FASTUS: A Finite-state Processor for Information Extraction from Real-world Text*

Douglas E. Appelt, Jerry R. Hobbs, John Bear, David Israel and **Mabry Tyson**
SRI International
333 Ravenswood Ave.
Menlo Park, CA 94025
USA

Abstract

Approaches to text processing that rely on parsing the text with a context-free grammar tend to be slow and error-prone because of the massive ambiguity of long sentences. In contrast, FASTUS employs a nondeterministic finite-state language model that produces a phrasal decomposition of a sentence into noun groups, verb groups and particles. Another finite-state machine recognizes domain-specific phrases based on combinations of the heads of the constituents found in the first pass. FASTUS has been evaluated on several blind tests that demonstrate that state-of-the-art performance on information-extraction tasks is obtainable with surprisingly little computational effort.

1 Introduction

There are many reasons to want to process large numbers of natural-language texts automatically with high speed and accuracy. The best information retrieval systems can retreive texts that have a reasonable likelihood of being relevant to one's general concerns. However, getting the information out of the text that is required for solving someone's problem still requires a human to actually read the texts — a time consuming process that can easily swamp the available resources. We have developed the FASTUS system (the acronym is a slight permutation of Finite State Automata-based Text Understanding System) to address the need for a system that extracts prespecified information from a text with high speed and accuracy. This system has been tested in the MUC-4 evaluation of text processing systems [Sundheim, 1992] and has demonstrated

- High Performance (44% Recall and 55% Precision on a blind test of 100 Texts, which was among the best scores in the evaluation)
- Short domain-specific development time (three and a half weeks for the domain of terrorist incidents)
- Fast processing time (texts were processed at the rate of more than 2,000 words per minute)

1.1 Two Types of System

One can distinguish between two types of natural language systems: *information extraction* systems and *text understanding* systems. In information extraction,

- Only a fraction of the text is relevant; in the case of the MUC-4 terrorist reports, probably only about 10% of the text is relevant.
- Information is mapped into a predefined, relatively simple, rigid target representation; this condition holds whenever entry of information into a database is the task.
- The subtle nuances of meaning and the writer's goals in writing the text are of no interest.

This contrasts with text understanding, where

- The aim is to make sense of the entire text.
- The target representation must accommodate the full complexities of language.
- One wants to recognize the nuances of meaning and the writer's goals.

FASTUS is designed for the former case. Although the task is limited, it is nevertheless an important task that has many real-world applications.

It is tempting to view the information extraction task as a simple special case of the text understanding task, and approach the problem by applying a system designed for text understanding to the simpler task. We originally attempted to apply the TACITUS system [Hobbs et al., 1993] to this task. TACITUS is a text-understanding

*This research was funded by the Defense Advanced Research Projects Agency under Office of Naval Research contracts N00014-90-C-0220, and by an internal research and development grant from SRI International.

system that attempts to recover all implicit information from the text by abductive inference, using a semantic analysis of each sentence as the statement to be abductively proved, and the assumptions necessary to complete the proof comprising the additions to the system's knowledge base from understanding the sentence. This attempt was mildly successful (we managed to achieve 24% Recall and 40% Precision in an objective evaluation on the same task), however, the system was slow, since it spent much time on irrelevant text.

The TACITUS experience demonstrates why it was wrong to approach the information extraction task as a "traditional" computational linguistics problem. Even though many techniques to increase the robustness of the system were employed [Hobbs et al., 1993], the fact remains that parsing of context-free grammars is relatively slow, and this slowness is evident when encountering sentences that are 60 to 100 words in length, as happens not infrequently in real-world texts. Although TACITUS employed heuristics to guide the parsing of such long sentences, and statistics could be employed by a parser to find the most likely analysis first [Magerman and Weir, 1992], the fact remains that mistakes will be made, and they will have an impact on performance. A further problem was that the system had to apply computationally intensive reasoning techniques to every sentence. Since much of the text was irrelevant this meant that much of this effort was wasted. A statistical relevance filter was employed, but this filter had to have high recall and hence low precision to avoid excluding truly relevant text from consideration, and therefore admitted much irrelevant text to further processing.

1.2 Finite-State Models of English

If there are problems in applying context-free parsing to real-world text, then an efficient text processor might make use of weaker language models, i.e., regular or finite-state grammars. Every computational linguistics graduate student knows, from the first textbook that introduces the Chomsky hierarchy, that English has constructs, e.g., center embedding, that cannot be described by any finite-state grammar. This fact has no doubt biased researchers away from serious consideration of possible applications of finite-state grammars to difficult problems.

Church [1980] was the first to advocate finite-state grammars as a processing model for language understanding. He contended that, although English is clearly not a regular language, memory limitations make it impossible for people to exploit that context-freeness in its full generality, and therefore a finite-state mechanism might be adequate in practice as a model of human linguistic performance. A computational realization of memory limitation as a depth cutoff was implemented by Black [1989].

More recently, Pereira and Wright [1991] have developed methods for constructing finite-state grammars from context free grammars that overgenerate in certain systematic ways. The finite-state grammar could be applied in situations, e.g. as the language model in a speech understanding system, where computational considerations are paramount.

At this point, the limitations of the application of finite-state grammars to natural-language processing have not yet been determined. We believe this research establishes that these simple mechanisms can achieve more than has previously been thought possible.

2 A Description of the Task

SRI International participated in the recent MUC-4 evaluation of text-understanding systems [Sundheim, 1992], the fourth in a series of evaluations. The methodology chosen for this evaluation was to score a system's ability to fill in slots in templates summarizing the content of newspaper articles on Latin American terrorism. The articles ranged from one third of a page to two pages in length. Many articles described multiple incidents, while other texts were completely irrelevant.

The following are some relevant excerpts from a sample terrorist report, which is the source of most of the examples in this paper.

> San Salvador, 19 Apr 89 (ACAN-EFE) – [TEXT] Salvadoran President-elect Alfredo Cristiani condemned the terrorist killing of Attorney General Roberto Garcia Alvarado and accused the Farabundo Marti National Liberation Front (FMLN) of the crime.
>
> . . .
>
> Garcia Alvarado, 56, was killed when a bomb placed by urban guerrillas on his vehicle exploded as it came to a halt at an intersection in downtown San Salvador.
>
> . . .
>
> Vice President-elect Francisco Merino said that when the attorney general's car stopped at a light on a street in downtown San Salvador, an individual placed a bomb on the roof of the armored vehicle.
>
> . . .
>
> Guerrillas attacked Merino's home in San Salvador 5 days ago with explosives. There were seven children, including four of the vice president's children, in the home at the time. A 15-year-old niece of Merino's was injured.
>
> . . .
>
> According to the police and Garcia Alvarado's driver, who escaped unscathed, the attorney general was traveling with two bodyguards. One of them was injured.

This text is taken from a set of one hundred messages used in the final evaluation of MUC-3 in May 1991.

Some of the corresponding database entries for the killing of Garcia Alvarado are as follows:

Date:	- 19 Apr 89
Location:	El Salvador: San Salvador
Incident Type:	Bombing
Perpetrator ID:	"urban guerrillas"
Perp. Org.:	"FMLN"
Confidence:	Suspected or Accused by Authorities: "FMLN"
Physical Target:	"vehicle"
Effect:	Some Damage: "vehicle"
Human Target:	"Roberto Garcia Alvarado"
Description:	"attorney general": "Roberto Garcia Alvarado" "driver" "bodyguards"
Effect:	Death: "Roberto Garcia Alvarado" No Injury: "driver" Injury: "bodyguards"

The principal measures in the MUC-4 evaluation were recall and precision. *Recall* is the number of answers the system got right divided by the number of possible right answers. It measures how comprehensive the system is in its extraction of relevant information. *Precision* is the number of answers the system got right divided by the number of answers the system gave. It measures the system's accuracy. For example, if there are 100 possible answers and the system gives 80 answers and gets 60 of them right, its recall is 60% and its precision is 75%.

3 Overview of the FASTUS Architecture

The input text is first preprocessed to ensure that the text is in a standardized format for the remainder of the processing. Spelling correction is done at this point as well. The preprocessed text is then given to the FASTUS system proper.

The operation of FASTUS is composed of four steps:

1. Triggering
2. Recognizing Phrases
3. Recognizing Patterns
4. Merging Incidents

These steps are described in the next four sections. A postprocessing phase then converts the incident structures generated by FASTUS into the format required for the MUC-4 templates.

The system is implemented in CommonLisp and runs on both Sun and Symbolics machines.

3.1 Triggering

In the first pass over a sentence, trigger words are searched for. There is at least one trigger word for each pattern of interest that has been defined. Generally, these are the least frequent words required by the pattern. For example, in the pattern

> take <HumanTarget> hostage

"hostage" rather than "take" is the trigger word. There are at present 253 trigger words.

In addition, the names of people identified in previous sentences as victims are also treated, for the remainder of the text, as trigger words. This allows us, for example, to pick up occupations of victims when they occur in sentences with no other triggers, as in

> Hector Oqueli and Gilda Flores were assassinated yesterday.
>
> Gilda Flores was a member of the Democratic Socialist Party (PSD) of Guatemala.

Finally, on this pass, full names are searched for, so that subsequent references to surnames can be linked to the corresponding full names. Thus, if one sentence refers to "Ricardo Alfonso Castellar" but does not mention his kidnapping, while the next sentence mentions the kidnapping but only uses his surname, we can enter Castellar's full name into the template.

The performance of FASTUS on the example message is illustrative of its performance in general. In that message, 21 of 30 sentences were triggered. Thirteen of the 21 triggered sentences were relevant. Two of the 9 sentences not triggered were actually relevant.

3.2 Recognizing Phrases

The problem of syntactic ambiguity is AI-complete. That is, we will not have systems that reliably parse English sentences correctly until we have encoded much of the real-world knowledge that people bring to bear in their language comprehension. For example, noun phrases cannot be reliably identified because of the prepositional phrase attachment problem. However, certain syntactic constructs can be reliably identified. One of these is the noun group, that is, the head noun of a noun phrase together with its determiners and other left modifiers. Another is what we are calling the "verb group", that is, the verb together with its auxiliaries and any intervening adverbs. Moreover, an analysis that identifies these elements gives us exactly the units we most need for recognizing patterns of interest.

Pass Two in FASTUS identifies noun groups, verb groups, and several critical word classes, including prepositions, conjunctions, relative pronouns, and the words "ago" and "that". Phrases that are subsumed by larger phrases are discarded. Overlapping phrases are rare, but where they occur they are kept.

Choosing the longest subsuming phrase could lead to incorrect analyses in some cases involving lexical ambiguity between nouns and verbs. The present tense forms of such verbs are the same as the corresponding nouns; therefore a noun phrase could be constructed taking the misidentified verb as its head. This ambiguity problem was solved by the simple expedient of assigning a lower preference to any constituent with a present-tense verb form. Because the source of texts for this task was newspaper articles about past events, the use of present tense verb phrases in relevant sentences was quite rare.

In domains in which this expiedient solution to lexical ambiguity is inapplicable, a part of speech tagger could be employed to help resolve the ambiguity. We implemented and considered using a part-of-speech tagger, but we found that there was no clear improvement in performance, and it would have doubled the time the system took to process a message.

Noun groups are recognized by a 37-state nondeterministic finite state automaton. This encompasses most of the complexity that can occur in English noun groups, including numbers, numerical modifiers like "approximately", other quantifiers and determiners, participles in adjectival position, comparative and superlative adjectives, conjoined adjectives, and arbitrary orderings and conjunctions of prenominal nouns and noun-like adjectives. Thus, among the noun groups recognized are

> approximately 5 kg
> more than 30 peasants
> the newly elected president
> the largest leftist political force
> a government and military reaction

Verb groups are recognized by an 18-state nondeterministic finite state machine. They are tagged as Active, Passive, Gerund, and Infinitive. Verbs are sometimes locally ambiguous between active and passive senses, as the verb "kidnapped" in the two sentences,

> Several men kidnapped the mayor today.
> Several men kidnapped yesterday were released today.

These are tagged as Active/Passive, and Pass Three resolves the ambiguity if necessary.

Certain relevant predicate adjectives, such as "dead" and "responsible", are recognized, as are certain adverbs, such as "apparently" in "apparently by". However, most adverbs and predicate adjectives and many other classes of words are ignored altogether. Unknown words are ignored unless they occur in a context that could indicate they are surnames. The complete grammars of noun groups and verb groups are given by Hobbs et al., [1992b].

Lexical information is read at compile time, and a hash table associating words with their transitions in the finite-state machines is constructed. There is a hash table entry for every morphological variant of a word. The TACITUS lexicon of 20,000 words is used for lexical information. Morphological expansion of these words results in 43,000 morphological variants in the hash table. During the actual running of the system on the texts, only the state transitions accessed through the hash table are seen.

In the example message, 243 of 252 phrases, or 96.4%, were correctly recognized. Of the 9 mistakes, 5 were due to nouns being misidentified as verbs or verbs as nouns. The other 4 mistakes were due to simple bugs of the type that frequently creep into code during development.

3.3 Recognizing Patterns

The input to Pass Three of FASTUS is a list of phrases in the order in which they occur. Anything that is not included in a phrase in the second pass is ignored in the third pass. Patterns of interest are encoded as finite state machines, where state transitions are effected by phrases. The state transitions are driven off the head words in the phrases. That is, a set of state transitions is associated with each relevant head word-phrase type pair, such as "mayor-NounGroup", "kidnapped-PassiveVerbGroup", "killing-NounGroup", and "killing-GerundVerbGroup". In addition, some nonhead words can trigger state transitions. For example, "bomb blast" is recognized as a bombing.

We implemented 95 patterns for the MUC-4 application. Among the patterns are the following ones that are relevant to the example message:

> killing of <HumanTarget>
> <GovtOfficial> accused <PerpOrg>
> bomb was placed by <Perp>
> on <PhysicalTarget>
> <Perp> attacked <HumanTarget>'s
> <PhysicalTarget> with <Device>
> <HumanTarget> was injured
> <HumanTarget>'s body

As patterns are recognized, incident structures are built up. For example, the sentence

> Guerrillas attacked Merino's home in San Salvador 5 days ago with explosives.

matches the pattern

> <Perp> attacked <HumanTarget>'s
> <PhysicalTarget> in <Location>
> <Date> with <Device>

This causes the following incident to be constructed.

Incident:	ATTACK/BOMBING
Date:	14 Apr 89
Location:	El Salvador: San Salvador
Instr:	"explosives"
Perp:	"guerrillas"
PTarg:	"Merino's home"
HTarg:	"Merino"

The incident type is an attack or a bombing, depending on the Device.

A certain amount of "pseudo-syntax" is done while patterns are being recognized. In the first place, the material between the end of the subject noun group and the beginning of the main verb group must be read over. There are patterns to accomplish this. Two of them are as follows:

> Subject {Preposition NounGroup}* VerbGroup
>
> Subject Relpro {NounGroup | Other}* VerbGroup {NounGroup | Other}* VerbGroup

The first of these patterns reads over prepositional phrases. The second over relative clauses. The verb group at the end of these patterns takes the subject noun group as its subject. There is another pattern for capturing the content encoded in relative clauses:

> Subject Relpro {NounGroup | Other}* VerbGroup

Since the finite-state mechanism is nondeterministic, the full content can be extracted from the sentence

> The mayor, who was kidnapped yesterday, was found dead today.

One branch discovers the incident encoded in the relative clause. Another branch marks time through the relative clause and then discovers the incident in the main clause. These incidents are then merged.

A similar device is used for conjoined verb phrases. The pattern

> Subject VerbGroup {NounGroup | Other}* Conjunction VerbGroup

allows the machine to nondeterministically skip over the first conjunct and associate the subject with the verb group in the second conjunct. Thus, in the sentence

> Salvadoran President-elect Alfredo Cristiani condemned the terrorist killing of Attorney General Roberto Garcia Alvarado and accused the Farabundo Marti National Liberation Front (FMLN) of the crime.

one branch will recognize the killing of Garcia and another the fact that Cristiani accused the FMLN.

The second sort of "pseudo-syntax" that is done while recognizing patterns is attaching genitives, "of" complements, and appositives to their heads, and recognizing noun group conjunctions. Thus, in

> seven children, including four of the vice-president's children

the genitive "vice-president's" will be attached to "children". The "of" complement will be attached to "four", and since "including" is treated as a conjunction, the entire phrase will be recognized as conjoined noun groups.

In the example message, there were 18 relevant patterns. FASTUS recognized 12 of them completely. Because of bugs in implemented patterns, 3 more patterns were recognized only partially.

A rudimentary sort of pronoun resolution is done by FASTUS. If (and only if) a pronoun appears as a Human Target, an antecedent is sought. First the noun groups of the current sentence are searched from left to right, up to four phrases before the pronoun. Then the previous sentences are searched similarly for an acceptable noun group in a left-to-right fashion, the most recent sentence first. This is continued until a paragraph break is encountered, and if nothing is found by then, the system gives up. A noun group is an acceptable antecedent if it is a possible human target and agrees with the pronoun in number. This algorithm worked in 100% of the relevant cases in the first 200 messages of the development set. However, in its one application the example message, it failed. The example is

> According to the police and Garcia Alvarado's driver, who escaped unscathed, the attorney general was traveling with two bodyguards. One of them was injured.

The algorithm incorrectly identifies "them" as "the police".

3.4 Merging Incidents

As incidents are found they are merged with other incidents found in the same sentence. Those remaining at the end of the processing of the sentence are then merged, if possible, with the incidents found in previous sentences.

For example, in the first sentence of Message 48 of TST2, the incident

Incident:	KILLING
Perp:	–
Confid:	–
HTarg:	"Roberto Garcia Alvarado"

is generated from the phrase

> killing of Attorney General Roberto Garcia Alvarado

while the incident

Incident:	INCIDENT
Perp:	FMLN
Confid:	Suspected or Accused by Authorities
HTarg:	–

is generated from the clause

> Salvadoran President-elect Alfredo Cristiani ... accused the Farabundo Marti National Liberation Front (FMLN)

These two incidents are merged, by merging the KILLING and the INCIDENT into a KILLING, and by taking the union of the other slots.

Incident:	KILLING
Perp:	FMLN
Confid:	Suspected or Accused by Authorities
HTarg:	"Roberto Garcia Alvarado"

Merging is blocked if the incidents have incompatible types, such as a KIDNAPPING and a BOMBING. It is also blocked if they have incompatible dates or locations.

There are fairly elaborate rules for merging the noun groups that appear in the Perpetrator, Physical Target, and Human Target slots. A name can be merged with a description, as "Garcia" with "attorney general", provided the description is consistent with the other descriptions for that name. A precise description can be merged with a vague description, such as "person", with the precise description as the result. Two precise descriptions can be merged if they are semantically compatible. The descriptions "priest" and "Jesuit" are compatible, while "priest" and "peasant" are not. When precise descriptions are merged, the longest string is taken as the result. If merging is impossible, both noun groups are listed in the slot.

There were 13 merges altogether in the processing of the example message. Of these, 11 were valid.

One of the two bad merges was particularly unfortunate. The phrase

> ... Garcia Alvarado's driver, who escaped unscathed, ...

correctly generated an attack incident with no injury to the human target, the driver:

Incident:	ATTACK
Perp:	–
PTarg:	–
HTarg:	"Garcia Alvarado's driver"
HEffect:	No Injury

This was merged with the attack on Merino's home

Incident:	BOMBING
Perp:	"guerrillas"
PTarg:	"Merino's home"
HTarg:	"Merino"
HEffect:	–

to yield the combined incident

Incident:	BOMBING
Perp:	"guerrillas"
PTarg:	"Merino's home"
HTarg:	"Merino": "Garcia Alvarado's driver
HEffect:	No Injury

That is, it was assumed that Merino was the driver. The reason for this mistake was that while a certain amount of consistency checking is done before merging victims, and while the system knows that drivers and vice presidents-elect are disjoint sets, the fact that Merino was the vice president-elect was recorded only in a table of titles, and consistency checking did not consult that table.

4 Controlling the FASTUS System

In the course of designing the system, we parameterized a number of characteristics of the system's operation because we believed that the parameterized behavior would reflect tradeoffs in recall versus precision. Subsequent testing revealed that many of these parameters result in both higher recall *and* higher precision when in one state or the other, and therefore we left them permanently in their most advantageous state. Those parameters that seemed to affect recall at the expense of precision were set to produce an optional test run in which we attempted to maximize the system's recall. The effect of these parameters could be described in general as distrusting the system's filters' ability to eliminate templates for incidents that were defined by the MUC-4 rules as being of no interest, including military incidents, incidents in uninteresting countries, and incidents that occurred more than two months before the date of the article. We observed a small but measurable increase in recall at the expense of precision by distrusting our filters.

Here we summarize some of the more interesting results we obtained regarding optimizing our parameter settings for the MUC-4 evaluation.

- *Conservative Merging.* A major emphasis of MUC-4, largely because of its scoring algorithm, was the proper individuation of incidents. When the Conservative Merging option is selected in FASTUS, the system would not merge incidents that had nonoverlapping targets with proper names. When not selected, any merges consistent with the incident types were permitted. Testing revealed that merging should *always* be conservative.

- *Civilian Target Requirement.* Incidents that involved only the military were of no interest in MUC-4. The Civilian Target Requirement filter would reject any template that did not have at least one nonmilitary target, including templates that identified a perpetrator, but no physical or human target at all. This option appears to produce a recall-

precision tradeoff of about one or two points. That is, recall improved at the expense of precision if we distrusted our system and assumed that there really were civilian targets but that they were missed by the system.

- *Subjectless Verb Groups.* This parameter would allow the system to generate an incident structure from a verb together with its object, even if its subject could not be determined. Although early tests showed a recall-precision tradeoff, subsequent and more thorough testing indicated that this should always be done.

- *Military Filtering.* This heuristic causes the system to eliminate all military targets from templates, on the belief that we may have incorrectly merged a military incident with a civilian incident and incorrectly reported the union of the two. Tests show that this filtering improves precision slightly.

- *Spelling Correction.* This parameter controls how much spelling correction the system does. Our experiments indicated that spelling correction hurts, primarily because novel proper names get corrected to other words, and hence are lost. We tried a weaker version of spelling correction which would correct only misspelled words that did not occur on a large list of proper names that we had assembled. This showed an improvement, but spelling correction still had a small negative effect. This was also a surprising result, and we were not willing to abandon spelling correction, and ran all tests with weak spelling correction enabled, although to some extent a complete lack of spelling correction is compensated for by the presence of common misspellings of important domain words like "guerrilla" and "assassinate" in the lexicon.

- *Stale Date Filtering.* This parameter causes filtering of any template that has a date that is earlier than two months before the date of the article. Eliminating this filtering produces an increase in recall at the expense of precision, the magnitude of which depends on how well our date detection currently works. We would expect about a one-point tradeoff.

5 Results in the MUC-4 Evaluation

On a blind test of 100 texts, we achieved a recall of 44% with precision of 55% using the most rigorous penalties for missing and spurious fills. On a different blind test of 100 texts covering incidents from a different time span than the training data, we observed, surprisingly, an identical recall score of 44%; however our precision fell to 52%. It was reassuring to see that there was very little degradation in performance when moving to a time period over which the system had not been trained.

We also conducted a test in which we attempted to maximize the system's recall by not filtering military targets, and allowing incidents with stale dates. On the first test set, this led to a two-point increase in recall at the expense of one point in precision. On the second test set, our recall did not increase, although our precision fell by a point. These results were consistent with our observations during development, although our failure to produce even a small increase in recall on the second test set was somewhat disappointing.

Only General Electric's system [Jacobs et al., 1992] performed significantly better (a recall of 62% and a precision of 53% on the first test set), and their system has been under development for over five years. Given our experience in bringing the system to its current level of performance in three and a half weeks, we feel we could achieve results in that range with another month or two of effort. It is unlikely that human coders would achieve an agreement of more than around 80% on this task. Thus, we believe this technology can perform 75% as well as humans, and considerably faster.

The system is extremely fast. The entire set of 100 messages, ranging from a third of a page to two pages in length, required 11.8 minutes of CPU time on a Sun SPARC-2 processor. The elapsed real time was 15.9 minutes, but observed time depends on the particular hardware configuration involved.

In more concrete terms, this means that FASTUS can read 2,375 words per minute. It can analyze one text in an average of 9.6 seconds.

6 Conclusions

FASTUS was more successful than we ever dreamed when the idea was originally conceived. In retrospect, we attribute its success to the fact that its processing is extremely well suited to the demands of the task. The system's domain-level processing works successfully because the input from phrase recognition is already reliably processed. Phrase recognition does only the linguistic processing that can be done reliably and fast, ignoring all the problems of making attachment decisions, and the ambiguity introduced by coordination and appositives. This input is adequate for the domain-level processing because the domain is sufficiently constrained that, given this initial chunking, the relevant information can be reliably detected and extracted.

The advantages of the FASTUS system are as follows:

- It is conceptually simple. It is a set of cascaded finite-state automata.

- The basic system is relatively small, although the dictionary and other lists are potentially very large.

- It is effective. Only General Electric's system performed significantly better than FASTUS, and it has been under development for a number of years.

- It has very fast run time. The average time for analyzing one message is less than 10 seconds. This is nearly an order of magnitude faster than comparable systems.
- In part because of the fast run time, it has a very fast development time. This is also true because the system provides a very direct link between the texts being analyzed and the data being extracted.

FASTUS is not a text understanding system. It is an information extraction system. But for information extraction tasks, it is perhaps the most convenient and most effective system that has been developed.

One of the lessons to be learned from our FASTUS experience is that an information extraction task is much easier than anyone ever thought. Although the full linguistic complexity of the MUC texts is very high, with long sentences and interesting discourse structure problems, the relative simplicity of the information-extraction task allows much of this linguistic complexity to be bypassed—indeed much more than we had originally believed was possible. The key to the whole problem, as we see it from our FASTUS experience, is to do exactly the right amount of syntax, so that pragmatics can take over its share of the load. For the information extraction task, we think FASTUS displays exactly the right mixture.

References

[Black, 1989] Black, Alan W., "Finite State Machines from Feature Grammars," in Tomita, ed., *International Workshop on Parsing Technologies,* 1989, pp. 277–285.

[Church, 1980] Church, Ken W., *On Memory Limitations in Natural Language Processing*, MIT Laboratory of Computer Science Technichal Report MIT/LCS/TR-245, 1980.

[Hobbs et al., 1992a] Hobbs, Jerry R., Douglas E. Appelt, John Bear, Mabry Tyson, and David Magerman, 1992a. "Robust Processing of Real-World Natural-Language Texts", in *Text-Based Intelligent Systems: Current Research and Practice in Information Extraction and Retrieval*, P. Jacobs, editor, Lawrence Erlbaum Associates, Hillsdale, New Jersey, pp. 13-33.

[Hobbs et al., 1992b] Hobbs, Jerry R., D. Appelt, J. Bear, D. Israel, and M. Tyson, FASTUS: A system for Extracting Information from Natural-Language Text, SRI International Technical Note No. 519, 1992.

[Hobbs et al., 1993] Hobbs, Jerry R., Mark Stickel, Douglas Appelt, and Paul Martin, "Interpretation as Abduction", to appear in *Artificial Intelligence*, 1993.

[Jacobs et al., 1992] Jacobs, P., G. Krupka, L. Rau, L. Childs, and I. Sider, "GE NLTOOLSET: Description of the System as Used for MUC-4," Proceedings of the MUC-4 Workshop, 1992, pp. 177–185.

[Magerman and Weir, 1992] Magerman, D., and C. Weir, "Probabilistic Prediction and Picky Chart Parsing," Proceedings of the Fifth DARPA Workshop on Speech and Natural Language, February, 1992.

[Pereira and Wright, 1991] Pereira, Fernando, and R. Wright, "Finite-State Approximation of Phrase Structure Grammars," Proceedings of the 29th Meeting of the ACL, 1991, pp. 246–255.

[Sundheim, 1992] Sundheim, Beth, ed., 1992. *Proceedings*, Fourth Message Understanding Conference (MUC-4), McLean, Virginia, June 1992. Distributed by Morgan Kaufmann Publishers, Inc., San Mateo, California.