

EXTRACTION OF ITEMS FROM ABSTRACTS

Fujio Nishida, Giichi Kishimoto and Shinobu Takamatsu
Department of Electrical Engineering,
Faculty of Engineering, University of Osaka Prefecture,
Sakai, Osaka, Japan 591

This paper presents a method of extracting prescribed items from a summary or an abstract of technical papers. An experimental system is now under development. The internal expression of an input sentence is transformed into a standard form which has the standard case-labels for extraction of items. The part of remaining sentences is combined with the adjacent sentence into an item section every main item. [1] And the prescribed items are extracted efficiently.

1. EXTRACTED ITEMS

This paper is primarily concerned with a textual analysis from the viewpoint of extraction of items in a specified form for filing. The kind of texts is chosen to a summary of some technical papers, the length of which is rather short. The first thing to be done will be the determination of the specification table of the extracted items rather than the construction of the rewriting rules. The main items to be extracted are generally chosen to the labels of main frames appearing in many summaries of technical papers which can be also considered to produce their respective details of the main frames of summaries by various transformation. For the whole of various kinds of technical papers, however, it is a difficult and significant problem to choose the main items and to construct a well organized tree or graph of these summaries. Hence, we confine the file of the main object to a specific one, for example, one of software systems such as language processing systems. In this scheme, a choice of main items is taken as shown in Table 1.

```
PRED-F: _ (PRED-F: *, AG-S: t1 t2, PAT: _ ##,
SO: _ ##, G: _ ##, M: _ ##)
AG-S: t1 (OBJ-S: *, COMP-S: t2)
INSTR-S: _ (PRED-F: _, AG-S: *, PAT: _, SO: _,
G: _, M: _, INSTR-S: _ #, USED-S: t1 t2)
USED-S: _ (PRED-F: _, AG-S: *, PAT: _, SO: _,
G: _, M: _, INSTR-S: t1 t2, USED-S: _ #)
```

Table 1. A Specification Table

In the above, two capital-letter sequents connected with a hyphen '-' are a case-label and a category of a term inserted to the case respectively. The specification of the category of a case is assumed optional. The abbreviated

large symbols denote OBJECT, COMPONENT, PREDICATE, AGENT, PATIENT, SOURCE, GOAL, MODAL or manner, INSTRUMENT, USED system or location, PARTICIPANT, CHARACTERISTIC, SYSTEM, FUNCTION and PROCESS. Each underline denotes a slot of a case and a small letter t denotes a term. Each term takes a form

C: t
or C: t (--, K: t, C₁: t₁, C₂: t₂, --),

which means a term t plays the role denoted by a case or a case-category label C in a specified upper case frame and it is sometimes modified by several terms in a case frame enclosed by parentheses.

In a modifying frame, the modified term and the case frame are described together with a predicative term when it is necessary to show the role of the modified term clearly. In the Table 1, the symbol * in a modifying frame denotes the modified term prefixed to the frame. The double underlined term specifies or defines the main case-label of the term prefixed to the modifying frame and are used for identification of the term. In a completed form for output, however, they are deleted for efficiency. The nested modification of a term is restricted to specified terms with the symbol '#' or '##' in order to avoid to store duplicated information and non-essential one. The term with the symbol '#' can be modified by a modifying frame which involves a term in the same case-label as the prefixed case-label recursively. The term with the symbol '##' can be modified by any modifying frame so long as it is not duplicated information.

A main item is defined as a pair of a case-label and the characterized term with a number instead

of modifying frames appearing in the leftmost of the specification table. For a case-label, there are sometimes several terms in a union or serial form, where the union form is an unordered set of terms with a form 'C: $t_1 \cup t_2 \cup \dots$ ', while the serial form is an ordered set of terms with a form 'C: $t_1 \wedge t_2 \wedge \dots$ ', placed by a temporal or causal relation.

2. NORMALIZATION

Prior to extracting some specified items from summaries, it is preferable that each sentence of a summary is normalized to some extent. First, each input of a summary is transformed into the internal form by parsing. Each word of a pronoun or a common noun is characterized by attaching a number by means of the anaphoric analysis though the attached number is omitted except section 4 for simplicity. Then the internal form is transformed into the standard form which takes a concrete predicate word as a governor at every partial predicate or function expression and does not involve any term of the same name as a case-label. Every transformation is done by referring to case-labels and categories of terms though the precise conversion procedures are omitted for simplicity. The outline of the conversions is described as follows.

2.1 Transform To Predicate-governor Forms

This is a form of a predicate expression every part of which generally takes a concrete predicative word as a governor if the part involves a concrete predicative word. For example, let us take an example of an internal form which involves a term of a function form that contains a predicative word p corresponding to a clause or a phrase.

(PRED: p_1 , ---, K_0 : t_1 (K_1^T : p_2 (K_2 : t_2 , --))).

The predicate-governor form of the above is

(PRED: p_1 , ---, K_0 : t_1 (PRED: p_2 , K_1^T : $*$, K_2 : t_2 , --)).

If a function form lacks a predicative word, the term of the pseudo case of a predicative word is left empty.

[Example 1]

K : t_1 (COMP: t_2) $\rightarrow$ K : t_1 (OBJ: $*$, COMP: t_2),

where the symbol ' $\rightarrow$ ' shows the left side is to be replaced by the right side.

A term of the same name as a case label K is also transformed to an expression using the case-label K .

[Example 2]

K (OBJ: t) $\rightarrow$ $\$(OBJ: t, K: \$)$,

where the symbol '\$' denotes a certain term to be replaced by a concrete term if it appears later.

2.2 Transform To Non-complementary Forms

A predicate part of a complement form is transformed to a non-complementary form by a case-transformation.

[Example 3]

(PRED: make, AG: t_1 , PAT: t_2 (PRED: $*$, **t**), G: t_3)
 $\rightarrow$ (PRED: t_2 , **t** , INSTR: t_1 , M: t_3),

where a bold letter **t** denotes a sequence of pairs of case-labels and terms.

2.3 Removal Of Formal Predicative Words

A formal predicative word such as 'be based on' or 'comprise' is removed, and the case-labels prefixed to the dependant terms are replaced by a more concrete case label implied by the removed predicative word.

[Example 4]

(PRED: use, AG: t_1 , PAT: t_2 ,
 PURPOSE: t_3 (PRED: $*$, **t**))
 $\rightarrow$ (PRED: t_3 , **t** , AG: t_1 , INSTR: t_2).

The conversion rules in article 2.2 and 2.3 are implemented for each word in a word dictionary.

3. EXTRACTION OF ITEMS

From the normalized sentences of a summary the items in the specification table are extracted. First, the subject sentence is taken from a sentence of the summary which involves a predicative word of category 'process' and contains several terms involved in the main items of the specification table as well as in the title. The cases of these items are identified using the method described in section 1 and several case-transformation rules such as

(OBJ-S: t_1 , COMPOSITE-S: t_2)
 $\rightarrow$ (OBJ-S: t_2 , COMP-S: t_1),
 (PRED-F: t_1 , AG-S: t_2 , INSTR-S: t_3
 (PRED-F: t_4 , AG-S: $*$))
 $\rightarrow$ (PRED-F: t_4 , AG-S: t_3 , USED-S: t_2).

(PRED-F: t_1 , AG-S: *)).

Thereafter, the details of the main items are arranged as indicated in the specification table, combined with the main items and put into the edition area for output.

Subsequently, the remaining sentences are processed from the first sentence. First, the main item to which the current sentence belongs is identified. Second, the part to be extracted is arranged to the specified form. For example, if a sentence contains a serial relation, it is arranged as follows;

(PRED-F: t_1 , AG-S: t_2 , PAT: t_3 ,
MEANS-F: t_4 (PRED-F: *, t) $\rightarrow$ (PRED-F: t_4 ,
 t) $\wedge$ (PRED-F: t_1 , AG-S: t_2 , PAT: t_3)).

Then, the following sentence is taken and it is asked whether or not this sentence belongs to the same main item as the preceding one. If it does, the part to be extracted is arranged, combined with the preceding extracted part as an item section and rearranged using some temporal relations or combined rules such as

(K_0 : t , K_1 : t_1) $\cup$ (K_0 : t , K_2 : t_2)
 $\rightarrow$ (K_0 : t , K_1 : t_1 , K_2 : t_2).

If the following sentence does not belong to the same item as the preceding one, the extracted part of the preceding item section are marked for later scanning, and its copy is combined and rearranged with the data in the edition area for output. Then the current sentence is processed as a new item sentence. At the end of the summary, the scanning restarts from the beginning of the summary for extracting the specified detail from the remaining parts of each sentence. Each term is stored from the output form into a tree-like file of categories together with several sequents of the prefixed case-labels and the document number.

4. EXAMPLE

4.1 An Example

(1) This paper describes TORUS, a natural language understanding system which serves as a front end to a data base management system in order to facilitate communication with a casual user. (2) TORUS uses a semantic network for understanding input statements and for deciding what information to output in response. (3) The semantic network stores general knowledge about the problem domain. (4) A number of functions of TORUS make it possible to integrate the meaning

of an input statement to the semantic network, and to select a portion of the semantic network which stores information that must be output.

4.2 Standard Forms

- (1) (PRED-P: describing, AG: paper'1,
PAT: TORUS
(PRED-F: understanding, AG: *,
PAT: natural-language'1)
(PRED-F: communicating, INSTR: *, M: easy,
COAGENT: casual-user'1)
(AG: *, USED: front-end'1 (AG: *,
USED: system'1 (PRED-F: managing,
AG: *, PAT: data-base'1))))),
- (2) (PRED-F: understanding, AG: TORUS, INSTR:
semantic-network'1, PAT: input-statement'1)
 $\cup$ (PRED-F: deciding, AG: TORUS, INSTR: semantic-
network'1 PAT: information (DET: what)
(PRED-F: output, PAT: *,
ENVIRONMENT: response)),
- (3) (PRED-F: storing, AG: semantic-network'1,
PAT: general-knowledge'1
(OBJ: *, PART: problem-domain'1)),
- (4) (PRED-F: integrating, AG: TORUS, PAT:
meaning'1 (CH: *, OBJ: input-statement'2),
G: semantic-network'1, M: possible)
 $\cup$ (PRED-F: selecting, AG: TORUS, PAT: portion'1
(OBJ: *, COMPOSITE: semantic-network'1
(PRED-F: storing, AG: *, PAT: information'1
(PRED-F: output, PAT: *, M: must)))
M: possible).

4.3 Extracted Items

(PRED-F: understanding (PRED-F: *, AG: TORUS,
PAT: natural-language)
 $\cup$ integrating (PRED-F: *, AG: TORUS,
PAT: meaning (CH: *, OBJ: input-statement),
G: semantic-network, M: possible)
 $\cup$ selecting (PRED-F: *, AG: TORUS, PAT: portion
(OBJ: *, COMPOSITE: semantic-network))
 $\cup$ deciding (PRED-F: *, AG: TORUS, PAT: informa-
tion (DET: what) (PRED-F: output, PAT: *,
ENVIRONMENT: response)),
AG-S: TORUS,
INSTR-S: semantic-network (PRED-F: storing,
AG: *, PAT: general-knowledge, USED: TORUS)
(PRED-F: storing, AG: *, PAT: information),
USED-S: front-end (AG: *, USED-S: system
(PRED-F: managing, AG: *, PAT: data-base),
INSTR: TORUS)).

REFERENCE

[1] Hobbs, J.R. "Coherence and Interpretation in English Texts." In Proc. IJCAI-77. MIT, Cambridge, Mass., August, 1977, pp. 110-116.