Layout and Language: lists and tables in technical documents

Shona Douglas and **Matthew Hurst**
Language Technology Group
Human Communication Research Centre
University of Edinburgh
Email: S.Douglas,Matthew.Hurst@edinburgh.ac.uk
Telephone: +44 131 650 4439

Abstract

In this paper, we describe some of the interactions between layout and language we have been dealing with in recent applied NLP projects. We present two complementary views of lists and tables, intended to bridge the gap between considering them as a type of running text (which linguistics knows how to deal with) and as a multi-dimensional relation represented in two dimensions, which may have many reading-paths (which linguistics doesn't know how to deal with). Stated or inferred linguistic and world knowledge in the text surrounding tables and lists provides a context for the interpretation of a set of tuples extracted from tables or lists together with heuristics about how multi-dimensional information is projected on to two dimensions.

1 Introduction

It is often assumed for purposes of computational linguistic description and practical natural language processing that layout and punctuation can be ignored: they are removed in pre-processing, and not referred to in linguistic processes. Equally, in automatic tagging of text structure (increasing with the advent of markup such as SGML), linguistic factors are not considered. As computational linguistics faces up to the challenge of 'real texts', and as 'logical markup' aims to capture more and more of the information content of texts, however, problems arise with the neat partition between layout and language. In this paper, we describe some of the interactions between layout and language we have been faced with in recent applied NLP projects.[1] These projects are:

- a controlled language checker for Perkins Approved Clear English (PACE)[2] — the application domain is engine workshop manuals (EWM);
- information extraction in construction industry specification documents (CISAU)[3]

The phenomena in question appear extremely common in many types of technical documents.

2 Layout effects in technical text

The main layout categories we will be interested in here are **lists** and **tables**. These categories are typically indicated graphically by layout effects such as **relative vertical spacing** (for table rows) and **relative horizontal spacing** or tabulation (for various levels of lists and for table columns). (In addition, lists may have labels based on some numbering series, or 'anonymous' labels such as bullets.) Such indicators can be thought of as 'primitive' markup — it is inseparable from any readable rendering of the text; it is surface-based, and the relationship with actual text categories[4] is not always straightforward.

[1]We are grateful to Perkins Engines and BICC plc for the use of documents in these projects, and to David Quinn of BICC plc for discussions on the subject of tables processing.

[2]See (Douglas, 1996) for an overview.

[3]CISAU: Construction Industry Specification Analysis and Understanding, in collaboration with BICC plc, with partial funding from SERC/DTI project IED4/1/5818.

[4]The term is Nunberg's: (Nunberg, 1990) p6 and *passim*

These are layout categories that one can increasingly expect to find marked up in technical documents in SGML (Goldfarb, 1990), whether as a result of authoring in an SGML environment or of post-processing using a programmable pattern-based system such as OmniMark (OmniMark, 1993), customized to recognise patterns in the primitive markup that indicate particular text categories. In what follows, we will generally assume that such markup is present in the text, and that getting it there is a straightforward matter; for the most part in the paper we are dealing with what a consideration of layout can do for language processing.[5]

3 Interpreting lists and tables in context

In this section, we present a view of the general problem of list and table processing, focussing on the information extraction task in the CISAU domain.

3.1 A relational view of lists and tables: some terms and examples

Some lists behave just like in-line text: they can be processed from left to right with no difficulty, and contain only syntactically well-formed self-contained sentences. In many of these cases, the list format has been used to mark off a set of sentences with a particular textual function, such as a sequence of instructions. This is a typical phenomenon of text grammar in the sense used by Nunberg ((Nunberg, 1990)): it affects only the argument structure of the text, and the list items can (for the most part) be processed for propositional content as if they were in-line text, by throwing away any list-item indicator glyphs such as bullets or numbers, and ignoring indentation. Such lists do not interest us further in this paper. What we are dealing with here is cases where list or tabular structures must be interpreted to obtain the full propositional content of the text. The following examples will form the basis for the subsequent discussion.

One way of looking at lists and tables is as **parallel structures** with typically a lot of ellipsis or gapping, with some layout indicators to ease segmentation. Examples 1 and 2 appear next to each other in one of the texts from the CISAU domain:

(1)
```
SULPHATE CONTENT OF MIXES: The total sulphate content of the
constituents of each mix must not exceed 4% by weight of
the cement in the mix.
```

(2)
```
CHLORIDE CONTENT OF MIXES: the total chloride ion content of the
constituents of each mix, expressed as a percentage by weight of
cement (including GGBS or PFA if used) in the mix, must not exceed the
following:
Prestressed concrete:                                          0.1
Concrete made with sulphate resisting  Portland cement or supersulphated
cement:                                                        0.2
Concrete made with Portland cement, Portland blastfurnace
cement or combinations of GGBS or PFA with ordinary Portland
cement and containing embedded metal:                          0.4
```

The list in example 2 could be thought of as equivalent to a parallel set of sentences, each with syntactic/semantic structure similar to that of example 1. There are some interesting and typical differences between the individual sentence form and the form that actually appears with the list:

- There is frequent use of a pragmatic anaphor such as *the following* referring to part or all of the subsequent list structure;
- Values are stated elliptically, without their units (*percentage by weight...*), which are abstracted out into the lead-in text. This requires lexicalisation, such as *expressed as*, of a relationship presented in 1 simply by adjacency.

Examples 3 and 4, constructed to express the same information as example 2 in tabular form, do not actually appear in a real document, but examples exactly like both in form are common.

[5] In (Douglas, 1995), we were concerned with what language processing might be able to do for logical document structure recognition.

(3) CHLORIDE CONTENT OF MIXES: the total chloride ion content of the constituents of each mix, expressed as a percentage by weight of cement (including GGBS or PFA if used) in the mix, must not exceed the following:

Mix	%
Prestressed concrete	0.1
Concrete made with sulphate resisting Portland cement or supersulphated cement	0.2
Concrete made with Portland cement, Portland blastfurnace cement or combinations of GGBS or PFA with ordinary Portland cement and containing embedded metal	0.4

(4) CHLORIDE CONTENT OF MIXES:

Mix	Maximum total chloride ion content (% by weight of cement, including GGBS or PFA if used)
Prestressed concrete	0.1
Concrete made with sulphate resisting Portland cement or supersulphated cement	0.2
Concrete made with Portland cement, Portland blastfurnace cement or combinations of GGBS or PFA with ordinary Portland cement and containing embedded metal	0.4

Inspection of these examples suggests that another way of thinking about lists, and particularly about tables, is as **relations**. A table can be thought of as a set of **tuples** expressing a multi-dimensional **relation** among a number of **domains** (sets) of **values**[6], mapped on to a two-dimensional representation of rows and columns. (This contrasts with the one-dimensional stream of text suitable for linguistic processing.) Lists can be thought of under this interpretation as rather simple tables.[7]

The list in example 2 relates values from two domains, one populated by *mixes*, the other by *percentages*; example 3 is exactly like it, except for the introduction of lines to mark off table cells, and a **label** for each domain as it is expressed as a table column. In both these examples, the **lead-in text** is important in specifying the **semantic content** of the relation; to the extensional definition comprising the tuples themselves is added the statement of **relationship** lexicalised as *must not exceed.*

Comparing examples 3 and 4, we can see that there is a range of options for distributing the semantic content of a relation between lead-in text and table domain labels. Note how the specific relation between the domains, expressed as a verb in the preceding examples (*must not exceed*), is nominalised to fit the conventional syntactic form for table labels (*maximum*).

3.2 Information from lists and tables

In the CISAU application domain, the interpretation task to be performed on these lists and tables is simple information extraction. We take advantage of the predominant argument type of this genre of specification documents, which we conceptualize as a form of 'assignment', similar to that in programming languages. Our aim is to fit each assignment in the text into a **template** that contains various elements represented in terms

[6]We can think of this as corresponding to relations in relational database terms. See, for example, (Ullman, 1988).

[7]Note that the potential for lists to have recursive sublist structure is an exception to the general similarity with tables, which will not be dealt with further here.

of the sublanguage world model, a simple part-of/type-of knowledge representation (KR). The elements we are looking for are

- an **entity** (in the case of our examples, various instances of CONCRETE; in this application, these are really entity types, mass or generic nouns);
- an **attribute** which the KR accepts as appropriate for that entity (in the examples, CHLORIDE CONTENT);
- a **unit** or type for the attribute (in the examples, %, with supplementary specifying information *by weight...*);
- a **value** which the assignment gives to the attribute (in the examples, the actual percentage numbers);
- a **relationship** expressing the semantic content of the assignment (in the examples, "<="; in other cases simple equality is the relation, or ranges or sets of possible values may be given; in less numeric examples, there may be more general sorts of relationship, corresponding to general verbal semantics).

In the next section, we look more closely at the characteristics of table cells, labels, and lead-in text; in section 5, we present a heuristic approach to disentangling table and list domains, based on those characteristics.

4 Lead-in text, labels, and value cells

We have seen that lead-in text and table labels often contain further specification of the semantic content of the relation, which may be essential to fill our templates. Accordingly, although it is not distinguished by layout from the rest of the document, and is ignored by most markup conventions, we consider lead-in text to be an integral part of the list or table structure.[8] In this section we will look more closely at its structure and function.

The information we are interested in isolating from the lead-in text and labels is

- the matching of references in the lead-in text with domains (labelled or unlabelled) in the table or list, thus specifying the domain structure of the relation;
- the further specification of the relationship between the table domains, using any relationship given in the lead-in text between the various domain references.

The set of values occurring in a table domain is in general a subset of those in one of the world model domains — for instance, the mixes in examples 2–4 are a subset of possible concrete mixes in the CISAU world model. In interpreting a table, we can say that the relation it embodies defines a relation in the world model; for CISAU, that is expressed as a set of templates as described above.

The type-of model of the domain is a hierarchy of concepts which we can use to distinguish between table cells that are domain labels (and equivalent descriptions in lead-in text) and cells that contain domain values. In principle, a domain consists of a set of values with a common ancestor or supertype, to which all the values bear the same relation in the hierarchy. Such a supertype is the interpretation of the label of a domain, or that part of lead-in text that fills the same function. If the node `MATERIAL` in the world model has below it in the hierarchy `WOOD`, `CONCRETE`, etc., then `Material` or its cognates will be likely to be a label for the subordinate classes.

Lead-in text and the list items or table rows that follows it behave a good deal like Nunberg's colon expansion[9]: the lead-in text is usually a lexical sentence, and the list usually acts as a text adjunct, marked off with a colon. The lead-in text often contains a phrase that acts as a **deep textual anaphor**: deep in the sense of (Hankamer, 1976) that it requires pragmatics to resolve, and textual, in the sense that the referent is not an external entity, but a portion of the text in which the anaphor occurs, in this case the following list, or some part of it. Typical forms are

[8] As a result, we would claim that proper logical markup of list and table structures in something like SGML should include the lead-in text — identifying this requires NLP.

[9] Discussed in (Nunberg, 1990), p25ff.

following, listed below, given below, ...

In our texts, we have seen these phrases used in two different ways. The first is in modifying a noun phrase which is a description of one of the domains in the table or list. For example, *the following tools*, or *the tools listed below*, precedes a table relating tool names to part numbers. In such cases, the co-referents appear to be the values of the 'target' domain in the list or table (where there is more than one domain, the target domain can be thought of as the thing being looked up using the other domain values as keys).

In the second use, the phrase refers as a unit to the list items following the lead-in. For example *in the following*, or *in the following list*, or even *in the following procedures* are found (more often in lists with purely text grammatical functions than the pseudo-lexical ones considered here, perhaps).

Another form of anaphor used is

as follows

which occurs as a modifier to a verb in the lead-in, for example *The different parts in situ concrete construction are specified in separate sections as follows:*, preceding a list relating topics to section numbers. It can perhaps be thought of as referring to the actual semantic content of the relation, or, as in the second use above, to the list as a whole.

Nunberg remarks about colon expansions:

> The only constraints on the content of the colon expansion appear to be semantic: can the lexical content be plausibly interpreted as an elaboration or expansion of some element in the preceding lexical clause?
>
> (Nunberg, 1990)p30

The ordinary prose colon expansion he is concerned with is a pure text category construct, in that it expresses only argument structure. The lists we are interested in here, however, are related to the lexical grammar (in his sense) in that the layout structure must be used to reconstruct the actual propositional content of the text.

We would like to use the lead-in text to construct a specification of a parallel set of sentences, and thus to derive a fully specified set of relational propositions. But the 'antecedents' of table items in lead-in text cannot be readily resolved using syntactic means. There is, for instance, no discernible pattern relating grammatical roles in the lead-in sentence — or topicalisation, or which domain is marked out as 'target' with phrases like *the following* — with the reading order in which domains appear in two-domain lists or tables. We have noted some consistency *within* a document, as a matter of style, but this does not seem to generalise. Accordingly, we use a heuristic approach, based on the kind of KR relations observed among the elements of table and list structure.

5 Processing tables and lists with heuristics

Tables can become extremely complex in layout when the underlying relation has arity greater than two, and an autonomous grammar of how relations can be mapped on to two dimensions is needed to allow the disentangling of the interleaved cells. In (Douglas, 1995), we presented a relatively detailed account of how such a grammar of possible layouts combines with the function of tables in determining a reading order, or decision structure, for the information they contain.[10] These two factors determine how domain values and labels are laid out as cells in a 2-D table. In relation to lists and very simple tables with only two domains, the complexity of this variation is severely curtailed, and so we present here a simplified form of the layout part of the heuristics, while extending the treatment to consider lead-in text which was not dealt with in (Douglas, 1995).

First, the lead-in text is parsed and interpreted to identify a set of candidate **descriptions**, which are nominal groups that may describe domains, and any relationships between them. The cells of the table

[10]See (Wright, 1982), on the idea of decision structure; Wright's view of texts as information artefacts can be related perhaps to Nunberg's idea of how text categories are used to create text that can simultaneously accomodate different (notional) readers.

(domain values or labels—they cannot be distinguished *a priori*) are also parsed and interpreted to give a semantic type for each cell. Then a phase of **heuristic analysis** identifies the elements of table content:

- The domains are identified.

  First the columns of values are grouped. The **semantic type** of the domain is determined by finding a superclass for the semantic classes of all the cells in the column.

  If the top-most element in the column fails to share this superclass, it may be nominated as a label for this domain, if it is has the same semantic type as the superclass of the rest of the cells.

- The set of descriptions (from the lead-in text) is matched against the identified domains.

  A description matches a domain if the semantic type of the description is the same as the semantic type of the domain.

- The match between the descriptions and the domains is used to identify any semantic relationships specified in the lead-in text between the descriptions; this relationship is now assigned to the relation.

A template for the CISAU application can be constructed for each tuple in the relation, using the matched descriptions and the semantic relationship between them, if any.

6 Conclusions and further work

In this paper, we have argued that the task of reconstructing the propositional content of technical texts with complex layout features requires an autonomous view of those structures which includes any relevant associated text such as lead-ins and captions. Our proposed reconstruction makes use of heuristics relating layout structure with linguistic and world knowledge.

If we take Nunberg's distinction between text grammar and lexical grammar, then list structures are like commas, in that they can appear as indicators of both text categories (e.g., lists of sentences describing a sequence of procedures) and as segmenting guides for constructing the propositional content of the text — the job of the lexical grammar. While Nunberg seems to suggest that lexical grammar is what is already dealt with by (computational) linguistics, we would say from our work so far that a consideration of the autonomous syntax of lists and tables is necessary to constrain the processing of text, in just the same way as the constraint that text must be processed from left to right.

It is clear that we have only touched the surface of these issues, and we are convinced that this currently greatly under-researched field will repay a great deal more study. In the immediate future, we will be collaborating with BICC plc on further implementation and investigation of the approach outlined here.

References

Shona Douglas, Matthew Hurst and David Quinn. Using natural language processing for identifying and interpreting tables in plain text. In *Fourth Annual Symposium on Document Analysis and Information Retrieval*, 1995.

Shona Douglas and Matthew Hurst. Controlled language support for perkins approved clear english (pace). In *Proceedings of the First International Workshop on Controlled Language Applications*, 1996.

Exoterica. *OmniMark Programmer's Guide*. Exoterica Corporation, Ottawa, Canada, 1993.

Charles F. Goldfarb. *The SGML Handbook*. Clarendon Press, Oxford, 1990.

Jorge Hankamer and Ivan Sag. Deep and surface anaphora. *Linguistic Inquiry*, 7(3):391–428, 1976.

Geoffrey Nunberg. *The Linguistics of Punctuation*. Center for the Study of Language and Information, Stanford, 1990.

Jeffrey D. Ullman. *Principles of Database and Knowledge-Base Systems*, volume 1. Computer Science Press, 1988.

Patricia Wright. *A User-Oriented Approach to the Design of Tables and Flowcharts*, pages 317–340. Educational Technology Publications, 1982.