Coordination-aware Dependency Parsing (Preliminary Report)

Akifumi Yoshimoto* **Kazuo Hara†** **Masashi Shimbo*** **Yuji Matsumoto***

*Nara Institute of Science and Technology
Ikoma, Nara, Japan
{akifumi-y,shimbo,matsu}@is.naist.jp

†National Institute of Genetics
Mishima, Shizuoka, Japan
kazuo.hara@gmail.com

Abstract

Coordinate structures pose difficulties in dependency parsers. In this paper, we propose a set of parsing rules specifically designed to handle coordination, which are intended to be used in combination with Eisner and Satta's dependency rules. The new rules are compatible with existing similarity-based approaches to coordination structure analysis, and thus the syntactic and semantic similarity of conjuncts can be incorporated to the parse scoring function. Although we are yet to implement such a scoring function, we analyzed the time complexity of the proposed rules as well as their coverage of the Penn Treebank converted to the Stanford basic dependencies.

1 Introduction

Even for state-of-the-art dependency parsers, the recovery of dependencies involving coordinate structures remains difficult. According to Nivre et al. (2010), the accuracy in parsing the construction known as right node raising, which includes a coordinate structure such as "president and chief executive of the company," is less than fifty percent.

Apart from dependency parsers, there are methods specialized for coordination structure analysis, which attempt to identify coordinate structures from the similarity of conjuncts (Kurohashi and Nagao, 1994; Hara and Shimbo, 2007; Hara et al., 2009).

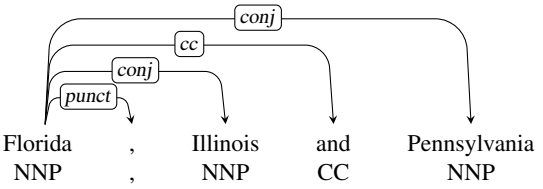
Our final goal is to improve the accuracy of parsing sentences containing coordination, by taking advantage of the above methodologies. In the same vein, Hanamoto et al. (2012) used dual decomposition to combine an HPSG parser with Hara et al.'s (2009) model.

In this paper, we take a different approach: we augment Eisner and Satta's (1999) parsing rules for normal dependencies, with a new set of rules to specifically handle coordinations. This design reflects the fact that Eisner and Satta's "split-head" (Johnson, 2007) rules construct dependency trees on the left and the right of a word independently, and thus the entire span of conjuncts cannot be captured during the course of construction.

Using the Penn Treebank (PTB) (Marcus et al., 1993) after converting the structures to Stanford basic dependency representation (de Marneffe and Manning, 2008), we verified that the proposed rules cover 94% of the sentences involving coordinations. Moreover, about half the failure of the remaining sentences was caused by unusually annotated structures, rather than a weakness in the proposed rules.

2 Dependency structure for coordination

In the Stanford basic dependencies, coordination is represented as a dependency structure in which the first conjunct is the head of the dependency[1], with the second and subsequent conjuncts being the dependents of the first. Commas and coordinate conjunctions (such as "and", "or") separating conjuncts are also the dependents of the first conjunct, with labels *punct* and *cc*, respectively.



Although not explicitly stated in the Stanford manual, in the converted PTB corpus, the first conjunct

[1]The Stanford dependencies manual (de Marneffe and Manning, 2008) merely states that the first conjunct is *normally* the head, but we take this rule as definitive in this paper. The error analysis of our method based on this assumption is given in Section 5.

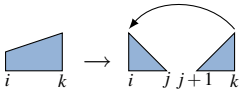
Proceedings of the 14th International Conference on Parsing Technologies, pages 66–70,
Bilbao, Spain; July 22–24, 2015.

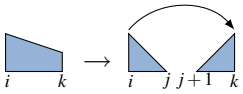
is, in most cases, also the head of extra punctuations (e.g., a comma in front of "and" in "*A*, *B*, and *C*"), or parentheses (e.g., "he says" in "*A*, *B* and, he says, *C*") that occur inside coordinate structure.

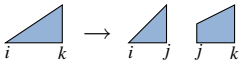
3 The Eisner-Satta algorithm

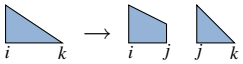
In this section, we briefly review the grammar rules used in the Eisner-Satta algorithm (1999), which our proposed rules are designed to augment. For details about the algorithm, see the original paper, as well as (Koo and Collins, 2010; Johnson, 2007).

The Eisner-Satta algorithm builds a parse tree in a bottom-up manner using the following four rules, each of which yields a span of different type shown next to the respective rule.

(1) INCOMPLETELEFT 

(2) INCOMPLETERIGHT 

(3) CCOMPLETELEFT 

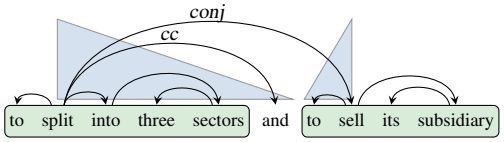
(4) COMPLETERIGHT 

In these rules, i, j, and k indicate word indices. Symbol ℓ is also used for an index later. Let $w(i)$ denote the ith word in a sentence. Each word $w(i)$ is initially associated with COMPLETELEFT  and COMPLETERIGHT  . A root node is added as the leftmost node of a sentence, with only COMPLETERIGHT.

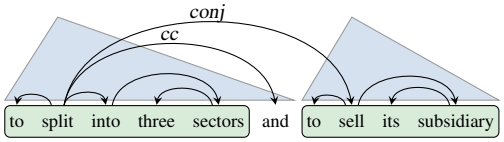
4 Parsing rules for coordination

We augment Eisner and Satta's split-head dependency parsing rules with a set of additional rules that are tailored for similarity evaluation of conjuncts. Split-head rules lead to an efficient $O(n^3)$ parsing algorithm for a sentence of length n, but the way in which a parse tree is built does not allow evaluation of conjunct similarity as done in Hara et al. (2009).

When a structure "*A* and *B*" is parsed with the Eisner-Satta algorithm, rule (2) is used for creating a dependency arc (with label *conj*) between conjuncts *A* and *B*. At that moment, the available spans are COMPLETERIGHT whose head is *A*, and COMPLETELEFT whose head is *B*. However, these spans do not correspond to the entire spans of the respective conjuncts. See the figure below for illustration.



Therefore, we introduce new constituents specifically for coordination, which group both sides of the head of conjuncts, as in:

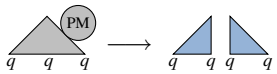


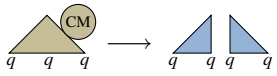
4.1 Punctuations and conjunctions

Let q, r, and s signify word indices whose values are restricted by the position of punctuations or conjunctions[2].

We first group COMPLETELEFT and COMPLETERIGHT for a punctuation or a conjunction as a single constituent. We call the resulting constituents PUNCTMARK (or PM for short) and CCMARK (CM), respectively.

Rule (5) can be applied only if $w(q)$, the word at index q, is a commas or a semicolons, and rule (6) only if $w(q)$ is a coordinate conjunctions, such as "and," "or," "and/or," "&," "but," "plus," and "yet."

(5)  $O(p)$

(6)  $O(p)$

As shown next to the rules, the time needed to apply them is $O(p)$, where p ($\ll n$) is the number of coordinate conjunctions and punctuations in a sentence; i.e., the values q can potentially take on.

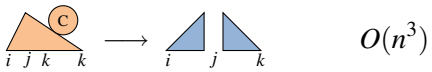
4.2 Rules for cascading conjuncts

In Hara et al. (2009), the plausibility score of a coordinate structure is defined in terms of the similarity between the head conjunct and each of the remaining conjuncts. In the Stanford basic dependencies, the head of a coordinate structure is the first conjunct by default. Thus, to enable computation similar to Hara et al., the end position of the first conjunct must be maintained throughout the

[2]Depending on the rules, indices q, r, s may not represent the exact position of punctuations or conjunctions, but their previous or succeeding positions.

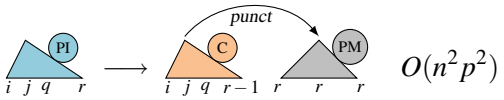
construction of dependency relations for a coordinate structure. This leads to an additional index associated with constituents, but in most cases, it is constrained by the position of punctuations and conjunctions.

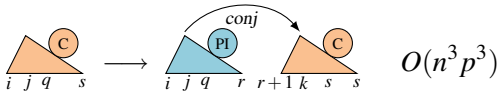
Let us introduce constituent CONJ (or C for short) to represent a conjunct. The first rule for CONJ simply groups the left and right dependents for a conjunct, as follows.

(7)  $O(n^3)$

As we explain shortly, CONJ is also used to represent a partially-built coordinate structure. The index k inside the span is used for maintaining the end of the first (head) conjunct, such that the similarity between the first and the subsequent conjuncts can be computed. However, in the above rule, the end position of the span is equal to the end of the conjunct; thus, k appears twice on the left-hand side (lhs) of the rule.

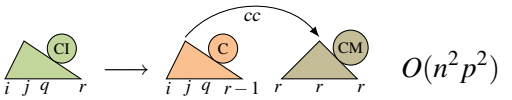
Rules (8)–(9) below deal with a series of conjuncts separated by PUNCTMARK, e.g., "*A*, *B*, *C*, ..." The new constituent, PUNCTINCOMPLETE (PI), represents the sequence CONJ PUNCTMARK.

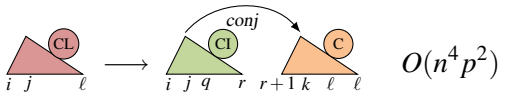
(8)  $O(n^2p^2)$

(9)  $O(n^3p^3)$

In these rules, q, the index for the end of the first conjunct[3] is inherited by their lhs, as mentioned earlier. With q at our disposal, we can compute the score for the derivation by (9), by taking into account the similarity between the first conjunct on the span (i,q) with head j, and the subsequent conjunct on $(r+1,s)$ with head k.

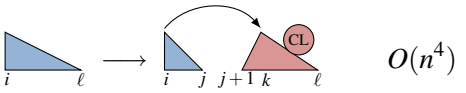
The following two rules terminate a coordinate structure, when CCMARK and the last CONJ are encountered. We introduce new constituents CCINCOMPLETE (CI) and COMPLETE (CL).

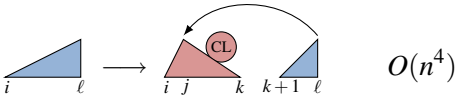
(10)  $O(n^2p^2)$

(11)  $O(n^4p^2)$

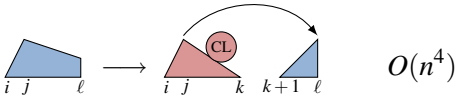
4.3 Interfacing complete coordination with outer parse trees

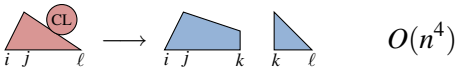
Rules (12)–(13) connect a completed coordination COMPLETE to the structures made with the standard Eisner-Satta rules.

(12)  $O(n^4)$

(13)  $O(n^4)$

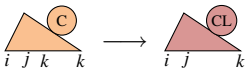
We also need rules to handle the case where COMPLETE takes a modifier.

(14)  $O(n^4)$

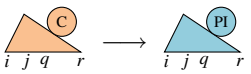
(15)  $O(n^4)$

The lhs of rule (14) has a new constituent, which can be regarded as a concatenation of COMPLETELEFT and INCOMPLETERIGHT in Eisner and Satta's rules. Notice also that although the lhs of rule (15) is represented by COMPLETE, it simply means a constituent having both left and right dependents, and loses the meaning of a coordination as a whole.

Furthermore, to deal with recursive coordinations, such as "(*A* & *B*) & *C*," we allow COMPLETE to be CONJ again.[4]

(16) 

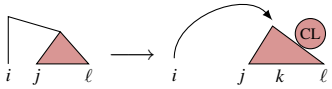
PUNCTINCOMPLETE is also allowed to be CONJ, to tolerate patterns like ", and" (note the comma preceding "and").

(17) 

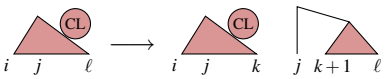
[3]Note that in rules (8)–(11), the index for the end of the first conjunct is denoted by q, meaning that it can take only $O(p)$ different values. This opposes rule (7) in which it was denoted by k, whose range is $O(n)$. This change owes to the fact that the end of the first conjunct is constrained by rules (8) or (10), which require the first CONJ to be followed by a PUNCTMARK or CCMARK; thus the end index for the first conjunct has only $O(p)$ possibilities.

[4]The computational complexity for unary rules is not shown, as it is assumed that their right-hand side is expanded and binarized before application of these rules, in which case the complexity is the same as those of the expanded rules.

There are also cases in which COMPLETE depends upon another COMPLETE, which occurs with a construct such as "$(V_1$ & $V_2)(N_1$ & $N_2)$," where V_i and N_i represent verbs and their objects (nouns), respectively.

(18) 

$O(n^4)$

(19) 

$O(n^4)$

Here, we used the "hook trick" (Eisner and Satta, 1999; Huang et al., 2005) to reduce the incurred time complexity from $O(n^5)$ to $O(n^4)$.

4.4 Time complexity

If we neglect the number p of punctuations and conjunctions in a sentence, parsing a sentence of length n requires $O(n^4)$ time, which is the same as that of Hara et al.'s (2009) coordination analysis method. For a sentence not containing coordinate conjunctions, the run time is $O(n^3)$.

4.5 A note on spuriousity

Rule (17) causes spurious ambiguity when there are exactly two conjuncts and a comma precedes a coordinate conjunction; e.g., "A, and B." In this case, the dependency between A and the comma can be made with not only rule (17), but also with the standard Eisner-Satta rules. Rule (16) also causes spuriousity in a similar situation. However, these spuriousities can be easily removed by imposing restrictions on rules (7) and (16), such that they are not applicable if $w(k)$ is a comma or semicolon. In the experiment in the next section, these restrictions are used.

5 Coverage of coordination rules

5.1 Experimental setup

We converted the WSJ part of PTB into Stanford basic dependencies[5], and verified the coverage of the proposed rules. Of the 43,948 sentences in WSJ sections 2–23, dependency arcs labeled *conj*, which indicate the presence of coordination, were contained in 40%, or 17,695 sentences.

Dependency structure for coordinations can also be derived with the standard Eisner-Satta rules, but we are only interested in the coverage of the proposed rules in WSJ. Hence we treated *conj*-labeled arcs as only derivable with the proposed rules, but not with Eisner-Satta. We thus essentially made Eisner and Satta's rules to fail to derive dependencies on all the sentences containing *conj*-dependencies, and evaluated the coverage of the proposed rules over these sentences. Except for this constraint on *conj*, we ignored all the dependency labels.

[5] we used the Stanford parser (converter) v. 3.5.2, with `-basic` and `-conllx` options to generate gold dependencies.

Count	Type	Description
7	R	multi-word *cc*
5	R	non-projective dependency
5	A	multiple *conj* arcs following *cc*
4	A	head of *punct* is not the first conjunct
3	A	*conj* on the left side of the head
1	R	*conj*s not separated by *cc* or *punct*
1	R	parenthesis (arc labeled *parataxis*)

Table 1: Causes of non-derivable dependencies in WSJ Section 22. We classified these into two types: Error type A is due to errors/inconsistencies in the gold annotation, and type R can be attributed to the deficiency for the proposed rules.

5.2 Result

Of the 17,695 sentences containing *conj*-labeled dependencies in WSJ sections 2–23, the proposed rules were able to derive correct dependencies for 94%, or 16,659 sentences.

Table 1 lists the types of failures of our parsing rules and their frequencies in WSJ section 22. About half of the failures were due to inconsistencies or errors in the gold annotations.

6 Conclusion

In this paper, we have proposed a set of dependency parsing rules specifically designed to handle coordination, to be used in combination with Eisner and Satta's rules. The new rules enable the parse scoring function to incorporate the syntactic and semantic similarity of conjuncts. While we are yet to implement such a scoring function, we analyzed the time complexity of the proposed rules and their coverage in the Penn Treebank.

Acknowledgments

This work was supported by JSPS Kakenhi Grant Nos. 24500193 (KH), 15H02749 (MS), and 26240035 (YM).

References

Marie-Catherine de Marneffe and Christopher D. Manning. 2008. Stanford typed dependencies manual. http://nlp.stanford.edu/software/dependencies_manual.pdf. Revised in April 2015.

Jason Eisner and Giorgio Satta. 1999. Efficient parsing for bilexical context-free grammars and head automaton grammars. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics (ACL '99)*, pages 457–464, College Park, Maryland, USA.

Atsushi Hanamoto, Takuya Matsuzaki, and Jun'ichi Tsujii. 2012. Coordination structure analysis using dual decomposition. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL '12)*, pages 430–438, Avignon, France.

Kazuo Hara and Masashi Shimbo. 2007. A discriminative learning model for coordinate conjunctions. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL '07)*, pages 610–619, Prague, Czech Republic.

Kazuo Hara, Masashi Shimbo, Hideharu Okuma, and Yuji Matsumoto. 2009. Coordinate structure analysis with global structural constraints and alignment-based local features. In *Proceedings of the 47th Annual Meeting of the ACL and the 4th IJCNLP of the AFNLP (ACL-IJCNLP '09)*, pages 967–975, Singapore.

Liang Huang, Hao Zhang, and Daniel Gildea. 2005. Machine translation as lexicalized parsing with hooks. In *Proceedings of the 9th International Workshop on Parsing Technology (IWPT '05)*, pages 65–73, Vancouver, British Columbia, Canada.

Mark Johnson. 2007. Transforming projective bilexical dependency grammars into efficiently-parsable CFGs with unfold-fold. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics (ACL '07)*, pages 168–175, Prague, Czech Republic.

Terry Koo and Michael Collins. 2010. Efficient third-order dependency parsers. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1–11, Uppsala, Sweden.

Sadao Kurohashi and Makoto Nagao. 1994. A syntactic analysis method of long Japanese sentences based on the detection of conjunctive structures. *Computational Linguistics*, 20(4):507–534.

Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Joakim Nivre, Laura Rimell, Ryan McDonald, and Carlos Gómez Rodríguez. 2010. Evaluation of dependency parsers on unbounded dependencies. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling '10)*, pages 833–841, Beijing, China.