FASTUS: A Cascaded Finite-State Transducer for Extracting Information from Natural-Language Text

Jerry R. Hobbs

Artificial Intelligence Center

SRI International

Menlo Park, California

Abstract

FASTUS is a system for extracting information from natural language text for entry into a database and for other applications. It works essentially as a cascaded, nondeterministic finite-state automaton. There are five stages in the operation of FASTUS. In Stage 1, names and other fixed form expressions are recognized. In Stage 2, basic noun groups, verb groups, and prepositions and some other particles are recognized. In Stage 3, certain complex noun groups and verb groups are constructed. Patterns for events of interest are identified in Stage 4 and corresponding "event structures" are built. In Stage 5, distinct event structures that describe the same event are identified and merged, and these are used in generating database entries. This decomposition of language processing enables the system to do exactly the right amount of domain-independent syntax, so that domain-dependent semantic and pragmatic processing can be applied to the right larger-scale structures. FASTUS is very efficient and effective, and has been used successfully in a number of applications.

1 Introduction

FASTUS is a (slightly permuted) acronym for Finite State Automaton Text Understanding System. It is a system for extracting information from free text in English, Japanese, and potentially other languages as well, for entry into a database and for other applications. It works essentially as a set of cascaded, nondeterministic finite-state transducers. Successive stages of processing are applied to the input, patterns are matched, and corresponding composite structures are built. The composite structures built in each stage provides the input to the next stage.

In Section 2 we describe the information extraction task, especially as exemplified by the Message Understanding Conference (MUC) evaluations (Sundheim 1991, 1992, 1993), which originally motivated the system design. We also discuss the important distinction between information extraction systems and text understanding systems. Section 3 describes the overall architecture of the FASTUS system, and Sections 4 through 8 describe the individual stages. The principal barrier to the widespread use of information extraction technology is the difficulty in defining the patterns that represent one's information requirements. Sections 9 through 12 discuss three successive approaches we have taken to this problem. Section 13 summarizes the advantages of the FASTUS approach.

2 The Information Extraction Task

There are a large number of applications in which a large corpus of texts must be searched for particular kinds of information and that information must be entered into a database for easier access. In the applications implemented so far, the corpora have typically been news articles or telegraphic military messages. The task of the system is to build templates or database entries with information about who did what to whom, when and where.

This task has been the basis of the successive MUC evaluations. In MUC-1 in June 1987, and MUC-2 in May 1989, the corpora were telegraphic messages about naval operations. The task definition for the evaluations took shape over the course of these two efforts.

The corpus for MUC-3 in June 1991 and MUC-4 in June 1992 consisted of news articles and transcripts of radio broadcasts, translated from Spanish, from the Foreign Broadcast Information Service. The focus of the articles was Latin American terrorism. The articles ranged from one third of a page to two pages in length. The template-filling task required identifying, among other things, the perpetrators and victims of each terrorist act described in an article, the occupations of the victims, the type of physical entity attacked or destroyed, the date, the location, and the effect on the targets. Many articles described multiple incidents, while other texts were completely irrelevant.

The task in MUC-5 in July 1993 was to extract information about joint ventures from business news, including the participants in the joint venture, the resulting company, the ownership and capitalization, and the intended

activity.

A typical text is the following:

> Bridgestone Sports Co. said Friday it has set up a joint venture in Taiwan with a local concern and a Japanese trading house to produce golf clubs to be shipped to Japan.
>
> The joint venture, Bridgestone Sports Taiwan Co., capitalized at 20 million new Taiwan dollars, will start production in January 1990 with production of 20,000 iron and "metal wood" clubs a month.

This text is used as an example in the description below of the FASTUS system.

The information to be extracted from this text is shown in the following templates:

TIE-UP-1:

Relationship:	TIE-UP
Entities:	"Bridgestone Sports Co."
	"a local concern"
	"a Japanese trading house"
Joint Venture Company:	"Bridgestone Sports Taiwan Co."
Activity:	ACTIVITY-1
Amount:	NT$20000000

ACTIVITY-1:

Activity:	PRODUCTION
Company:	"Bridgestone Sports Taiwan Co."
Product:	"iron and 'metal wood' clubs"
Start Date:	DURING: January 1990

Seventeen sites participated in MUC-5. It was conducted in conjunction with the ARPA-sponsored Tipster program, whose objective has been to encourage development of information extraction technology and to move it into the user community.

More recently we have implemented systems for extracting information about labor negotiations (the MUC-6 dry-run) and management succession events (MUC-6).

The principal measures for information extraction tasks are recall and precision. *Recall* is the number of answers the system got right divided by the number of possible right answers. It measures how complete or 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 correctness or 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%.

In addition, a combined measure, called the F-score, is often used. It is an approximation to the weighted geometric mean of recall and precision. The F-score is defined as follows:

$$F = \frac{(\beta^2+1)PR}{\beta^2 P+R}$$

where P is precision, R is recall, and β is a parameter encoding the relative importance of recall and precision. If $\beta = 1$, they are weighted equally. If $\beta > 1$, precision is more significant; if $\beta < 1$, recall is.

It is important to distinguish between two types of natural language systems: *information extraction* systems and *text understanding* systems. In information extraction,

- generally only a fraction of the text is relevant; for example, in the case of the MUC-4 terrorist reports, probably only about 10% of the text was 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 at best secondary 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 an information extraction system, rather than a text understanding system. Our original motivation in developing FASTUS was to build a system that was appropriate to the information extraction task.

Although information extraction is not the same as full text understanding, there are many important applications for information extraction systems, and the technology promises to be among the first genuinely practical applications of natural language processing.

3 Overview of the FASTUS Architecture

The key idea in FASTUS, the "cascade" in "cascaded finite-state automata", is to separate processing into several stages. The earlier stages recognize smaller linguistic objects and work in a largely domain-independent fashion. They use purely linguistic knowledge to recognize that portion of the syntactic structure of the sentence that linguistic methods can determine reliably, requiring little or no modification or augmentation as the system

is moved from domain to domain. These stages have been implemented for both English and Japanese.

The later stages take these linguistic objects as input and find domain-dependent patterns among them.

The current version of FASTUS may be thought of as using five levels of processing:

1. Complex Words: This includes the recognition of multiwords and proper names.
2. Basic Phrases: Sentences are segmented into noun groups, verb groups, and particles.
3. Complex Phrases: Complex noun groups and complex verb groups are identified.
4. Domain Events: The sequence of phrases produced at Level 3 is scanned for patterns for events of interest to the application, and when they are found, structures are built that encode the information about entities and events contained in the pattern.
5. Merging Structures: Structures arising from different parts of the text are merged if they provide information about the same entity or event.

As we progress through the five levels, larger segments of text are analyzed and structured.

This decomposition of the natural-language problem into levels is essential to the approach. Many systems have been built to do pattern matching on strings of words. One of the crucial innovations in our approach has been dividing that process into separate levels for recognizing phrases and recognizing event patterns. Phrases can be recognized reliably with purely syntactic information, and they provide precisely the elements that are required for stating the event patterns of interest.

4 Complex Words

The first level of processing identifies multiwords such as "set up", "trading house", "new Taiwan dollars", and "joint venture", and company names like "Bridgestone Sports Co." and "Bridgestone Sports Taiwan Co.". The names of people and locations, dates, times, and other basic entities are also recognized at this level.

Languages in general are very productive in the construction of short, multiword fixed phrases and proper names employing specialized microgrammars, and this is the level at which they are recognized.

Not all names can be recognized by their internal structure. Thus, there are rules in subsequent stages for recognizing unknown possible names as names of specific types. For example, in

> XYZ's sales
>
> Vaclav Havel, 53, president of the Czech Republic,

we might not know that XYZ is a company and Vaclav Havel is a person, but the immediate context establishes that.

5 Basic Phrases

The problem of syntactic ambiguity is AI-complete. That is, we will not have systems that reliably parse natural-language 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 domain-dependent processing.

Stage 2 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.

The first sentence in the sample joint venture text is segmented by Stage 2 into the following phrases:

Company Name:	Bridgestone Sports Co.
Verb Group:	said
Noun Group:	Friday
Noun Group:	it
Verb Group:	had set up
Noun Group:	a joint venture
Preposition:	in
Location:	Taiwan
Preposition:	with
Noun Group:	a local concern
Conjunction:	and
Noun Group:	a Japanese trading house
Verb Group:	to produce
Noun Group:	golf clubs
Verb Group:	to be shipped

Preposition: to
Location: Japan

"Company Name" and "Location" are special kinds of noun group.

Noun groups are recognized by a finite-state grammar that 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 people
the newly elected president
the largest leftist political force
a government and commercial project

Verb groups are recognized by a finite-state grammar that tags them as Active, Passive, Gerund, or 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 Stage 4 resolves the ambiguity if necessary.

Predicate adjective constructions are also recognized and classified as verb groups.

The grammars for English noun groups and verb groups used in MUC-4 are given in Hobbs et al. (1992); although these grammars have subsequently been augmented for domain-specific constructs, the core remains essentially the same.

Unknown or otherwise unanalyzed words are ignored in subsequent processing, unless they occur in a context that indicate they could be names.

The breakdown of phrases into nominals, verbals, and particles is a linguistic universal. Whereas the precise parts of speech that occur in any language can vary widely, every language has elements that are fundamentally nominal in character, elements that are fundamentally verbal or predicative, and particles or inflectional affixes that encode relations among the other elements (Croft, 1991).

6 Complex Phrases

In Stage 3, complex noun groups and verb groups that can be recognized reliably on the basis of domain-independent, syntactic information are recognized. This includes the attachment of appositives to their head noun group,

> The joint venture, Bridgestone Sports Taiwan Co.,

the construction of measure phrases,

> 20,000 iron and "metal wood" clubs a month,

and the attachment of "of" and "for" prepositional phrases to their head noun groups,

production of 20,000 iron and "metal wood" clubs a month.

Noun group conjunction,

a local concern and a Japanese trading house,

is done at this level as well.

In the course of recognizing basic and complex phrases, entities and events of domain interest are often recognized, and the structures for these are constructed. In the sample joint-venture text, entity structures are constructed for the companies referred to by the phrases "Bridgestone Sports Co.", "a local concern", "a Japanese trading house", and "Bridgestone Sports Taiwan Co." Information about nationality derived from the words "local" and "Japanese" is recorded. Corresponding to the complex noun group "The joint venture, Bridgestone Sports Taiwan Co.," the following relationship structure is built:

Relationship:	TIE-UP
Entities:	–
Joint Venture Company:	"Bridgestone Sports Taiwan Co."
Activity:	–
Amount:	–

Corresponding to the complex noun group "production of 20,000 iron and 'metal wood' clubs a month", the following activity structure is built up:

Activity:	PRODUCTION
Company:	–
Product:	"iron and 'metal wood' clubs"
Start Date:	–

Also in the Complex Phrase level of processing, complex verb groups are recognized. Consider the following variations:

GM *formed* a joint venture with Toyota.

GM *announced it was forming* a joint venture with Toyota.

GM *signed an agreement forming* a joint venture with Toyota.

GM *announced it was signing an agreement to form* a joint venture with Toyota.

Although these sentences may differ in significance for some applications, they were equivalent in meaning within the MUC-5 application and would be in many others. Rather than defining each of these variations, with all their syntactic variants, at the domain pattern level, the user should be able to define complex verb groups that share the same significance. Thus, "formed", "announced it was forming", "signed an agreement forming", and "announced it was signing an agreement to form" are all equivalent, at least in this application, and once they are defined to be so, only one Stage 4 pattern needs to be expressed.

Various modalities can be associated with verb groups. In

GM will form a joint venture with Toyota.

the status of the joint venture is "Planned" rather than "Existing". But the same is true in the following sentences.

GM plans to form a joint venture with Toyota.

GM expects to form a joint venture with Toyota.

GM announced plans to form a joint venture with Toyota.

Consequently, as patterns are defined for each of these complex verb groups, the correct modality can be associated with them as well.

Verb group conjunction, as in

Terrorists ***kidnapped and killed*** three people.

is handled at this level as well.

Our current view is that this stage of processing corresponds to an important property of human languages. In many languages some adjuncts are more tightly bound to their head nouns than others. "Of" prepositional phrases are in this category, as are phrases headed by prepositions that the head noun subcategorizes for. The basic noun group together with these adjuncts constitutes the complex noun group. Complex verb groups are also motivated by considerations of linguistic universality. Many languages have quite elaborate mechanisms for constructing complex verbs. One example in English is the use of control verbs; "to conduct an attack" means the same as "to attack". Many of these higher operators shade the core meaning with a modality, as in "plan to attack" and "fail to attack".

7 Clause-Level Domain Events

The input to Stage 4 of FASTUS is a list of complex phrases in the order in which they occur. Anything that is not included in a basic or com-

plex phrase in Stage 3 is ignored in Stage 4; this is a significant source of the robustness of the system. Patterns for events 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, each pair of relevant head word and phrase type—such as "company-NounGroup", "formed-PassiveVerbGroup", "bargaining-NounGroup", and "bargaining-PresentParticipleVerbGroup"— has an associated set of state transitions.

In the sample joint-venture text, the domain event patterns

<Company/ies> <Set-up> <Joint-Venture> with <Company/ies>

and

<Produce> <Product>

are instantiated in the first sentence, and the patterns

<Company> <Capitalized> at <Currency>

and

<Company> <Start> <Activity> in/on <Date>

are instantiated in the second. These four patterns result in the following four structures being built:

Relationship:	TIE-UP
Entities:	"Bridgestone Sports Co."
	"a local concern"
	"a Japanese trading house"
Joint Venture Company:	–
Activity:	–
Amount:	–

Activity:	PRODUCTION
Company:	–
Product:	"golf clubs"
Start Date:	–

Relationship:	TIE-UP
Entities:	–
Joint Venture Company:	"Bridgestone Sports Taiwan Co."
Activity:	–
Amount:	NT$20000000

(This is an augmentation of the previous relationship structure.)

Activity:	PRODUCTION
Company:	"Bridgestone Sports Taiwan Co."
Product:	–
Start Date:	DURING: January 1990

A certain amount of "pseudo-syntax" is done in Stage 4. 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 set of patterns for capturing the content encoded in relative clauses, of the form

Subject Relpro {NounGroup | Other}* VerbGroup

The finite-state mechanism is nondeterministic. With the exception of passive clauses subsumed by larger active clauses, all events that are discovered in this stage of processing are retained. Thus, 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. That is, when the first verb group is encountered, all its complements and adjuncts are skipped over until a conjunction is encountered, and then the subject is associated with a verb group, if that is what comes next. 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.

In addition, irrelevant event adjuncts in the verb phrase are read over while relevant adjuncts are being sought.

8 Merging Structures

The first four stages of processing all operate within the bounds of single sentences. The final level of processing operates over the whole text. Its task is to see that all the information collected about a single entity or relationship is combined into a unified whole. This is one of the primary ways the problem of coreference is dealt with in our approach.

The three criteria that are taken into account in determining whether two structures can be merged are the internal structure of the noun groups, nearness along some metric, and the consistency, or more generally, the compatibility of the two structures.

In the analysis of the sample joint-venture text, we have produced three activity structures. They are all consistent because they are all of type PRODUCTION and because "iron and 'metal wood' clubs" is consistent with "golf clubs". Hence, they are merged, yielding

Activity:	PRODUCTION
Company:	"Bridgestone Sports Taiwan Co."
Product:	"iron and 'metal wood' clubs"
Start Date:	DURING: January 1990

Similarly, the two relationship structures that have been generated are consistent with each other, so they are merged, yielding,

Relationship:	TIE-UP
Entities:	"Bridgestone Sports Co."
	"a local concern"
	"a Japanese trading house"
Joint Venture Company:	"Bridgestone Sports Taiwan Co."
Activity:	-
Amount:	NT$20000000

Both of these cases are examples of identity coreference, where the activities or relationships are taken to be identical. We also handle examples of inferential coreference here. A joint venture has been mentioned, a joint venture implies the existence of an activity, and an activity has been mentioned. It is consistent to suppose the activity mentioned is the same as the activity implied, so we do. The Activity field of the Tie-Up structure is filled with a pointer to the Activity structure.

For a given domain, there can be fairly elaborate rules for determining whether two noun groups corefer, and thus whether their corresponding entity structures should be merged. A name can corefer with a description, as "General Motors" with "the company", provided the description is consistent with the other descriptions for that name. A precise description, like "automaker", can corefer with a vague description, such as "company", with the precise description as the result. Two precise descriptions can corefer if they are semantically compatible, like "automaker" and "car manufacturer".

9 FastSpec: A Declarative Specification Language

In the first version of FASTUS (Hobbs et al., 1992), the finite-state transducers were represented in a table of state changes with blocks of code associated with the final states. Only the developers were able to define patterns in this system. The next version, used in MUC-5 (Appelt et al., 1993), had a graphical interface for defining state changes and allowed blocks of code to be associated with transitions. Only a small group of cognoscenti were able to use this system.

In 1994 we defined and developed a declarative specification language called FastSpec. It enabled the easy definition of patterns and their associated semantics, and made it possible for a larger set of users to define the patterns.

FastSpec allows the definition of multiple grammars, one for each phase. The terminal symbols in the grammar for a phase correspond to the objects produced by the previous phase, and their attributes can be accessed and checked. The rules have a syntactic part, expressing the pattern in the form of a regular expression, with attribute and other constraints permitted on the terminal symbols. They also have a semantic part, which specifies how attributes are to be set in the output objects of the phase.

The following is a fragment of a grammar for verb groups in the Basic Phrase Recognizer:

```
VG --> VG2 Adv* V-en:1;
       head = (obj 1);
       active = T;
```

```
        aspect = perf;;

VG2 --> VG1 ''have'';;

VG2 --> V[have]:1 (Not);
        tense = (tense 1);;

VG1 --> Modal:1 (Not) Adv*;
        tense = (tense 1);;

Not --> ''not'';
        negative = T;;
```

This covers a phrase like "could not really have left". `V-en` and `Adv` refer to words that are past participles and adverbs, respectively. `V[have]` indicates some form of the verb "have". The use of indices like ":1" allows us to access the attributes of terminal symbols. The semantics in these rules sets the features of `active`, `aspect`, `tense`, and `negative` appropriately, and sets `head` to point to the input object providing the past participle.

The following is one rule in a grammar for the Clause-Level Event Recognizer for the labor negotiations domain used in the dry run of MUC-6 in April 1995.

```
Event -->
    Event-Adj*  NG[org]:1 (Compl)
    VG[active,resume-word]:2  NG[talk-word]
    {''with'' NG[org]:3 | Event-Adj}*;
        type = Talk;
        parties = (List (obj 1) (obj 3));
        talk-status = Bargaining;;
```

This says that when an organization resumes talks with an organization, it is a significant event. `Event-Adj` is matched by temporal, locative, epistemic and other adverbial adjuncts. `Compl` is matched by various possible noun complements. This rule creates an event structure in which the event type is `Talk`, the parties are the subject and the object of "with" matched by the patterns, and the talk status is `Bargaining`.

FastSpec has made it immensely easier for us to specify grammars.

10 Compile-Time Transformations

For an application such as MUC-5 in which we had to recognize the products made by companies, we would want a pattern that would recognize

> GM manufactures cars.

But in addition to writing a rule for this pattern, we would have to write rules for all the syntactic variations of the simple active clause, to recognize

> Cars are manufactured by GM.
> ... GM, which manufactures cars.
> ... cars, which are manufactured by GM.
> ... cars manufactured by GM.
> ... for GM to manufacture cars.
> ... for cars to be manufactured by GM.
> GM is a car manufacturer.

Moreover, in each of these patterns we would need to allow the occurrence of temporal, locative, and other adverbials. Yet all of these variations are predictable, and every time we want the first pattern we want the others as well.

This consideration led us to implement what can be called "compile-time transformations". Expensive operations of transformation are not done while the text is being processed. Instead, the transformed patterns are generated when the grammar is compiled. We have implemented a number of parameterized metarules that specify the possible linguistic variations of the simple active clause, expressed in terms of the subject, verb, and object of the active clause, and having the same semantics. Then domain-specific patterns are defined that provide particular instantiations of the metarules.

The metarule for the basic active clause, as in "The company resumed talks", is

```
Event -->
    Event-Adj* NG[??subj]:1 VG[active,??head]:2 NG[??obj]:3
    {P[??prep] NG[??pobj]:4 | Event-Adj}*;
    ??semantics;;
```

Once the variables `??subj`, `??head`, `??obj`, `??prep`, and `??pobj` are defined by the user, they are plugged into this rule and a new specific rule is generated. Each of these variables is a (list of) lexical or other attributes, and when they are plugged into the metarule, they define a pattern that is constrained to those attributes. Adverbials are recognized by matching a sequence of input objects with `Event-Adj`. Indices are associated with each of the arguments of the head's predication, and these can be used in the semantics specified for particular pattern.

The metarule for passives, as in "Talks were resumed", is

```
Event -->
    NG[??obj]:3 VG[passive,??head]:2
    {P[??prep] NG[??pobj]:4 | Event-Adj}*;
    ??semantics;;
```

The object still has the index 3, so that the same semantics can be used for the passive as for the active.

The metarule for relative clauses with a gapped subject, as in "the company, which resumed talks ...", is

```
Event -->
    NG[??subj]:1 P[relpro] VG[active,??head]:2 NG[??obj]:3
    {P[??prep] NG[??pobj]:4 | Event-Adj}*;
    ??semantics;;
```

The metarule for nominalizations, as in "the company's resumption of talks", must appear in the Complex Phrase Recognizer and has the form

```
ComplexNG -->
    (NG[??subj]:1 P[gen]) NG[??head]:2 (''of'' NG[??obj]:3)
    {P[??prep] NG[??pobj]:4 | Event-Adj}*;
    ??semantics;;
```

Here all the arguments are optional. We could simply have the bare nominal.

In addition to the basic patterns, middle verbs and symmetric verbs are handled. Middle verbs are verbs whose object can appear in the subject position and still have an active verb.

They resumed the talks.
The talks resumed.

The metarule that implements the middle "transformation" is as follows:

```
Event -->
    NG[??obj]:3 VG[active,??head]:2
    {P[??prep] NG[??pobj]:4 | Event-Adj}*;
    ??semantics;;
```

Symmetric verbs are verbs where an argument linked to the head with the preposition "with" can be moved into a subject position, conjoined with the subject. For example,

The union met with the company.

The union and the company met.

The meeting between the union and the company.

To handle this there are patterns in the Complex Phrase Recognizer that recognize a conjunction of the subject and the prepositional argument, when the verb is designated symmetrical:

```
NG[??subj] ''and'' NG[??pobj]
```

This is then given a special attribute symconj, and in the Clause-Level Event Recognition phase, complex noun groups with this property are sought as subjects for symmetric verbs.

```
Event -->
    Event-Adj* NG[symconj] VG[active,??head]:2
    NG[??obj]:3 Event-Adj*;
    ??semantics;;
```

With this set of metarules, defining the necessary patterns becomes very easy. One need only specify the subject, verb, object, preposition, and prepositional object, and the classes of metarules that need to be instantiated, and the specific rules are automatically generated. For example, the specification for "resume" would be

```
Transformations:  Middle, Basic:
    1: Subj = org;
    2: Head = resume-word;
    3: Obj = talk-word;
       Prep = ''with'';
    4: PObj = org;
       Semantics =
         <type = Talk;
          parties = (list (obj 1) (obj 4));
          talk-status = Bargaining;;>;
```

In the semantics, we set the type of event to be `Talk` and the talk status to be `Bargaining`. The parties are those referred to by the subject (1) and the prepositional object (4).

Our experience with this aspect of the FASTUS system has been very encouraging. During the preparation for MUC-6, it took us only about one day to implement the necessary clause-level domain patterns, because of the compile-time transformations.

11 Atomic versus Molecular Approaches

There are two approaches that have emerged in our experience with FASTUS. They might be called the "atomic" approach and the "molecular"

approach. Both approaches are made easier by FastSpec and the compile-time transformations.

In the atomic approach, the system recognizes entities of a certain highly restricted type and assumes that they play a particular role in a particular event, based on that type; then after event merging it is determined whether enough information as been accumulated for this to be an event of interest. This approach is more noun-driven, and its patterns are much looser. It is most appropriate when the entity type is highly predictive of its role in the event. The labor negotiations domain was of this character. When one sees a union, it can only go into the union slot of a negotiation event.

In the molecular approach, the system must recognize a description of the entire event, not just the participants in the event. This approach is more verb-driven, and the patterns tend to be tighter. It is most appropriate when the syntactic role of an NP is the primary determinate of the entity's role in the event. The terrorist domain of MUC-3 and MUC-4, the joint venture domain of MUC-5 and the management succession domain of MUC-6 were of this character. You can't tell from the fact that an entity is a person whether he is going into or out of a position at an organization. You have to see how that person relates to which verb.

The distinction between these two approaches can be used as a conceptual tool for analyzing new domains.

12 Adapting Rules from Examples

The FastSpec language and the compile-time transformations make it easier for linguists and computer scientists to define patterns. But they do not enable ordinary users to specify their own patterns. One way to achieve this would be to have automatic learning of patterns from examples provided by the user. We have begun in a small way to implement such an approach.

We need a way for the user to supply a mapping from strings in the text to entries in the template. This can be accomplished by having a two-window editor; the text being annotated or analyzed is in one window, the template in the other. The user marks a string in the text, and then either copies the string to a template entry or enters the set fill that is triggered by the string. Such a system is first of all a convenient text editor for filling data bases from text by hand. But if the system is trying to deduce the implicit rules the user is responding to to make the fills, then the system is automatically constructing an information extraction system as well.

We have implemented a preliminary experimental version of such a system, and are currently developing a more advanced one. We assume that the user somehow provides a mapping from text strings to template entries and that the semantics of the rule is completely specified by such a mapping. Moreover, we are only handling the case where the new rule to be induced is a specialization of an already existing rule, in the sense that

<Location> "-" "based"

is a specialization of

<Noun> "-" <Past-Participle>

In general, the problem of rule induction is very hard. What we are doing is a tractable and useful special case.

The first problem is to identify the phase in which the new rule should be defined. To do this, we identify the highest-level phase (call it Phase n) in which the constituent boundaries produced by the phase correspond to the way the user has broken up the text. A new rule is then hypothesized in Phase $n+1$. For example, if the user has marked the string "the union resumed talks with the company" and placed "the union" in one slot and "the company" in another, then Phase n is the Complex Phrase Recognizer, since it provides those noun groups as independent objects. On the other hand, if the string is "the union's resumption of talks with the company", then the Complex Phrase Recognizer will not do, since it combines at least "the union" and possibly "the company" into the same complex noun group as "resumption". We have to back up one more phase, to the Basic Phrase Recognizer, to get these noun groups as independent elements.

In the current version, we determine what Phase $n+1$ rule matches the entire string and then construct as general as possible a specialization of that rule. For the semantics of the specialized rule, we encode the mapping the user has constructed.

Determining the correct level of generalization of the hypothesized rule is a difficult problem. There are some obvious heuristics that we have implemented, such as generalizing "38" to Number and "Garrick" to Person. But should we generalize "United Steel Workers" to Union or to Organization?

Our current approach is to be conservative and to experiment with various options.

Once the rule is hypothesized it will be presented to the user in some form for feedback and validation. How best to implement this is still a research issue.

13 Conclusions

Finite-state technology is sometimes characterized as *ad hoc* and as *mere* pattern-matching. However, our approach of using a *cascade* of finite-state machines, where each level corresponds to a linguistic natural kind, reflects important universals about language. It was inspired by the remarkable fact that very diverse languages all show the same nominal element - verbal element - particle distinction and the basic phrase - complex phrase distinction. Organizing a system in this way lends itself to greater portability among domains and to the possibility of easier acquisition of new patterns.

The advantages of the FASTUS system are as follows:

- It is conceptually simple. It is a set of cascaded finite-state automata.
- It is effective. It has been among the leaders in recent evaluations.
- It has very fast run time.
- In part because of the fast run time, it has a very fast development time. This is also true because the system provides a 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 many information extraction tasks are much easier than anyone ever thought. Although the full linguistic complexity of the texts is often 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 many information extraction tasks, we think FASTUS displays exactly the right mixture.

While FASTUS is an elegant achievement, the whole host of linguistic problems that were bypassed are still out there, and will have to be addressed eventually for more complex tasks, and to achieve higher performance on simple tasks. We have shown one can go a long way with simple techniques. But the hard problems cannot be ignored forever, and scientific progress requires that they be addressed.

Acknowledgments

The FASTUS system is the result of collaborative work with Douglas Appelt, John Bear, David Israel, Megumi Kameyama, Karen Myers, David Martin, Mark Stickel, and Mabry Tyson. The FASTUS system was originally built

under SRI internal research and development funding. Specific applications and improvements have been funded by the (Defense) Advanced Research Projects Agency under Office of Naval Research contract N00014-90-C-0220, Office of Research and Development contract 94-F157700-000, and Naval Command, Control and Ocean Surveilliance Center contract N66001-94-C-6044, and by the US Army Topographic Engineering Center under contract no. DACA76-93-L-0019. Specific developments have also been funded by commercial contracts.

References

[1] Appelt, Douglas E., Jerry R. Hobbs, John Bear, David Israel, Megumi Kameyama, and Mabry Tyson, 1993a. "The SRI MUC-5 JV-FASTUS Information Extraction System", *Proceedings*, Fifth Message Understanding Conference (MUC-5), Baltimore, Maryland, August 1993.

[2] Appelt, Douglas E., Jerry R. Hobbs, John Bear, David Israel, and Mabry Tyson, 1993b. "FASTUS: A Finite-State Processor for Information Extraction from Real-World Text", *Proceedings*. IJCAI-93, Chambery, France, August 1993.

[3] Croft, William, 1991. *Syntactic Categories and Grammatical Relations: The Cognitive Organization of Information*, University of Chicago Press, Chicago, Illinois.

[4] Hobbs, Jerry R., Douglas E. Appelt, John Bear, David Israel, and Mabry Tyson, 1992. "FASTUS: A System for Extracting Information

from Natural-Language Text", SRI Technical Note 519, SRI International, Menlo Park, California, November 1992.

[5] Sundheim, Beth, ed., 1991. *Proceedings*, Third Message Understanding Conference (MUC-3), San Diego, California, May 1991. Distributed by Morgan Kaufmann Publishers, Inc., San Mateo, California.

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

[7] Sundheim, Beth, ed., 1993. *Proceedings*, Fifth Message Understanding Conference (MUC-5), Baltimore, Maryland, August 1993. Distributed by Morgan Kaufmann Publishers, Inc., San Mateo, California.