Recognizing and Interpreting Tables

Mabry Tyson, Douglas Appelt, Jerry R. Hobbs,
John Bear, David Israel, and Megumi Kameyama
Artificial Intelligence Center
SRI International
333 Ravenswood Ave.
Menlo Park CA 94025, USA
`tyson@ai.sri.com`

Abstract

Tables are a pervasive problem in the real-world corpora that are the focus of information extraction applications. Moreover, they constitute a problem of significant linguistic interest. In this paper we present a general method for recognizing and interpreting tables in text, and describe its implementation in a particular application.

1 The Problem

In recent years there has been an explosion in research on text understanding and on extracting information from real-world texts such as newspaper articles, as exemplified by the MUC evaluations (Sundheim 1992, Sundheim 1993). Tables are very common in these corpora, and very often the tables are rich in the information of interest. For example, a typical recent edition of the Wall Street Journal had twenty tables on a wide variety of subjects. Moreover, unless the tables are tagged with SGML, it is a problem that cannot be avoided. If a system attempts to process a table as though it were ordinary text, serious mistakes would be made. The row and column structure of the table will not be respected, and spurious constituents spanning internal boundaries inside the table will be recognized. Ignoring tables is simply not an option.

Yet there has been relatively little work on recognizing and interpreting tables. In part, this has been because priorities were elsewhere, in part because the problem was not deemed of sufficient *linguistic* interest. In part, it may be because the problem was viewed as too hard.

In fact, the problem of recognizing and interpreting tables is of significant linguistic interest. It presents in a pure form some of the most important problems in local pragmatics and discourse structure. When we encounter a table such as Table 1,

Nancy Kerrigan	Disneyland Pizza Hut
Michael Jordan	Nike
Charles Barkley	Reebok

we have to determine the relations among the elements in a single row in a way that is consistent across all the rows, and we must fill in missing elements. Determining the relations among the elements in a single row is an instance of a problem that is pervasive in local pragmatics—finding the relation among adjacent elements in the text. Discovering the implicit relation between the two nouns in a compound nominal is a common example of this problem. Recognizing the consistency of the the relation across rows in the table is a particularly pure form of the problem of recognizing parallelism among adjacent segments of discourse. Suppose this table were preceded by the "pretabular" sentence,

> Major athletes have endorsement contracts for over $1M as follows:

We would have to determine in what way each row of the table instantiates this summary statement. This is, in a pure form, the problem of determining how successive segments of text all instantiate or exemplify one general or summary statement.

Discovering this relation can be critical in information extraction tasks. The pretabular sentence may characterize a set of events of interest while the table lists those entities that participated in such an event. Failing to recognize the relation between the pretabular sentence and the items in the table would mean failing to recognize one event for each row of the table.

In brief, the task of recognizing and interpreting tables is of immense importance in information extraction applications, and is a problem of substantial linguistic interest in its own right. It is an excellent locus for research on discourse structure, because the fundamental issues arise there with particular purity.

The methods described here were developed in connection with a project for extracting information from unformatted military messages, and have been implemented in the Message Handler System, based on

FASTUS (Hobbs et al., 1997). But the problems that were encountered and the techniques developed generalize to other varieties of text. We are currently testing some of the techniques on business sections of American newspapers.

We discuss first the method for recognizing when there *is* a table embedded in the text. This can be thought of as specifying the *syntax* of tables. We then discuss how we interpret tables. This can be thought of as specifying the *semantics* of tables.

2 Recognizing Tables

A table is a two-dimensional array. It consists of two or more *records*, frequently corresponding to the rows, where each record consists of two or more *fields*, frequently corresponding to the vertical columns. The information in a field will be referred to as an *item*. Records may span multiple lines of text. In the current implementation, fields are pieces of text vertically aligned over multiple records. Our corpus consists of military texts in ASCII text (as opposed to typeset text) so vertical alignment is recognized by the columns (character offsets from the beginning of each line) of the characters in the fields.

Consider the following table, with its pretabular sentence, headings and subheadings:

```
FIELD EXERCISES WERE CONDUCTED BY THE FOLLOWING UNITS:

     UNIT             HOME BASE           LOCATION

21 MAY 94:
   1ST MECH INF BN   FT SAM HOUSTON         LAFAYETTE
   2ND MECH INF BN     FT LEWIS      BATON ROUGE

22 MAY 94:
   3RD MECH INF BN     MONTEREY          LAFAYETTE
```

Table 2

The first problem we face is recognizing this as a table in the stream of text, rather than as a long sentence with lots of spaces between words. This can be more difficult than it first might seem, because lots of spaces *do* sometimes occur between words and often in text spaces line up in successive lines accidentally.

```
This text provides an example which could be misinterpreted.  There
are three fields.  The second field ends with this sentence.  One
field stops here.  The problem occurred often in our corpus.
```

We also must recognize that outline formats are not tables.

Moreover, the edges of the fields can be quite ragged, sometimes by accident, sometimes by design, as when a list of decimal numbers is justified on the decimal point. In our corpus, tables were sometimes hastily typed and the alignment is erratic.

Our algorithm first analyzes each line of the text. Lines are classified in numerous ways: blank, short, centered, colon- or dash-terminated, separator (e.g., a repeated string of the same characters), or possible outline subdivision line (eg. "`A. Introduction`"). The multiple space gaps are recorded in a data structure.

We then scan the text (ignoring blank lines and always breaking at text section boundaries) looking for sets of consecutive lines of potential tables. We look for at least two successive lines for which the following conditions hold:

1. If two or more spaces occur between words in a line, that qualifies as a potential field boundary (a "gap"). If the lines already identified as potentially part of a table have a gap, then there must be at least a single space that is within or abuts the gap. The gap from the previous lines is intersected with the gap from this line.

2. In the set of lines, there must be no vertical overlap between fields. That is, different fields in different records may not have any column in common.

3. In the set of lines, if a line has any characters in a field, then those columns must intersect the columns occupied in other lines. That is, the same occupied field in different records must all have some vertical intersection. This can lead the system to recognize two fields where it originally recognized just one. In the table of presidents, only one field (the entire line) would be recognized from the first line but the subsequent lines allows the system to recognize that the first line could have been parsed as two fields.

4. There are at least two fields.

Rules 2 and 3 are relaxed for the first line of tables. It is often the case that headings may be longer than the data under them, which may result in a heading overlapping another field. Furthermore, headings may be centered while the data is left justified which can result in a short heading and short data failing to overlap. A heading line must have the same number of fields as the table below it and each field of the heading must overlap the corresponding field of the table. The following table of vehicles demonstrates problems with recognizing headings as parts of tables.

```
    UNIT      VEHICLES      TYPES

1ST MECH INF BN  10  TANKS
```

```
2ND MECH INF BN  24  TANKS, APCS, TRUCKS
3RD MECH INF BN   3  TRUCKS
```

Table 3

Headings sometimes are present, and sometimes are not. During *interpretation*, the first record may be recognized as a heading and treated differently than the data of the other records.

The limitations of this approach are obvious. We cannot recognize a vertical list of items as a table, since there is only one field. Tables with fields that are so ragged that they overlap will not be recognized. However, including these cases would have significantly degraded the precision of the algorithm, since there would have been a large number of false positives.

In addition, the algorithm will not recognize separate fields where there is only one space between them. This can happen especially when the items in one field have a fixed length. The following is an example in which we recognize two fields rather than three:

```
William Henry Harrison 1841 1841
 Died of pneumonia in office
John Tyler             1841 1845
James Knox Polk        1845 1849
```

Table 4

However, we can often recover from such cases because we allow multiple items in a field during interpretation.

Once potential tables and their fields have been identified, there are several more problems. Rows of a table can be interrupted with both subheadings and remarks, as in the tables of exercises and of presidents.

We allow for these by splicing together potential tables with compatible fields that are separated by a single line that doesn't fit the fields of either table. This allows tables to span subheadings, single-line remarks, and single rows that are misaligned enough not to match the fields of the other rows.

The next problem is to distinguish subheadings from remarks. An interruption in a table is classified as a subheading if one of the following conditions holds:

1. The line ends in a colon, dash, or multiple dashes.
2. The line begins to the left of the first column of the table.
3. The line is centered.

Otherwise the interruption is classified as a remark. As will be seen in Section 3, the interpretation of subheadings constitutes an important part of the interpretation of tables. At present, remarks are ignored, although more properly they should be interpreted as normal sentences, but in the context provided by the previous record in the table.

Tables often have multiple-line fields within a record. Sometimes these multiple lines represent lists of items and sometimes they constitute a single item whose representation was too long to fit in the field. To confound this situation, tables also often have empty fields within a record. Recognizing when a line represents its own record rather than a continuation line of a previous record is done by noting how many fields exist in a line.

UNIT	VEHS	EQUIPMENT	COMMENT
1ST MECH INF BN	10	TANKS	
2ND MECH INF BN	24	TANKS	
		APCS	
		TRUCKS	
3RD MECH INF BN	3	TRUCKS	
4TH MECH INF BN	0		EQUIPMENT WAS ALREADY
			COMMITED ELSEWHERE

Table 5

When our system finds a line with only a single field in the middle of a table with multiple fields, that field is considered to be a continuation of the field from the previous line. Thus the EQUIPMENT field of the record for the 2ND MECH INF BN will be a sequence consisting of “TANKS”, “APCS”, and “TRUCKS” while the COMMENT field of the record for the 4TH MECH INF BN will be a sequence consisting of “EQUIPMENT WAS ALREADY” and “COMMITED ELSEWHERE”. The interpretation process will need to determine whether the sequences are separate items or a single item.

If there are multiple fields in a single line, then that line is considered to be its own record rather than a continuation line. While this worked well for our corpus, a more general solution would require interleaving recognition and interpretation.

Missing fields in a table are sometimes intended to indicate that the information is unknown or irrelevant as in the multiline table. Other times fields (especially those on the left) are omitted as a kind of ellipsis or as a form of subtitle. We currently treat missing fields as missing data.

Not every sentence that immediately precedes a table is a *pretabular* sentence in the sense that it describes the contents of the table. During Table Recognition we attempt to identify those immediately preceding sentences that *are* pretabular sentences. An immediately preceding sentence is classified as pretabular if no paragraph break occurs between it and the table and one of the following conditions holds:

1. It ends in a colon or a dash.
2. It contains the words “the following” or ends with the words “as follows”.

In the tables that were recognized, there were 11 with subheadings. The program recognized 8 of these as subheadings, with no false positives, for a recall of 73% and a precision of 100%. There were 9 tables with remarks or overflow lines.

3 Interpreting Tables

We describe the process of interpreting tables at a general level that should be applicable to a wide variety of tables in a wide variety of text types. The implementation of the process was optimized specifically for military messages, and our examples come primarily from military messages. In Section 4 we discuss some problems that have to be faced as the implementation is extended to other domains.

The process of interpreting tables consists of five steps:

1. Subheads are distributed through subsumed items.
2. The types of items are recognized, subject to consistency across that field in all the records.
3. Limited use is made of headers to further disambiguate the types of the items.
4. The most plausible relation among the items in the records is hypothesized, subject to consistency throughout the table.
5. The pretabular sentence is interpreted, allowing for parameters for the table items.

Each of these steps is discussed in turn.

Subheads. The subheads that were identified in the recognition phase are distributed through the records that they subsume, becoming another field in those records. For example, Table 1 is transformed into the following:

```
FIELD EXERCISES WERE CONDUCTED BY THE FOLLOWING UNITS:

                       UNIT             HOME BASE           LOCATION

        21 MAY 94   1ST MECH INF BN   FT SAM HOUSTON          LAFAYETTE
```

21 MAY 94	2ND MECH INF BN	FT LEWIS	BATON ROUGE
22 MAY 94	3RD MECH INF BN	MONTEREY	LAFAYETTE

Table 22

Recognizing Entity Type. The rules in the system that are used to recognize types of entities are applied to the individual items in the table. Any constraints on where the entity names can be located in a sentence are lifted. Where the item has an ambiguous type, multiple readings are maintained at this stage. In our application, the entity types includes dates, units, facilities and locations.

The system then seeks to resolve the ambiguities by maximizing the consistency of all the entities in the field across the records in the table. That is, for each field it is determined which type is consistent with the maximum number of items. For example, in the above table, it may be that MONTEREY is listed in the lexicon as both a facility and a location. Since the other two items are both facilities, the facility reading would be chosen for MONTEREY.

When it is impossible to interpret the items in a field in a consistent fashion, certain coercions are permitted. One of these coercions for the military message application is a coercion from locations to facilities. Suppose, for example, that MONTEREY were listed in the lexicon only as a location. Then since the other two items in that field are facilities and there is a legal coercion from locations to facilities, that coercion would be applied.

Although we do not recognize columns separated by only one space, we overcome this shortcoming by allowing more than one entity in each field of the table. For example, in Table 3, we would allow the field of the second column to contain the items "Year Year". These would then be treated as if they were two separate items.

Using Headers for Disambiguation. Limited use is made of headers for disambiguating the type of items in a field. The top line of the table has the same structure as every other line, but it is viewed as a possible header line, labelling each of the fields. If the items in that line are not recognizable entity names but are recognizable names of entity types, then the line is assumed to be a header. In some cases the name of the entity type provides the information for determining the entity type in that field. For example, in the fourth field of Table 3, LAFAYETTE could be either a ship or a location. The fact that the header says LOCATION leads us to conclude that the entities are locations, not ships. Similarly, UNIT leads us to conclude that the items in the second field are units, and HOME BASE that the items in the third field are facilities.

The headers are used only for disambiguation, rather than as the principal evidence for the type of the items in the field. The reasons for this are that frequently there are no headers, and that often when there are they do not provide an unambiguous or even interpretable identification of the entity type.

Hypothesizing the Most Plausible Relation. The central problem in interpreting tables is in discovering the implicit relation that obtains among the items in each record of the table, in a way that is consistent across the records of the table. By the time this stage of processing is reached, the items have been disambiguated as to type, in a way that is consistent across the records, so there is no problem in determining a relation that is consistent across the table. We only need to find the most likely relation among the set of entity types that occurs in the records.

This process necessarily requires a domain model. The model we use is a fairly general one—a labelled graph in which the nodes correspond to entity types and the arcs correspond to possible relations between them. The system places the entity types found in the records of the table in this graph and then seeks to minimize the spanning tree covering these nodes. The minimal spanning tree corresponds to the hypothesized relation among the entities. (This is called the Domain Information for the records of the table.) Where a connection cannot be formed, the item in the table is ignored. This seems appropriate; very often some information in the table falls outside the domain model.

In the military message application, the system must recognize the names of units, locations, facilities, equipment, and times, among other things. It must recognize relations among these entities, and it must recognize events in which these entities participate.

There is a limited set of possible relations among entities, such as

> Unit is at Facility.
> Unit is at Location.
> Unit moves from Location to Location.
> Facility is at Location.
> Equipment is at Facility or Location.
> Unit has Equipment.
> An “at” relation or a “move” event is at a Time.

In Table 3, the minimal spanning tree linking up times, units, and locations is one corresponding to the unit being at the location at a time. Since it is known that the facilities listed are not at the locations listed, no connection can be found between the facilities and the other items. This field is discussed further in the next section.

This stage of the processing can be viewed as a limited, tractable form of abduction. The items in the table constitute the data to be explained. The graph encodes the possible explanations. Choosing the minimal spanning tree is a way of choosing a minimal, or a best, explanation.

Pretabular Sentences: The pretabular sentences are parsed and analyzed just as every other sentence in the text is. In ordinary text the system looks for particular patterns that represent events and relationships and builds up the corresponding Domain Information. The one difference is that where in ordinary sentences we are looking for patterns involving names of entity *tokens*, in the pretabular sentences we are also allowing patterns with entity *types*. For example, the pretabular sentence of Table 3 is just like the ordinary sentence

FIELD EXERCISES WERE CONDUCTED BY THE 1ST MECH INF BN.

For such an ordinary sentence the system builds a Domain Information encoding the type of event and the participants in the event. For a pretabular sentence, the same structure is built, but the entity types are only parameters, and the structure is instantiated once for each record of the table, with the parameterized entity replaced with the corresponding entity of the same type in that record. That is, for each record in the table, the Domain Information for that record is unified with the Domain Information for the pretabular sentence.

Our processing of pretabular sentences is done almost as an afterthought, after the relation among the elements in the table has already been hypothesized. One might think that instead the system should use the pretabular sentence to drive the interpretation of the table. There are three reasons this would not be a good idea. First, very often there is no pretabular sentence describing its structure. Second, most of the time the table contains more information than is described in the pretabular sentence. For example, in Table 3 the records contain fields for times, units, facilities, and locations, but only units are mentioned in the pretabular sentence. Third, the pretabular sentence frequently refers to entity types that are not found in the table.

4 Planned Improvements

There are several limitations to the present system that we intend to improve upon in the near future.

In the recognition of tables, we plan to extend the treatment to cover separators between columns other than spaces. Many military messages, for example, use slashes or other characters, and make no attempt to align the items in a column. In newspaper articles the space between items in one column and the next is filled with a row of dots.

In general, as we extend our table interpretation methods from military messages to business news, we expect to encounter and accommodate a variety of different structures of tables. In addition, we plan to extend the treatment to cover vertical lists of items (one-column tables).

One of the principal shortcomings of the interpretation procedures is the limited use we make of the information in headers. In Table 3 we are able to recognize that the items in the third field are facilities because of the header HOME BASE, and the items in the fourth field are locations because of the header LOCATION. But we do not now treat the headings as names of relationships, which in this table they are. It is necessary to be open to the possibility of the heading being the name of a relationship, and then to be able to identify the "principal" column of the table so we know what entity the relationship is with. In Table 3, for example, it is necessary to know that the UNIT field is the principal one, so that the LOCATION relation is between units and the locations, rather than, say, between the home bases and the locations.

Another use of headers is to provide a third entity, frequently a time, in the relationship, as in the following table:

Hourly Compensation
For Production Workers
(As a percentage of U.S. costs)

			Jan. 31
	1985	1986	1985
Germany	75	103	120
Japan	50	73	79
South Korea	11	12	12

In this table, the problem is first to recognize from the title (a kind of pretabular sentence) that the elements in the matrix are percentages, and then to discover the relation among the country, the year, and the percentage, as they fit into the description of a relation provided by the header.

A common kind of table is one in which one record lists the name of a property or relation and the second record gives the value, as in

Height	5'10''
Weight	175 lbs
Eyes	brown

We do not at present handle this case.

Another kind of problem involves the use of free text in fields of a table, as in the following example from the Wall Street Journal:

```
      A Chronology of the Stock-Trading Scandal
   May 12, 1986 -- SEC charges Dennis Levine of Drexel
Burnham Lambert Inc. with making $12.6 million since mid-1980
from insider trading. SEC also names as defendant Bernhard
Meier, Mr. Levine's broker at Bank Leu International in
Nassau, Bahamas.
   May 13, 1986 -- Mr. Levine is arrested and charged with
obstructing justice for attempting to destroy records. He is
released on a $5 million bond.
```

Here it is necessary to be able to process each free text item within the context provided by the date field, and to recognize the relations inherent in the structure of the table.

In conclusion, we have articulated a conception of the processing of tables, a pervasive and important phenomenon in real-world text, as an instance of a more general problem in local pragmatics and discourse structure. We have translated this conception into an implementation in a tractable manner in a specific domain. We have used this implementation with significant success in a practical application.

References

[1] Hobbs, Jerry R., Douglas E. Appelt, John Bear, David Israel, Megumi Kameyama, Mark Stickel, and Mabry Tyson, 1997. "FASTUS: A Cascaded Finite-State Transducer for Extracting Information from Natural-Language Text", in E. Roche and Y. Schabes, eds., *Finite State Devices for Natural Language Processing*, MIT Press, Cambridge, Massachusetts, pp. 383-406.

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

[3] Beth Sundheim (editor). 1993. *Proceedings, Fifth Message Understanding Conference (MUC-5)*, Baltimore, Maryland, June 1992. Distributed by Morgan Kaufmann Publishers, Inc., San Mateo, California.