

依存関係を用いた漸進的構文解析の効率化

村瀬 隆久^{†1} 松原 茂樹^{†2†4} 外山 勝彦^{†3†4} 稲垣 康善^{†3}

^{†1}名古屋大学工学部 ^{†2}名古屋大学言語文化部 ^{†3}名古屋大学大学院工学研究科

^{†4}名古屋大学統合音響情報研究拠点 (CIAIR)

{murase,matu,toyama,inagaki}@inagaki.nuie.nagoya-u.ac.jp

1 はじめに

即時応答機能を備えた対話処理システムを実現するためには、話者による入力に従って順次、解析を行う漸進的解析手法が必要である。漸進的に解釈するためには、語の入力毎にそれまでの入力に対する解析木を随時作成する必要がある。それを実現するための手法として漸進的チャート解析法が提案されている [2]。しかし、この手法では、未入力部分の予測をもとに解析木を作成するため、1 文単位の解析手法に比べて、作成される解析木数が増加し、さらに解析処理時間が増大するという問題が生じる。

そこで本稿では、文脈自由文法を用いた構文解析と、意味情報を用いた構文的曖昧性解消を、漸進的かつ同期的に行うことによって解析処理時間を抑える手法を提案する。本手法では、あらかじめ獲得した単語間の依存関係情報を、漸進的チャート解析における文法適用可能性に関する制約として利用することにより、意味的に不適当な解析木の作成を回避し、漸進的構文解析を効率化できる。

これまでにも、意味情報を制約として用いることによって、意味的に不適当な部分解析木を削除する構文解析手法が提案されている [3, 4]。一方、本手法では漸進的チャート解析によって、随時、それまでの入力に対する解析木を作成するため、解析木全体に対して意味情報を活用することができる。

本稿の構成は以下の通りである。2節では、漸進的チャート解析とその効率化について述べる。3節では、依存関係情報を用いた漸進的チャート解析法を提案する。4節では、本手法を用いた解析実験の結果について報告する。

2 漸進的構文解析とその効率化

2.1 漸進的チャート解析

文脈自由文法に対する構文解析手法の一つである漸進的チャート解析 [2] では、解析途中の結果をチャート [1] と呼ばれるグラフで表現する。チャートの節点は、入力文の単語と単語の間に付けられた番号であり、2つの節点 $i-1, i$ の間は入力文の左から i 番目の入力単語に対応する。節点間に張られる弧には、項と呼ばれる構文構造がラベルとして付けられる。 X を範疇とすると項 $[?]_X$ を未決定項と呼ぶ。特に項の中の最も左に位置する未決定項を最左未決定項と呼ぶ。未決定項を含む項をラベルとしてもつ弧を活性弧と呼び、そうでない弧を不活性弧と呼ぶ。

漸進的チャート解析では、項 $[?]_S$ をもつ弧が節点 $0, 0$ の間に張られたチャートを初期状態として、 i 番目の語 w_i が入力されたとき以下の手続きを順に実行する。

- a) 辞書引き 語 w_i の範疇が A ならば、項 $[w_i]_A$ をラベルにもつ弧をチャートの節点 $i-1, i$ の間に追加する。
- b) 文法規則の適用 チャートの節点 $i-1, i$ の間に張られた弧の項が $[\dots]_{A_1}$ であり、かつ、文法規則 $A \rightarrow A_1 A_2 \dots A_n$ が存在するならば、項 $[[\dots]_{A_1} [?]_{A_2} \dots [?]_{A_n}]_A$ をラベルとしてもつ弧をチャートの節点 $i-1, i$ の間に追加する。この操作を可能な限り繰り返す。
- c) 項の置き換え チャートの節点 $0, i-1$ の間に張られた弧の項 σ の最左未決定項を $[?]_X$ とする。このときチャートの節点 $i-1, i$ を結ぶ弧の項 τ の範疇が X ならば、 σ の最左未決定項を τ で置き換えた項をラベルとしてもつ弧を、チャートの節点 $0, i$ の間に追加する。

漸進的チャート解析では、語が入力される毎にそれまでの入力に対する解析木を作成することができ、漸進的解釈のための構文解析手法として適している。

2.2 漸進的構文解析の効率化

漸進的構文解析の結果、一般に、複数の解析木が作成されるが、これは、その文に含まれる構文的曖昧性に起因する。構文的曖昧性は、文中で語と語の間の依存関係、すなわち、修飾、非修飾関係が複数考えられるときに発生する [3]。図 3 に示した文法と辞書を用いて、英語文

Ken studied English at school yesterday. (1)

を漸進的チャート法に従って解析すると、語 “at” が入力された時点で、図 1、図 2 に示す 2 つの解析木が生成される。図 1 の解析木は “at” が “studies” に構造上依存することを、図 2 の解析木は “at” が “English” に構造上依存することをそれぞれ表している。しかし、意味的には一般に、“at” は “studies” に依存し得るが、“English” に依存することはないと考えられる。よって、図 2 の解析木は、構文的には適当であるが、意味的には不適当であるといえる。

作成された解析木において、その中の語と語の間の依存関係が常に成り立つとは限らない。正しくない依存関係を含むような、つまり、意味的に不適当な解析木の作成を回避することにより解析を効率化できる。

3 依存関係情報を用いた漸進的チャート解析

前節で述べたように、漸進的構文解析を効率化するために、意味情報を用いた構文的曖昧性解消を早い段階で行うことによって、作成される解析木の数を削減する方法が考えられる。本節では、単語間の依存関係情報を、漸進的チャート

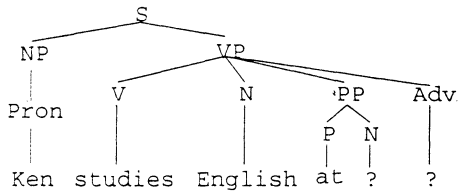


図 1: 英語文(1)の解析木 1

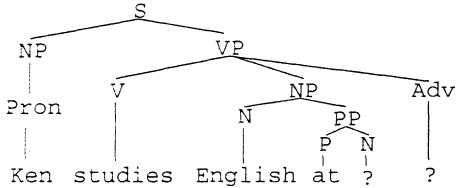


図 2: 英語文(1)の解析木 2

解析における文法適用可能性に関する制約として用いることによって不適当な解析木の作成を回避する手法について述べる。本手法では、解析木全体に対して、構造上定まる語と語の間の依存関係を決定し、それらが意味的に適当であるかどうかを、あらかじめ獲得した単語間の正しい依存関係情報を用いることによって調べる。意味的に不適当であればその解析木を削除する。

以下、3.1節では、解析木から構造上成り立つべき依存関係を決定する方法について述べ、3.2節では、構造上の依存関係と意味的に正しい依存関係情報を用いて解析木を削除する手法について述べる。

3.1 文法と項への制約

本手法で用いる文法は文脈自由文法であり、各文法規則の右辺の範疇列の中には、記号*が付与された範疇がただ一つ存在するものとする。以下では、そのような範疇を主辞と呼ぶ。また、右辺の主辞でない範疇を補語と呼ぶ。それらの間には、補語が主辞に依存するという依存関係を定めることができる。

また、文法規則

$$X \rightarrow Y_1 \cdots Z^* \cdots Y_n \quad (2)$$

の適用により、項

$$[[\cdots]_{Y_1} [\cdots]_{Z^*} \cdots [\cdots]_{Y_n}]_X \quad (3)$$

が作られるとき、 $[\cdots]_{Z^*}$ を項(3)の主辞項と呼ぶ。

定義 1 (語彙的主辞) 語彙的主辞は、次の 1. ~ 3. のいずれかである。

1. w_i を語、 X を範疇とする。このとき、項 $[w_i]_X$ の語彙的主辞は w_i である。

文法	辞書
S → NP VP	Pron → Ken
NP → Pron	N → English / school
NP → N PP	V → studies
PP → P N	P → at
VP → V NP Adv	Adv → yesterday
VP → V N PP Adv	

図 3: 構文解析のための文法と辞書(1)

2. 未決定項 $[?]_X$ の語彙的主辞は“?”である。
3. $\alpha_i = [\cdots]_{A_i} (i = 1, \dots, n)$ とする。文法規則 $A \rightarrow A_1 \cdots A_i^* \cdots A_n$ が存在し、 α が項 $[\alpha_1 \cdots \alpha_i \cdots \alpha_n]_A$ であるとき、 α の語彙的主辞は主辞項 α_i の語彙的主辞である。

語彙の主辞は、その項の中心的意思を担う語であり、すべての項は語彙の主辞を持つ。

あらかじめ主辞と補語が定められた文法規則を用いて作成された解析木では、構成要素間の依存関係を求めることができる。また、各構成要素には語彙的主辞が存在する。構成要素間の依存関係は、語彙的主辞間においても成立するため、語彙的主辞間の依存関係が構造上定まる。すなわち、文法規則(2)の適用によって作られた項(3)において、 $[\cdots]_{Y_j} (1 \leq j \leq n)$ の語彙的主辞は $[\cdots]_Z$ の語彙的主辞に依存する。

3.2 依存関係情報を用いた漸進的チャート解析

本手法では、解析木から定まる構造上の依存関係が、あらかじめ獲得した単語間の意味的な依存関係情報と適合するかどうかを、漸進的チャート解析における項の置き換え操作の時点で調べることにし、解析木を削除する。

漸進的チャート解析では、項の置き換え操作によって解析木を作り上げていくため、置き換え操作の実行可能性に対して条件を追加することにより、不適当な解析木の作成を回避することができる。例えば、図4において、項 α の最左未決定項 γ を項 δ で置き換えたこととすると、図5が示す依存関係が構造上成り立つことになる(項 β_i の語彙的主辞を w_{β_i} 、置き換えられた項 δ の語彙的主辞を w_δ とする。)。よって、この依存関係が意味的に成り立つかどうかを、置き換え操作前に調べることにし、項の置き換え可能性を判断できる。図4では、項 α の最左未決定項 γ が主辞項であるため、語彙的主辞 $w_{\beta_i} (1 \leq i \leq m)$ が語彙的主辞 w_δ に構造上依存することが、意味的に成り立つかを調べる。また図6において、項 α の最左未決定項 γ を項 δ で置き換えたこととすると、図7が示す依存関係が構造上成り立つことになる。図6では、項 α の最左未決定項 γ が主辞項 β_n より右に位置するため、語彙的主辞 w_δ が語彙的主辞 w_{β_n} に構造上依存することが、意味的に成り立つかを調べる。

すなわち2.1節のc)で述べた置き換え操作に対して以下の追加条件を加え、それを満たすならばc)を実行し、解析木を作る。

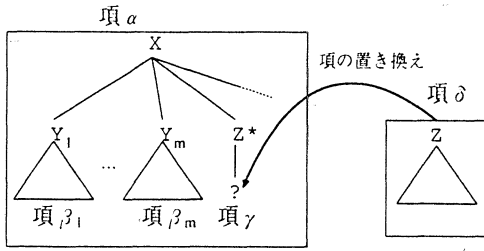


図 4: 置き換え操作 1

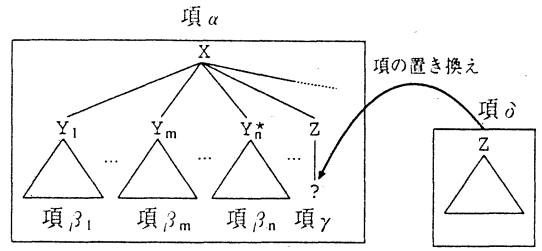


図 6: 置き換え操作 2

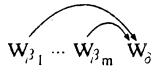


図 5: 構造上の依存関係 1

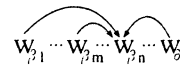


図 7: 構造上の依存関係 2

追加条件 τ の語彙の主辞が“?”である。または、あらかじめ獲得した語と語の間の正しい依存関係情報を用いて、以下のいずれかのケースを満たす。ただし、 $w_a \Rightarrow w_b$ で w_a が w_b に依存することを表すこととする。また σ の最左未決定項 $[?]_X$ を δ とし、項

$$\rho = [\alpha_1 \cdots \alpha_{l-1} \delta \alpha_{l+1} \cdots \alpha_n]_A \quad (\delta = \alpha_l)$$

の主辞項を α_h ($1 \leq h \leq n$) とする。さらに、項の列

$$\{\alpha_1 \cdots \alpha_{l-1} \delta \alpha_{l+1} \cdots \alpha_n\} \quad (\delta = \alpha_l)$$

を L とする。

ケース 1) L において、 $1 \leq h \leq l-1$ であるとき、 τ の語彙の主辞 w_τ と、 α_h の語彙の主辞 w_{α_h} との間に依存関係情報 $w_\tau \Rightarrow w_{\alpha_h}$ がある。

ケース 2) L において、 $h = l$ であるとき、 τ の語彙の主辞 w_τ と、 α_i の語彙の主辞 w_{α_i} ($i = 1, \dots, l-1$) との間に依存関係情報 $w_{\alpha_i} \Rightarrow w_\tau$ ($i = 1, \dots, l-1$) があるならば、 $\delta := \rho$ とし、項の列 L

$$\{\alpha_1 \cdots \alpha_{l-1} \delta \alpha_{l+1} \cdots \alpha_n\} \quad (\delta = \alpha_l)$$

に対して、追加条件を満たす。

ケース 3) L において、 $l+1 \leq h \leq n$ である。

3.3 解析例

英語文 (1) に対する解析例は以下の通りである。なお、ここでは、図 9 に示した文法と辞書を用いる。図 8 に示すように、語 “at” が入力された時点で、(a) と (b) の間で項の置き換えを行うと、“at” が “English” に依存するという構造上の依存関係が成り立つことになる。このとき、項

[[English] $_N$ ·[?]PP]において、最左未決定項 [?]PP は主辞項 [English] $_N$ の左に位置するため、追加条件の中のケース 3 により、項 [English] $_N$ の語彙の主辞 “English” と、項 [[at] $_P$ ·[?]N]PP の語彙の主辞 “at” の依存関係について調べる。このとき、あらかじめ獲得した単語間の依存関係情報から、“at” は “English” に意味的に依存しないと判断されるなら、(a) と (b) の間で項の置き換え操作を行わないので、図 2 の解析木を生成しない。

4 実験

本稿で提案した手法の有効性を評価するために、それを UltraSparcII ワークステーション (メモリ 512MB, CPU 248MHz) 上に、GNU Common Lisp 2.2.2 を用いて実装し、解析実験を行った。実験では、文献 [7] に収録された辞書 (438 語)、文法 (224 規則、非終端記号数 93)、及び、英語文 40 文 (平均語数 12.2 語) を用いた。各文における単語間の正しい依存関係は、あらかじめ人手で付与した。実験では、1 単語入力毎の、生成された解析木数、及び、解析処理時間について、従来の漸進的チャート法と依存関係を用いた漸進的チャート法との間で比較した。

従来手法に対して本手法が作成する解析木数の比を図 10 に示す。図 10 から、文が長くなると、作成された解析木数の割合は小さくなるのがわかる。これは、長い文では、構文的曖昧性が大きくなり、そのため意味的に不適當な解析木が多く作成されるからであり、それに対して本手法の効果が現れている。

一方、従来手法に対する本手法の解析処理時間の比を図 11 に示す。本手法では、語と語の間の依存関係のチェックに時間を要するため、削除される解析木数の割合が小さいときには、両手法の解析処理時間はそれほど変わらないものの、解析木数の割合が大きくなると、本手法の効果は大きくなる。

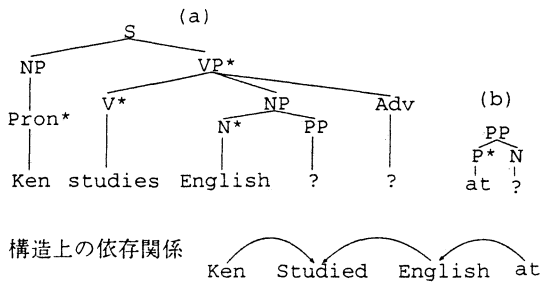


図 8: 解析例

文法	辞書
S → NP VP*	Pron → Ken
NP → Pron*	N → English / school
NP → N* PP	V → studies
PP → P* N	P → at
VP → V* NP Adv	Adv → yesterday
VP → V* N PP Adv	

図 9: 構文解析のための文法と辞書 (2)

5 おわりに

本稿では、単語間の依存関係情報を利用した漸進的構文解析の効率化手法を提案した。本手法では、解析木から定まる依存関係が意味的に正しいかどうかを、あらかじめ獲得した単語間の依存関係に関する情報を用いて調べることによって、意味的に不適当な解析木の作成を回避することができる。作成される解析木数を削減することができるため、解析処理時間を抑えることができる。実験を通してその有効性を確認した。

本稿で提案した手法では、語が入力される毎に、作成された項に対して語彙の主辞を与えるため、項の語彙の主辞となる語が早い段階で入力されれば、早期に項の語彙の主辞を決定することができ、語と語の依存関係を調べることができる。そのため本手法は、英語のような主辞先行言語に対して、より有効であると考えられる。

これまでに、漸進的構文解析を効率化する手法として、構文構造に対して行っていた処理をグループ単位で行い、実行する処理量を削減することにより、解析を高速化する手法 [5] や、いくつかの解析木を同一化できるようにあらかじめ文脈自由文法を変換することによって、作成される解析木数を抑える手法 [6] などが提案されている。それらの手法を本手法に導入することにより、一層の効率化が期待できる。

参考文献

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

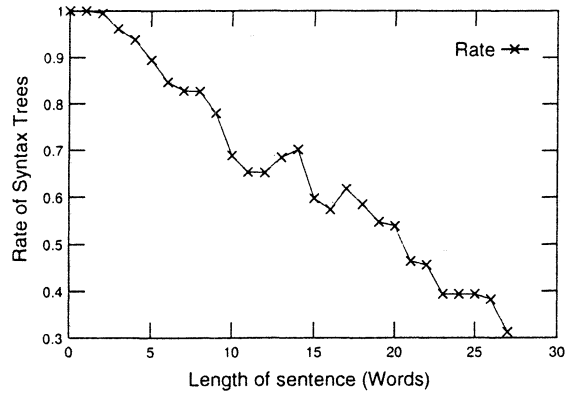


図 10: 作成された解析木数の比

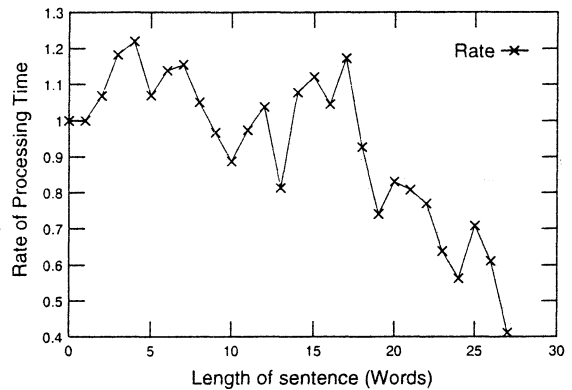


図 11: 解析処理時間の比

- [2] Matsubara, S., Asai, S., Toyama, K. and Inagaki, Y.: Chart-based Parsing and Transfer in Incremental Spoken Language Translation, *Proc. of 4th Natural Language Processing Pacific Rim Symposium*, pp.521-524 (1997).
- [3] 今一 修, 松本 裕治, 藤尾 正和: 統計情報と文法制約を統合した統語解析手法, *自然言語処理*, Vol.5, No.3, pp.67-83 (1998).
- [4] 渡辺 日出雄: 係り受け関係を用いた CFG 構文解析の枝刈手法, *情報処理学会第 59 回全国大会 (2)* pp.345-346 (1999).
- [5] 浅井 悟, 松原 茂樹, 外山 勝彦, 稲垣 康善: 構文構造のグループ化における漸進的チャート解析の高速化, *言語処理学会第 5 回年次大会* pp.88-91(1999).
- [6] 松原 茂樹, 村瀬 隆久, 加藤 芳秀, 外山 勝彦, 稲垣 康善: 文脈自由文法の変換に基づく漸進的構文解析の効率化, *電気関係学会東海支部連合大会* pp.635(1999).
- [7] Tomita, M: Efficient Parsing for Natural Language: A Fast Algorithm for Practical Systems (1985).