three discourse understanding capabilities mentioned previously
2. briefly discuss directions for future discourse research.

This paper is organized as follows: First, seven sites describe their discourse modules or discourse-related capabilities that were used for the official MUC-3 run [4]. The two final sections discuss the identified tasks, along with the solutions offered by different sites and plans for future research.

BBN'S PLUM: THE DISCOURSE COMPONENT

The discourse component of BBN's PLUM message processing system (also described in BBN's System Summary in these proceedings) is responsible for extracting events of interest from what is normally very fragmentary syntactic and semantic information. Using a model of the relevant events in the domain (e.g., "murder"), it attempts to derive any information which was not already found by the semantic interpreter. The primary output of the discourse module is a sequence of frame-like event structures, which are in turn the input to the template generator.

The discourse module of PLUM is new (as are all the other PLUM components except the parser), and in flux—what is reported here is its state as of the MUC-3 test. This module is a significant departure from the discourse module in BBN's Janus [3] and the related module in BBN's Delphi [4, 5]. The discourse components of those systems, which were developed primarily for question-answering applications in limited domains, are able to take advantage of having complete analyses of the input sentences. For example, syntactic information is used to constrain intra-sentential anaphora [19]; complete semantic representations (including quantification information) are used in generating discourse entities in a principled way [2]; and centering heuristics [7, 12] are used for tracking focus, improving anaphora resolution.

[3] For a more detailed characterization of the MUC-3 corpus see the introduction to these proceedings.

[4] If the description includes an existing module not used for MUC-3, then a brief explanation as to why it was not used is given.

A fundamental characteristic of PLUM, however, is the emphasis on fragmentary processing at all levels, and the assumption that a non-trivial amount of an input message may not be understood. This led us to focus more on approaches that depend less on reliable complete understanding of the text. Thus, for example, most of the mechanisms mentioned above from our previous work are not yet present in PLUM. Instead we are taking advantage of expectations encoded in representations of the events in the domain in order to reconstruct missing information of relevance to the templates from less than perfect syntactic and semantic processing[5].

Our discourse module operates on the output of the semantic interpreter, which operates on fragments resulting from parsing and fragment combination. Fragments are non-overlapping and together span all the input. The interpreter assigns a list of semantic forms to each fragment, where a form has 4 fields: a category (e.g., KNOWN-ENTITY), a variable (unique identifier, e.g., ?45), a semantic type (e.g., PERSON), and a list of predicates (e.g., (NAME-OF ?48 'FMLN')). For a particular fragment, the generated list of forms contains all the semantic information local to the fragment. The discourse module performs the following primary steps in generating discourse events:

1. For each semantic form (processed roughly in the order they arise in the text):
    - reference resolution to a previous entity is attempted if the form corresponds to a pronoun,
    - co-reference with a previous proper name is checked if the form is that of a proper name,
    - if the input corresponds to a new entity stemming from a noun phrase, it is added to a table of entities,
    - the predicates in the form are added to a discourse predicate database which supports unification of variables (used in reference resolution) and quick lookup, and
    - if this form is relevant, the current list of discourse events is updated—this may involve creating an event, filling slots, and/or merging events as necessary.
2. Once all semantic forms have been processed, heuristic rules attempt to fill in any unfilled event slots.

Possible referents for a pronoun are looked up in the entity lists (which themselves can be resolved pronouns) corresponding to the current and previous sentences—they must match the pronoun in semantic type and number. Given the assumption that understanding is fragmentary, we expect that searching further would increase the possibility of obtaining an incorrect referent, though we have not yet verified this. A global list of the proper names corresponding to persons or unknown entities is kept, and any new name is looked up in the list for a possible co-referent. When a reference/referent pair is found, their variables are unified in the database. In case of reference ambiguity the closest referent is used; in the future we would like to maintain parallel discourse databases which would represent any ambiguity or divergent inference.

A semantic form may affect the list of current discourse events in various ways. It may create a new event if it is one of the semantic types of interest and it passes any arbitrary tests associated with the corresponding discourse event type (e.g., a terrorist incident may not have a terrorist target). It may also fill a slot of an event. Event slots are usually filled in directly, by finding a predicate in the semantics which corresponds directly to a discourse-event slot, and relates the form which 'triggered' the event with a filler. Normally only information local to the fragment will be available for filling slots in this way, since the semantic interpreter operates at the fragment level (through the unification done during reference resolution, some non-local slot-filling information may also be available). As an example, in the first sentence in

DEV-MUC3-0001, '**THE ARCE BATTALION COMMAND HAS REPORTED THAT ABOUT 50 PEASANTS OF VARIOUS AGES HAVE BEEN KIDNAPPED** BY TERRORISTS OF THE FARABUNDO MARTI NATIONAL LIBERATION FRONT [FMLN] IN SAN MIGUEL DEPARTMENT,'

[5]Current work in the ATIS (Air Traffic Information Service) application of Delphi is also making use of event expectations in order to better use information from context, and to aid in fragmentary fall-back processing.

the portion delimited by ** is parsed into a single fragment. One of its relevant semantic forms and the discourse event it generates follow:

```
(KNOWN-EVENT ?45 KIDNAPPING
  ((TI-PERP-OF ?45 ?33) (OBJECT-OF ?45 ?18)
   (PERFECT-TENSE ?45 "HAVE") (PRESENT-TENSE ?45) (PASSIVE ?45)))

Discourse event:
(KIDNAPPING trigger:  ?45  "HAVE BEEN KIDNAPPED"
  slots: (TI-PERP-OF: ?33 "TERRORISTS")
         (OBJECT-OF:  ?18 "ABOUT 50 PEASANTS"))
```

Once a new event is created and all the local slots found by the semantic interpreter are filled in, it will be merged (if possible) with the closest existing event of the same type whose filled slots are compatible. Information which would make two events incompatible may be present in the new (or old) event's surrounding text, but to be recognized (as of MUC-3) it needs to be parsed into the same fragment as the event trigger *and* the semantic interpreter needs to interpret it (and its relation to the trigger) correctly. Note that the merging process frequently has the effect of performing resolution of definite references—for example a later mention, e.g., 'the kidnapping', will get merged with an earlier event if the information found with the two is compatible. However, only events of the same type are merged, which currently prevents a vague reference from merging with a previous specific event, i.e., if instead of 'the kidnapping' the reference is 'the attack'. Analyzing further the conditions under which such things should be merged is one of the topics we will address for MUC-4. As an example, continuing DEV-MUC3-0001, in the second sentence the portion in ** was parsed into a fragment:

'**ACCORDING TO THAT GARRISON, THE MASS KIDNAPPING TOOK PLACE ON 30 DECEMBER** IN SAN LUIS DE LA REINA'.

A new event first gets created for this kidnapping and is then merged with the previous kidnapping event.

Finally once all the forms have been processed, heuristic rules are applied to try to fill in any unfilled discourse event slots. Most of these rules simply look outward from the forms which triggered the event (more than one form triggers an event when event merging has occurred) looking for any form of the correct semantic type and which passes arbitrary tests defined for that slot. A score is assigned to indicate how far away the filler was found. A global parameter defines a cutoff distance for the search—for MUC-3 the search went no further than adjacent paragraphs. When multiple possible fillers are found, the template-filler will pick the one with the highest score (the closest one), and in the case of ambiguity will arbitrarily pick one. Note that this heuristic slot filling is done *after* any event merging has occurred. Because we have not yet determined the reliability of this fall-back heuristic, only the information found directly by the semantic interpreter is considered reliable and therefore used during merging. Continuing with our example, below is a portion of the final kidnapping event after the whole message is processed:

```
(KIDNAPPING (?45 ?73) "HAVE BEEN KIDNAPPED" "THE MASS KIDNAPPING"
  slots: (TI-PERP-OF:?33 score: 0 "TERRORISTS"
                  :?32 score: 1 "FARABUNDO MARTI NATIONAL LIBERATION FRONT")
         (EVENT-LOCATION-OF: ?54 score: 2 "SAN MIGUEL DEPARTMENT")
         (TI-RESULT-OF: ?182 score: 6 "WERE WOUNDED")
         (OBJECT-OF: ?18 score: 0 "ABOUT 50 PEASANTS"))
```

A score of 0 indicates a filler which was found directly by the semantics, 1 is a filler found in the same fragment as a trigger (though it was not connected by the semantics), and 2 is a filler found in the same sentence, but a different fragment. The injury result from 'WERE WOUNDED' is incorrect (it belongs with a separate incident), the score of 6 indicates it was found in an adjacent paragraph.

Although many aspects of our discourse module are preliminary, we were impressed with how well it performed for MUC-3. However, there are many areas we want to pursue in developing the module further,

for example, we plan to add some tracking of tense and time, explore further the issues of merging of events, and add a representation of ambiguity at various levels.

GE'S NLTOOLSET: DPM AND TRUMPET

Discourse processing is the ability to understand connected text. This includes the ability to recognize fragments of text that describe individual events. We do discourse-related processing in two stages: before parsing, the Discourse Processing Module (DPM) [20], produces an initial segmentation of the input story into fragments relevant for different events; then, after parsing, the top-down expectation module (TRUMPET) [28] uses special domain knowledge to connect the representations of individual sentences relating to the same event.

Our intuition is that discourse should drive text understanding, including parsing and interpretation. Placing DPM before parsing is the first step toward this mode of text understanding. DPM and TRUMPET currently overlap somewhat in their discourse-related processing. We maintain them as independent modules in order to ensure fewer errors by employing multiple strategies, to evaluate the performance of the system in different configurations and to experiment with different control strategies.

DPM

DPM is a heuristically driven program that identifies segments of a story describing different domain events. The input to DPM is a story with sentence and paragraph boundaries marked. DPM outputs text segments, where a segment is a set of fragments of the story that could describe a single event.

First, DPM searches for phrases that identify domain events. For example, the expression *set the building on fire* identifies an *arson* event, and the expression *serious condition* may signify the continuation of any event. *Primary phrases* are used to decide whether a new event starts or the previous event continues. *Secondary phrases* are used to decide the closing boundaries of the fragments judged as relevant to the previous event. Three factors influence the decision as to whether a subsequent paragraph or sentence continues describing an event or starts describing a new event: (1) whether an event type that the current primary phrase has identified is different from the previous one; for example, *bombing* versus *robbery* (*attack* is the only type that is in general consistent with all other event types); (2) the definiteness of a primary phrase; for nouns, it usually coincides with the definiteness of the noun phrase that describes an event; (3) the amount of text with no secondary phrases that separates the current and previous primary phrases. The result of this first step is a set of segments, each containing only one text fragment.

Second, the program uses several heuristics to further split the segments by identifying new events. For example, one heuristic is based on the occurrence of cue phrases, e.g. *meanwhile*, combined with the occurrence of a primary phrase not in definite form.

Third, the program attempts to merge certain segments, i.e., replace them with a new segment which combines all their text fragments. For example, DPM merges two consecutive segments if they have consistent types and the first primary phrase in the second segment is in definite form. In certain cases, DPM recognizes events whose description is embedded in the description of other events. In this case, the order of the segment fragments does not correspond to the original text.

We have tested DPM on 500 stories from the MUC-3 corpus (400 from DEV and 100 from TST1). We attribute the good performance of DPM to the fact that it combines rudimentary anaphora resolution and a rudimentary notion of topic shift with other clues about the discourse structure.

TRUMPET

TRUMPET is a domain-driven program that coordinates with the parser in order to interpret connected text. It uses the following discourse-related features: (1) a simple anaphora resolution mechanism; (2) basic knowledge of causality that connects events with their effects; (3) knowledge of common scenarios in a domain that connect events with their subevents; and (4) rudimentary temporal and spatial reasoning that determines event compatibility.

The input to TRUMPET is the conceptual representation of each sentence. As its output, TRUMPET produces a set of structures containing information about distinct domain events.

For each sentence, TRUMPET applies rules to map the conceptual representation into domain *event structures*, or *EStructs*. Primary EStructs roughly correspond to the MUC-3 templates (e.g. *kidnapping*), and secondary EStructs represent effect (e.g. *injury*) or subevents (e.g. *hostage-release*). Each EStruct contains expectations about what information may also appear in the text. These expectations weakly represent cause-effect and event-subevent relationships which are used to connect EStructs together. CEStruct, a set of connected EStructs, corresponds to a single domain incident. TRUMPET focuses its processing on the active CEStruct of the message (i.e. the most recently encountered), and stores previously deactivated CEStructs for final post-processing.

After the representation of a sentence is mapped to EStructs, TRUMPET tries to connect them. Next, TRUMPET checks the compatibility of the current primary EStructs with those of the active CEStruct. For MUC-3, two primary EStructs are incompatible if either (1) their incident types do not match; (2) their dates are different; or (3) their locations are different. An *attack* incident type matches any other incident type. If the current EStructs are compatible, then TRUMPET merges them with the active CEStruct. By merging EStructs, TRUMPET is in effect resolving certain references. If the current EStructs are not compatible, TRUMPET deactivates the current CEStruct and creates a new active CEStruct which is initialized with the current EStructs.

TRUMPET performs well when the text contains descriptions of domain events which include "distinguishing" conceptual information. However, TRUMPET's performance suffers in the absence of such information, since it ignores discourse information in the text.

Interaction between DPM and TRUMPET

TRUMPET has difficulty distinguishing consecutive events of compatible types if it does not receive enough key information to properly judge compatibility. This may happen either when parsing or interpretation errors occur, or when key information is missing or distributed throughout the text. DPM addresses these problems as well as increases our capability to recognize discourse information in the text. TRUMPET uses the text segmentation produced by DPM to separate conceptual representations concerning different events. However, in certain cases, TRUMPET may override this segmentation in an attempt to overcome DPM limitations or errors. We present four examples that demonstrate both the capabilities of the two modules and their interaction.

1. DPM prevents TRUMPET from merging two different events. Message DEV-MUC3-0340 contains consecutive descriptions of two bombing events. DPM recognizes that these events are different based on the indefiniteness of *explosion* and marks the boundary as shown below. TRUMPET would fail to distinguish these events due to its rudimentary spatial reasoning component, which would fail to differentiate the two locations.

THE MOST SERIOUS INCIDENT WAS CAUSED BY UNIDENTIFIED PERSONS WHO THREW A GRENADE AT A PRIVATELY OWNED PASSENGER BUS ON KILOMETER 6 OF THE ROAD LINKING GUATEMALA CITY WITH PUERTO BARRIOS, ON THE ATLANTIC COAST. ONE PERSON WAS KILLED AND AT LEAST FIVE WERE WOUNDED IN THE ATTACK. THE FATALITY WAS AN UNIDENTIFIED WOMAN.

boundary

ANOTHER EXPLOSION TOOK PLACE SIMULTANEOUSLY IN ZONE 5 OF GUATEMALA CITY, IN FRONT OF THE CHURCH OF THE HOLY PRIEST OF ARS, WHOSE PARISH PRIEST, JOSE MARIA RUIZ (FATHER CHEMITA), WAS PREPARING TO SAY MASS. [the continuation of the description omitted]

2. DPM prevents TRUMPET from over-splitting. Message TST2-MUC3-0004 contains a long description of one *bombing* event, with an embedded single-sentence description of an *attack* and a *murder*. DPM correctly recognizes the continuation of the *bombing* event and prevents TRUMPET from treating these descriptions as two distinct *bombing* events. Unlike the previous example, TRUMPET would not be able to merge the continuation with the first segment because it fails to recognize that the descriptions have the same target.

[long description of a bombing event omitted]

boundary

THE "ZARATE WILLKA" GROUP CLAIMED RESPONSIBILITY IN AUGUST 1988 FOR THE ATTACK AGAINST THEN U.S. SECRETARY OF STATE GEORGE SHULTZ AND THE KILLING OF TWO MORMON MISSIONARIES.

boundary

TODAY'S BOMB EXPLOSION DAMAGED THE SHELVES OF A BOOK STORE, THE "PEOPLE'S PERUVIAN BANK," THE STATE BANK [NOT FURTHER IDENTIFIED]; THE MARISCAL BALLIVIAN BUILDING, AND OTHER SHOPS, ALL OF WHICH ARE LOCATED NEAR THE U.S. EMBASSY. [the continuation of the description omitted]

3. TRUMPET further refines DPM segmentation. Message TST1-MUC3-0040 contains the descriptions of four distinct events as shown below. DPM produces a single segment: (1) the descriptions of the first two (*bombing* and *bombing*) and the last two events (*attack* and *bombing*) are not split since they are within the same sentence; and (2) the fourth event cannot be distinguished because *blew up* is neither definite or indefinite. TRUMPET is able to distinguish the three bombings because they differ in location and time. However, TRUMPET is unable to distinguish the *attack* from the third *bombing* because it has no explicit temporal or spatial information.

[text omitted]

boundary

ON 2 SEPTEMBER, A CAR BOMB DESTROYED THE INSTALLATIONS OF "EL ESPECTADOR" NEWSPAPER IN BOGOTA AND 5 WEEKS LATER ANOTHER CAR BOMB CAUSED SIMILAR DAMAGE TO THE "VANGUARDIA LIBERAL" NEWSPAPER IN BUCARAMANGA, IN NORTHEASTERN COLOMBIA.

SEVERAL SECURITY FORCES HEADQUARTERS HAVE ALSO BEEN HIT BY THESE ATTACKS AND AN AVIANCA AIRLINES PLANE BLEW UP IN MIDAIR ON 27 NOVEMBER KILLING ALL 107 PASSENGERS AND CREW.

boundary

[text omitted]

4. TRUMPET overcomes DPM over-splitting. Message TST1-MUC3-0083 contains a long description of one *murder* event, with an embedded description of a *kidnapping* event. The text segmentation of DPM is shown below. DPM fails to recognize that the descriptions of the *murder* after the *kidnapping* are a continuation of the same *murder*. TRUMPET is able to rectify the problem by merging the last two segments with the first one, because the descriptions are not incompatible and they have the same target.

SAN SALVADOR, 26 APR 89 (ACAN-EFE) - [TEXT] A SALVADORAN COURT YESTERDAY PRESENTED A SWORN DECLARATION BY PRESIDENT JOSE NAPOLEON DUARTE IN WHICH HE LINKS RIGHTIST LEADER ROBERTO D'AUBUISSON TO THE MARCH 1980 ASSASSINATION OF ARCHBISHOP OSCAR ARNULFO ROMERO.

[DPM deleted irrelevant text]

THE LIST OF PLOTTERS INCLUDES RIGHTIST DEPUTY D'AUBUISSON, FOUNDER OF THE NATIONALIST REPUBLICAN ALLIANCE (ARENA) – THE WINNER IN THE RECENT ELECTIONS – AND CAPTAIN ALVARO SARAVIA, WHOM THE PRESIDENT ACCUSED OF BEING THE MASTERMINDS OF THE ASSASSINATION OF THE CLERGYMAN.

boundary

THE LIST ALSO INCLUDES THE NAMES OF CIVILIANS INVOLVED IN THE KIDNAPPING OF SEVERAL SALVADORAN BUSINESSMEN EARLY IN THE DECADE.

boundary

ACCORDING TO DUARTE'S DECLARATION, 1ST INFANTRY BRIGADE SOLDIERS ARRESTED THE 24 COUP PLOTTERS IN 1980, AND SEVERAL DOCUMENTS WERE SEIZED FROM THEM, INCLUDING A NOTEBOOK WITH PLANS FOR KILLING ROMERO.

boundary

[DPM deleted irrelevant text]

THE ARCHBISHOP WAS KILLED BY A SNIPER WHILE SAYING MASS AT A HOSPITAL FOR CANCER PATIENTS.

Future Directions

We plan to further transition toward discourse-driven message understanding. We want to be able to take advantage of the type of a story and in some cases, e.g., journal articles, to recognize and utilize their structure. We plan to develop a representation of the stories that would allow the system to draw logical and plausible inferences necessary to handle questions about the story that people genrally are capable of answering. Producing MUC-3 templates should be properly subsumed by such a representation. Such research will significantly extend current capabilities of our system, also for other tasks or domains [6]. We will also investigate the possibility of doing scenario analysis on a larger scale. Other potential implementational plans include improving temporal and spatial reasoning and distributive anaphora resolution.

ITP: DISCOURSE PROCESSING

The hallmark of the ITP approach is detailed, formal linguistic analysis. ITP handles discourse phenomena on a number of levels: the formal semantic level with Discourse Representation Theory [21] [1] and anaphora resolution, the pragmatic level with naive semantics and coherence, and the discourse structure level with segmentation and topic identification.

Discourse Representation Theory directly displays in logical form the difference between given and new information. Objects and events which have already been introduced into a discourse are given reference markers which are mapped to the entity (in a world model) which a discourse entity denotes. Those entities which are not accessible for anaphor resolution are structurally noted in the representation. While DRT does not by any means solve all of the problems of formal semantic representation, it refreshingly recognizes as a formalism that meaning extends beyond the sentence.

A Discourse Representation Structure (DRS) consists of reference markers (one for each entity introduced into the discourse), and conditions (predications of those entities). The critical point in the context of data

[6] We hope that one of the future MUC's will reformulate the task to the capability of answering certain questions that are answerable after reading a story. This would more closely correspond to understanding of a story

extraction is that formal semantic properties of the discourse are directly displayed in the DRS. For example, a negated verb phrase is represented in a subDRS which is inaccessible to anaphor resolution and can be ignored in reasoning. The sentence 'The government did not find a bomb' results in a DRS

drs1: { the1 }	government(the1), not(drs2)	drs2: { a1, e1 }	bomb(a1), e1 find(the1,a1)

The truth conditions for this DRS involved neither a bomb nor a 'finding' event in the actual world. Similarly, sentences with semantically negative verbs such as 'deny' are translated into formal representations with the appropriate truth conditions.

DRT is designed to formalize a variety of phenomena which complicate anaphor resolution. The ITP anaphor resolution algorithm goes far beyond the mechanisms offered by DRT, incorporating surface syntactic properties (a preference for resolving to prior main clause subjects, etc), the formal semantic properties reflected in accessibility properties of DRS's, and naive semantic properties. The use of naive semantics in anaphora resolution is illustrated in a text like 'The government charged the guerrillas with illegal acts. They had stolen food'. In finding the antecedent of 'they', with three possible plural NP's as candidates, ITP's word sense disambiguation algorithm uses naive semantics to find the verbal reading of the verb charge. 'charge with' is then an accusation making 'guerrillas' agent of 'acts'. Naive semantics knows that 'acts' refers to events, excluding that NP as antecedent of 'they'. The algorithm prefers prior subjects, but it also reasons about implications of events represented in the naive semantic lexicon. Those implications suggest that as 'stealing' is an illegal act. Thus plausibly the intended subject of 'steal' is 'guerrillas' rather than 'government', as they are the agents of illegal acts in the prior sentence.

In DRT different types of entities are distinguished: individuals, events, states, propositions and so on. The ITP approach to discourse makes use of these distinctions for anaphor resolution and coherence. In the example above, the noun 'act' gets an event type reference marker rather than a individual type reference marker, providing the mechanism for excluding it as antecedent of a pronoun which is subject of a verb requiring sentients as subject. Conversely, in a text like 'There were bombings last night in San Isidro. The attacks took place after midnight', 'bombings' and 'attacks' both get event type reference markers, so that the anaphor resolution algorithm looks for plausible events rather than individuals as potential antecedents of 'attack'. The coherence relation assignment algorithm [8] similarly seeks to find relationships between event or state entities.

ITP segmentation is the result of detailed linguistic analysis from syntax through formal semantics to discourse structure. For MUC-3 ITP implemented a mechanism for finding temporal and locative expressions in the text. For each clause the event reference marker in the DRS had a temporal and/or locative predication added if any new information was added in the clause. This made segmentation very simple: new time, new place or segmenting clue phrase (such as 'meanwhile' or 'in summary') means new segment. These three indicators signal that a new segment has begun. Thus in message 99, new segments began at sentences 4,8,11 and 14. Clues were:

Sentence 1: the place changes from San Isidro to Orrantia Sentence 8: the time changes with the phrase 'in the past' Sentence 11: the time changes with the phrase 'some three years ago' Sentence 14: the time changes with the adverb 'today'

These clues were recognized automatically by comparing the temporal and locative predications of the event just previously added to the DRS, with the temporal and locative predications of events introduced in the current sentence. Segmenting phrases are recognized as adverbial predications of an event introduced by the current sentence. Antecedents of anaphors are sought only within the segments.

The five sister segments discovered by this algorithm (see tree in the ITP System Summary) enabled the template-filling code to accurately and automatically segregate the various terrorist events from each other and correctly posit three terrorist events and three templates. The template reasoning code was invoked

segment by segment. All information in the same segment was assumed to apply to the same terrorist incident, if any. This approach avoids problems of 'template merging' which plague systems which invoke a terrorism template for every terror word in the text.

The ITP approach to segmentation employed in MUC-3 used only three of the six factors which are important clues to discourse structure: 1) segmenting clue phrase 2) change of time, 3) change of place, 4) certain changes of tense, 5) certain changes of aspect, 6) shift of topic. These clues to discourse structure were determined in an empirical study of a novel and a corpus of newspaper articles, following a survey of the discourse literature.

For MUC-3, the algorithm was simplified because of the syntactic complexity of the texts. Use of topic identification would have permitted an even finer-grained, more correct segmentation. This is because in many texts a dominating topic segment can contain antecedents of anaphora, and play a role inside dominated segments. In a proper anaphora resolution algorithm, antecedents should be sought in the current segment and certain dominating segments. The tree ITP constructed for message 99 does not fully recover the discourse structure of message 99, which actually has a dominating segment consisting of sentences 1,2,3 and 14. Sentence 15 'pops' up to the dominating segment [13] [14]. If this dominance structure is represented, it is then possible to find the correct antecedent for 'the attacks' in sentence 5, which refers to the attack on the two embassies. Its antecedent can be found in sentence 1 of the dominating segment.

Future research will enable the ITP discourse segmentation algorithm to recover discourse dominance relations and become sensitive to tense, aspect and topic.

LSI'S DBG MESSAGE UNDERSTANDING SYSTEM: DISCOURSE PROCESSING

LSI's approach to discourse processing is based on the notion of **text grammar**, which was originally defined by Propp [27] and has more recently been elaborated by van Dijk [9], [10], and others.

A text grammar is comprised of a set of rules for analyzing or generating texts conforming to a given rule set. In generation, the rules embody the text plan. In analysis (e.g., understanding the text of military messages in various domains), the rules embody a set of expectations about the structure and content of a discourse that can provide a valuable knowledge source for resolving object, temporal, and spatial references and for coping with unexpected inputs (new or erroneous material). The rules embody a set of expectations about the structure and content of a discourse at all levels of granularity, ranging from phrase to message or text level.

Our notion of text grammar has served us well in previous work on military message corpora, as described in our system summary in these proceedings. Detailed descriptions of the text grammars for two Air Force corpora are contained in Montgomery and Glover [25] and in Stalls et al. [29]. For MUC3, however, text grammar rules were implemented in the DBG (Data Base Generator) system only at the message level at the time of final testing (May 1991), since our resources had been focused on a major redevelopment of the grammar and parsing mechanism to a syntactic analyzer based on Government Binding principles (as described in our system summary and site report in these proceedings).

The DBG system output for the processing of each message is a text level knowledge representation for the message content consisting of a set of instantiated event and object frames called templates. [7] This set of templates is an instantiation of the text grammar that reflects the structure and content of the given message.

[7] It is important to note that the term template in the DBG system is a label for the generic message level semantic and pragmatic representational units, not an application-oriented structure like the MUC templates. The DBG templates are the glass box output or internal representational output, as opposed to the MUC templates, which are black box outputs mapped to the external representation required by a given application: currently, extraction of specific data elements for describing terrorist incidents in 9 Latin American countries.

The information contained in these internal templates is then mapped to the external application-oriented data structures, in this case the MUC-3 templates.

The first internal template generated by our system is a **Report** template. The **Report** template is a 'meta template', which considers the message itself as a reporting event over and above the actual events described in the message. Such header information can be extremely complex, as in military messages for the air activity and space event domains (see our system summary and references). The header may even supply information assumed, but not provided explicitly within the accompanying message text, such as specific equipment designations, which are referred to only by generic names in the text. Because our text grammar for MUC-3 was only partially implemented at the time of testing, just a few of the report attributes identified (e.g., date of the report and country of the reporting source) were actually recorded in the **Report** templates.

In addition to header slots containing information about the reporting event itself, the **Report** template contains a set of slots that point to the internal (DBG) event/action templates generated for the text. These slots may be qualified by attributes reflecting degree of certitude concerning the event pointed to (e.g., 'probable', 'possible', 'apparently', etc.). In turn, the event/action templates pointed to by the **Report** template contain slots that contain or point to objects and actions related to the given event (e.g., agent, patient), (which may also be qualified by degree of certitude expressions, as described above). So, for example, for Message TST2-MUC3-0055, a 'kidnap' event/action template was generated, with an agent slot filled by by 'commando of the pro-che guevara group tupac amaru revolutionary movement' and a patient slot filled by 'delgado parker'. All of these templates are instantiated based on object and event/action frames in the semantic frame hierarchy, which includes class membership, part-whole, and other semantic relations among frames and slots.

The text grammar provides rules for **Action** and **Object** template generation and instantiation, as well as rules for locating information in the message text. The Template Unexpected Input Module (TUX) of the DBG system, which tries to fill empty template slots using unexpected information encountered in the text, uses discourse information about text (e.g., form of the information, its location within a particular text segment) to assist in determining whether a particular unused input string is a possible valid instantiation for a particular slot. This module was not implemented in the redesigned DBG system at the time of MUC-3 testing, however we have used it previous versions of our system to recover coordinate locations and other specific expected information occurring in unexpected syntactic and semantic contexts in military domains. For MUC-3, the TUX module could be used, for example, to retrieve such expected information as the possible identities of perpetrators and victims from event summary sentences even when the syntactic parse fails badly. This capability is particulary valuable for MUC-3 because many such sentences are syntactically quite unwieldy.

In the MUC3 corpus, the class of messages containing the majority of the relevant material consists essentially of reports of events in journalistic style, where the initial sentence summarizes the most salient features of the event, and following sentences give more details. In this respect, this discourse structure is similar to the air activity message corpus described in Stalls et al. [29]. In this domain, apart from the subject line or title sentence, which is essentially a press-style headline summarizing the first sentence, the text grammar or discourse model for an air activity report comprises two types of sentences:

1. a *summary* sentence type, which gives an overview of the activity;
2. a *flight event* sentence type, which describes the individual flight events that comprise the activity.

Each unique activity description in a message will begin with a *summary* sentence, followed by 0 - n *flight event* sentences. These sentence types are syntactically and semantically distinct, and both are quite complex. Also – although some messages describe a single activity carried out by one aircraft or one group of aircraft – more typically, two or more activities involving two or more groups of aircraft are described. Thus references to aircraft, activities, time, and space become extremely complex to sort out, paralleling

the problem of resolving references to the various sets of incidents/humans/groups/inanimate objects, etc., represented in the MUC messages.

Another complication found in messages of this type is the introduction of contextually-relevant but non-primary incidents for the sake of comparison or assessment. An example is the following sentence from Message DEV-MUC3-0008:

```
CASTELLAR IS THE
SECOND MAYOR THAT HAS BEEN MURDERED IN COLOMBIA IN THE LAST 3 DAYS.
```

In [25], we discuss the implications for processing of such statements of assessment as they occur in the space event corpus, as, for example, in: 'Terrex 592 is the third low density crop enhancement/fifth generation corn agricultural satellite (BR02E) launched this year'.

A text grammar for the MUC-3 domain is difficult to construct because there is little similarity of structure across messages of a given type. Military messages, for example, are typically designed to convey particular types of information precisely, whereas the broader, journalistic function of the MUC-3 messages makes the particular details mentioned less predictable, and stylistic variation appears to be more highly valued in a non-military domain, such as MUC-3. Some of the MUC messages seem to have been translated from Spanish, which introduces additional variations. Moreover, the sentences in the MUC-3 messages are characterized by a non-hypotactic or 'loose' syntactic structure, relying heavily on postposed apposition, coordination, and subordination, which makes them cumbersome to process. An example is the MUC-3 *summary* sentence from Message DEV-MUC3-0008:

```
RICARDO ALFONSO CASTELLAR, MAYOR OF ACHI, IN THE NORTHERN DEPARTMENT OF BOLIVAR, WHO WAS KIDNAPPED ON 5 JANUARY,
APPARENTLY BY ARMY OF NATIONAL LIBERATION (ELN) GUERRILLAS, WAS FOUND DEAD TODAY, ACCORDING TO AUTHORITIES.
```

In comparison, modifiers of noun phrases in the air activities domain (whether attributive or event-related) tended to be preposed and tightly constructed, as in the following example of a *summary* sentence:

```
ONE UGANDAN KAMPALA AIR ARMY HEAVY BOMBER AVIATION DIVISION (KA123) GULU (0335N 03315E) B-60 (BUFF C)
AND TWO UGANDAN KAMPALA AIR ARMY HEAVY BOMBER DIVISION (KA456) MASAKA (0000N 00000E) BASED,
TORORO (0000N 00000E) DEPLOYED, F-TYPES (BUNNY) CONTINUE ACTIVE OVER THE GULF OF NDEGE ON 8 AUGUST 2023.
```

The summary sentences in the two domains exhibit other differences as well. In the air activities example above, extremely complex descriptions are used to specify the objects constituting the main topic of the message, which are referred to anaphorically by very simple expressions later in the discourse (typically: 'the B-60', 'the F-types'). In the MUC-3 texts, the simpler description (e.g., 'the kidnapping') often appears first, as a sort of summary of a more detailed description appearing later in the text.

In addition, the MUC texts contain many distributive referring expressions, indicating a whole sequence of events, which might be limited to that described in the foregoing paragraphs of a given text, or might also include events described earlier in other texts, such as 'bloodbath' or 'violence' (e.g., 'deputies ask for end to violence'). This type of referring expression did not occur in the air activities messages, although we have noted similar anaphora in other military domains (e.g., 'additional activity was observed in the coastal area').

The difficulty of distinguishing events in multiple event reports is more complex for the MUC-3 messages than for some other domains. The description of an event in the MUC-3 corpus may consist of an NP (e.g., 'Galan's murder', a sentence, a paragraph, several paragraphs, or an entire message. In contrast, in military messages the paragraphs or sections describing separate events or sets of events are generally separately numbered. Even where this convention is violated, it is usually possible to detect event boundaries based on the text grammar (e.g., encountering a *summary* type sentence in the air activities messages). Also, in

military message corpora, message types exist to report a type of event or activity that continues to occur. In the MUC domain, on the other hand, the prominent events and objects–terrorist incidents and human beings–are by their nature unpredictable.

Because of the homogeneity of military message content and structure, a text grammar is an especially valuable tool. It can predict the structure of the knowledge representation (i.e., the internal template set) for a given message/event type that can be identified early on in processing (e.g., launch or deorbit messages in the space event corpus), as well as where in the text to find what kinds of information. This same degree of predictability cannot be achieved for the more heterogeneous MUC-3 corpus, for the reasons given above.

This does not mean, however, that a text grammar approach has little to contribute to processing a corpus like MUC-3. On the contrary, such a corpus requires an array of well-developed analytical tools because no single approach has as yet been shown to be completely successful. It appears that a text grammar for this type of corpus has to be more dynamic than that for military messages, and will probably be more useful for some text types than for others (e.g., event reports vs. speeches). Rather than relying on message type or paragraph numbering, the text grammar could identify summary sentences or paragraphs (usually the first paragraph of a message or the first describing a 'new' event within a message) for example, for critical event reports. As noted previously, these text segments usually contain basic information about the event type, the perpetrators, and the targets of the attack. Later sentences and paragraphs then fill in the details. Finally, there may be one or more assessment sentences, which compare the primary event to other similar events. Also, certain critical event reporting combinations are more common than others, such as kidnapping-murder, and multiple bombings.

Although the utility of a text grammar approach has yet to be established for this corpus, we envision such an approach to be useful in sorting out which event particular information in a text describes; guiding analysis of syntactic and semantic structure of different sentence types (e.g., summary sentences introducing a new event vs. assessment sentences); and providing a model for text level analysis and generation/consolidation of knowledge representations for events and entities discussed in the text. This approach can also characterize domain differences (e.g., the types of anaphoric reference described above), and is therefore useful in extensions to new domains. Our research is focusing on these areas.

Although a text grammar approach will not solve all of the discourse processing problems presented by the MUC-3 corpus, it does provide a framework for directing the text understanding process based on textual expectations at the phrase, sentence, and discourse levels.

NYU: INTERSENTENTIAL PROCESSING IN PROTEUS

The main task of intersentential processing in our system is to identify references to the same objects and events and to merge this information to form a single richer description of each object and event.

This merging is performed by two separate components: a reference resolution component (following semantic analysis) and a frame-merging component of the template generator. Reference resolution is intended primarily to handle references by definite noun phrases and pronouns to entities and events introduced previously in the discourse. This component is independent of the domain and discourse structure, although it does refer to the concept hierarchy for the domain. The frame-merging component, on the other hand, has been coded specifically for the current task and is intended to handle the particular discourse patterns of the newspaper articles (the brief introductory description of an event which may be followed by one or more elaborations).

Reference Resolution

As we describe in our system summary, reference resolution is applied to the output of semantic analysis. At this point, the logical form is a set of nested EVENT and ENTITY structures. Each noun phrase in the text will have been mapped by semantic analysis into an event (if it is a nominalization) or an entity (otherwise). Reference resolution examines each such event or entity to determine whether it can be an anaphoric reference to some prior event/entity. Each potential anaphor is compared to prior entities or events (starting with the most recent), looking for a suitable antecedent such that the class of the anaphor (in the concept hierarchy) is equal to or more general than that of the antecedent, the anaphor and antecedent match in number, the restrictive modifiers in the anaphor have corresponding arguments in the antecedent, and the non-restrictive modifiers (e.g., apposition) of the anaphor are not inconsistent with those of the antecedent.

If a suitable antecedent is found, references to the new entity/event (the anaphor) are replaced by references to the antecedent. In addition, for entities, if the anaphor contains non-restrictive modifiers for which there is no corresponding value in the antecedent, information from the anaphor is added to the antecedent. For example, message TST1-0065 mentions 'TWO ITALIAN ENGINEERS' in the opening sentence and later refers to 'THE TWO ITALIANS, ROBERTO ROASCIO OF MILAN AND MARIO ACCURSO OF ROME', so reference resolution needs to add the information about names to the prior entity, which gives information about occupations.

Special tests are provided for names (people may be referred to a subset of their names) and for referring to groups by typical members ('terrorist column' ... 'terrorists'). In the latter case reference resolution establishs a 'part-of' link between the group (the 'column') and its members ('terrorists'); this link is later used to permit frame merging.

Frame Merging

Reference resolution will handle explicit indications of prior reference, such as 'The building was bombed last week. The FMLN claimed responsibility for the attack.' (Here attack will be resolved to the bombing). On the other hand, it will not help in dealing with the repeated elaborations of events which are typical of newspaper style: 'The EXXON building was attacked last night. Five terrorists from the Shining Path fired two rockets at the EXXON building.' These cases are handled as part of template generation.

Template generation begins by converting EVENT structures for those events which are relevant to the final reporting task into FRAME structures. In this process, the event-type (ultimately, the incident type of the template), the date, and the location are reduced to normalized forms so that it will be easier to determine whether two FRAMEs might refer to the same event.

Once these FRAMEs are generated, the system attempts to merge them if one of three criteria are met:

- they involve a common target

- they come from the same sentence

- they consist of an attack followed by a possible effect of an attack (damage, death, etc.)

The merging will not take place, however, if the FRAMEs are incompatible with respect to type of attack, date, or location.

These heuristics are effective when targets are explicitly mentioned. For example, in DEV-0011 we are told in several ways about the death of Lopez Albujar:

AUTHORITIES HAVE REPORTED THAT FORMER PERUVIAN DEFENSE MINISTER GENERAL ENRIQUE LOPEZ ALBUJAR DIED TODAY ... AS A RESULT OF A TERRORIST ATTACK. LOPEZ ALBUJAR ... WAS RIDDLED WITH

BULLETS ... . HE WAS SHOT EIGHT TIMES IN THE CHEST. THE FORMER MINISTER WAS RUSHED TO THE AIR FORCE HOSPITAL WHERE HE DIED.

Reference resolution will first apply to resolve the pronouns. FRAME merging can then combine 'SHOT' and 'RIDDLED WITH BULLETS' (which are both mapped by semantics into SHOOT events). Richer heuristics will be needed for cases without explicit targets (e.g., TST1-0027: '...A BOMB ATTACK OCCURRED EARLY THIS MORNING SOMEWHERE NEAR THE PLACE WHERE BOLIVIAN PRESIDENT JAIME PAZ ZAMORA AND PERUVIAN PRESIDENT GARCIA ARE TO MEET TODAY. NAVY LIEUTENANT JUSTO MARTINEZ ... REPORTED THAT A BOMB EXPLODED IN THE PERUVIAN TOWN OF YUNGUYO ... .'.

Shortcomings

The most serious shortcoming of the system is its lack of domain and world knowledge. This is reflected in the serious gaps still existing in our set of lexico-semantic models. It is also reflected in the extremely limited rules we have at present for coalescing events. The only two semantically-based criteria we have, as noted just above, are that two events have the same target or that an attack is followed by a description of a possible effect of an attack. Simple examples of knowledge the system should include in order to merge events properly: if you are riding in a vehicle and the vehicle is blown up, you might be injured or die; if you are carrying explosives, you may be trying to bomb something. (Of course, the knowledge the system might potentially need is open ended, but we may hope that a core of knowledge associated with attacking things will be of substantial benefit).

Given the paucity of knowledge that we were able to enter into our system in the time alloted for MUC-3, we decided not to attempt to build explicit discourse structures (which would in many cases be dependent on this knowledge) at this time. As a result, we often failed to combine events into a single template when appropriate. If we are able to enrich the knowledge structures adequately for MUC-4, it may then make sense to build discourse links as a separate step prior to template generation.

Our system is also somewhat weak in handling set/element relationships. Specifically, it will not relate a plural noun phrase to the individuals with which it coreferential unless these individuals appear syntactically conjoined in the text. In other words, it will not form new groupings of individuals. Thus, it would not handle 'X was attacked. Y was attacked. The attacks were conducted by Z.' Nor is the system able to figure out that the 'TWO VEHICLES' mentioned in TST1-MUC3-0099 are the CAR-BOMB and USSR EMBASSY VEHICLE, since these antecedents do not appear conjoined.

SRI'S TACITUS: DISCOURSE PROCESSING

The TACITUS system employs the general method of abductive explanation to understand texts [16]. This method of explanation is quite well suited to the narrative texts of the MUC-3 domain, because the texts consist almost entirely of declarative sentences that are intended to convey information to the reader. TACITUS does not have an explicit discourse processing module, and does not currently employ any theory of discourse structure, but rather relies on the assumption that the correct resolution of anaphora and individuation of events will be a consequence of generating the best explanation for the truth of its constituent sentences, subject to minimizing the extension of certain predicates. The justification for this assumption is described below.

Abductive Interpretation

TACITUS processes each sentence S incrementally, seeking the best explanation for why that sentence would be true, given its domain knowledge, $\mathcal{D}$, and all of the text that it has processed up to the given point in the message, $\mathcal{T}$. An explanation consists of finding some minimal set of premises, $\mathcal{A}$, such that $\mathcal{D} \cup \mathcal{T} \cup \mathcal{A} \vdash L_S$, where L_S is the logical form of sentence S. The minimality of alternative sets of premises is evaluated by adding the assumption cost of each literal in $\mathcal{A}$. The assumption costs of each literal comprising the logical form is assigned initially in accordance with heuristics reflecting the relative importance of that literal's contribution to the interpretation. Assumption costs can be passed from consequent literals to antecedent literals in Horn clause axioms by means of weighting factors associated with each literal. When the best interpretation of a sentence is found, the set of premises is added to the text theory $\mathcal{T}$, which then forms part of the base theory for the interpretation of the next sentence in the text. At any time, the contents of the set $\mathcal{T}$ can be examined by the analysis component of the system to generate a set of templates reflecting the text as it has been analyzed up to that point.

In addition to proving and assuming literals, an important part of the abduction proof involves minimizing the extension of certain predicates through factoring. If $\exists x P(x)$ is an assumption, and $\exists y P(y)$ is a goal, then it is possible to *factor* the literals through unification, setting the resulting assumption cost to the minimum assumption cost of the two literals. This factoring operation entails the assumption that $x = y$, which amounts to assuming that individuals that share property P are identical.

Only a subset of the predicates in a domain should be minimized. For example, causation is a relation that holds among many events. Simply because we know that event e_1 causes e_2 and e_3 causes e_4 is rather poor grounds for assuming that e_1 and e_2 are the same. Apparently, causation is not one of the predicates that should be factored. However, predicates corresponding to natural kinds probably refer to specific entities, and are good candidates for minimization. Similarly, predicates relating event types to tokens should be minimized. If an article mentions a kidnapping twice, then it is often reasonable to assume that the same event is being described.

Clearly, assumption of a goal literal is not a sound operation, because the assumption might be inconsistent with the base theory. This means that every set of assumptions must be checked for internal consistency and consistency with the base theory. In general this is a computationally expensive process (in the cases that it is even decidable. TACITUS therefore uses a restricted theory to check consistency of a set of assumptions so that the consistency check can be computed with very little effort. Any assumption set is rejected as inconsistent that meets any of the following criteria:

- $P(a)$ and $Q(a)$ are both assumptions, and a class hierarchy indicates that P and Q have disjoint extensions.
- $P(a)$ and $Q(a)$ are both assumptions, and P and Q are distinct predicates corresponding to proper names.
- If s_1 and s_2 are both sets, and s_1 and s_2 are identified through factoring, and s_1 and s_2 have different cardinality.

This basic assumption and consistency checking mechanism drives all of the discourse processing in the current TACITUS system.

Why does Minimization Work?

It may not be obvious that minimization of events and individuals of a given natural kind should lead to a correct interpretation of the text. After all, there is no a priori justification in the world for assuming

that two individuals of a given type are the same. Strictly on the basis of probability, it is in fact highly unlikely. The minimization heuristic relies on the fact that one is interpreting a coherent text that conforms to the Gricean maxim of relevance. By assuming that a text is coherent, one can assume that the events an individuals mentioned are related in some systematic way. The abductive interpretation of the text makes these relations explicit as part of the process of explaining the truth of the sentences. Minimization of selected relations is one way of maximizing the connections of each sentence with the text that preceeds it.

Discourse Processing in TACITUS

To illustrate the basic principles of discourse interpretation in TACITUS we refer to the following message from the MUC-3 training corpus:

LIMA, 30 MAR 89 – [TEXT] A CARGO TRAIN RUNNING FROM LIMA TO LOROHIA WAS DERAILED BEFORE DAWN TODAY AFTER HITTING A DYNAMITE CHARGE. INSPECTOR EULOGIO FLORES DIED IN THE EXPLOSION.

THE POLICE REPORTED THAT THE INCIDENT TOOK PLACE PAST MIDNIGHT IN THE CARAHUAICHI-JAURIN AREA.

Resolving NP Anaphora

Interpreting the first sentence of this text requires making certain assumptions based on general world knowledge. For example, the knowledge base expresses the fact that dynamite is an explosive substance, a 'substance object' compound nominal referers to an object that is composed of the substance, an object that is composed of an explosive substance is a bomb, and hitting a bomb results in an explosion, and explosions cause damage, and derailing is damage. By minimizing the extension of damage, we conclude that the hitting of the dynamite charge caused an explosion (which is a terrorist bombing incident), and the bombing caused the train to derail.

The next sentence mentions a death in an explosion. Correct interpretation of this sentence requires association between the explicitly mentioned explosion and the explosion resulting from hitting the dynamite charge. Otherwise the system may conclude that two bombing events were involved. Minimization of exploding events results in the correct resolution.

Resolving Pronominal Anaphora

The descriptive content of pronouns is very limited, since it includes only number, gender, and animateness information. In general this descriptive content is insufficient alone to facilitate identification of the antecedent. In addition, syntactic relationships can rule out certain coreferentiality possibilities that would be consistent with the descriptive content of the pronoun, and make others more likely. Therefore, TACITUS uses a differnt method for resolving pronominal references than for NPs with noun heads.

Hobbs [17] describes an algorithm for pronominal anaphora resolution based only on criteria of syntactic structure and matching of basic selectional constraints. A statistical study by Hobbs demonstrated that this algorithm correctly identifies pronominal antecedents 91.7% of the time in the texts he studied. TACITUS employs this algorithm to produce an ordered disjunction of coreference possiblities as part of the logical form. During abductive interpretation, the variable representing the referent of the prounoun is bound to the the first alternative on this list, and if this binding passes the consistency check, it is assumed to be the correct resolution. If not, successive bindings are chosen from progressively less likely alternatives as determined by the resolution algorithm, until a consistent interpretation is found. This resolution method

thus allows the syntactic algorithm to be improved by the incorporation of pragmatic information, although no evaluation has yet been undertaken to quantify the success of this approach.

Individuating Events

TACITUS does not employ any means of individuating events other than the general heuristic of finding a consistent interpretation with the minimal number of events of each type. There are a number of problems posed by the MUC-3 domain that require some extensions to this basic minimization strategy. In interpreting the final sentence of the above example, TACITUS relies on its knowledge that any type of event can be described as an incident. Minimization of events can be done by resolving the 'incident' to either the implicit explosion, the death of Flores, or the derailing, and thus associating the locative and temporal information contained in this sentence with one of the events we already know about. Since all of these events are essentially concurrent, the template generation process can correctly fill the template, no matter which event is chosen as the resoution of 'incident.'

Events in the MUC-3 domain have multiple agents, objects, and instruments. Therefore, two events of the same type with different agents and objects can consistently be collapsed into a single event with multiple agents and objects. The reliable individuating criteria are time and location. However, the temporal reasoning necessary to accurately determine whether two intervals or locations are different can be quite complex, and do not fit within the class hierarchy or simple consistency checking mechanisms that are employed by the abductive theorem prover. Therefore, these sorts of inconsistencies are not detected during pragmatics processing and must be left for the final analysis-template filling phase. At this stage of development, there is no opportunity to backtrack to earlier stages of processing if the consistency checking mechanism is not powerful enough to detect the relevant inconsistencies. The incorporation of locative and temporal consistency checking into the abductive proof process is a current topic of investigation.

Problems for Future Research

A serious problem with this general approach to discourse processing is the combinatorics of the problem of minimizing the predicates while at the same time searching for the cheapest abductive proof. Each time the theorem prover attempts to prove a goal, it could have three choices: it can prove it, assume it, or factor it with something it already assumed. It is easy to see that each choice point increases the size of the search space exponentially.

One approach to dealing with this combinatorial problem is to limit the choices by processing only sentences that pass a statistical relevance filter. Another strategy along these lines is careful selection of the predicates to be minimized. If predicates are related by a chain of entailments, only the most general predicates in the chain should be considered for minimization.

Another problem is that the approach requires a relatively rich knowledge base for consistency checking. When information outside the scope of the knowledge base is encountered, the minimization strategy is generally too agressive. The absence of information allows it to assume that almost anything is the same as anything else without contradiction. Knowledge base construction is, of course, part of the long-term effort we are addressing.

Finally, methods must be found for expanding the consistency checking to include various temporal and locative inconsistencies, as well as some other problems. For example, current consistency checking methods have trouble dealing with singular and plural entities properly, as well as collective anaphora.

One type of discourse related resolution problem that the approach outlined here cannot solve is anaphora that depends on syntactic parallelism to resolve, because all information about parallel structure of phrases is lost by the time the abductive reasoning process operates. However, the actual number of texts in which

this consideration is crucial for performing the MUC-3 task seems to be quite small, and therefore the shortcoming is not a severe handicap for this task.

Our current experience with the TACITUS system suggests that this simple but powerful method of abductive interpretation can be quite successful at handling many of the discourse problems that arise in this task. In many cases, anaphora is correctly resolved, and correct causal relationships between actions are postulated. Although the system in its current state of implementation still makes mistakes, these mistakes can often be traced to inadequacies in the knowledge base. While the ultimate success of the general approach is still an open question, current experience suggests that there is still much room for improvement before inherent limitations are reached.

UMASS' CIRCUS: CONSOLIDATION

The UMass system (see [24] and our system summary in these proceedings) as used for MUC-3 is composed of a conceptual sentence analyzer, CIRCUS [23], and a discourse analysis component called 'consolidation'. These two modules work together in a pipelined fashion: CIRCUS generates conceptual meaning representations for individual sentences and consolidation maps these representations onto a set of response templates. We use two distinct methods for generating templates from parser output: rule-based consolidation and case-based consolidation.

Our case-based reasoning (CBR) module is an optional component that may be used to augment the output of rule-based consolidation. We used the CBR component in our official MUC-3 test run because it increases recall by generating templates that rule-based consolidation may have missed. However, the increased recall comes at the expense of precision because many of the additional templates turn out to be spurious. We demonstrated this recall/precision tradeoff by doing an optional MUC-3 test run without the CBR component; as expected, our system had lower recall but better precision (see our site report in these proceedings).

Rule-Based Consolidation

Rule-based consolidation merges the meaning representations produced by CIRCUS into a set of response templates. This process consists of 4 phases: constructing task-specific representations, partitioning, rule-based merging, and normalization.

The CIRCUS parser produces task-independent meaning representations (*concept nodes*) for individual sentences. Concept nodes are frame-like structures that are triggered by relevant words or phrases and filled by local syntactic constituents using semantic constraints and preferences. For example, the following concept node is generated from the sentence below it:

```
TYPE = MURDER
ACTOR = (FMLN commandos)
VICTIM = (Salvadoran leftist leader Hector Oqueli Colindres)
```

'Salvadoran leftist leader Hector Oqueli Colindres was killed by FMLN commandos.'

Consolidation, however, must be able to reason with task-specific knowledge so it immediately converts each concept node into a task-specific knowledge structure called a *c-structure*.[8] During this process, explicit memory objects are created for victims, physical targets, perpetrators, dates, and locations. A simple

[8] short for 'consolidation structure'

pronoun resolution algorithm also tries to locate pronominal referents in preceding sentences; if it succeeds then the referent is substituted for the pronoun in the c-structure. The following c-structure is generated from the concept node above:

```
$MURDER
   ACTOR = $Perp-1
           ID = (FMLN commandos), ORG = (FMLN),
           WORD-SENSES = (ws-terrorist ws-organization),
           CONFIDENCE = nil, NEW-INFO = nil
   VICTIM = $Victim-1
           ID = (Hector Oqueli Colindres), TITLE = (Salvadoran leftist leader),
           NATIONALITY = El Salvador, NUM = 1, TYPE = (ws-proper-name ws-politician)
           EFFECTS = (death)
```

The second phase of consolidation, *partitioning*, identifies groups of c-structures that belong to the same incident. The c-structures are divided into partitions that reflect a weak organization of the text. All c-structures within a single partition are assumed to refer to the same incident, but different partitions may or may not correspond to the same incident. Partitions are created by exploiting textual cues, including specific phrases and patterns, as well as domain-dependent heuristics. There are four classes of textual cues: **new-event-markers**[9] introduce a new incident (e.g. 'meanwhile'), **generic-event-markers** suggest a generic or irrelevant event (e.g. 'wave of'), and **separate-event-markers** identify references to multiple events within a single sentence (e.g. 'the day before'). Domain-dependent heuristics are also applied to infer boundaries between multiple events. For example, if a partition contains two event types that were not in the preceding partition then we infer that this partition refers to a new incident. During the partitioning phase, c-structures that represent irrelevant events are discarded and c-structures that contain certain types of summary information are removed from the merging process and put aside to be used elsewhere.

Once the c-structures have been partitioned, they are sequentially merged into a set of response templates. This merging process is guided by a rule base of 139 rules.[10] Most rules are condition-action pairs where the condition specifies whether a particular c-structure and template are compatible and the action dictates how to merge the c-structure into the template. There are also default rules and special rules to generate a new template instead of merging the c-structure with an existing template. As templates are created, they are pushed onto a context stack. Given a c-structure to be merged, the rules are applied to each template on the stack, in turn, until one template is found to be compatible with the c-structure or until the stack is exhausted. If a compatible template is found then the rule fires, merges the c-structure into that template, and moves the template to the top of the stack.[11] If no compatible template is found then default rules decide whether a new template should be created from the c-structure.

Each rule has its own criteria for judging whether a c-structure and template are compatibile. Some rules require only that the respective dates and locations are consistent whereas other rules may also require compatible targets, victims, instruments, etc. When a rule fires, memory objects that refer to the same entity are unified as a side effect of the merging process; this involves proper name resolution, updating type and nationality information, etc.

Rule-based merging also involves maintaining families of events. Texts often contain information about multiple events that were perpetrated by the same people on the same day and in the same location. We call this a *family* of events. To keep these events together, each template is tagged with a family id number. This is where the partitions come into play. All c-structures within a partition are forced to merge with templates in the same family. For example, if the first c-structure in a partition is merged into a template in family #2, then the remaining c-structures in that partition must also be merged into templates in family #2. Therefore the first c-structure in a partition effectively commits the entire group to a particular family.

[9]These include possible-new-event-markers that signal a context switch only under specific conditions.

[10]There is a separate subset of rules for each type of c-structure.

[11]Hence, this is not strictly a stack since templates are not always removed from top. However the templates are checked for compatibility from top to bottom.

The last stage of consolidation is *normalization*. This phase integrates summary information, normalizes families to ensure that all incidents in the same family share perpetrators, dates, and locations, adds default slot fillers, and discards templates that are deemed to be irrelevant. Currently, we only deal with summary information about similar incidents that took place in multiple locations. If the rule-based merging process has not already created templates for each of these incidents, then the missing templates are generated.

Case-Based Consolidation

Our system also has an optional case-based reasoning (CBR) component that can be used to augment the set of response templates generated by rule-based consolidation. CBR allows us to benefit from the development corpus by examining how parser output correlated with key templates in previous texts. The CBR module currently contains 254 cases drawn from 383 texts. A case is constructed from each key template by determining which concept nodes contain slot fillers that belong in the key.[12] For example, suppose a MURDER key template contains a perpetrator and human target that correspond to the perpetrator slot filler in a $murder concept node from sentence 1 and a victim slot filler in a $murder concept node from sentence 2, respectively. The following case would be constructed:

(**MURDER** (0 (perp)) (1 (victim)))

This case represents concept nodes in adjacent sentences that contain a perpetrator and victim, respectively.

The concept nodes generated by the parser are compared with the cases in the case base. Some subset[13] of cases will be retrieved. Each retrieved case recognizes a pattern of concept nodes that resulted in a key template in a previous message; therefore, it recommends that this type of template should be generated for this text. If rule-based consolidation has not already generated such a template, then the CBR module will create one from the concept nodes that retrieved the case. In this manner, CBR can suggest additional templates that rule-based consolidation may have missed or incorrectly discarded.

Future Work

There are several problems with our system that suggest directions for future research. In general, consolidation does not have access to information that was not picked up by a concept node. For instance, secondary information about victims (titles, types, full names, etc.) is often provided by text that surrounds the sentence(s) that actually describe the event. Our system currently has no facility for getting this information. Along different lines, although our rule base served us well in MUC-3, we suspect that we could do better by organizing events in a more intelligent manner. For example, our system might merge a bombing of target X with a different but compatible bombing near the top of the stack despite the fact that a bombing template for target X is already on the stack but has gotten buried underneath a different bombing. An intelligent memory organization would allow us to find the best match more naturally. In addition, it should allow us to handle forms of summary information that could not be handled easily by our current architecture.

We are also excited about the prospects for case-based consolidation. Our CBR module was developed very late in the project so there are still many avenues to explore. One possible approach is to integrate rule-based and case-based consolidation. By starting with only a small set of rules, the cases could be used to augment the rule base and expand its coverage. We would also like to experiment with different retrieval algorithms, case representations, and case bases. We find CBR to be particularly appealing because it holds great promise for improving portability and scale-up problems; since cases can be acquired automatically, a successful case-based solution could be transferred quickly and easily to new domains.

[12] The current implementation only looks at the perpetrator, human target, and physical target slots in the key.

[13] Possibly empty if no similar cases are retrieved

OVERVIEW OF SELECTED ASPECTS OF DISCOURSE

I: Recognizing a single event and distinguishing multiple events

The tasks of **recognizing a single event** and **distinguishing multiple events** were considered to be the most important aspect of discourse-related processing. For the messages in the MUC-3 corpus, these tasks are particularly difficult because the description of a single event may be as short as one sentence or as long as an entire message (several paragraphs), and because many messages report on multiple events.

The capability of **recognizing a single event** is important both for overgeneration and recall. If the system fails to recognize that the text describes the same event, it will produce one or more spurious templates. At the same time, it will also distribute the information about a single event across multiple templates. Since only one template will be scored against the key template, the system effectively gets no credit for information that it correctly extracted but attributed to a different event.

All systems except ITP and SRI recognize a single event primarly by checking the compatibility of frame-like data structures. These structures contain all the information, understood or inferred from the text, about the events encountered so far. Events are assumed to be compatible if they are of compatible types, have the same targets, and if their locations and times are not incompatible. The event compatibility check usually involves heuristics. For this task, UMass uses an elaborate set of rules which reflect relevant properties of the MUC-3 corpus and in certain cases embody knowledge about text structure. If two events are compatible, then the structures that represent them are merged. This merging is usually done on the fly, with possible further merging at a final post-processing stage.

GE, NYU and UMass depart somewhat from the above scheme. GE enhances merging with an initial pre-parsing segmentation of a story into portions of text that are relevant for different events. This segmentation is based on heuristics that combine weak anaphora resolution and a weak notion of topic shift with some clues about the discourse structure.

After semantic analysis, NYU performs pronoun and definite anaphora resolution. If a definite reference is encountered and its resolvent is found, then the event continues. Merging of compatible data structures is attempted only after the definite anaphora component.

After parsing but before merging, UMass segments texts using textual cues and heuristics that identify boundaries between multiple events.

For SRI, a sentence continues the description of an event if an abductively derived explanation can be constructed for it. Merging of events (but not frame-like data structures) takes place down inside the abductive reasoning process. The system hypothesizes that an event may be identical to one it already knows about, which if true would produce a better explanation than hypothesizing multiple events, provided that the identification is consistent. Checking consistency of the facts that it knows about those events has similar effect as checking the compatibility of frame-like data structures.

The ITP system skips merging. It performs a post-parsing segmentation of a story based on the compatibility of time and location of the events, as well as some clues about the discourse structure. An event continues (a new segment is not created) if there is no information about the incompatibility of time or location. In such cases, other systems will also recognize the continuation of an event and merge relevant data structures.

The capability of recognizing the continuation of the description of a single event is closely related to **distinguishing multiple events**. It is especially difficult for the systems to distinguish multiple events of the same type or to distinguish attacks from other events. As in the case of recognizing a single event, the most popular technique is to avoid merging incompatible events. Two events are assumed to be different if their types, locations, times or targets are not compatible.

As before, GE, SRI and UMass depart somewhat from this scheme. Before the merging stage of the processing, GE and UMass attempt to segment the text into pieces that correspond to different events. Both systems use cue phrases combined with heuristics about patterns of key phrases in order to identify event boundaries. There are three differences between GE and UMass text segmentation: (1) a GE text segment is intended to correspond to a single event and a UMass text segment to a group of events that share certain properties (e.g. perpetrator or location), (2) GE initial text segmentation never results in multiple segments for a single sentence (this is done later by the merging component) and UMass sometimes produces multiple segments for a single sentence, (3) GE segments texts before parsing, and UMass after parsing.

SRI does not employ a separate merging module. Instead, event merging takes place inside the abductive reasoner. Two events are considered to be different if they are of different types, or if it is inconsistent with what the system knows to assume that they are the same.

ITP does not use the merging technique and relies on its post-parsing text segmentation. The similarity in text segmentation is in the fact that GE, UMass and ITP use cue phrases for this task. The differences are in the fact that GE and UMass merge events, and ITP avoids the merging technique, and that GE and UMass text segmentation is based on patterns of key phrases [14] and ITP text segmentation is based on parsed sentences.

All three systems, GE, ITP and UMass, use temporal and spatial information for this task. ITP uses it while segmenting texts, and GE and UMass employ the same information in their merging modules.

II: Resolving references: definite references, pronouns and proper names

GE (to some extent), ITP and NYU do **definite reference resolution** as an independent activity. Only resolution of noun phrases that involve relevant words (key words) is attempted. GE indirectly resolves definite references at the pre-processing stage by using heuristics about key word patterns. ITP resolves references based on surface syntactic constraints, referent accessability and knowledge about the world and relations between entities in different situations. NYU uses interpreted sentences and performs a type subsumption check when looking for the resolvent.

Other systems do not perform definite reference resolution. Instead, references are oftentimes resolved as the side-effect of some other activity. For BBN, this activity is merging compatible data structures. For SRI, it is constructing an abductive explanation. If an entity is referred to by a unique set of properties, then resolving the referent to some known entity that shares all of the properties will correspond to the cheapest proof obtained with factoring, other things being equal.

No systems handles distributive references; for example, when the phrase *the attacks* refers to a subset of attacks described earlier in the text. Similarly no system even attempts to resolve references to events assumed to be familiar to the reader from other messages.

All systems perform **pronoun resolution** to various degrees. Typically checked properties when resolving pronouns are *number* and some semantic type constraints. For example, the referent for *she* must be *singular* and of type *person*. Some systems also employ *gender* check when searching for the antecedent. For example, the referent for *she* must have *female* property. ITP uses knowledge about the world and relations between entities in different situations to resolve pronouns. For SRI, a candidate for a referent must satisfy certain syntactic constraints and selectional restrictions.

Most systems maintain a global list of all **proper names** encountered in the text. Many can handle partial matches and recognize that *Luis Galan* may be referred to as *Senator Galan* or *Mr. Galan*.

[14] Patterns are expressions of a regular language whose grammar contains shallow rules. Using patterns does not mean avoiding parsing but rather doing efficient parsing of a very restricted regular language. When we say "parsing" in this paper we mean parsing for a context-free grammar with context-sensitive extensions that possibly allows for left- or right recursion.

Reference resolution is usually done on the fly. Some systems delay this decision for certain references until the end of the message. For pronominal anaphora, SRI maintains a preferentially ordered list of candidate referents. If the top element in the list does not pass consistency check, then the next one is considered.

III: Discourse representation

Most systems take a combination of **bottom-up and top-down approach to discourse**. The bottom-up aspect consists in interpreting individual sentences independently and key phrase-based activation of frame-like data structures. The top-down aspect consists in realizing expectations encoded as slots, usually with some restrictions on fillers in the activated data structures. Merging compatible data structures, text segmentation into portions describing different events and search for the best abductive explanations are the three exhibited methods to compute a coherence relation that ties pieces of text together.

Usually, individual sentences are processed independently. Satisfying expectations is one way to influence the uderstanding of a subsequent sentence by the previously understood text. Another way of doing such an incremental understanding is building an abductive proof that may depend heavily on the previous context.

Most systems do not produce an explicit **discourse representation**. Systems store the understood relevant aspects of the text in a variety of ways. The most common are frame-like data structures that store information about some objects or events. Some systems consider only events or objects relevant for the MUC-3 templates. Others, like BBN, create semantic representations of all the input that is processed. BBN, GE, LSI and UMass store understood text in such frame-like structures.

In case of LSI, these structures are instantiated by a text grammar. For some messages, they encode expectations about text structure and content. NYU stores a list of sentence interpretations (logical forms) with resolved references which are subsequently transformed into frame-like structures. ITP uses a formal discourse representation, Kamp's discourse representation structures. The main advantage of this representation is making explicit the availability of an entity for anaphora resolution. SRI stores a list of logical forms of the sentences along with a list of premises that, combined with domain knowledge, entail all the previously understood sentences.

Some general observations

A characteristic property of understanding MUC-3 messages is **locality**. Missing fillers are looked for in the immediate neighborhood of the key phrases that activated the data structure, typically no further than the same or adjacent paragraph. Antecedents of anaphora are looked for in the same segment. Search for a pronoun resolvent typically does not go beyond the current and previous sentence. Ambiguities, for example, in reference resolution, are not propagated, but instead are resolved immediately. It is difficult for some systems to use information contained in a sentence different from the one containing the key phrase that activated the structure. As the result, systems have troubles with relating information distributed throughout the text. Some of this information may be recovered through merging or reference resolution.

Few systems recognize and are able to take advantage of **discourse structure**. For example, the distribution of relevant information may depend on the style of a message. An editorial report may contain a one-paragraph summary of the events followed by a more detailed description, but in an interview, information about the same events may be distributed throughout the text. In certain cases, not recognizing discourse structure does not hurt the performance, because the same effect may be achieved through some other computation. For example, if the brief introductory description of an event is followed by one or more elaborations, then merging may be an indirect way of recognizing elaboration of the same event. UMass deals with summary information in the case of similar events that took place in multiple locations.

FUTURE RESEARCH

For most systems, the decision about the future research is based on the desire to perform well on MUC-4 and to qualitatively improve systems' capabilities of performing well on other tasks or in different domains. The two do not always coincide. For example, understanding temporal relations in a text is crucial not only for MUC, but in general. However, NYU verified that for the TST1 set of stories, a part of this task, undestanding semantics of temporal subordinate conjunctions, e.g., *while* or *since*, will not contribute to the increase in performance. More data characterizing the MUC-3 corpus, for example, frequency of occurence of different phenomena, is needed.

Interestingly enough, only LSI listed improvement of its parser as a focus for future research. All other systems perceive handling discourse-related phenomena as the most promising area, possibly because discourse capabilities allow one to balance the above mentioned trade-off.

Recognizing a single event and distinguishing multiple events have been recognized by most sites as a central problem of message understanding, both for MUC and in general. Many planned extensions can be viewed as means of solving thes problems. They include reference resolution (particularly, distributed definite anaphora and vague references), temporal and spatial reasoning, better strategies for resolving ambiguities, more semantic criteria for merging events, and expanding existing knowledge bases.

Some sites consider discourse structure to be a promising path to explore. One identified structure is relatively common for journal articles. It consists of an introductory paragraph that provides some general or salient information about a group of events, and several paragraphs that describe individual events and provide more specific and possibly new information. The capability of taking advantage of such structures involves, among others, the capability of computing more sophisticated temporal and spatial relations.

Some systems plan to develop a more adequate discourse representation that would subsume a MUC task. This involves search for computationally feasible coherence relations.

Text understanding is **computationally very expensive**. Most systems control this complexity by reading only relevant fragments of text. This usually means sacrificing completeness, because a portion of text judged irrelevant may contain the antecedent of an anaphor that would allow the connection of text fragments. Processing only relevant text fragments also means relying more heavily on **heuristics**. Some sites plan to investigate more reliable methods, for example, statistical methods, of identifying the relevance of a text fragment.

CONCLUSION

We discussed discourse-related processing in the context of MUC-3 conference. MUC-3 participants exhibited a striking commonality both in their choice of central problems and approaches to their solutions. Three discourse-related tasks, (1) identifying portions of text that describe different domain events, (2) resolving references, and (3) discourse representation, have been identified as crucial not only for MUC but for text understanding in general. We compared the approaches of different sites, stressing both similarities and differences [15].

MUC-3 participants represented in this paper plan to invest their effort in improving or extending their discourse-related capabilities. The identified tasks are both challenging and expensive, in terms of time and other resources. They involve not only implementation of known techniques, but also extensive original research.

[15] Space limitations do not allow us to perform a more detailed analyis, but we have included references to the relevant literature.

References

[1] N. Asher. A typology for attitude verbs and their anaphoric properties. *Linguistics and Philosophy*, 10:125–198, 1987.

[2] D. Ayuso. Discourse entities in Janus. In *Proceedings of the 27th Annual Meeting of the ACL*, pages 243–250, 1989.

[3] D. Ayuso, G. Donlon, R. Bobrow, D. MacLaughlin, L. Ramshaw, V. Shaked, and R. Weischedel. Research and development in natural language understanding as part of the Strategic Computing Program–Final Report. Vol. 3: A guide to IRUS-II application development. BBN Report 7191, BBN Systems and Technologies Corp., Cambridge, MA, 1990.

[4] R. Bobrow, R. Ingria, and D. Stallard. Syntactic and semantic knowledge in the Delphi unification grammar. In *Proceedings of the DARPA Speech and Natural Language Workshop*, pages 230–236, June 1990.

[5] R. Bobrow, R. Ingria, and D. Stallard. The mapping unit approach to subcategorization. In *Proceedings of the DARPA Speech and Natural Language Workshop*, February 1991.

[6] Michael Brady and Robert C. Berwick (editors). *Computational models of discourse.* MIT Press, Cambridge, 1986.

[7] S. Brennan, Friedman M., and C. Pollard. A centering approach to pronouns. In *Proceedings of the 25th Annual Meeting of the ACL*, pages 155–162. ACL, 1987.

[8] K. Dahlgren. Coherence relation assignment. In *Proceedings of Cognitive Science Society*, pages 588–596, 1989.

[9] T. A. Dijk. *Some aspects of text grammar.* Mouton, the Hague, 1972.

[10] T. A. Dijk. *Macrostructures.* Lawrence Erlbaum, Hillsdale, NJ, 1980.

[11] Ralph Grishman and Tomasz Ksiezyk. Causal and temporal text analysis: the role of the domain model. In *Proceedings of COLING-90*, pages 126–131, 1990.

[12] B. Grosz, A. Joshi, and S. Weinstein. Providing a unified account of definite noun phrases in discourse. In *Proceedings of the 21st Annual Meeting of the ACL*, pages 44–50. ACL, 1983.

[13] B. Grosz and C. Sidner. Attention, intentions and the structure of discourse. a review. *Computational Linguistics*, 7:85–98, 1986.

[14] B. Grosz and C. Sidner. Attention, intentions and the structure of discourse. a review. *Computational Linguistics*, 12:175–204, 1986.

[15] Erhard Hinrichs. Tense, quantifiers, and contexts. *Computational Linguistics*, 14:3–14, 1988.

[16] J. Hobbs, M. Stickel, P. Martin, and D. Edwards. Interpretation as abduction. In *Proceedings of the 26th Annual Meeting of the ACL*, pages 95–103, 1988.

[17] Jerry R. Hobbs. Resolving pronoun references. *Lingua*, 44:311–338, 1978.

[18] Jerry R. Hobbs. Coherence and coreference. *Cognitive Science*, 3:67–90, 1979.

[19] R. Ingria and D. Stallard. A computational mechanism for pronominal reference. In *Proceedings of the 27th Annual Meeting of the ACL*, pages 262–271. ACL, 1989.

[20] Lucja Iwańska. Discourse processing module. Technical Report forthcoming, GE Research and Development Center, Schenectady, NY, 1991.

[21] H. Kamp. A theory of truth and semantic representation. In T. Groenendijk, Janssen, and M. Stokhof, editors, *Formal Methods in the Study of Language*. Mathematisch Centrum, Amsterdam, 1981.

[22] Alex Lascarides and Nicholas Asher. Discourse relations and defeasible knowledge. In *Proceedings of the 29th Annual Meeting of the ACL*, pages 55–62, 1991.

[23] W. Lehnert. Symbolic/Subsymbolic Sentence Analysis: Exploiting the Best of Two Worlds. In J. Barnden and J. Pollack, editors, *Advances in Connectionist and Neural Computation Theory, Vol. 1*. Ablex Publishers, Norwood, NJ, 1990.

[24] W. Lehnert, C. Cardie, D. Fisher, E. Riloff, and R. Williams. The CIRCUS System as Used in MUC-3. Technical Report forthcoming, University of Massachusetts, Amherst, MA, 1991.

[25] C. A. Montgomery and B.C. Glover. A sublanguage for reporting and analysis of space events. In *Analyzing language in restricted domains*. Lawrence Erlbaum, Hillsdale, NJ, 1986.

[26] Rebecca J. Passonneau. A computational model of the semantics of tense and aspect. *Computational Linguistics*, 14:44–60, 1988.

[27] V. Propp. *Morphology of the folktale*. Indiana University Press, Bloomington, 1958.

[28] Lisa Rau and Paul Jacobs. Integrating top-down and bottom-up strategies in a text processing system. In *Proceedings of Second Conference on Applied Natural Language Processing*, pages 129–135, Morristown, NJ, Feb 1988. ACL.

[29] B. G. Stalls, R. E. Stumberger, and C.A. Montgomery. Long range air (lra) data base generator (dbg). Technical Report RADC-TR-89-366, Rome Air Development Center, Griffiss AFB, NY, 1990.