DAG Conversion for Penn Treebank

陳宙斯　小町守
東京都立大学
{chen-zhousi@ed., komachi@}tmu.ac.jp

Abstract

Natural languages contain complicated structures beyond trees. Penn Treebank (PTB) [1] is one of the most frequently studied large-scale annotated phrase structure corpus in a format for continuous trees. It has a co-indexing system that allows structures beyond continuous trees [2]. This paper addresses a full conversion leveraging the co-indexing system to recover directed acyclic graphs (DAG) for PTB. Due to a similar annotation scheme, our DAG conversion is applicable to Chinese Treebank (CTB). However, much fewer DAGs from CTB are obtained.

1 Introduction

Constituency parsing is a task for phrases and their relations. Most research for this task is supervised, where parsers follow the annotation schemes of corpora. Many constituency corpora are annotated in bracketing format that limits the parsing structure to continuous trees, which are typically described in context-free grammars (CFG). Syntactically annotated corpora PTB, CTB, Keyaki Treebank (KTB) [3], NINJAL Parsed Corpus of Modern Japanese (NPCMJ) [4] and shared task SPMRL 2013 [5] contain abundant examples in bracketing format for multiple well-studied languages.

However, natural languages can hardly be described in CFG [6]. Enforcing CFG causes grammar exceptions and degrades parsing performance [7, 8]. Syntactic and semantic grounds foster complicated structures. On one hand, syntactic movements create discontinuity by moving phrases away from their siblings, such as "*what it is*" where fronting "*what*" makes the clause nominal with a discontinuous verb phrase "*is what*". Another example in Figure 1 shows a discontinuous conversion for PTB [9] which offers a new challenge where constituency is better described in linear context-free rewriting system (LCFRS) parsers. However, this work left some syntactic movements (e.g., gapping and part of right-node raising) unstudied [2].

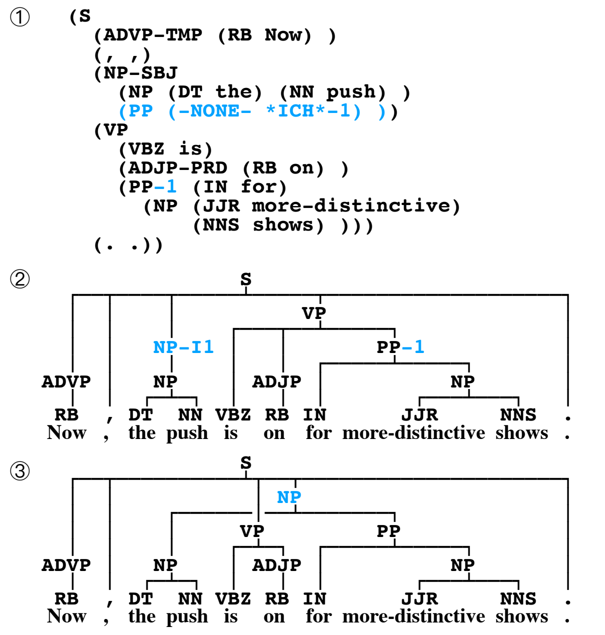
①
```
(S
  (ADVP-TMP (RB Now) )
  (, ,)
  (NP-SBJ
    (NP (DT the) (NN push) )
    (PP (-NONE- *ICH*-1) ))
  (VP
    (VBZ is)
    (ADJP-PRD (RB on) )
    (PP-1 (IN for)
      (NP (JJR more-distinctive)
        (NNS shows) )))
  (. .))
```

Figure 1　Recovering a discontinuous phrase from PTB with co-indexed traces (blue). Top: bracketing annotation; middle: visualization of the top; bottom: conversion after the middle.

On the other hand, semantic ground involves semantic frames (i.e., predicate-argument structure) which can also hardly be captured by CFG and even LCFRS. Semantic frames are not addressed in the conversion for discontinuous trees in PTB [9] because of the large amount of potential DAGs after full conversion. For example, the passive voice "*it is watched by John*" has a semantic frame watch(John, it) where "*John*" is the logical subject (i.e., syntactic object) and "*it*" is the logical object (i.e., syntactic subject) of predicate "*watch*". Syntactic and semantic relations jointly lead to the complex wiring of DAG.

In this paper, we fully leverage PTB's co-indexing system [2] to recover its DAG structures. Our work can be regarded as a continuum from [9], contributing another new challenging corpus via converting the treebank into a graphbank [1]. CTB is checked but needs further study.

1) See our code at https://github.com/tmu-nlp/UniTP.

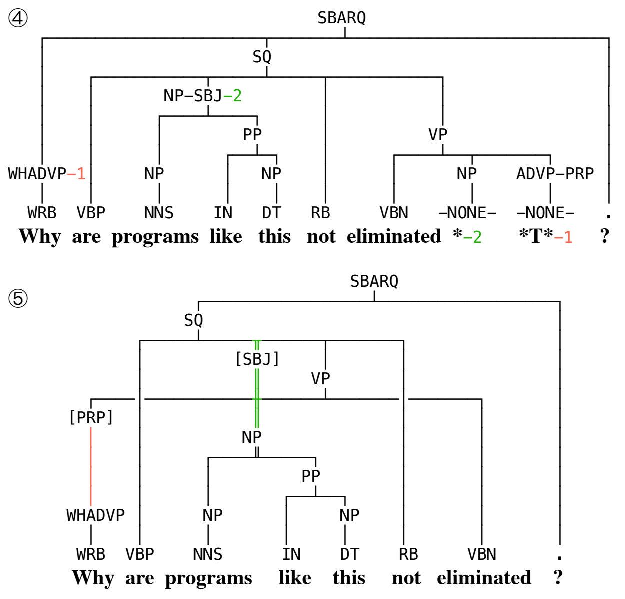
④
SBARQ
SQ
NP-SBJ-2
PP
VP
WHADVP-1
NP
NP
NP
ADVP-PRP
WRB VBP NNS IN DT RB VBN -NONE- -NONE- .
Why are programs like this not eliminated *-2 *T*-1 ?
⑤
SBARQ
SQ
[SBJ]
VP
[PRP]
NP
PP
WHADVP
NP
NP
WRB VBP NNS IN DT RB VBN .
Why are programs like this not eliminated ?

Figure 2 Recovering DAG from PTB with co-indexed traces (colored). Double lines denote multi-attachment. Top: visualization of bracketing annotation; bottom: conversion after the top.

2 Related Work

As a pioneering practice, English PTB selected the context-free bracketing format and later introduced the co-indexing system for non-context-free phenomena of discontinuity and multi-attachment [2]. Some corpora adopted the format with their own co-indexing systems for their languages [3, 4, 5], whereas other languages with free word order, such as German and Czech, either chose more expressive presentations for discontinuity (e.g., Export and XML formats [10, 11]) or adopted lexical dependency parsing instead of constituency.

The conversion of PTB for constituency has two directions. One direction is for discontinuity [9] by partially utilizing the co-indexing system; the other is for coordination [7, 8]. These two directions cover different aspects of the co-indexing system and still keep PTB tree-based. However, presenting PTB directly through DAG [2] leads to grammatical discontinuity and coordination.

Dependency parsing had addressed DAG [12] earlier, which has deeper semantic motivations (i.e., predicate-argument relations). Furthermore, semantic parsing directly addresses those semantic frames in DAG via logical expressions [13]. As a constituency corpus, PTB gets converted into dependency DAG parsing. To our knowledge, however, direct research on constituency DAG parsing is still lacking, probably because of lacking such a corpus.

Algorithm 1: DAG conversion with trace.

```
1  $T \leftarrow$ {*, *T*, *ICH*, *EXP*, *RNR*};
2  foreach sentence $s$ do
3    foreach node $c$ from $s$ co-indexed with null elements $N$ of
       trace types in $T$ do
4      foreach $n$ in $N$ do
5        attach $c$ to the grandparent of $n$ (i.e., destination $d$);
6        if $c$ properly dominates $d$ then
7          find the node labeled with PRN on the path from
             $c$ to $d$, call it $e$;
8          detach $e$ from its current parent;
9          if $c$ is not the root of $s$ then
10           attach $e$ to the parent of $c$;
11     if none of $N$ belongs to type * then
12       detach $c$ from its original parent;
13   go to Algorithm 2 for intra-sentential gapping;
14   delete all null elements;
15   delete all non-terminals without children;
16   remove all remaining indices from node labels;
```

3 Conversion

Our DAG conversion is achieved by modification of each original tree, which involves the reattachment of existing subtrees and the creation of new nodes. The process contains two algorithms. Algorithm 1 is extended from [9], which allows multi-attachment for the reattachment with co-indexed trace to null element and thus produces DAG. Algorithm 2 deals with intra-sentential gapping, where a phrase in coordination acts as a template for the other gapping phrases to imitate.

For the convenience of further study, we maintained all functional tags (e.g., -SBJ and -TMP) at either their original attachment or new reattachment locations.

3.1 Trace with Null Element

We consider five types of null elements for reattachment in PTB. Mark ✔ denotes a fully exploitation by [9], ▲ a partially exploitation, and ✘ no exploitation.

✘	*	*A-movement.*
✔	*T*	*A-bar movement.*
✔	*ICH*	*Interpret constituent here.*
✔	*EXP*	*Expletive.*
▲	*RNR*	*Right node raising.* ([9] picks only the closest null element for reattachment.)

We use an instance in Figure 2 to exemplify Algorithm 1. Node c in line 3 stands for each non-terminal node with a co-indexing number. For example, node NP-SBJ-2 re-

Algorithm 2: DAG conversion for gapping.

```
1  Function SHARE(template $t$, gapping $g$):
2      foreach not co-indexed direct child $r$ of $t$ do
3          if $r$ is a complement of $t$ and $r$ is not a child of $g$ then
4              attach $r$ to $g$;
5  Function IMITATE(template $t$, gapping $g$):
6      SHARE($t$, $g$);
7      foreach common co-index $c$ do
8          $c$ co-indexes $(i, j)$-th descendants $(t^i, g^j)$ of $(t, g)$;
9          if $i = j$ then
10             if $i = j = 1$ then
11                 continue;
12             else
13                 IMITATE($t^1$, $g^1$);
14         else
15             create a new child $\bar{g}$ for $g$ with the label of $t^1$;
16             remove $g^1$ from $g$ and attach $g^1$ to $\bar{g}$;
17             IMITATE($t^1$, $\bar{g}$);
18 if the only template phrase $t$ in sentence $s$ exists then
19     foreach corresponding gapping phrase $g$ in $s$ do
20         IMITATE($t$, $g$);
```

attaches to null element *-2 without detachment, creating a DAG by lines 11 & 12. Although null elements N contain only one *-2, N may contain multiple for DAG. Lines 6–10 are for canceling cycles created by PRN [9].

Among all five types, ***RNR*** and * are major sources for DAG with the loop in line 4 of Algorithm 1. Specifically, ***RNR*** and * differ in detachment. Type ***RNR*** detaches node c from the original parent for reattachment, while type * keeps the original syntactic attachment during reattachment in lines 11 & 12. Following [9], the reason for detachment of all non-* nodes is that their parent is a pseudo-attachment that does not have any syntactic role. In contrast, non-* nodes do have [2]. Besides the above differences, our improvement includes lines 11–13 for gapping.

3.2 Intra-sentential Gapping

PTB co-indexes each coordination with either "-" for a complete phrase (i.e., a template usually at the initial place) or "=" for one or more incomplete phrases (i.e., gapping with remnant components) to imitate the template. The template and gapping phrases are commonly conjunct in no more than one coordination structure in a sentence.

Algorithm 2 has a recursive IMITATE function making each gapping phrase resemble its template. In PTB, gapping phrases must have incomplete structures of small heights. Each IMITATE entrance lets a gapping phrase regain one level of missing complements from the template via the SHARE function. The missing structures are recovered by lines 15 & 16 by forcing co-indexed nodes to grow to the same height with the same structure.

	Phrase Level: Number of Parents						Sentence Level	
Type	**1**-ary	**2**-ary	**3**-ary	**4**	**5**	**6**	Reatt.	DAG
T	15,837	71	3	-	-	-	13,584	82
***T*-PRN**	840	2	-	-	-	-	842	4
ICH	1,265	2	-	-	-	-	1,239	3
RNR	2	203	5	1	-	-	209	210
EXP	657	1	-	-	-	-	651	1
*****	13	15,590	1,906	217	22	8	15,992	17,756
Gapping	372	457	36	8	-	4	311	534
Total	55,226 reatt. of 918,730						26,164	16,324

Table 1 Statistics of PTB after our DAG conversion. PTB (2.0) has in total of 49,208 parsed sentences.

We exemplify the process in Figure 3. At first, S for *Dean Writer new recommends 80 %* is found to be the template t with S for *Goldman 65 %* being an imitator phrase. The modifier ADVP is not shared to the imitator because it is not a complement for the formation of S. Then, a new substructure VP (in yellow) is created by lines 14 & 15 and the second IMITATE entrance creates two VPs in two respective S nodes. In contrast, VBZ is shared by VPs because it is the complement for them. The process works in a top-down style for every substructure paired with the template and gapping phrases.

3.3 Error and Exception

The annotation of PTB involved a large amount of human labor. Thus, a few errors and exceptions are inevitable. During our re-implementation of [9], we manually adjusted the annotation of 26 sentences for reproducibility.

For our DAG conversion, we observed 10 additional errors and exceptions, which involve incorrect trace types, co-indices that lead to circles, ill-formed coordination, and ***EXP*** coming across gapping. Nevertheless, their population is relatively small and we provide our solution for each of them in our code.

3.4 Adjustment for CTB

CTB uses a similar annotation style to PTB. However, there are certain differences:

- Fewer trace types: ***T***, ***RNR***, and *.
- Templates are also marked by "=".

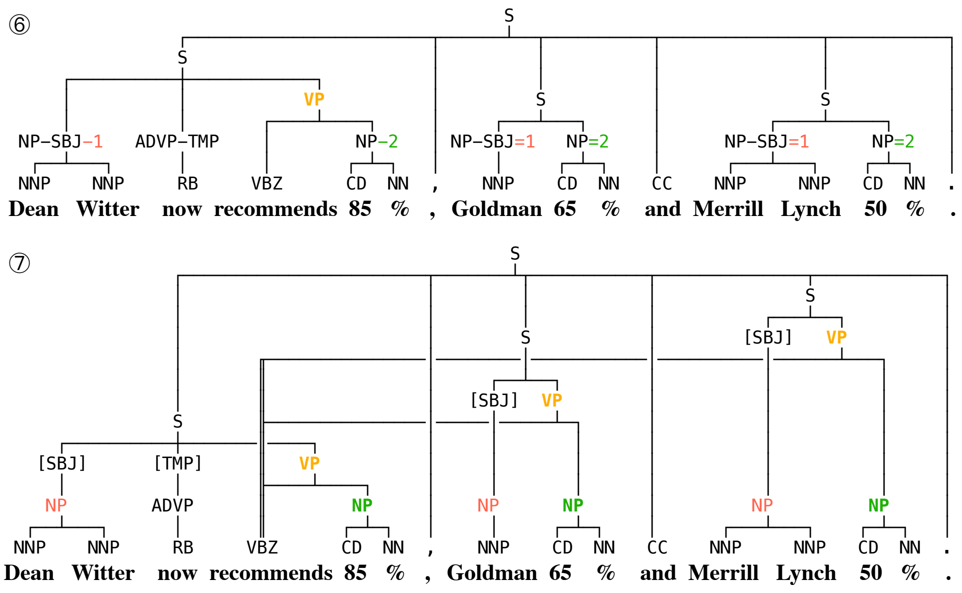
⑥
S
S
VP
NP-SBJ-1
ADVP-TMP
NP-2
NP-SBJ=1
NP=2
S
NP-SBJ=1
NP=2
NNP NNP RB VBZ CD NN , NNP CD NN CC NNP NNP CD NN .
Dean Witter now recommends 85 % , Goldman 65 % and Merrill Lynch 50 % .
⑦
S
S
[SBJ] VP
S
[SBJ] VP
S
[SBJ] [TMP] VP
NP ADVP NP
NP NP
NP NP
NNP NNP RB VBZ CD NN , NNP CD NN CC NNP NNP CD NN .
Dean Witter now recommends 85 % , Goldman 65 % and Merrill Lynch 50 % .

Figure 3 Recovering DAG from PTB with co-indexed intra-sentential gapping (colored).

Type	Phrase Level: Number of Parents					Sentence Level	
	1-ary	**2**-ary	**3**-ary	**4**-ary	≥**5**-ary	Reatt.	DAG
T	3,214	36	6	-	-	3,132	43
ICH	26	-	-	-	-	24	0
RNR	999	48	7	1	1	989	58
*	2	2,971	55	4	1	2,877	3,031
Gapping	-	16	7	1	1	25	25
Total	10,426 reatt. of 2,476,071					6,883	2,992

Table 2 Statistics of CTB after our DAG conversion. CTB (9.0) has in total of 132,080 parsed sentences.

Thus, Algorithm 1 is still applicable for CTB but Algorithm 2 needs adjustment to first decide which phrase is the template. Our decision is the conjunct phrase that possesses the tallest height and thus contains the complete structure to imitate. Sometimes, the results can indicate some templates on the right side of the coordination.

4 Discussion

Statistics. As a new graphbank, we show the statistics of a number of parents and their conversion types in Tables 1 & 2. Discontinuous tree conversion [9] can be seen as a subset that only has the 1-ary column of each table. For both DAG conversions of PTB and CTB, types * and ***T*** are the major sources of reattachment, while *, **gapping**, and ***RNR*** are the top three DAG provider. However, both reattachment and DAG are still sparse, especially for CTB, which makes DAG constituency parsing very challenging.

For CTB, we did not leverage the potential null element of type ***pro*** (i.e., dropped subject or object) because it does not contain any co-index. We left the further DAG conversion for CTB as a future study.

Comparison to combinatory categorical grammar. As a parsing formalism, combinatory categorical grammar (CCG) also addressed gapping phenomena in coordination [14]. The solution of CCG is through compound lexical tags (i.e., supertagging) operated by combinatory rules, whereas ours is not a parsing formalism but a graphbank in DAG with each node being an atomic constituency label.

Specifically, CCG assigns each word with a string of constituency labels and operating slashes that define how the word combines with adjacent words to form a parse. Apart from CCG's ability for gapping, its parsing formalism is context-free by assuming that necessary contextual information is locally encoded into each lexical string. Our graphbank simply provides the DAG structure without assumption. Moreover, operating slashes can only handle binary combinatory operations, in contrast to our conversion's flexible number of child phrases.

5 Future Work

We proposed a DAG conversion approach for PTB one of the most frequently studied large-scale treebank, by fully leveraging its co-indexing trace system. However, we observed much fewer DAGs from CTB probably because of uncovered phrases without co-indexing. Nevertheless, DAGs in many other treebanks are to be recovered by carefully applying the instruction in their guidelines.

Acknowledgement

References

[1] Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a Large Annotated Corpus of English: The Penn Treebank. **Computational Linguistics**, 19(2):313–330, 1993.

[2] Mitchell P. Marcus, Grace Kim, Mary Ann Marcinkiewicz, Robert MacIntyre, Ann Bies, Mark Ferguson, Karen Katz, and Britta Schasberger. The Penn Treebank: Annotating Predicate Argument Structure. In **Human Language Technology, Proceedings of a Workshop**, page 114–119. Morgan Kaufmann, 1994.

[3] Alastair Butler, Tomoko Hotta, Ruriko Otomo, Kei Yoshimoto, Zhen Zhou, and Hong Zhu. Keyaki Treebank: phrase structure with functional information for Japanese. In **Proceedings of Text Annotation Workshop, 2012**, 2012.

[4] National Institute for Japanese Language and Linguistics. NINJAL Parsed Corpus of Modern Japanese, 2016. http://npcmj.ninjal.ac.jp/.

[5] Djamé Seddah, Reut Tsarfaty, Sandra Kübler, Marie Candito, Jinho D. Choi, Richárd Farkas, Jennifer Foster, Iakes Goenaga, Koldo Gojenola Galletebeitia, Yoav Goldberg, Spence Green, Nizar Habash, Marco Kuhlmann, Wolfgang Maier, Joakim Nivre, Adam Przepiórkowski, Ryan Roth, Wolfgang Seeker, Yannick Versley, Veronika Vincze, Marcin Wolinski, Alina Wróblewska, and Éric Villemonte de la Clergerie. Overview of the SPMRL 2013 Shared Task: A Cross-Framework Evaluation of Parsing Morphologically Rich Languages. In **Proceedings of the Fourth Workshop on Statistical Parsing of Morphologically-Rich Languages**, pages 146–182. Association for Computational Linguistics, 2013.

[6] Geoffrey K. Pullum and Gerald Gazdar. Natural Languages and Context-Free Languages. **Linguistics and Philosophy**, 4(4):471–504, 1982.

[7] Jessica Ficler and Yoav Goldberg. Improved Parsing for Argument-Clusters Coordination. In **Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics**, 2016.

[8] Jessica Ficler and Yoav Goldberg. Coordination Annotation Extension in the Penn Tree Bank. In **Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics**, page 834–842, 2016.

[9] Kilian Evang. Parsing Discontinuous Constituents in English. **Mémoire de master, University of Tübingen**, 2011.

[10] Sabine Brants, Stefanie Dipper, Peter Eisenberg, Silvia Hansen-Schirra, Esther König, Wolfgang Lezius, Christian Rohrer, George Smith, and Hans Uszkoreit. TIGER: Linguistic Interpretation of a German Corpus. **Research on language and computation**, 2(4):597–620, 2004.

[11] Wojciech Skut, Thorsten Brants, Brigitte Krenn, and Hans Uszkoreit. A Linguistically Interpreted Corpus of German Newspaper Text. **CoRR**, cmp-lg/9807008, 1998.

[12] Kenji Sagae and Jun'ichi Tsujii. Shift-Reduce Dependency DAG Parsing. In **22nd International Conference on Computational Linguistics, Proceedings of the Conference**, pages 753–760, 2008.

[13] Federico Fancellu, Sorcha Gilroy, Adam Lopez, and Mirella Lapata. Semantic Graph Parsing with Recurrent Neural Network DAG Grammars. In **Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing**, pages 2769–2778, 2019.

[14] Nathaniel Little. Reevaluating Gapping in CCG: Evidence from English & Chinese. In **Proceedings of the 10th International Workshop on Tree Adjoining Grammar and Related Frameworks**, pages 25–34, 2010.