The *Diderot* Information Extraction System

Jim Cowie, Takahiro Wakao, Louise Guthrie, Wang Jin
Computing Research Laboratory, New Mexico State University
&
James Pustejovsky, Scott Waterman
Computer Science Department, Brandeis University
Email: jcowie@nmsu.edu

ABSTRACT

Diderot is an information extraction system built at CRL and Brandeis University over the past year. It was produced as part of our efforts in the Tipster project. Diderot has already been converted from one subject domain to another and versions of the system have been made for two languages. The same system architecture has been used for English and Japanese and a comparison is made of the processing done at each stage for the two languages. The domain of the texts used is the business area of joint ventures.

1. Introduction

Natural Language Processing (NLP) is beginning to appear viable for a number of large scale tasks related to information analysis. This is in part due to improved data resources, for example machine readable dictionaries and large scale corpora and part due to the increased speed of machines and the reduced cost of storage. These coupled with the fact that simple methods can accomplish much of the basic tasks of Information Extraction and the realization that linguistically rich and complex systems can prove extremely brittle when confronted with novel constructions account for this optimistic attitude. Applications are emphasizing robustness, portability and modularity. There has also been a new emphasis on the actual evaluation of systems against some fixed measures of performance. This is particularly evident in the Tipster project for which the system described here was developed.

Information Extraction (IE) is the name now being applied to a process analogous to reading a document and filling out a form which captures the crucial information. Specifically, IE requires the structuring of information from texts into a record format suitable for creating data-bases, or for use in other computer based applications. The information extracted is normally names of people and organizations, place names the types of these entities and specific relationships between them. Successful automatic extraction depends on limiting the subject domain being considered and limiting the scope of the information being extracted. The grammars used for this task are normally limited in their coverage and hence parsing techniques must be robust. Within their limited scope information extraction systems now appear to be approaching levels of performance which allow the production of adequately accurate information. This has been evidenced by the steadily improving performance of the text extraction systems which have been evaluated at successive Message Understanding Conferences[1, 2].

2. Tipster - Portability and Re-configurability

The Tipster research initiative is intended as a first stage in the development of working IE systems. The approach adopted is similar to that of the earlier Message Understanding Conferences[6]. A large number (1,000) of domain specific texts are supplied along with a complex set of rules specifying what information is to be extracted. The texts are accompanied by accurate templates (filled structures) which have been completed by human analysts. This data is used to train and test systems being developed to perform the task automatically. Periodically, a new set of texts are released and the IE systems must produce a set of templates for the new texts. These are then scored automatically against human produced templates.

It is the intention of the funders of the research that after the initial two years some combination of the best methods produced by the participating reseach groups be used to produce an actual working system. To this end the specification of the requirements for Tipster are, in a way, an attempt to pose a set of NLP problems. The two largest of these problems are the development of methods which can be adapted in both of two directions; language independence and domain independence. The first is encouraged by requiring that the extraction task be carried out for texts in two languages (English and Japanese). Note that this is not a translation task; Japanese text is used to fill the extracted fields for Japanese extraction. The second dimension is driven by the requirement that information is extracted for two subject areas - broadly business and micro-electronics.

3. Project Objectives

Our objectives in this research have been as follows:

- to develop and implement a language-independent framework for lexical semantic representation, and develop and implement a robust integration of that framework into a language-independent theory of semantic processing;
- to investigate and implement language independent techniques for automating the building of lexical knowledge bases from machine readable resources;
- to implement statistical tools for the tuning of lexical structures to specific domains;
- to implement to use of language independent statistical techniques for identifying relevant passages of documents for more detailed analysis;
- to develop and implement a set of robust multi-pass finite-state feature taggers;
- to develop and implement the equivalent methods for Japanese.

The most innovative aspects of the project involve three different components of the system:

- the use of multi-pass finite-state feature tagging, which takes advantage of specialized grammars parsing dates, locations, person and company names in both English and Japanese.
- the automatic construction of the core lexicon for a particular language, and the use of statistical methods applied to corpus to create a specific domain lexicon.
- the automatic conversion to DCG parse rules of lexical cospecification information, tuned against the corpus for each lexical item.

Parsing is accomplished in two stages;

1. Multi-pass FS feature tagging
2. Conventional DCG parsing based on lexical cospecification information.

4. Technical Aspects of the System

There are two major components to the DIDEROT system, one off-line the other on-line. The first involves the automatic construction of the lexicon and partial syntactic forms for a given language and domain, along with subsequent tuning and refinement. The second component involves the actual tagging, parsing, merging, and template generation from the texts.

This component facilitates the derivation of vocabulary automatically in languages from machine readable sources, and uses statistically-based techniques to determine the relevant domain-dependent vocabulary of words and phrases from text samples.

Statistical techniques can be used extensively to assist in the development of the various lexicons of the TIPSTER system. Initially, a simple frequency count of the tokens in the initial corpora can be used to highlight those words which should be targeted, essentially determining what core vocabulary elements to tune against the corpus.

4.1. Deriving Lexical Entries from Machine-Readable Dictionaries

The lexical knowledge base consists of lexical items called generative lexical structures (GLSs), after Pustejovsky[8]. This model of semantic knowledge associated with words is based on a system of generative devices which is able to recursively define new word senses for lexical items in the language. For this reason, the algorithm and associated dictionary is called a *generative lexicon*. The lexical structures contain conventional syntactic and morphological information along with detailed typing information about arguments

The creation of the GLS lexicon begins with the printer's tape of the Longman Dictionary of Contemporary English (LDOCE), Proctor[7]. This was parsed and analysed by the CLR lexical group to give a tractable formatted version of LDOCE called LEXBASE[4]. LEXBASE contains syntactic codes, inflectional variants, and boxcodes, selectional information for verbs and nouns, indicating generally what kind of arguments are well-formed with that lexical item. A GLS entry is automatically derived from LEXBASE by parsing the LEXBASE format for specific semantic information [10]. The most novel aspect of this conversion involves parsing the example sentences as well as parenthetical texts in the definition. This gives a much better indication of argument selection for an item than do the the boxcodes alone. For example, the verb *establish* is converted into the following GLS entry as a result of this initial mapping.

```
gls(establish,
  syn([type(v),
    code(gcode_t1),
    eventstr([]),
    ldoce_id(establish_0_1),
    caseinfo([subcat1(a1),
```

```
        subcat2(a2),
        case(a1,np),
        case(a2,np)]),
   inflection([ing(establishing),
         pastp(established),
         plpast(established),
         singpast1(established),
         singpast2(established),
         singpast3(established),
         past(established),
         pl(establish),
         sing1(establish),
         sing2(establish),
         sing3(establishes)])]),
 qualia([formal(['set up']),
     telic([]),
     const([]),
     agent([])]),
 args([arg1(a1,
      syn([type(np)]),
      qualia([formal([code_h]),
          telic([]),
          const([]),
          agent([])])),
    arg2(a2,
      syn([type(np)]),
      qualia([formal([code_2,shop]),
          telic([],
          const([]),
          agent([])]))]),
 cospec([[A1,*,self,*,A2],
     [A2,*,past(self),*,by,A1]]),
 types([]),
 template_semantics([]).
```

The two arguments to the verb *establish* are minimally typed as a result of the conversion from LDOCE, this information being represented as a *type-path* for each argument, Pustejovsky and Boguraev[11]. For example, the subject is typed as a `human` and the object as a type-path relating `solid` and the specific type encounted in definition, `shop`.

The syntactic and collocational behavior of a word is represented in the `cospec` (cospecification) field of the entry. The `cospec` of a lexical item can be seen as paradigmatic syntactic behavior for a word, involving both abstract types as well as lexical collocations. This field is created automatically by reference to the syntactic codes of the verb, as represented in LDOCE, in this case T1 (i.e., basic transitive). That is, the `cospec` encodes explicit information regarding the linear positioning of arguments, as well as semantic constraints on the arguments as imposed by the typing information in the qualia. The syntactic representation of a word's environment may appear flat, but the semantic interpretation is based on a unification-like algorithm which creates a much richer functional structure. Theoretically, the expressive power of converting the `cospecs` of a GLS into DCG parse rules is equivalent to the power of a Lexicalized Tree Adjoining Grammar with collocations (Shieber[12]).

4.2. Tuning GLS Entries against a Corpus

It is very often the case that the lexical structures derived from MRDs do not reflect the behavior of the actual words in a given corpus. There are two major ways in which the lexical structure may differ from its actual corpus use:

- syntactic mismatches
- selectional mismatches

In the joint venture domain, there is a sense of *establish* as a three argument verb, just in case the second argument is a relational noun such as *joint venture* or *consortium*.

> IBM will establish a joint venture with a local company.

Using statistical techniques described in Pustejovsky[9] and similar to those in Grishman and Sterling[3], we can automatically identify the contexts and type restrictions for this use of the verb in the corpus.

An example of a selectional mismatch comes with cases of *type coercion*, where the verb expects one type but an apparently inconsistent type appears:

> Mannheim Industries announced a joint venture with Maykoe Inc.

Here, the verb *announce* is typed from LDOCE as taking a sentence, but it appears in the corpus with an NP. The acquisition technique presented in Pustejovsky[9] shows how coercive contexts can be identified automatically from corpus analysis.

Another consideration when tuning lexical forms against a corpus is the proper representation of domain-specific syntactic and semantic typing. We are currently restructuring the lexical database to allow for greater portability across domains. Corpus-tuned entries will carry the more generic domain independent syntactic forms through inheritance in the GLS database, as the following lexical structures illustrate.

```
gls(create_verb_lcp),
  syn(...),
  args([...,
    arg2(A2,
      syn([type(np)]),
      qualia([formal([code_2,artifact])])]),
    arg3(A3,
      syn([type(np)]),
      qualia([formal([instrument])])])]),
  qualia([formal([trans_verb_lcp)]),
  cospec([[A1,*,self,*,A2],
     [A2,*,self,*,by,A1]]),
  ...).

gls(tie_up_lcp),
  syn(...),
  args([arg1(A1,
      syn([type(np)]),
      qualia([formal([code_2,organization)])]),
    arg2(A2,
      syn([type(np)]),
      qualia([formal([code_2,organization)])]),
    arg3(A3,
      syn([type(np)]),
     qualia([formal([code_2,joint_organ)])])]),
  qualia([formal([create_verb_lcp)]),
  cospec([[A1,*,self,*,A2,*,with,A3],
   [A1,and,A3,*,self,*,A2],
   [A1,together,with,A3,*,self,*,A2],
   [A2, is, to, be, self, *, with, A3]]),
  ...).

gls(establish,
  syn(...),
  args(...),
  qualia([formal(tie_up_lcp)]),
  cospec([
   [A1, *, signed, *, agreement, *, self, A2],
   [A1, *, self,*,joint,venture, A2,with,A3],
   [self, include, A2],
   [A2, was, self, with, A3]]),
 types(tie_up_verb),
 template_semantics(pt_tie_up,tie_up([A1,A3],
        A2,_,existing,_))).
```

The hierarchical structuring of lexical information follows in a fairly principled way the theory of Lexical Conceptual Paradigms as outlined in Pustejovsky[8].

5. Production of the Lexically-Driven Partial Parser

The generative lexical structures are "universal" in character in the same way that phrase structural descriptions provide a general, expressive language for describing the syntactic structures for widely different linguistic behaviors from different languages. In general, the lexical structures described here can be thought of as providing for the shallowest possible semantic decomposition while still capturing significant generalizations about how words relate conceptually to one another. The lexical entries for Japanese follow exactly the same specifications, with the same degree of flexibility.

The partial grammar is derived from the tuned GLS entries. Prolog Definite Clause Grammar rules are produced automatically from the patterns given in the GLS cospecification. The rules are then compiled into the working version of the system. A similar process also produces lists of words with GLS entries for the tagging programs.

For example this partial GLS entry will give rise to two parse rules.

```
gls(join,
 syn([type(v),
 code(gcode_t1),
 eventstr([]),
 ldoce_id(join_1_2),
....
arg3(A3,
 syn([type(np)]),
 qualia([formal([code_h,organization]),
 telic([]),
 const([]),
 agent([])]))]),
cospec([[A1,*,self,*,A2,*,with,A3],
 [A1,and,A3,*,self,*,A2],
 [A1,together,with,A3,*,self,*,A2]]),
 types(tie_up_verb),
 template_semantics(pt_tie_up,tie_up([A1,A3],
      A2,_,existing,_))).
```

The single pattern - `[A1,*,self,*,A2,*,with,A3]` gives the rule -

```
rule(join,
 template_semantics(pt_tie_up,
    tie_up([A1,A3],A2,_,existing,_)))
  -->
     glsphrase(A1,[type(np)],
          formal([code_h,organization])),
     ignore,
     term(gls(_,type([join,v]),_)),
     ignore,
     glsphrase(A2,[type(np)],
          formal([code_u,organization])),
```

```
    ignore,
    word(with),
    glsphrase(A3,[type(np)],
         formal([code_h,organization])).
```

6. Statistical Filtering Techniques

Statistical information is used to predict whether a paragraph holds important information and is relevant to completing a template. This allows the parser to skip non-relevant paragraphs or even entire texts.

We have developed a procedure for detecting document types in any language which is founded on a sound statistical basis using probabilistic models of word occurrence. The system requires training texts for the types of documents to be classified and may operate on letter grams of appropriate size or on actual words or word-grams of the language being targeted. We have and developed optimal detection algorithms from automatically generated 'word' lists.

The theoretical results on which the method [5] is based assure us that documents can be classified correctly if appropriate sets of words can be chosen for each document type. Intuitively, sets of words are needed which appear much more often in one text type than the other. We refer to these as 'distinguishing' words. In fact, the words do not need to appear in either text type very often to have near perfect probability of correct classification for each document type. If we consider the case of two document types, from our theoretical results we know, for example, that if words appear with probability 0.1 in one text type and with a much lower probability in the other (say (0.03) then we can classify documents of size 200 'words' correctly about 99% of the time.

We have used a simple heuristic to find the sets of distinguishing words, and with this method we have easily been able to discover word sets with the properties necessary to ensure our theoretical results hold. Each document type is modeled with a multinomial distribution having three parameters. The parameters represent the conditional probability that a word occurs in the distinguishing word set for the first type document, the conditional probability that a word occurs in the distinguishing word set for the second type document, and the conditional probability that a word occurs in neither distinguishing set, given the document type.

The method has been extended to the identification of relevant paragraphs for the Tipster joint-venture texts. In this case, each paragraph is treated as a 'document' and the 'document types' are relevant and non-relevant. We define 'relevant' in this case to be a paragraph which contains some information required by the Tipster template in the joint venture domain. This is a more difficult problem than the one described above for two document types such as Music and Business, due to several factors: the non homogeneous nature of the texts (many of the joint venture articles contain paragraphs about a range of topics), the difficulty of deriving training sets of text, the fact that the sample size is very small in the case of a paragraph, and the similarity of the two document types ('relevant' and 'non-relevant' is a much finer distinction than what appears in two distinct subject areas). The process uses two sets of words: one which occurs with high probability in the texts of interest and the other which occurs in the 'non-interesting' texts. Word lists are derived automatically by finding those words in the relevant training set which occurred within a threshold of most frequently occurring words in the relevant paragraphs and not in the non-relevant paragraphs. A set of non-relevant words is obtained by the same method but reversing the roles of relevant and non-relevant paragraphs.

The sets of relevant and irrelevant paragraphs which were used for training, were produced automatically using offset files which are produced as a product of the human template filling activity. Unfortunately, only a small number of valid files were available. (English - 77 relevant and 370 irrelevant paragraphs, Japanese - 28 relevant and 165 irrelevant paragraphs). The Japanese texts were segmented into words using Juman. A scoring system was implemented to test the system's predictions. This allows us to tune the relevancy test and relevant word list based on the systems performance.

Our best results so far are -

```
English:  recall=83% precision=87%
Japanese: recall=85% precision=100%
```

7. System Overview

An outline of the functions of the main system modules are given here. This is intended to provide a context for the more detailed description of each module which follows. The structures of the Japanese and English systems are very similar. In the examples of intermediate output either Japanese or English may be shown. The system architecture is shown in figure 1.

The input text to the system is processed by three independent pre-processing modules:

- A chain of finite-state feature taggers - these mark: names, organization names, place names, date expressions and other proper names (depending on the domain),

Figure 1: System Overview

- A part of speech tagger,
- A statistically based determiner of paragraph relevance.

Each of these modules produces a set of Prolog facts. This output then passes into the head of a chain of processes each of which gives rise to further refinements of the text:

- Merge - Here semantic tags, which may mark phrasal units, are merged with POS tags, which mark individual words.
- Compound noun recognizer - this groups words and phrases into compound nouns using POS and semantic information.
- Parser - the relevant paragraph information is used to select which sentences to process further. The sentences containing the marked up noun-phrase groups are then parsed to produce a partially completed representation of the relevant semantic content of the sentence (frames).
- Reference resolver - the frames are then merged based on name matching and noun compounds beginning with definite articles.
- Template formatter - this transforms the resolved frames into the final output form.

7.1. Semantic Tagging

This component is based on a pipeline of programs. These are all written in 'C' or flex. It marks all of the following —

- Organization names
- Human names
- Place names
- Date expressions

The tagging programs use three distinct methods —

- Direct recognition of already known unambiguous names, using a longest string match.
- Recognition using textual patterns only.
- Two pass method marking ambiguous, but potential names, and subsequently verifying they fit a pattern.

The whole process is data dependent —

- 50,000 known company names (NTDB, Disclosure)
- 5,000 first and last names (various university on-line phone books)
- Filtered for words appearing in Longman's Dictionary - this avoids problems such as recognizing 'WILL GO' as a human name.

The system uses the case of letters used when available. The final text is tagged using SGML like markers.

The remainder of the words in a text after proper name tagging is completed are marked as one of —

```
MATSUSHITA ELECTRIC INDUSTRIAL CO. HAS
REACHED AGREEMENT WITH ROBERT BOSCH GMBH
OF WEST GERMANY ON THE ESTABLISHMENT OF A
JOIN VENTURE FOR THE PRODUCTION OF VIDEOTAPE
RECORDERS (VTRS) IN THAT COUNTRY, INFORMED
SOURCES SAID THURSDAY.

<\organ> MATSUSHITA ELECTRIC INDUSTRIAL CO.
{type('COMP')} <\endorgan> HAS REACHED AGREEMENT
WITH <\organ> ROBERT BOSCH GMBH {type('COMP')}
<\endorgan> OF <\country> WEST GERMANY
{type(['WEST GERMANY','NAME'])} <\endcountry>
ON THE ESTABLISHMENT OF A JOIN VENTURE
FOR THE PRODUCTION OF VIDEOTAPE RECORDERS (VTRS)
IN THAT COUNTRY, INFORMED SOURCES SAID <\date>
THURSDAY {date('08 DEC 82','081282')} <\enddate>.
```

Figure 2: Proper Name Recognition - An Example

- Closed class words (derived from LDOCE)
- Generative Lexical Structure verbs or nouns (derived from the GLS lexicon)
- res (other words with no specific attributes)

At this point all the tags are converted into Prolog facts:

```
sen(1,1,[
organ('MATSUSHITA ELECTRIC INDUSTRIAL CO.',
    type('COMP')),
cs('HAS',closed(has,[pastv,presv])),
gls('REACHED',type([reach,v])),
gls('AGREEMENT',type([agreement,n])),
cs('WITH',closed(with,[prep])),
organ('ROBERT BOSCH GMBH',type('COMP')),
cs('OF',closed(of,[prep])),
country('WEST GERMANY', .. ])),
cs('A',closed(a,[determiner])),
```

The Japanese system preprocesses the article to change the original encoding (Shift JIS) to EUC for a given article. The original and unsegmented text goes through a series of taggers for known names, i.e. organizations, places, GLS verbs. This process is exactly the same as in the English system. The next step is to tag organization, personal and place names which are not known to the system. These are detected by using local context, using Japanese-specific patterns, which use particles, specific words and the text tags to recognize the unknown names. In addition, date expressions are tagged and changed into the normalized form. Date expressions in the Japanese articles seem straightforward, for example, '20 nichi' (20 day) is used even if the document date is 21st and 20th can be expressed as 'yesterday', and this convention 'XX day' (where XX is a number) to express a date is consistently used in the articles. Era names such as '昭和' (Showa) or '平成' (Heisei) are Japanese specific and the year in the era, e.g. '昭和６０年' (Showa 60th year), is correctly recognized and normalized. Here is the first sentence of a typical article after the tagging process.

```
<\organ>東京海上火災保険
   {type(['COMPANY'])}<\endorgan>は
<\date>今月{date(850101)}<\enddate>から
<\organ>大和証券
    {type(['COMPANY'])}<\endorgan>と
<\gls>提携して{type(['提携する',v])}<\endgls>
保険と年金の機能を兼ね備えた新商品
{ヒューマン}を売り出した.
```

Just as for the English system this is then converted into the form of Prolog facts ready to be read into the merging phase.

7.2. Part-Of-Speech Tagging

English text is also fed through the POST part of speech tagger. This attaches the Penn Treebank parts of speech to the text. The output is converted to Prolog facts. The Japanese text is segmented with part-of-speech information by the JUMAN program, which was developed by Kyoto University. The following is the result for exactly the same sentence. The segmented units are converted to Prolog facts ready to input to the next stage.

```
juman('東京','proper_noun').
juman('海上','proper_noun').
juman('火災','normal_noun').
juman('保険','normal_noun').
juman('は','topic_particle').
juman('今月','normal_noun').
juman('から','case_particle').
juman('大和','normal_noun').
juman('証券','normal_noun').
juman('と','case_particle').
juman('提携','noun_verb').
juman('して','verb').
juman('保険','normal_noun').
```

7.3. Merging and Noun Phrase Grouping

The semantic and syntactic information are merged to give lexical items in the form of triples. The merging is done in such a way that if it is not possible to match up words (eg due to different treatments of hyphens) a syntactic tag of 'UNK' is allocated and merging continues with the next word.

Noun phrases are identified by scanning back through a sentence to identify head nouns. Both semantically and syntactically marked units qualify as nouns. The grouping stops when closed class words are encountered. A second forward pass gathers any trailing adjectives. The main use of the noun phrase in the present system is to attach related strings to company names to help with the reference resolution. In the next stages of development they will also become useful for identifying the industry type. Similar processing is carried out for Japanese.

```
gls_noun_phrase([cs('THE', ... ,['DT']),
num('THIRD',num(3),['JJ']),
country('JAPANESE', ... ,['JJ']),
res('ELECTRIC',atom(electric),['JJ']),
res('APPLIANCE',atom(appliance),['NN']),
gls('CONCERN',type([concern,n]),['NN'])])

noun_phrase([res('COLOR',atom(color),['NN']),
gls('TELEVISION',type([television,n]),['NN']),
res('SETS',atom(sets),['NNS'])])
```

7.4. Parsing

The parser has the GLS co-specification patterns built into it. It uses these and ancillary rules for the recognition of entities to fill a frame format which was given as an application specific field in the GLS entry. The frame formats provide a bridge between the sentence level parse and the final template output. Given the grouped text, the DCG parser generated from the cospecification patterns parses the text and finds entities and tie-ups in it. It uses the patterns in the cospec section of the GLS verb as well as tags attached to the text to detect entities (organizations, persons), and tie-ups. The following shows an example of detected entity and a tie-up. The sub list [_,_,_] holds location information (city, state and country) when this is detected as a side effect of parsing. Note that the third entity in the tie-up is supposed to be a joint-venture company which is not mentioned in this particular text, and that is why all the slots for the entity are unspecified.

```
entity(3,2,['業界',最大,'手','東京','海上'],_,
  [_,_,'東京'],_,_)

tie_up(1,1,
  [entity(1,1,_,['東京海上火災保険'],
      [_,_,_],_,'企業'),
   entity(1,1,_,['大和証券'],
      [_,_,_],_,'企業')],
   entity(1,1,_,_,[_,_,_],
      _,_),
   'パートナー','現在','現行',850101)
```

7.5. Reference Resolution

The task of this component is to gather all the relevant information scattered in a text together. The major task is to resolve reference or anaphora. For the current application only references between tie-up events, between entities, and between entity relations are considered.

Since entities are expressed in noun phrases, the references for entities are resolved by resolving the reference between noun phrases. Since the entity can either be referred by definite or indefinite noun phrase or by name, it is necessary to detect the reference between two definite or indefinite noun phrases, between two names, as well as between one name and one definite or indefinite noun phrase. All entities are represented as frames of the form:

```
entity(Sen#, Para#, Noun-phrase, Name,
       Location, Nationality,
       Ent-type, alias-list, np-list).
```

The reference between two entities is resolved by looking at the similarity between their names and/or their noun phrases. Since companies are often referred by their nationality or location, the Location and Nationality slot fillers in the entity frame also contribute to the reference resolution. Some special noun phrases which refer to some particular role of a tie-up (*the newly formed venture in particular*) are also recognized and resolved. For example, a phrase which refers to the child entity, such as *'the new company'* or *'the venture'*, will be recognized and merged with the child of the tie-up event in focus. A stack of entities found in the text is maintained.

Definite noun phrases can only be used for local reference. So they can only be used to refer to entities involved in the tie-up event which is in focus. On the contrary, names can be used for both local and global reference, so they can refer to any entity referred to before in the text.

When a reference relation between two entities is resolved they are merged to create one single entity which contains all the information about that particular entity.

Since a tie-up is generally mentioned by a sentence rather than a noun phrase, the reference of tie-up event is resolved by resolving the reference between its participants and some other information mentioned about the event. Other heuristics are also applied. These mostly forbid merging. For example two tie-ups cannot be merged if their dates are different. Similarly entities with different locations will not be merged. Currently, there are two types of text structures which are considered. In the first type, one tie-up-event is in focus until the next one is mentioned and after the new one is mentioned the old one will not be mentioned again. In the second type, a list of tie-up-events are mentioned shortly in one paragraph, and more details of each event are given sequentially later. Finally, when the reference between two tie-ups is resolved they will also be merged to form a single tie-up event.

The final result is a set of new frames which are linked in such a way as to reduce the requirement on the final stage of maintaining pointer to the various objects.

With the exception of the use of definite articles the reference resolution process for Japanese is identical to English. The resolved entities, entity-relation, and tie-up for a typical text are shown below.

```
final_entity(1,_,['東京海上火災保険'],
   [_,_,_],_,'企業',[])
final_entity(2,_,['大和証券'],
   [_,_,_],_,'企業',[])
final_entity(3,_,_,
   [_,_,_],_,_,[])
final_rel(1,[1,2],3,'パートナー','現在')
final_tie_up(1,[1,2],3,'パートナー','現在',
                        '現行',850101)
```

The system uses character-based rules for identifying aliases. For example, if a company name starts with '日立' (Hitachi) as in '日立 製作所' (Hitachi Manufacturing), then the system looks for the string '日立' (Hitachi) or the first two characters of the company name as its alias.

7.6. Template Formatting

The final stage generates sequence numbers and incorporates document numbers into the labels. It also eliminate objects which are completely empty. A sample of the final output from the English system is shown below

```
<TEMPLATE-2300-1> :=
   DOC NR: 2300
   DOC DATE: 091282
   DOCUMENT SOURCE: Jiji Press Ltd.
   CONTENT:<TIE_UP_RELATIONSHIP-2300-1>
<TIE_UP_RELATIONSHIP-2300-1> :=
   TIE-UP STATUS: existing
   ENTITY: <ENTITY-2300-1>
           <ENTITY-2300-2>
   JOINT VENTURE CO:<ENTITY-2300-3>
<ENTITY-2300-1> :=
   NAME: MATSUSHITA ELECTRIC INDUSTRIAL CO.
   TYPE: COMPANY
   ENTITY REL:<ENTITY_REL-2300-1>
<ENTITY-2300-2> :=
   NAME: ROBERT BOSCH GMBH
   TYPE: COMPANY
   ENTITY REL:<ENTITY_REL-2300-1>
<ENTITY-2300-3> :=
   TYPE: COMPANY
   ENTITY REL:<ENTITY_REL-2300-1>
<ENTITY_REL-2300-1> :=
   ENTITY1: <ENTITY-2300-1>
            <ENTITY-2300-2>
   ENTITY2: <ENTITY-2300-3>
   REL OF ENTITY2 TO ENTITY1: CHILD
   STATUS: CURRENT
```

8. Conclusions

We feel that Diderot, given its only recently completed state, shows potential. At the twelve month evaluation the precision of the English system was 55% and the Japanese 60%. The corresponding recall measures were 13 and 22%. The recall measure reflects to some degree a lack of coverage in the system. The system is robust and provides a good starting point for the application of more sophisticated techniques. Given appropriate data it should be possible to produce a similar system for a different domain in a matter of weeks. The development of the Japanese system has been carried out in parallel with the English system and it is encouraging to note that the overall structure of both systems is the same.

9. Acknowledgements

The system described here has been funded by DARPA under contract number MDA904-91-C-9328.

We would like to express our thanks to our colleagues at BBN who have shared their part of speech tagger (POST) with us. Thanks also to Kyoto University for allowing us to use the JUMAN segementor and part of speech tagger.

Diderot is a team effort and is a result of the work of many people. The following colleagues at CRL and Brandeis have contributed time, ideas, programming ability and enthusiasm to the development of the Diderot system; Federica Busa, Peter Dilworth, Ted Dunning, Steve Helmreich, Eric Iverson, Fang Lin, Bill Ogden, Dihong Qiu, Gees Stein, Rong Wang, Yorick Wilks.

References

1. DARPA. *Proceedings of the Third Message Understanding Conference (MUC-3)*, San Mateo, CA, 1991. Morgan Kaufmann.
2. DARPA. *Proceedings of the Fourth Message Understanding Conference (MUC-4)*, San Mateo, CA, 1992. Morgan Kaufmann.
3. Ralph Grishman and John Sterling. Acquisition of selectional patterns. In *Proceedings of the 14th International Conference on Computational Linguistics (COLING92)*, Nantes, France, 1992.
4. Louise Guthrie, Rebecca Bruce, Gees C Stein, and Fuliang Weng. Development of an application independent lexicon: Lexbase. Technical Report MCCS-92-247, Computing Research Laboratory, New Mexico State University, 1992.
5. Louise Guthrie and Elbert Walker. Some comments on classification by machine. Technical Report MCCS-92-935, Computing Research Laboratory, New Mexico State University, 1992.
6. Wendy Lehnert and Beth Sundheim. An evaluation of text analysis technologies. *AI Magazine*, 12(3):81–94, 1991.
7. P Proctor, editor. *Longman Dictionary of Contemporary English*. Longman, Harlow, 1978.
8. James Pustejovsky. The generative lexicon. *Computational Linguistics*, 17(4), 1991.
9. James Pustejovsky. The acquisition of lexical semantic knowledge from large corpora. In *Proceedings of the DARPA Spoken and Written Language Workshop*. Morgan Kaufmann, 1992.
10. James Pustejovsky, Sabine Bergler, and Peter Anick. Lexical semantic techniques for corpus analysis. *Computational Linguistics*, 1993.
11. James Pustejovsky and Bran Boguraev. Lexical knowledge representation and natural language processing. *Artificial Intelligence*, 1993.
12. Y. Schabes and S. Shieber. An alternative conception of tree-adjoining derivation. In *Proceedings of 30th Annual Meeting of the Association for Computational Linguistics*, 1992.