# Shift-reduce parser における pp attachment の disambiguation *

3P−4

小野晋　西澤信一郎　中川裕志

横浜国立大学 工学部

## 1 はじめに

英語の文章を他の言語に翻訳する時に構文解析が必要であるが、このときの重大な問題の一つに pp attachment の ambiguity の問題がある。この問題は文脈に依存しており syntactic に解決することは厳密には不可能である。本稿では、shift-reduce parser における pp attachment の ambiguity の解決に文脈情報を利用する方法を提案する。

## 2 shift-reduce parser の問題点

以下のような CFG があるとする。

s → np vp　　np → art n　　np → n　　np → np pp

vp → vct np　　vp → vp pp　　pp → p np

この CFG を用いて shift-reduce parser[1] の state と arc を作り次のような文を解析することを考える。PS は parser stack で、IS は input stack とする。

(1) I read the book about economy.

従来の shift-reduce parser で (1) の文を解析すると、

| PS | IS | action |
|---|---|---|
| (st(0)) | (I read ⋯) | |
| (st(5),n,st(0)) | (read the book ⋯) | reduce |
| (st(0)) | ([np] read the book ⋯) | shift |

となった後、

| PS | IS | action |
|---|---|---|
| (st(0)) | ([np] read the book ⋯) | shift |
| (st(1),np,st(0)) | (read the book ⋯) | shift |

とするか

| PS | IS | action |
|---|---|---|
| (st(0)) | ([np] read the book ⋯) | shift |
| (st(8),np,st(0)) | (read the book ⋯) | fail |

とするかで conflict が発生する。というのは、st(0) は

st(0):[s → * np vp]　[np → * art n]

[np → * n]　[np → * np pp]

で、st(1) は

st(1):[s → np * vp][vp → * vct np][vp → * vp pp]

で、st(8) は

st(8):[np → np * pp][pp → * p np]

となっているからである。np の後に pp が来ない場合は前者のように shift すべきで、そうでない場合 (例えば 'I' の部分が 'The man of science' である場合) は後者のように shift すべきである。

(1) の文をさらに解析すると、

| PS | IS | action |
|---|---|---|
| (st(6),vct,st(1), ⋯) | ([np] about economy.) | shift |

となり、再び conflict が発生する。この文の場合は、

| PS | IS | action |
|---|---|---|
| (st(6),vct,st(1),np, ⋯) | ([np] about ⋯) | shift |
| (st(8),np,st(6),vct, ⋯) | (about economy.) | shift |

とすべきであるが、もし 'about economy' の部分が 'on Sunday' だったら、

| PS | IS | action |
|---|---|---|
| (st(6),vct,st(1), ⋯) | ([np] on ⋯) | shift |
| (st(7),np,st(6),vct,st(1), ⋯) | (on Sunday.) | reduce |

とすべきである。このとき、st(7) は、

st(7):[vp → vct np *]

である。この conflict は syntactic には解決できない。

この問題は左再帰 rule により起こる。しかし、左再帰 rule を避けるように CFG の句構造規則を作ると規則の数がより多くなってしまう。また、そうしたとしても pp の直前まで解析が終った時点で shift-reduce conflict(shift も reduce も可能な状態) は発生する。syntactic な解析だけで正しい解析結果を得るためには、ambiguity に対して並列的な処理をすることになるが、その場合、多くの計算の無駄が生じる。

## 3 shift-reduce parser の改良

shift-reduce parser の state の作り方を一部変更することで、左再帰 rule が CFG に含まれていても conflict が発生しないようにする方法がある。この章ではこれを説明した後、pp attachment の ambiguity の解決のために、文脈情報を parser に取り込む方法を提案する。

a,b は np,vp などの構成素で、$\alpha,\cdots,\zeta$ は構成素が複数 (0 個でもよい) 並んだものとする。従来の方法で state を作ると $i_m = [a \to \alpha b \beta * \gamma]$ というアイテムと $[b \to b\beta * \zeta]$ というアイテムは別な状態であるが、これらを同じ状態となるように state をつくるようにすれば、左再帰 rule による conflict は発生しなくて済む。

例えば、(1) の文の解析では、

| PS | IS | action |
|---|---|---|
| (st(0)) | (I read ⋯) | shift |
| ⋮ | | |
| (st(1),np,st(0)) | (read the ⋯) | shift |

となり、[np] に対して shift は一通りである。というのは

st(1):[s → np * vp]　[vp → * vct np]

[vp → * vp pp]　[np → np * pp]

[pp → * p np]

---

*Disambiguation of PP Attachment Problem in Shift-Reduce Parser

Susumu ONO , Shinichiroh NISHIZAWA and Hiroshi NAKAGAWA Faculty of Engineering , Yokohama National University

[1]shift-reduce parser は文献 [1] の形式のものを用いる。

となっているからである。(1) の 'I' の部分が 'The man of science' であるような文では、

| PS | IS | action |
|---|---|---|
| (st(0)) | ([np] of science read ···) | shift |
| (st(1),np,st(0)) | (of science read ···) | shift |

となり、主語の部分においては ambiguity が発生しない。

'read' の目的語の部分の解析においては、やはり ambiguity が生じる。(1) の文では、

| PS | IS | action |
|---|---|---|
| (st(7),vct,st(1), ···) | ([np] about ···.) | shift |
| (st(8),np,st(7),vct, ···) | (about economy.) | S or R |

となる。ここで、shift-reduce conflict(表では S or R と書くことにする) を生じる。この後、(1) の文では shift して 'the book' と 'about economy' をまとめて np にしなければならないが、'about economy' の部分が 'on Sunday' だとすると、reduce して 'read the book' で vp を作り、この vp と 'on Sunday' で vp を作るようにすべきである。

ところが、ここで (1) の文は、shift すると、

| PS | IS | action |
|---|---|---|
| (st(8),np,st(7),vct, ···) | (about ···.) | S or R |
| (st(11),p,st(8),np, ···) | (economy.) | shift |
| (st(5),n,st(11),p, ···) | (.) | reduce |
| (st(11),p,st(8),np, ···) | ([np].) | shift |
| (st(12),np,st(11),p, ···) | (.) | reduce |
| (st(8),np,st(7),vct, ···) | ([pp].) | S or R |

となる。そして、この状態で、再び、shift-reduce conflict が発生する。しかし、(1) の文で、

| PS | IS | action |
|---|---|---|
| (st(8),np,st(7),vct, ···) | (about economy.) | S or R |

と違い、

| PS | IS | action |
|---|---|---|
| (st(8),np,st(7),vct,st(1), ···) | ([pp].) | S or R |

では、すでに 'about economy' が p np になりさらに pp になっている。もし、この時に 'about economy' の意味的情報とここまでの文脈を用いれば、正確なバージングを行なうことができるはずである。この文では、名詞句の後に about に導かれる前置詞句が来た場合、これは adnominal になる可能性が大きいという英語の性質が使えるので、この後 shift すれば良いことが分かる。この後は、shift-reduce conflict を生ずることなく最後まで解析される。

'about economy' の部分が 'on Sunday' だとすると、read という動作動詞と前置詞句 'on Sunday' が時点を表すことから前置詞句は adverbial であると推定でき、reduce すれば良いことが分かる。その後の 'on Sunday' の pp が adverbial になることで、この後に出てくる shift-reduce conflict を解決できる。

動詞の後に pp の shift-reduce conflict が発生した時、一般にどのように shift-reduce parser を操作するかを説明する。そのために、pp と p についての '性質' を定義する。つまり、IS の先頭に pp または p が入って shift-reduce conflict が発生したときに [st(c),vp, ···] で IS が [[pp] ···] でない限り reduce すべきであるような pp または p には adverbial な性質があると定義する。この性質は、最初はどの pp にも p にもなく、意味の分析の結果 adverbial な pp または pp の head の p に対して与えられる。

前置詞句による shift-reduce conflict が発生した時のための以下のようなアルゴリズム PPD を提案する。

[PPD]

1. もし p に adverbial な性質があれば、reduce して、その後の分析を続ける。
2. もし p に adverbial な性質がなければ、前置詞で shift して、その後の分析を続ける。
3. PS が [st(b),np,st(a),p, ···] となったら、この前置詞句が adverbial か adnominal かを検討する。
   (a) もし shift-reduce conflict だったら、
      i. もし [st(b),np,st(a),p, ···] が必ず adverbial とはならないならば、PPD を実行する。
      ii. もし [st(b),np,st(a),p, ···] が必ず adverbial ならば、shift-reduce conflict のあったところまで逆向きに shift し、p の性質が adverbial とし、PPD を実行する。
   (b) もし reduce だったら、reduce して pp を生成し、この PP が adverbial か adnominal かを決める。
      i. もし pp が adverbial ならば、pp の性質が adverbial とし、PPD を実行する。
      ii. もし pp が adnominal ならば、shift して、その後の分析を続ける。

ここで、st(a),st(b),st(c) は適当な状態とする。

## 4 終りに

本稿では、shift-reduce parser において、pp attachment の ambiguity を解決するために文脈情報を取り込む方法を提案した。文脈情報をどのように使って ambiguity を解決するかは、今後の課題である。

## 参考文献


[1] James Allen : NATURAL LANGAGE UNDERSTANDING , 1988

[2] 三好秀夫 : 場所格表現の統語的／意味的分析 , ソフトウェア科学会全国大会 , 1991

[3] Masaru Tomita : An Efficient Augmented-Context-Free Parsing Algorithm , Computational Linguistics Vol.13 no.1-2 January-June 1987