自然言語処理 89−5
（1992. 5. 14）

痕跡を扱うためのチャート法の拡張

春野 雅彦* 松本 裕治* 長尾 真*

*京都大学工学部

英語、フランス語などでは関係節、受動文のような様々な言語現象において痕跡を持つ文が頻繁に現れ、その解析効率がシステム全体に与える影響は大きい。そのため計算言語学の分野では痕跡を取り扱うための様々な手法が提案されてきた。これらはユニフィケーション文法の枠組の中で痕跡を統一的に処理しようとする枠組と句構造文法の記法を拡張して痕跡の処理を直接パーサーで行なうものの二つの流れに分けることができる。ユニフィケーション文法による方法は確かに他の言語現象との整合性が良く、文法全体の見通しも良くなるという利点を持っているが、これを実用規模の大きさで実現するとなると大がかりな辞書が必要となるし、ユニフィケーションにかかるコストのためシステム全体の効率が低下する欠点も持っている。このことを考慮すると痕跡の処理はパーサーで行ないユニフィケーションは必要最小限に抑えるのが現実的であると思われる。本稿では、ボトムアップチャート法の拡張によって痕跡を処理できるアルゴリズムを提案する。この手法では関係節に現れるロスの制約も自然に取り扱うことができる。またチャート法に基づく手法であるため並列化が可能であるだけでなく、逐次に実行しても効率の良いプログラムを得ることができる。

An Extended Chart Parsing Algorithm for Gap Handling

Masahiko Haruno* Yuji Matsumoto* Makoto Nagao*

*Department of Electorical Engineering, Kyoto University

Various types of gap structures occur in European languages so often that gap handling method has great influence on the efficiency of a total system. Many approaches were proposed so far, and they could be divided into two directions. One is unification-based approach and the other is extension of context free grammar rules and parsing algorithm. The former can treat gaps and other phenomena in the same uniform way. But unification causes a serious problem of unefficiency. Therefore , it is practical to treat gap structures by modification of a parser and to use unification as little as possible. This paper proposes an extended chart parsing algorithm for gap handling, which also realizes Complex NP constraint easily. Since the mechanism is an extension of the chart algorithm, efficient implementation are realizable in sequnential as well as in parallel.

1 Introduction

Structures with gap such as relative clauses pose difficulties in parsing. In such constructions, an element is extracted from an embedded structure. Although the language phenomena concerning gaps are stated in a simple way, form of a phrase is ristricted by various syntactic constraints. It is therefore not practical to write rules each of which stands for the place of a gap explicitely.

Two types of gap handling approaches should be distinguished. One is unification-based method. HPSG[1], LFG[2] and other unification-based approaches are based on lexical feature descriptions consisting of attribute-value pairs, and treat all linguistic phenomena by a uniform operation called unification. Those approaches suffer from the problem of efficiency in parsing. There is actually a trade-off in efficiency between the direct representation of syntax as grammar rules and the representation of syntax by feature descriptions[3].

Another approach is to extend context free grammar rules and to devise a devoted parsing algorithm. Pereira proposed XGs[4] in which DCGs[5] are extended for gap handling. He showed a translation method of an XG into a Prolog program that implements a top-down back-tracking parser for the grammar. Hirschman studied a meta-rule treatment[6] of wh-constructions. She showed a way to translate meta-description of wh-constructions into Restriction Grammar rules, which is eventually implemented as a Prolog top-down back-tracking parser. The top-down strategy has an advantage for gap handling since prediction of the gap's occurence is relatively easy. An approach based on bottom-up parsing are also proposed[7], which uses the bottom-up left-corner parsing strategy with top-down prediction. This approach is also based on Prolog's depth-first execution mechanism. The top-down depth-first strategy is not efficient enough for a large scale grammar rules.

This paper presents a direct extension of Chart Parsing[8] so that the gap handling task is naturally incorporated within the efficient context-free parsing algorithm. We first regard gap phenomena as simple as an unrestricted extraction of an element from some grammatical category. Then we put restriction on the algorithm so that the constraints on gap phenomenon is implemented in a modular way. For such a constraint, we describe the Complex NP Constraint as a representative example.

The method can be implemented both in sequential and parallel ways since Chart Parsing scheme itself is independent of the execution order. We believe that the method is applicable to all parsers based on Chart Parsing.

Chapter2 describes our extended Chart Parsing algorithm by comparing it with the original Chart Parsing algorithm. In Chapter 3, the notation for grammar description and its extension for the Complex NP Constraint are explained.

2 Extended Chart Parsing Algorithm

2.1 The Chart Parsing Algorithm

This section shows the basic bottom-up chart algorithm[8][1]. The Chart algorithm is often explained by a graph. Two types of edges, called active and inactive edges, comprise the graph. The initial edges in the graph is the set of inactive edges corresponding to the words in the given sentence. A consecutive pair of words share a vertex initially. An inactive edge represents a partial parse tree whose descendants are completely filled. An active edge represents a tree some of whose right children are not yet filled.

Two basic operations completes the algorithm. One is to introduce an active edge, and the other is to fill an open place in an active edge with an inactive edge. These operations are described as Procedure-1 and Procedure-2 below. A data structure named a term is associated to each edge. An inactive edge has as the term its parse tree. An active edge also has the parse tree as the term, where the incomplete parts of the tree are unfilled and left as

[1] Although another type of Chart Parsing algorithm, the top down Chart Parsing is also introduced in Kay's paper, we only introduce the bottom-up version and its extension. Applying the extension to the top-down one is straightforward

open boxes. If an active edge represents a grammar rule $a \rightarrow b,\ c,\ d$, and the categories c and d are not yet filled, it has a term, [b [?]c [?]d]a, where the question marks represent open boxes.

Procedure-1: Let e_i be an inactive edge of category a incident from vertex v to vertex w. For all rules of the form $b \rightarrow c_1, c_2 \cdots c_n$ in the grammar such that $c_1 = a$, introduce a new edge e_a with the term [a [?]$c_2 \cdots$ [?]c_n]b, incident from v to w, provided that there is no such edge in the chart already.

Procedure-2: Let e_a and e_i be adjacent active and inactive edge. e_a is incident from vertex v and e_i is incident to vertex w. Let [?]α be the first open box in e_a. If e_i is of category α, create a new edge between v and w whose term is that of e_a with the first open box replaced by the term of e_i.

Both these rules are applied to all qualifying edges or pairs of edges, including any that arise as a result of applying these rules.

2.2 Grammar Description

We show here the notation of grammar rules for gap handling. Usually, a gap has a corresponding extracted category that appears somewhere in the same sentence. In the general case of context-free grammars, however, a gap can be seen as a missing element in a certain category which does not necessarily appear in the sentence. Thus, we assume the general form of a grammar rule with a gap is as follows:

```
    a --> b, c/g, d.
```

This rule specifies that the category `c` must have `g` as a gap. Followings are an example of natural language grammar rules:

```
(1) sentence --> np, vp.
(2) vp --> v, np.
(3) vp --> v.
(4) np --> det, noun.
(5) np --> np, relpronoun, sentence/np.
```

The grammar rule (5) means a gap of category **np** appears in **sentence**. Some extension to the grammar description is introduced in 2.5.2.

2.3 Extended Chart Parsing Algorithm for Gap Handling

This section describes how Chart algorithm is extended to handle gaps. For this purpose, an extra datum is attached to each edge. It is called a label and takes one of three kinds of values, $+gap$, $-gap$ or nil, where gap is a category which is expected as a gap. The meaning of each symbol is as follows: $+gap$ means that a gap of category gap is expected in the subsequent part of the chart, $-gap$ means that a gap of category gap has already been used within the edge, and nil means that the edge has nothing to do with gaps.

In the following extension, sets of new vertices are introduced dynamically. Vertices in a set have one-to-one correspondence with the original vertices. One set is introduced by an invocation of an active edge that expects a category containing a gap, though the set of whole corresponding vertices to the original ones are not introduced at once. They are introduced one by one only when they are required in the course of the algorithm. They are in fact imaginary ones which are introduced for the explanatory convenience. We use the notation like v' and w' for corresponding imaginary vertices to v and w. We use x and y to represent both imaginary and normal vertices in the procedure.

The initial configuration of the chart is the same as the original one except that every initial inactive edge has label nil. Procedure-1 and Procedure-2 are extension of those of the original Chart algorithm. Procedure-3 creates the first imaginary vertex by an invocation of a rule that contains a gap. Procedure-4 is to copy inactive edges to their corresponding imaginary positions. Procedure-5 is to fill the gap.

Procedure-1: Let e_i be an inactive edge of category a incident from x to y. For all rules of the form $b \rightarrow c_1, c_2 \cdots c_n$ in the grammar such that $c_1 = a$, introduce a new edge e_a with the term $[a\ [?]c_2 \cdots [?]c_n]b$ and with the same label as that of e_i, incident from x to y, provided that there is no such edge in the chart already.

Procedure-2: Let e_a and e_i be adjacent active and inactive edge. e_a is incident from v and e_i is incident to w. Let $[?]\alpha$ be the first open box in e_a. If e_i is of category α, create a new edge e_n between v and w whose term is that of e_a with the first open box replaced by the term of e_i.

The label L_n of e_n is defined according to the combination of the labels L_a of e_a and L_i of e_i:

If $L_a = nil$ and $L_i = nil$, then L_n is nil.

If $L_a = +gap$ and $L_i = +gap$, then L_n is $+gap$.

If $L_a = +gap$ and $L_i = -gap$, then L_n is $-gap$.

If $L_a = -gap$ and $L_i = nil$, then L_n is $-gap$.

Procedure-2': Let e_a and e_i be adjacent active and inactive edge. e_a is incident from v and e_i is incident to w. Let $[?]\alpha/\beta$ be the first open box in e_a. If e_i is of category α and of label $-\beta$, create a new edge e_n between v and w whose term is that of e_a with the first open box replaced by the term of e_i. The label of e_n is nil.

Procedure-3: This procedure is an exceptional case of Procedure-1 and Procedure-2, that is, when the following configuration occurs in either of above procedures it is overtaken by this procedure:

When an active edge e_a is to be introduced from v to w in either of Procedure-1 or Procedure-2 and the first open box in the term of e_a is to be a/b, create an imaginary vertex, w', corresponding to w, and introduce the active edge e_a' from v to w' instead of e_a. The label of e_n' is $+b$.

Procedure-4: Let e_a be an active edge whose label is $+gap$, incident to v', where v' is the corresponding imaginary vertex of v. If e_i is an inactive edge of category a and of label nil incident from v to w, create an inactive edge e_i' from v' to w' whose term is that of e_i and whose label is $+gap$.

Procedure-5: Let e_a be an active edge incident to w'. Let the category at first open box of its term be a and the label be $+cat$. If the category a is reachable[2] to the category cat, create an inactive edge from w' to w whose term is $[\phi]cat$ and whose label is $-cat$.

The label $+cat$ means that a category cat has to be filled in some subsequent position. This kind of labels are attached only to those edges adjacent to imaginary vertices. The label nil means that the current analysis has nothing to do with gaps. This label appears only with the edges between normal vertices. The label $-cat$ indicates that the gap of category cat has been filled within the edge. Every edge with a label $-gap$ is incident from an imaginary vertex to a normal vertex. Thus, Procedure-2 covers all of possible combinations of labels for adjacent active and inactive edges.

Once a gap rule is used and imaginary vertices are introduced, triggered by Procedure-3, the analysis at imaginary vertices returns to normal vertices only through Procedure-5. This guarantees that an introduced gap is definitely filled to complete the invoked gap containing rule and that it is filled exactly once. Note that different rules that contain a gap introduce distinct sets of imaginary vertices, so that the expected gap elements and the filled elements have one-to-one correspondence.

2.4 An Example

Figure 1 shows the graph representation of the chart when parsing 'The man who she loved died,' assuming the grammar rules in 2.2. Thick lines are inactive edges, dotted lines are active edges, and arrows are inactive edges whose labels are $-np$. Labels of edges adjacent to imaginary vertices, v_4', v_5' and v_6' are $+np$, and labels of other edges are nil. Table 1 is the chart. A row represents the information of an edge. Each of them consists of the number associated with the edge, vertices on the both ends of the edge, the term, the label, and the procedure number

[2] If there is a rule $\alpha \rightarrow \beta \cdots$, then α is defined to be directly reachable to β. The reachability relation is the reflexive and transitive closure of the directly reachability.

#	from	to	term	label	Procedure
1	v_1	v_2	det (the)	*nil*	
2	v_2	v_3	noun (man)	*nil*	
3	v_3	v_4	relpro (who)	*nil*	
4	v_4	v_5	np (she)	*nil*	
5	v_5	v_6	v (loved)	*nil*	
6	v_6	v_7	v (died)	*nil*	
7	v_1	v_2	[det,[?]noun]np	*nil*	1(#1)
8	v_1	v_3	[det,noun]np	*nil*	2(#7,#2)
9	v_1	v_3	[[det,noun]np,[?]relpro,[?]s/np]np	*nil*	1(#8)
10	v_1	v_4'	[[det,noun]np,relpro,[?]s/np]np	$+np$	3(#9,#3)
11	v_4'	v_4	[φ]np	$-np$	5(#10)
12	v_4	v_5	[np,[?]vp]s	*nil*	1(#4)
13	v_4	v_5	[np,[?]relpro,[?]s/np]np	*nil*	1(#4)
14	v_4'	v_5'	np	$+np$	4(#10,#4)
15	v_4'	v_5'	[np,[?]vp]s	$+np$	1(#14)
16	v_5	v_6	[v,[?]np]vp	*nil*	1(#5)
17	v_5'	v_6'	v	$+np$	4(#15,#5)
18	v_5'	v_6'	[v,[?]np]vp	$+np$	1(#17)
19	v_6'	v_6	[φ]np	$-np$	5(#18)
20	v_5'	v_6	[v,[φ]np]vp	$-np$	2(#18,#19)
21	v_4'	v_6	[np,[v,[φ]np]vp]s	$-np$	2(#15,#20)
22	v_1	v_6	[[det,noun]np,relpro,[np,[v,[φ]np]vp]s/np]np	*nil*	2'(#10,#21)
23	v_1	v_6	[[[det,noun]np,relpro,[np,[v,[φ]np]vp]s/np]np,[?]vp]s	*nil*	1(#22)
24	v_6	v_7	[v]vp	*nil*	1(#6)
25	v_1	v_7	[[[det,noun]np,relpro,[np,[v,[φ]np]vp]s/np]np,[v]vp]s	*nil*	2(#23,#24)

表 1: The Chart

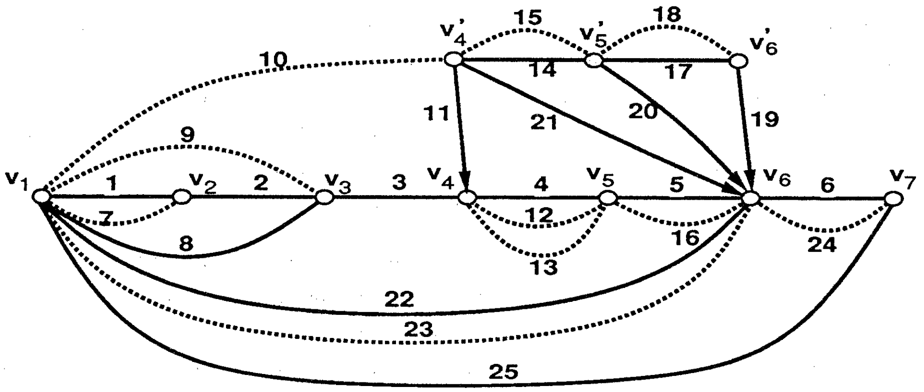

図 1: The Graph Representation of the Chart used to produce the edge. Each procedure number is associated with the edge number(s) used in the procedure. For instance, 1(#1) means that the edge is generated from the edge #1 by Procedure-1, and 2(#7,#2) means that the edge is generated from the edges #7 and #2 by Procedure-2. Terminal symbols are omitted in the terms in Table 1. You must especially pay attention to the edge #10 introduced by procedure3, which splits the parsing path from the ordinal chart.

2.5 Noun Phrase Constraints

2.5.1 The Complex NP Constraint

There are various constraints that restrict constructions involving gaps. This section gives a brief grammatical explanation of Ross' Complex NP(Noun Phrase) Constraint[9] as an example of such constraints. First, consider the following sentence:

> *The man who I read a statement which was about ϕ is sick.

The antecedent of the first relative clause "the man" can be regarded as being extracted from just after "about" leaving a gap there. This sentence is, however, ungrammatical since it violates Ross' Complex NP Constraint:

The Complex NP Constraint (Ross) Any element cannot be extracted from a sentence that is included in an NP with a lexically-filled nominal head.

The previous example is ungrammatical since the embedded relative clause "a statement which was about" has the lexically-filled nominal head "a statement" and the outer relative clause requires its gap to be in the sentence within the embedded relative clause.

Other examples that violate the constraint are:

> *The hat which I believed the claim that Otto was wearing ϕ is red.

> *Here is the snowball which I chased the boy who threw ϕ at our teacher.

The next sentence is sound since the embedded that-clause does not have a lexically-filled nominal head.

> The hat which I believed that Otto was wearing ϕ is red.

The extended Chart Parsing algorithm used with the grammar rules like in 2.2 accepts all of the above sentences since there is no restriction on the places to fill a gap. We will explain how such a constraint is realized in the extended algorithm.

2.5.2 Incorporation of the Complex NP Constraint

The way to realize the Complex NP Constraint is to prevent the gap information from being used in grammar rules that define a noun phrase satisfying the condition of The Complex NP Constraint. We introduce a special notation to specify such grammar rules. In the following sample grammar rules, a '$\Rightarrow$' specifies that the rule should not be invoked by inactive edges that have gap information.

```
(5') np ==> np, relpronoun, sentence/np.
```

The constraint is implemented in the extended Chart Parsing algorithm by adding the following modified definition of Procedure-1 for rules with '$\Rightarrow$'. Note that now Procedure-1 is applicable only to rules with '$\longrightarrow$'.

Procedure-1': Let e_i be an inactive edge of category a and of label *nil* incident from v to w. For all rules of the form $b \Rightarrow c_1, c_2 \cdots c_n$ in the grammar such that $c_1 = a$, introduce a new edge e_a with the term $[a\ [?]c_2 \cdots [?]c_n]b$ and with the label *nil*, incident from v to w, provided that there is no such edge in the chart already.

This means that grammar rules with '$\Rightarrow$' are never used for invocation by inactive edges whose label is either $+gap$ or $-gap$. This modification also enables the algorithm to parse nested occurrences of relative clauses correctly.

```
|: The man stood by the river which was calm.
parsed
                      sentence
                          |
   np-------------------------vp
   |                          |
det-noun  vp---------------------pp
 |   |     |                     |
 |   |    v1   prep----------------np
 |   |     |    |                   |
 |   |     |    |      np-----rel_marker---sentence
 |   |     |    |       |         |           |
 |   |     |    |   det-noun      |      np------vp
 |   |     |    |    |    |       |       |      |
 |   |     |    |    |    |       |       |  be_v--adj
 |   |     |    |    |    |       |       |   |     |
the man  stood  by  the river   which    gap was  calm

execution time          = 170 msec

|: The man knew the book which I read the statement about.
parsed
                            sentence
                                |
   np-------------------------------vp
   |                                |
det-noun  v2--------------------------np
 |   |    |                            |
 |   |    |      np----rel_marker------------sentence
 |   |    |      |         |                     |
 |   |    |   det-noun     |      np--------------vp
 |   |    |    |   |       |      |               |
 |   |    |    |   |       |      |   v2------------np
 |   |    |    |   |       |      |   |              |
 |   |    |    |   |       |      |   |        np----------pp
 |   |    |    |   |       |      |   |         |           |
 |   |    |    |   |       |      |   |   det---noun    prep--np
 |   |    |    |   |       |      |   |    |      |       |    |
the man  knew the book   which    i  read the statement about gap

execution time          = 180 msec

|: The man who I read a statement which was about is sick.

execution time          = 50 msec
```

図 2: 構文解析例

2.6 Examples

This section shows some examples parsed with our system. Take look at figure2. The first example shows a usual sentence with gap. The second and third examples show the realization of the Complex NP Constraint, of which third example violates the constraint. The system is implemented in SICStus Prolog 0.7 on SPARCstation 2.

3 Conclusion and Further Work

The extended Chart Parsing algorithm deals with the most basic construction of gap phenomena. It is guaranteed by the algorithm that the extracted element occurs only once as a gap. The nested cases of gap constructions can also be treated properly by introducing the special notation that restricts the extraction of elements from specified grammar rules. We need to study various other constraints and to identify the set of facilities necessary for the broader coverage of gap phenomena.

For instance, in the following grammar rule, the leftmost child np cannot be extracted as an antecedent of a relative clause.

```
np --> np, pp.
```

This rule may be rewritten to the following, assuming that the notation <*cat*> restrict *cat* to be filled by a gap:

```
np --> <np>, pp.
```

Gap constructions introduce another complication when they occur with coordination structure. An extracted element may have to be filled in more than one place in coordinated sentences. A general treatment of coordination is necessary so as to realize the facilities in a modular way, which requires further research.

An extension that is being undertaken is to incorporate the system with a facility to deal with other syntactic as well as semantic information. The current system supports grammar descriptions with DCGs, feature descriptions, and their unification facilities.

参考文献

[1] Pollard, C. and Sag, I.: "Information-Based Syntax And Semantics Vol.1," CSLI (1987).

[2] Kaplan, R.M. and Bresnan, J., "Lexical-Functional Grammar: A Formal System for Grammatical Representation," Chap.4 of 'The Mental Representation of Grammatical Relations' J. Bresnan (ed.), MIT Press, pp.173-281, 1982.

[3] Shieber, S.M.: "Using Restriction to Extend Parsing Algorithms for Complex-Feature-Based Formalisms," Proc. of 23rd Annual Meeting of ACL, pp.145-152, (1985).

[4] Pereira, F.: "Extraposition grammars," AJCL, Vol.7, No.4, pp.243-256, (1981).

[5] Pereira, F.: "Definite Clause Grammars for Language Analysis – A Survey of the Formalism and a Comparison with Augmented Transition Networks," Artificial Intelligence, Vol.13, pp.231-278, (1980).

[6] Hirschman, L.: "A Meta-Rule Treatment for English Wh-Constructions," in *Meta-Programming in Logic Programming* H. Abramson and M.H. Rogers (eds.), The MIT Press, pp.1-21, (1989).

[7] Konno, K. and Tanaka, H.: "Processing Left-extraposition in Bottom-up Parsing System (in Japanese)" Computer Software, Vol.3, No.2, pp.19-29, (1986).

[8] Kay, M.: "Algorithm Schemata and Data Structure in Syntactic Processing," Technical Report CSL-80-12, Xerox PARC, (1980).

[9] Ross, J.R.: "Constraints on Variables in Syntax," ph.D.dissertion, MIT. Bloomington, Indiana:IULC (1967).