実時間音声言語処理システムのための
漸進的構文解析に関する研究

加藤 芳秀

2003年1月

目 次

図目次

表 目 次

第1章　まえがき

1.1　自然言語の漸進的解釈

音声は人間にとって最も自然で手軽な意思伝達手段であり，人と計算機が音声を介してコミュニケーションできれば，計算機はより扱いやすいものになることが期待できる．音声を入出力インタフェイスとした計算機システムを実現するために，これまでに様々な分野においてそのための研究が行われてきた．音声処理の分野では，計算機による音声認識や音声合成技術が，自然言語処理の分野では，形態素解析，構文解析，意味解析，談話解析といった計算機による自然言語理解のための基礎技術が開発されている．これら両分野における技術の進展を背景に，音声翻訳システムや音声対話システムなどの音声言語処理システムが試作されるようになりつつある．

音声言語処理システムを実現するための重要な技術として，話し言葉の処理が挙げられる．これまで，自然言語処理の研究は主に書き言葉を対象としており，話し言葉に対して従来の自然言語処理技術を単に適用しても，話し言葉に特有である言い直しや言い誤り，言い淀みなどの文法から逸脱する表現，すなわち不適格表現に対して頑健に対処することは難しい．これまでに頑健な話し言葉処理手法として，言い直しを処理する手法 [Core and Schubert 1999, Funakoshi et al. 2002] や，日本語における助詞の省略を処理する手法 [山本ら 1992] などが提案されており，多くの成果が得られている．

書き言葉にはない話し言葉に関する別の重要な特徴として，その出現形態の違い

が挙げられる．すなわち，一般に，書き言葉が文字系列として一度に出現するのに対して，話し言葉は音素の時間軸上の連続として現れる．このことは，話し言葉をその出現に従って逐次処理することの必要性を示唆している．このような要請を満たすためには，自然言語文をその出現順序に従って走査していく途中段階で，順次，それまでに読んだ文の断片を解釈する枠組が必要であり，このような枠組は漸進的解釈とよばれている [Inagaki & Matsubara 1995, Milward & Cooper 1994].

漸進的解釈に関する研究は，

- 早期曖昧性解消
- 漸進的構文解析
- 人間の言語理解過程のモデル化
- 実時間音声言語処理システム

などの動機のもと，これまでにいくつかの研究が行われている．次節以降では，まず，漸進的解釈に関する研究動向を概観し，次に，本論文の目的と内容について述べる．

1.2 自然言語の漸進的解釈に関する研究動向

本節では，前節で述べた漸進的解釈に関する研究動機のそれぞれの項目ごとに，その研究動向を示す．

1.2.1 自然言語解析における早期曖昧性解消

自然言語処理における厄介な問題の一つとして，文の曖昧性解消の問題が挙げられる．自然言語解析において問題となる曖昧性として，構文解析における構造的曖

味性，意味解析における語義の曖昧性，文脈解析における照応の指示対象の曖昧性など多数存在するが，各解析処理において，これらの曖昧性を十分に解消できなければ，他の解析が処理しなければならない情報の量が増大し，解析全体の効率が低下してしまう．これらの曖昧性を解消するための効率的な方法として，早期曖昧性解消が提案されている．文を解析する途中段階で段階的に曖昧性を解消することにより，解析の途中で生じる曖昧性の増大を抑制することができ，効率的な処理が実現できる．

Mellish は，名詞句の指示対象の候補とそれに対する制約条件を保持し，各対象の充足可能性を段階的に計算することにより，指示対象の同定を行う枠組を提案している [Mellish 1985].

Hirst は，ポラロイド語というモデルを用いて，語義的曖昧性を解消する手法を提案している [Hirst 1988]．ポラロイド語とは，ポラロイド写真からの比喩で命名されたものであり，単語が入力されるたびに語義が段階的に定まってゆく過程を表現している．ポラロイド語は複数の語義を曖昧なまま保持しており，単語が入力されるたびに，ポラロイド語同士が相互作用することにより，曖昧性を解消する．

秋葉らは，一般化弁別ネットワークに基づく漸進的な語義曖昧性解消モデル [Okumura & Tanaka 1990] を用いて，日本語係り受け解析の係り受けに関する曖昧性を解消する方法を提案している [秋葉ら 1993]．係り受け解析の早期に意味解析を導入することにより，意味的に誤った係り受け構造を取り除くことができるため，解析を効率的に行うことができる．

1.2.2 漸進的構文解析

漸進的構文解析は，自然言語文をその単語の出現順序に従って解析し，入力途中の段階でその構文的関係を捉える手法である．実時間音声言語処理システムがユーザ発話の意味を同時的に理解するためには，それに先立って，発話途中の段階でそ

の構文的関係を捉えることが必要である．そのための手法として漸進的構文解析は必要不可欠な技術である．これまでに提案された漸進的構文解析手法として，文脈自由文法に基づく手法[秋葉 & 田中 1992, Matsubara et al. 1997]や範疇文法に基づく手法[Haddock 1987, Milward 1995]などがあるが，いずれの手法も，入力文中の単語が先頭から順に一単語ずつ入力されるごとに，それまでに入力された文の断片に対する構文構造を生成する．

[秋葉 & 田中 1992]や[Matsubara et al. 1997]の手法は，一文単位の構文解析法であるチャート解析[Kay 1980]を拡張したものである．チャート解析では，項と呼ばれるデータ構造により構文木を表現するが，その特徴は，構造が定まっていない部分を未決定項と呼ばれる特別な項により明示的に表現できる点である．漸進的構文解析では，文の入力途中の段階で構文構造を生成するため，未入力部分に対する構造が定まらないが，これを未決定項として表現することにより，文の断片に対する構文構造が生成できる．例えば，文“I saw the girl with the telescope.”における単語“saw”が入力された段階で，文の断片“I saw”に対して次のような項を生成する．

$$[[[I]_{pron}]_{np}[[saw]_v[?]_{np}]_{vp}]_s$$

この項は，“I”の品詞は$pron$（代名詞）で，それがnp（名詞句）を構成し，“saw”の品詞がv（動詞）で，“saw”の主語が“I”であることに加え，$[?]_{np}$が未決定項であり，“I saw”に続くnpの構造が定まっていないことを示している．未決定項を利用することにより，文の断片に対する構文構造を表現することが可能となり，これらの構文構造を段階的に生成することにより，漸進的な構文解析が実現できる．

範疇文法に基づく手法[Haddock 1987, Milward 1995]では，範疇文法における複合範疇を利用することにより，文の断片に対する構文構造を生成する．複合範疇は記法X/Yで表現されるが，これは範疇Xから範疇Yが欠けた範疇を意味する．上の例の“I saw”に対しては，sのnpの部分が欠けたs/npという範疇が与えられることになり，その構文構造を生成することができる．

1.2.3 人間の言語処理過程のモデル化

人間の言語理解の過程が連続的なものであり，発声された音声をほぼ同時的に理解しているという見方は，我々の直観にかなうものであり，人間の言語理解における漸進性は，認知科学の分野において広く認識されている．実際，人間の言語理解における漸進性の存在を示唆する実験結果がこれまでに報告されている [Marslen-Wilson 1973]．心理言語学の分野においては，人間の言語処理過程を明らかにすることを目的に，それを形式的にモデル化する研究が行われているが，このようなモデル化において漸進性の性質は重要な役割を果たしている．

Stablerは，人間の構文解析の過程において保持できる文法的関係の数は，関係の種類ごとに制限されるという仮説をたてているが，文献 [Stabler 1994] において，この仮説を漸進性の観点から議論している．Sturtらは，袋小路文 (garden path sentence) のような誤解析しやすい文を人間が理解する過程を漸進的構文解析の枠組に基づいてモデル化している [Sturt and Crocker 1996]．人間の発話生成の過程を，漸進性の観点からモデル化した研究なども行われている [Levelt 1989, 羽尻ら 1998]．

1.2.4 実時間音声言語処理システム

人と計算機との自然で柔軟なコミュニケーションを実現するために，ユーザによって発話される自然言語文を漸進的に解釈することが有用であると考えられる [Allen et al. 2001, Milward 1995]．ユーザ発話を漸進的に解釈することにより，システムはユーザの発話入力に対して同時進行的な出力が可能となり，効果的なインタラクションの実現が期待できる．

Nakanoらは，会議予約をタスクとする対話システムにおいて，ユーザの発話途中でその理解状況を提示する方法を提案している [Nakano et al. 1999]．このシステムでは，ユーザの発話途中において，発話を理解していることを示す相槌をシステム

が出力することにより，ユーザ発話の理解状況を提示する．これにより，ユーザはシステムが理解しているかどうかを確認しながら対話することが可能となる．

Matsubaraら [Matsubara et al. 1999b] や，Mimaら [Mima et al. 1998] は，漸進的な言語理解・生成に基づく同時通訳システムを提案している．これらのシステムは，原言語文の入力に対して，それを逐次的に解析し，目標言語文に変換することにより，同時的な翻訳を実現している．

1.3 本論文の目的と内容

本研究では，実時間音声言語処理システムにおいてユーザ発話を同時的に理解するために必要な技術の一つである漸進的構文解析に焦点を絞り，それを開発する．以下では，本論文の目的，ならびに研究内容について述べる．

1.3.1 本研究の目的と目標

従来の漸進的構文解析に関する研究は，文の断片に対する構文構造を如何にして表現するかということにその主目的をおいており，実時間音声言語処理システムに適用する場合に問題となる次のような点についてはほとんど検討されていない．

一つ目の問題は解析の正確さである．解析の正確さの観点から考えると，これらの漸進的構文解析手法により生成される構文構造は，それまでに入力された文の断片に対する構文構造として可能な構文構造の候補を生成するにすぎず，そのすべてが正しいわけではない．生成された構文構造の候補の中には，入力文全体から考えると明らかに誤っている構造も含まれている．漸進的構文解析を実時間音声言語処理システムの一つのモジュールとして組み込んだ場合，そのモジュールで生成された構文構造には誤った構文構造が含まれるため，それらをそのまま出力すると，それは他のモジュールの動作にも影響を及ぼすことになり，結果として，システム全

体の信頼性が損なわれてしまう．したがって，信頼性の高い実時間音声言語処理システムを実現するためには，高精度な漸進的構文解析が必要である．

もう一つの問題は，解析処理の実時間性の問題である．すなわち，実時間音声言語処理システムがユーザ発話の入力に対して，同時的な処理を行うためには，当然，その言語理解部においても発話を同時的に処理しなければならない．ところが，漸進的構文解析では，構文的な曖昧性に加えて，未入力部分に対する予測の曖昧性が生じるため，曖昧性が増大し，文単位の解析と比べて処理効率は必ずしもよくない．これまでに提案された漸進的構文解析手法の解析速度は，音声入力を同時的に処理するには十分ではなく，実時間音声言語処理システムに利用できるレベルに達していない．

そこで本研究では，実時間音声言語処理システムの実現を目的に，その言語理解モジュールを構成する漸進的構文解析手法を開発する．本研究の具体的な目標は，上で述べた従来の漸進的構文解析手法の問題の解決である．すなわち，解析精度の高い漸進的構文解析手法の開発，ならびに，音声入力と同程度の速度でそれを処理できる高速な漸進的構文解析手法の開発である．

1.3.2 本論文の内容

本論文ではまず第一に，漸進的構文解析の正確さの問題を解決する一つの方法として，構文解析モジュールの出力タイミングを遅らせ，後続の入力の情報を待ち，その情報を用いて解析結果から誤った構文構造を取り除く方法を提案する．漸進的構文解析が誤った構文構造を生成するのは，その構造が，文の断片の部分的な情報のみを用いて生成されたものだからである．したがって，入力が進行し，文の断片に後続する入力が定まるにつれて，構文構造の正誤は次第に判明すると考えられる．

このような構文構造の出力タイミングを遅らせるアプローチに従う場合，出力タイミングと解析の正確さの間にはトレードオフの関係が存在すると考えられるため，

そのタイミングを考慮しなければならないが，本論文では，出力タイミングを定める方法の一つとして，まず，正確さを優先する方法を提案する．漸進的構文解析において正しい構文構造が確定するタイミングについて，文脈自由文法を基礎にして検討する．まず，漸進的構文解析の解析過程で生成される構文構造の正しさについて定義し，それを解析途中で逐次判定することにより，正しい構文構造を確定する方法について述べる．本方法では，文脈自由文法に基づく漸進的構文解析手法である漸進的チャート解析 [Matsubara et al. 1997] に，正しい構文構造を確定する仕組みを導入する．漸進的チャート解析では，単語入力のたびに，それまでに入力された文の断片に対する構文構造を生成するが，本方法では，文の断片に対する構文構造を生成する一方で，それ以前の段階で生成されたすべての構文構造についてそれが正しいかどうかを判定する．文法によって受理可能な文が入力されることを前提とすれば，文法制約を用いることにより，解析途中の段階でそれに続いて入力されうる単語列を推測できる．その中のどの単語列が入力されてもそれが正しい構文構造となることが保証されれば，その時点でその構文構造を正しいものと確定する．

出力タイミングを制御し，解析の正確さを高める別の方法として，次に，漸進的構文解析において，構文構造が正しいことが確定するまで出力を待つのではなく，正しい構文構造となる可能性を評価し，その可能性がある程度高くなった段階でそれを出力する手法を提案する．入力途中の段階で生成された構文構造の正しさは，残りの入力に依存して定まるが，この手法では，各構文構造に対して，残りの入力がその構文構造を正しくするような単語列である確率を確率文脈自由文法 (probabilistic context free grammar, PCFG) に基づいて計算する．この確率を単語が入力されるごとに計算し，それが予め定めておいた閾値を超えた時点で，構文構造を出力する．これにより，ある程度の解析の正確さを保ちつつ正確さを優先した手法に比べて，より早いタイミングで構文構造を出力できる．

また第二に，処理効率の問題に対して，漸進的依存構造解析手法を提案する．本

手法は，単語の入力ごとに，文の断片に対する依存構造を計算する．依存構造は句構造などに比べて単純な構造であるが，近年，その重要性が認められており，実際，依存構造を利用して構文的曖昧性を解消する方法 [Collins 1996] や，依存構造に基づいて翻訳文を生成する方法 [Alshawi et al 2000, 松原ら 1999] などが提案されている．まず，文脈自由文法に基づく漸進的構文解析と，句構造からの依存構造計算を組み合わせることにより，漸進的な依存構造解析が実現できることを示す．つぎに，これと同等かつ，より効率的な依存構造解析を実現する手法として，到達可能性に基づく漸進的依存構造解析手法を提案する．本手法は，入力された文の断片に対する句構造を生成する必要がなく，効率的な解析が可能である．

本論文の構成は以下の通りである．

次の第 2 章では，本論文の内容を展開する準備として，本研究の基礎をなす漸進的チャート解析について説明する．漸進的チャート解析は，一文単位で構文解析を実行するチャート解析を拡張した手法であるが，漸進的チャート解析と通常のチャート解析を比較し，漸進的チャート解析が単語が入力されるごとに，それまでに入力された文の断片に対する構文構造を生成できることを示す．

第 3 章では，漸進的チャート解析により生成される文の断片に対する構文構造の妥当性を定義し，妥当であることが保証される構文構造を入力途中の段階で判定する方法を提案する．提案手法を実際の音声対話データに対して適用した解析実験について報告する．

第 4 章では，3 章の手法を確率的に捉え，妥当である確率の高い構文構造を判定する方法を提案する．解析実験を通して，解析の正確さをそれほど損なうことなく，3 章の手法より早いタイミングで構文構造を出力できることを示す．

第 5 章では，漸進的依存構造解析アルゴリズムを提案する．漸進的チャート解析に基づく手法と等価な依存構造を生成できることを理論的に示し，より効率的な解析が実現できることを実験により示す．

第6章では，本論文のまとめと，残された課題，今後の展望について述べる．

第2章　漸進的チャート解析

本章では，本研究の基礎となる漸進的チャート解析 [Matsubara et al. 1997] について説明する．この手法は，一文単位での構文解析を実現するチャート解析 [Kay 1980] を拡張し，単語が入力されるごとに，それまでに入力された文の断片に対する構文構造を生成できるようにした手法である．まず，通常のチャート解析について説明し，次に漸進的チャート解析について述べる．

2.1　チャート解析

チャート解析 [Kay 1980] は，文脈自由文法ベースの構文解析手法の一つである．チャート解析では，**チャート** (chart) と呼ばれるグラフ構造を用いて解析結果を保持する．チャートは節点の集合，及び弧の集合から構成される．**節点** (node) は入力文中の単語と単語の間に位置し，i 番目の単語 w_i と $i+1$ 番目の単語 w_{i+1} の間の節点は，番号 i でラベル付けされる．以下では，番号 i でラベル付けされた節点を単に節点 i と呼ぶ．**弧** (edge) は節点と節点を結び，文中でその弧が覆っている部分に対する構文木をラベルとしてもつ．この構文木は**項** (term) と呼ばれるデータ構造で表され，記法 $[\alpha]_X$ で表現される．ここで，X は**範疇**であり，α は単語，記号 ? あるいは項のリストである．$[\alpha]_X$ という項 σ に対して，X を σ の範疇と呼ぶ．$[?]_X$ のように ? を含む項は**未決定項** (undecided term) と呼ばれ，構造が決定されていないことを表す．項の中に出現する未決定項のうち，もっとも左に位置するものを**最左未決定項**と呼ぶ．弧にラベルとして付された項の中に未決定項が出現するとき，この弧を

活性弧 (active edge) と呼び．そうでないとき，**不活性弧** (inactive edge) と呼ぶ．

チャート解析は，次の3つの操作を実行する．以下では，チャートの節点 i と節点 j を結ぶラベルが σ である弧を (i, j, σ) と書く．

(操作 1) 辞書引き: i 番目の単語 w_i の範疇が X ならば，不活性弧 $(i-1, i, [w_i]_X)$ をチャートに追加する．

(操作 2) 文法規則の適用: 不活性弧 $(i, j, [\cdots]_X)$ がチャート中に張られているとき，文法規則 $A \rightarrow XY \cdots Z$ が存在するならば，弧 $(i, j, [[\cdots]_X[?]_Y \cdots [?]_Z]_A)$ をチャートに追加する．

(操作 3) 項の置き換え: (i, j, σ) をチャート中の活性弧とし，σ の最左未決定項を $[?]_X$ とする．このとき，チャート中に不活性弧 (j, k, σ') が存在し，項 σ' の範疇が X であるならば，σ の最左未決定項を σ' 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 i と k の間に追加する．

例えば，図2.1の文法と辞書を用いて，英語文

$$\text{I saw her aunt with the telescope.} \tag{2.1}$$

の解析を考える．

1番目の単語 "I" に対しては，その範疇は $pron$ であるので，不活性弧 $(0, 1, [I]_{pron})$ をチャートに追加する．この弧は不活性弧であるので，これに対して文法規則 $np \rightarrow pron$ を適用し，不活性弧 $(0, 1, [[I]_{pron}]_{np})$ をチャートに追加する．さらに，この弧に対して文法規則 $s \rightarrow np\ vp\ \$$ を適用し，活性弧 $(0, 1, [[[I]_{pron}]_{np}[?]_{vp}[?]_{\$}]_s)$ をチャートに追加する．以上の過程を図2.2に示す．なお，図中の実線は不活性弧を表し，点線は活性弧を表す．

3番目の単語 "her" に注目してみると，その範疇は pos であるので，不活性弧 $(2, 3, [her]_{pos})$ をチャートに追加し，この弧に対して文法規則 $np \rightarrow pos\ n$

文法		辞書	
s	→ np vp $	pron	→ I / her
np	→ pron	det	→ the
np	→ pos n	pos	→ her
np	→ det n	n	→ aunt / telescope
np1	→ pos n pp	p	→ with
np1	→ det n pp	vi	→ saw
pp	→ p np	vt	→ saw
vp	→ vi	$	→ .
vp	→ vt np pp		
vp	→ vt np1		

s :	文	np, np1:	名詞句
vp :	動詞句	pp	: 前置詞句
det:	冠詞	pron	:人称代名詞
pos:所有代名詞		n	: 名詞
vi :	自動詞	vt	: 他動詞
p :	前置詞	$	: 文末記号

図 2.1: 構文解析のための文法と辞書

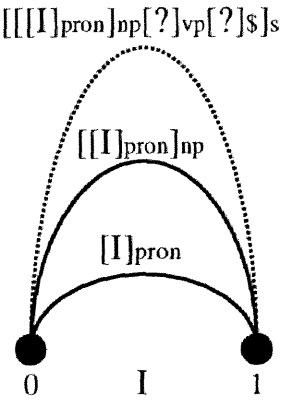

図 2.2: チャート解析における文法規則適用の例

を適用し，活性弧 $(2,3,[[her]_{pos}[?]_n]_{np})$ をチャートに追加する．次の単語 "aunt" の範疇は n であるので，不活性弧 $(3,4,[aunt]_n)$ がチャートに追加され，この弧と活性弧 $(2,3,[[her]_{pos}[?]_n]_{np})$ に対して項の置き換えの操作を適用し，弧 $(2,4,[[her]_{pos}[aunt]_n]_{np})$ がチャートに追加される．以上の過程を図 2.3 に示す．

2.2 漸進的チャート解析

チャート解析では，部分的に完成していない構文構造を活性弧として表現できるため，チャートを活用することは，漸進的な解析処理において有効であるが，次の点において漸進的チャート解析は通常のチャート解析と異なる．すなわち，漸進的チャート解析では，通常のチャート解析に対して，活性弧への文法規則の適用と活性弧による項の置き換えができるように，さらに次の 2 つの操作を導入している．

(操作 4) 活性弧への文法規則の適用 活性弧 (i,j,σ) がチャート中に張られているとき，σ の範疇が X であり，文法規則 $A \rightarrow XY\cdots Z$ が存在するならば，弧 $(i,j,[\sigma[?]_Y\cdots[?]_Z]_A)$ をチャートに追加する．

(操作 5) 活性弧の項による項置き換え (i,j,σ) をチャート中の活性弧とし，σ の最左

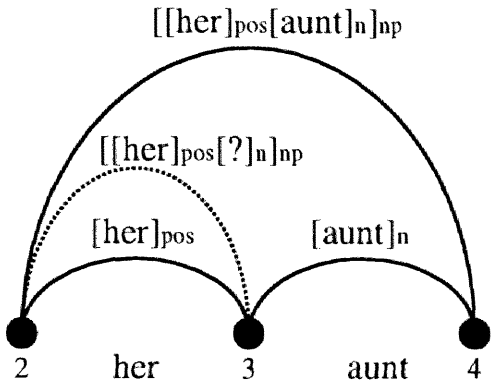

図 2.3: チャート解析における項の置き換えの例

未決定項を $[?]_X$ とする．このとき，チャート中に活性弧 (j, k, σ') が存在し，項 σ' の範疇が X であるならば，σ の最左未決定項を σ' 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 i と k の間に追加する．

これらの操作により，単語が入力されるごとに，それまでに入力された文の断片に対する構文構造を生成することが可能となる．次節で，解析処理の例を通してこれを説明する．

2.3 解析処理の例

例として，図2.1に示す文法と辞書を用いて，英語文 (2.1) の解析を考える．漸進的チャート解析における解析処理の過程を表2.1に示す．各行がチャートの弧に対応しており，#の欄は生成順に付けられた弧の番号，位置の欄は弧の始点と終点の対を表す．単語 “saw” が入力された時点までの解析処理を比較すると，通常の上昇型チャート解析では，単語 “I”，及び “saw” にそれぞれ文法規則を適用し，次のような弧

$$[I]_{pron} \tag{2.2}$$

表 2.1: "I saw her aunt with the telescope ." に対する漸進的な解析過程

入力語	チャート		
	#	位置	項
	1	0-0	$[?]_s$
I	2	0-1	$[[[I]_{pron}]_{np}[?]_{vp}[?]_{\$}]_s$
saw	3	0-2	$[[[I]_{pron}]_{np}[[saw]_{vi}]_{vp}[?]_{\$}]_s$
	4	0-2	$[[[I]_{pron}]_{np}[[saw]_{vt}[?]_{np}[?]_{pp}]_{vp}[?]_{\$}]_s$
	5	0-2	$[[[I]_{pron}]_{np}[[saw]_{vt}[?]_{np1}]_{vp}[?]_{\$}]_s$
her	6	0-3	$[[[I]_{pron}]_{np}[[saw]_{vt}[[her]_{pron}]_{np}[?]_{pp}]_{vp}[?]_{\$}]_s$
	7	0-3	$[[[I]_{pron}]_{np}[[saw]_{vt}[[her]_{pos}[?]_n]_{np}[?]_{pp}]_{vp}[?]_{\$}]_s$
	8	0-3	$[[[I]_{pron}]_{np}[[saw]_{vt}[[her]_{pos}[?]_n[?]_{pp}]_{np1}]_{vp}[?]_{\$}]_s$
aunt	9	0-4	$[[[I]_{pron}]_{np}[[saw]_{vt}[[her]_{pos}[aunt]_n]_{np}[?]_{pp}]_{vp}[?]_{\$}]_s$
	10	0-4	$[[[I]_{pron}]_{np}[[saw]_{vt}[[her]_{pos}[aunt]_n[?]_{pp}]_{np1}]_{vp}[?]_{\$}]_s$
with	11	0-5	$[[[I]_{pron}]_{np}[[saw]_{vt}[[her]_{pos}[aunt]_n]_{np}[[with]_p[?]_{np}]_{pp}]_{vp}[?]_{\$}]_s$
	12	0-5	$[[[I]_{pron}]_{np}[[saw]_{vt}[[her]_{pos}[aunt]_n[[with]_p[?]_{np}]_{pp}]_{np1}]_{vp}[?]_{\$}]_s$
the	13	0-6	$[[[I]_{pron}]_{np}[[saw]_{vt}[[her]_{pos}[aunt]_n]_{np}[[with]_p[[the]_{det}[?]_n]_{np}]_{pp}]_{vp}[?]_{\$}]_s$
	14	0-6	$[[[I]_{pron}]_{np}[[saw]_{vt}[[her]_{pos}[aunt]_n[[with]_p[[the]_{det}[?]_n]_{np}]_{pp}]_{np1}]_{vp}[?]_{\$}]_s$
telescope	15	0-7	$[[[I]_{pron}]_{np}[[saw]_{vt}[[her]_{pos}[aunt]_n]_{np}[[with]_p[[the]_{det}[telescope]_n]_{np}]_{pp}]_{vp}[?]_{\$}]_s$
	16	0-7	$[[[I]_{pron}]_{np}[[saw]_{vt}[[her]_{pos}[aunt]_n[[with]_p[[the]_{det}[telescope]_n]_{np}]_{pp}]_{np1}]_{vp}[?]_{\$}]_s$
.	17	0-8	$[[[I]_{pron}]_{np}[[saw]_{vt}[[her]_{pos}[aunt]_n]_{np}[[with]_p[[the]_{det}[telescope]_n]_{np}]_{pp}]_{vp}[.]_{\$}]_s$
	18	0-8	$[[[I]_{pron}]_{np}[[saw]_{vt}[[her]_{pos}[aunt]_n[[with]_p[[the]_{det}[telescope]_n]_{np}]_{pp}]_{np1}]_{vp}[.]_{\$}]_s$

$$[[I]_{pron}]_{np} \tag{2.3}$$

$$[[[I]_{pron}]_{np}[?]_{vp}[?]_{\$}]_{s} \tag{2.4}$$

$$[saw]_{vi} \tag{2.5}$$

$$[saw]_{vt} \tag{2.6}$$

$$[[saw]_{vi}]_{vp} \tag{2.7}$$

$$[[saw]_{vt}[?]_{np}[?]_{pp}]_{vp} \tag{2.8}$$

$$[[saw]_{vt}[?]_{np1}]_{vp} \tag{2.9}$$

を生成するが，一方，漸進的チャート解析では項 (2.4) の最左未決定項を，(2.7) や (2.8)，あるいは (2.9) で置き換えることにより，文の断片 “I saw” に対する項#3, #4, #5 をも生成する（図 2.4 参照）．これにより，文の構成素間の関係，例えば，動詞 “saw” の主語は “I” であるといった関係を捉えることができる．

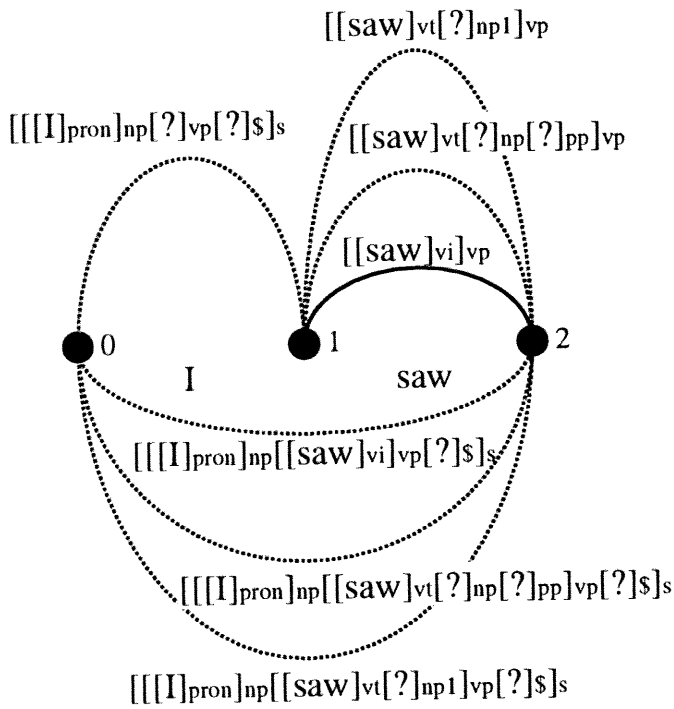

図 2.4: 漸進的チャート解析における項の生成の例

第3章　漸進的構文解析における構文構造確定手法

3.1　はじめに

漸進的構文解析では，単語が入力されるたびに，それまでに入力された文の断片に対する構文構造を生成するが，生成された構文構造は可能な構造の候補に過ぎず，そのすべてが正しいわけではない．構文構造の候補の中には，入力文全体から考えると明らかに誤っている構造も含まれている．

本章では，漸進的構文解析において正しい構文構造を確定する方法を提案する[Kato et al. 2000]．前章で述べた漸進的チャート解析を基礎にして議論を進める．まず，漸進的チャート解析の解析過程で生成される項の正しさについて定義し，その正しさを解析途中で逐次判定することにより，正しい項を確定する方法について述べる．本方法では，漸進的チャート解析により，それまでの入力に対する項を生成する一方で，それ以前の段階で生成されたすべての項についてそれが正しいかどうかを判定する．項の正しさが判定できない場合は，判定を遅延し，後続の入力を待ち，再度判定を試みる．文法によって受理可能な文が入力されることを前提とすれば，文法制約を用いることにより，解析途中の段階でそれに続いて入力されるうる単語列を推測できる．その中のどの単語列が入力されてもそれが正しい項となることが保証されれば，その時点でその項を正しいものと確定する．本方法をPennTreeBank[Marcus et al. 1993]のATISコーパスならびにATR音声言語データベース[浦谷ら 1994]の文に適用したところ，項の確定タイミングは文によってばら

ついており，早い段階で確定する文もあれば，入力が完了するまで構造を確定できない文も存在した．全体としては，文が長くなるにつれて確定タイミングは遅れる傾向にあり，ATISコーパスを用いた実験では，確定の遅れは最大で文の長さの約70%，平均で約31%であった．ATR音声言語データベースを用いた実験では，構文構造確定の遅れは，最大で文の長さの約71%，平均で約33%であった．

次の3.2節では，文の断片に対する項の正しさを定義する．3.3節では，漸進的チャート解析において項の正しさを判定する手法について述べる．3.4節では，実験の概要を述べ，その結果について報告する．

3.2 文の断片に対する構文構造の妥当性

2章では，漸進的チャート解析により，単語が入力されるごとに文の断片に対する項を生成できることを示した．一般に，文の断片に対して複数の項が生成されるが，それらの項のすべてが構文的役割を正しく捉えているわけではなく，単に構文構造の候補であるにすぎない．入力文全体から考えると明らかに誤っているものが含まれている場合がある．例えば，文の断片

I saw her (3.1)

の “her” の品詞には人称代名詞と所有代名詞の2つの可能性がある．図2.1の文法と辞書を用いて，これを漸進的チャート解析により解析すると，“her” が入力された段階で，“her” を人称代名詞として捉えたとして表2.1の項#6及び，所有代名詞として捉えた項#7, #8が生成される．(3.1) が英語文 (2.1) の断片であるとすると，“her” は明らかに所有代名詞である．その場合，項#6は “her” の構文的役割を誤って捉えており，項#7, #8は正しく捉えているということができる．

以下での議論を明確にするため，入力文全体に対して正しく構文的関係を捉えている項に関する定義を与える．まず，項の間の包含関係を定義する．

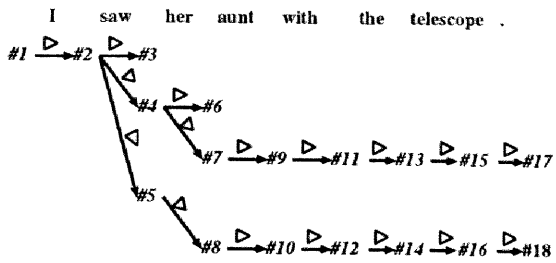

図 3.1: 項の間の包含関係

定義 3.1 (包含関係) 項σ, τに対して，ある項θが存在し，σの最左未決定項をθで置き換えることにより得られる項がτに等しいとき，σはτを**直接に包含する**といい，$\sigma \triangleright \tau$と書く．さらに，$\triangleright^*$を$\triangleright$の反射推移閉包とする．$\sigma \triangleright^* \tau$であるとき，$\sigma$は$\tau$を**包含する**という． □

σがτを包含するとは，σがτの部分を構成することを意味する．図 3.1 に，英語文 (2.1) に対して漸進的チャート解析で生成された表 2.1 の各項の間の包含関係$\triangleright$を示す．

文の断片に対する項が文全体から考えて正しく構文的関係を捉えている場合，その項は入力文全体に対する項を少なくとも一つ包含している．上の例において，項#7，及び#8 は英語文 (2.1) に対する項#17, #18 をそれぞれ包含している．本論文では，そのような項を**妥当な項**と呼ぶこととし，以下のように定義する．

定義 3.2 (妥当な項) 入力文$w_1 \cdots w_n$に対して項σが**妥当な項である**とは，次の *1.* ～*3.* を満たす項τが存在するときである．

1. $\sigma \triangleright^* \tau$.

2. τは$w_1 \cdots w_n$に対する項である．

3. τは未決定項を含まない． □

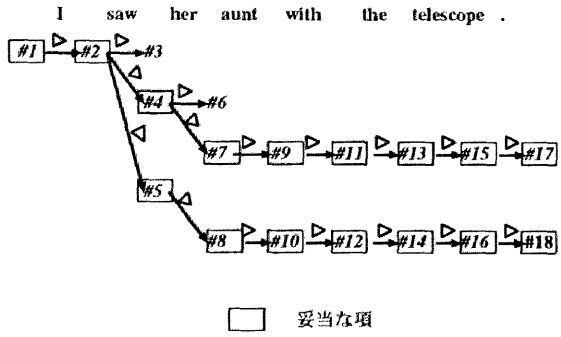

図 3.2: 妥当な項

例えば，英語文 (2.1) に対して項#7，及び#8 は妥当な項であり，#6 は妥当な項ではない．図 3.2 に，英語文 (2.1) に対して生成された表 2.1 の各項のうち，妥当な項であるものを示す．四角で囲まれたものは妥当な項である．

3.3 漸進的構文解析における構文構造の正当性判定

文の断片に対する項の妥当性は，その断片に後続する部分に依存している．前章の例の “her” には，人称代名詞であるか，あるいは所有代名詞であるかという曖昧性があり，それに伴って項#6〜#8 が生成されたが，それらの項の妥当性は “her” に続く入力に依存する．例えば，“her” に続く入力が “aunt with the telescope.” である，すなわち (3.1) が (2.1) の断片である場合には，#7 と#8 は妥当な項であり，#6 は妥当な項でない．一方，“her” に続く入力が “with the telescope.” である場合には，#6 は妥当な項であり，#7，#8 は妥当な項でない．このことは，項の妥当性の判定において後続の入力情報が必要であることを意味する．

入力が完了した時点で項の妥当性が判定できるのは，定義 3.2 より明らかである．問題は，入力途中の時点でそれを判定できるかどうかであるが，これは，後続の入力がどのようなものであっても妥当であることが確定している項を見つければよい．

以下では，後続の入力がどのようなものであっても妥当であることが確定している項を**正当な項**と呼ぶことにし，次のように定義する．

定義 3.3 (正当な項) 文の断片 $w_1 \cdots w_j$ に対して項 σ （σ は $w_1 \cdots w_i (i \leq j)$ に対する項）が正当であるとは，任意の文 $w_1 \cdots w_j w_{j+1} \cdots w_n$ に対して，σ が妥当なときである． □

3.3.1 項の正当性の判定

本節では，文の入力途中の時点で，項の正当性を判定する方法を提案する．この方法では，漸進的チャート解析において新たに単語が入力されるたびに，それ以前の段階で生成された項の正当性の判定を試みる．判定できない場合は，判定を遅らせ，次の段階で再度判定を試みる．

まず，文の断片 $w_1 \cdots w_j$ に続く入力として，どのような単語列が可能であるかについて考える．文の断片 $w_1 \cdots w_j$ に対する項は $w_1 \cdots w_j$ の構文的関係を示していると同時に，それに後続する入力としてどのようなものが可能であるのかをも表現している．次に定義する項の未決定範疇列がそれである．

定義 3.4 (項の未決定範疇列) 項 σ に出現する未決定項を左から順に並べた列を $\tau_1, \ldots, \tau_m$ とする．また，$\tau_k (1 \leq k \leq m)$ の範疇を X_k とする．このとき，$X_1 \cdots X_m$ を σ の**未決定範疇列**という． □

いま，$\alpha \stackrel{*}{\Rightarrow} w_{j+1} \cdots w_n$ は，範疇列 α から単語列 $w_{j+1} \cdots w_n$ が導出されることを示すものとする[1]．文の断片 $w_1 \cdots w_j$ に対する項 σ の未決定範疇列が α であるとは，$\alpha \stackrel{*}{\Rightarrow} w_{j+1} \cdots w_n$ であるような単語列 $w_{j+1} \cdots w_n$ が $w_1 \cdots w_j$ に続いて入力されたと

[1] $\stackrel{*}{\Rightarrow}$ の定義は以下の通りである．β，及び γ を範疇と単語からなる列とし，A を範疇とする．このとき，文法規則 $A \rightarrow \alpha$ が存在するならば，$\beta A \gamma$ は $\beta \alpha \gamma$ を**導出する**といい，$\beta A \gamma \Rightarrow \beta \alpha \gamma$ と書く．$\stackrel{*}{\Rightarrow}$ は $\Rightarrow$ の反射推移閉包である．

表 3.1: 項の未決定範疇列

単語	項	未決定範疇列
	#1	s
I	#2	vp \$
saw	#3	\$
	#4	np pp \$
	#5	$np1$ \$
her	#6	pp \$
	#7, #8	n pp \$
aunt	#9, #10	pp \$
with	#11, #12	np \$
the	#13, #14	n \$
telescope	#15, #16	\$
.	#17, #18	

き，σ が妥当であることを意味しており，言い換えると，$w_{j+1}\cdots w_n$ が $w_1\cdots w_j$ に続いて入力される可能性があることを表している．

例えば，項#4 の未決定範疇列は np pp \$ であるが，np pp \$ $\stackrel{*}{\Rightarrow} x$ を満たす単語列 x，例えば “her aunt with the telescope .” は “I saw her” に続いて入力される可能性があり，これが入力されたとき#4 は入力文に対して妥当な項となる．表 2.1 の項の未決定範疇列を表 3.1 に示す．

入力文が与えられた文法によって受理される文，すなわち適格文であれば，文の断片 $w_1\cdots w_j$ に対する項の中には必ず妥当な項が存在する．したがって，いま $w_1\cdots w_j$ に対する項の未決定範疇列が k 個あるとしてそれらを $\alpha_1,\cdots,\alpha_k$ とすると，後続の

入力 $w_{j+1}\cdots w_n$ は，

$$\begin{array}{c} \alpha_1 \stackrel{*}{\Rightarrow} w_{j+1}\cdots w_n \\ \vdots \\ \alpha_k \stackrel{*}{\Rightarrow} w_{j+1}\cdots w_n \end{array} \tag{3.2}$$

のいずれかを満たすような単語列に限られる．そうでないとすると，$w_1\cdots w_j$ に対する項で妥当な項となるものが存在しなくなり，入力文が適格であることに矛盾する．

文の断片 “I saw her” に対する項として，#6〜#8 が生成される．#6 の未決定範疇列は $pp\ \$$ であり，#7，#8 の未決定範疇列は $n\ pp\ \$$ である．$pp\ \$ \stackrel{*}{\Rightarrow}$ with the telescope .，$n\ pp\ \$ \stackrel{*}{\Rightarrow}$ aunt with the telescope . であるから，“with the telescope .” や “aunt with the telescope .” などは “I saw her” 以後に入力される可能性がある．

次に，文の断片 $w_1\cdots w_j$ に続く入力が単語列 $w_{j+1}\cdots w_n$ であるとした場合に，どのような項が妥当な項となるかについて考える．j 番目の語が入力された時点で生成された項，すなわち $w_1\cdots w_j$ に対する項が妥当な項となるのは，その未決定範疇列 α が $\alpha \stackrel{*}{\Rightarrow} w_{j+1}\cdots w_n$ を満たすときである．例えば，文の断片 “I saw her” に続いて “aunt with the telescope .” が入力される場合を考える．“I saw her” に対しては，項#6〜#8 が生成される．#7, #8 の未決定範疇列はともに $n\ pp\ \$$ であり，$n\ pp\ \$ \stackrel{*}{\Rightarrow}$ aunt with the telescope . であるので，#7, #8 は妥当な項となる．逆に，#6 の未決定範疇列は $pp\ \$$ であり，$pp\ \$ \stackrel{*}{\Rightarrow}$ aunt with the telescope . ではないので，#6 は妥当な項とはならない．

j 番目の語が入力される以前に生成された項，すなわち $w_1\cdots w_i (i<j)$ に対する項が入力文に対して妥当な項となるのは，$w_1\cdots w_j$ に対する項の中の妥当な項を包含するときである．これは，定義 3.2 と包含関係の推移性から成り立つ．例えば，英語文 (2.1) に対して語 “her” が入力された段階で生成された項#6〜#8 のうち，妥当な項は#7, #8 であるが，“I saw” に対する項の中の妥当な項#4，及び#5 は，それぞれ#7, #8 を包含している．逆に，妥当な項でない#3 は，#7 と#8 のどちらも包

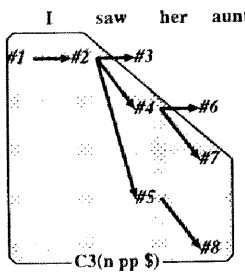

図 3.3: 英語文 (2.1) に対する項の集合 $C_3(n\ pp\ \$)$

含していない.

以上より，次の定理が成り立つ.

定理 3.5 文 $w_1 \cdots w_j w_{j+1} \cdots w_n$ の文頭から j 個の単語からなる文の断片 $w_1 \cdots w_j$ に対する項でその未決定範疇列が α であるような項を包含する項の集合を $C_j(\alpha)$ とする．このとき，$\alpha \stackrel{*}{\Rightarrow} w_{j+1} \cdots w_n$ ならば，$C_j(\alpha)$ の要素は $w_1 \cdots w_j w_{j+1} \cdots w_n$ に対して妥当な項である. □

例えば，英語文 (2.1) に対して，$C_3(n\ pp\ \$) = \{\#1, \#2, \#4, \#5, \#7, \#8\}$ である（図 3.3 参照）.

j 番目の語 w_j が入力されたときに，それ以前の段階で生成された項の正当性を判定する方法について述べる．文の断片 $w_1 \cdots w_j$ に対する項の未決定範疇列が k 個あるとしてそれらを $\alpha_1, \ldots, \alpha_k$ としたとき，$w_1 \cdots w_j$ に続く入力 $w_{j+1} \cdots w_n$ は，$w_1 \cdots w_j w_{j+1} \cdots w_n$ が適格文なら，(3.2) のいずれかを満たしている．よって定理 3.5 より，$C_j(\alpha_1), \ldots, C_j(\alpha_k)$ のいずれかは，$w_1 \cdots w_j w_{j+1} \cdots w_n$ に対して妥当な項からなる集合となる．したがって，次の条件を満たす σ は，$w_1 \cdots w_j w_{j+1} \cdots w_n$ が適格文になるいかなる $w_{j+1} \cdots w_n$ に対しても，文 $w_1 \cdots w_j w_{j+1} \cdots w_n$ の妥当な項である

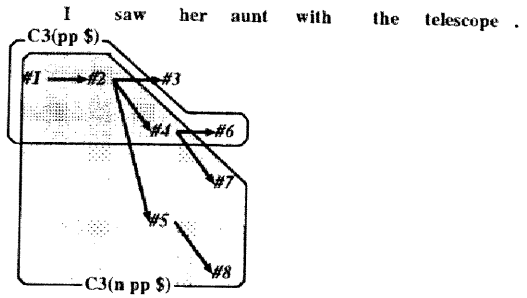

図 3.4: 正当な項の判定

ことが保証される，すなわち $w_1 \cdots w_j$ に対して正当な項である．

$$\begin{cases} \sigma \in C_j(\alpha_1) \\ \quad \vdots \\ \sigma \in C_j(\alpha_k) \end{cases} \tag{3.3}$$

(3.3) を満たす σ からなる集合は，

$$C_j(\alpha_1) \cap \cdots \cap C_j(\alpha_k) \tag{3.4}$$

である．つまり (3.4) は，$w_1 \cdots w_j$ に対して正当な項からなる集合である．ここで重要なのは，(3.4) が後続の入力 $w_{j+1} \cdots w_n$ とは無関係に求めることができる点である．すなわち，(3.4) を解析途中の各時点で随時計算することにより，入力途中の任意の段階で現れる項の正当性を判定できる．

例えば，英語文 (2.1) において，語 “her” が入力された時点で項#6〜#8 が生成されるが，#6 の未決定範疇列は $pp\ \$$ であり，#7，及び#8 の未決定範疇列は $n\ pp\ \$$ である．したがって，$\#6 \in C_3(pp\ \$)$ であり，$\#4 \triangleright^* \#6$ であるから，$\#4 \in C_3(pp\ \$)$ でもある．また，$\#7 \in C_3(n\ pp\ \$)$ であり，$\#4 \triangleright^* \#7$ であるので，$\#4 \in C_3(n\ pp\ \$)$ である．したがって，$\#4 \in C_3(n\ pp\ \$) \cap C_3(n\ pp\ \$)$ であるので，#4 は “I saw her” に対して正当な項である (図 3.4)．さらに，“aunt” が入力された時点では，“I saw her aunt” に対して，項#9，及び#10 が生成される．項#9, #10 の未決定範疇列は共に

表 3.2: 項の正当性を判定する過程

入力語	生成された項	正当と判定された項
	#1	#1
I	#2	#2
saw	#3, #4, #5	
her	#6, #7, #8	#4
aunt	#9, #10	#5, #7, #8, #9, #10
with	#11, #12	#11, #12
the	#13, #14	#13, #14
telescope	#15, #16	#15, #16
.	#17, #18	#17, #18

pp \$ である．#9, #10 $\in C_4(pp\ \$)$ であるから，いずれも “I saw her aunt” に対して正当な項であるとわかる．

3.3.2 正当性判定の例

英語文 (2.1) に対して生成された表 2.1 の項について，その正当性を判定する過程を表 3.2 に示す．

単語 “saw” が入力された時点では，項#3〜#5 が生成されるが，それらの正当性はこの段階では判定できない．単語 “her” が入力された時点では，項#6〜#8 が生成される．#6 の未決定範疇列は pp \$ であり，#7 と#8 の未決定範疇列は $n\ pp$ \$ である．$C_3(pp\ \$) = \{\#1, \#2, \#4, \#6\}$ であり，$C_3(n\ pp\ \$) = \{\#1, \#2, \#4, \#5, \#7, \#8\}$ であることから，項#6〜#8 のいずれもその正当性を判定できないが，“I saw” に対する項#4 は，$C_3(pp\ \$) \cap C_3(n\ pp\ \$)$ の要素であり，この時点で正当な項であることがわかる．単語 “aunt” が入力された時点では，項#9, #10 が生成される．#9, 及び#10 の未決定範疇列は pp \$ であり，#9$\triangleright^*$#9 であるので，#9 $\in C_4(pp\ \$)$ であ

る，すなわち，#9は正当な項である．同様にして，#10も正当な項であるとわかる．単語 “with” が入力された時点では項#11及び#12が，“the” が入力された時点では#13及び#14が，“telescope” が入力された時点では#15及び#16が，“.” が入力された時点では#17及び#18がそれぞれ生成されるが，その段階でそれらが正当な項であるとわかる．

このように，入力途中の段階で，項の正当性を判定することができる．項の正当性は，表3.2の “saw” が入力された段階で生成された項#3, #4, #5のように即座には決まらない場合があるが，正当性の判定を遅延することにより，入力途中の段階で判定できる．

3.3.3 正当性判定アルゴリズム

$w_1 \cdots w_j$ に対して正当な項からなる集合 (3.4) を計算するアルゴリズム，すなわち項の正当性判定アルゴリズムを図3.5に示す．なお，$I(w_1 \cdots w_i)(1 \leq i \leq j)$ を漸進的チャート解析により生成された $w_1 \cdots w_i$ に対する項の集合とし，$u(\sigma)$ を σ の未決定範疇列とする．

1行目から3行目までの操作は，$w_1 \cdots w_j$ に対する項の未決定範疇列の集合 U を計算する．4行目と5行目の操作は，$C_j(\alpha)(\alpha \in U)$ を保持するテーブル C_j の初期化である．6行目から11行目までの操作により，各 $C_j(\alpha)$ を計算し，その結果が $C_j[\alpha]$ に保持される．12行目の操作では，$C_j[\alpha]$ の積集合をとり，式 (3.4) を計算する．

このアルゴリズムが停止することを示すには，アルゴリズム内の **for** ループ中の操作の実行回数が有限であること，9行目の $\sigma' \rhd^* \sigma$ が有限ステップ数で判定できること，及び12行目の積集合を計算する操作が有限ステップ数で停止することを示せばよい．

まず，**for** ループ中の操作の実行回数が有限であることを示す．漸進的チャート解析により生成される項の数は有限であるので，$I(w_1 \cdots w_i)(1 \leq i \leq j)$ は有限集合であ

入力：項の集合 $I(w_1)$, $I(w_1w_2)$, ..., $I(w_1 \cdots w_j)$

出力：項の集合 O

1 $U := \phi$;

2 **for each** $\sigma \in I(w_1 \cdots w_j)$ **do**

3 $U := U \cup \{u(\sigma)\}$;

4 **for each** $\alpha \in U$ **do**

5 $C_j[\alpha] := \phi$;

6 **for each** $\sigma \in I(w_1 \cdots w_j)$ **do begin**

7 $C_j[u(\sigma)] := C_j[u(\sigma)] \cup \{\sigma\}$;

8 **for** $i := 1$ **to** j **do**

9 **for each** $\sigma' \in I(w_1 \cdots w_i)$ **such that** $\sigma' \rhd^* \sigma$ **do**

10 $C_j[u(\sigma)] := C_j[u(\sigma)] \cup \{\sigma'\}$;

11 **end**

12 $O := \bigcap_{\alpha \in U} C_j[\alpha]$;

図 3.5: 項の正当性判定アルゴリズム

る．したがって，2行目，6行目，及び9行目のforループ中の操作の実行回数は有限である．1行目から3行目の操作で計算されたUの要素数は，高々$|I(w_1 \cdots w_j)|$個と有限である．したがって，4行目のforループ中の操作の実行回数も高々$|I(w_1 \cdots w_j)|$回と有限である．8行目のforループ中の操作の実行回数はj回と有限である．

$\bigcup_{i=1}^{j} I(w_1 \cdots w_i)$上の関係$\rhd^*$は，$\bigcup_{i=1}^{j} I(w_1 \cdots w_i)$が有限集合であるから，有限ステップ数で計算できる．したがって，9行目の$\sigma' \rhd^* \sigma$は有限ステップ数で判定できる．

Uは有限集合であるので，12行目の積集合を計算する操作が有限ステップ数で停止することを示すには，各$C_j[\alpha](\alpha \in U)$が有限集合であることを示せば十分である．以下にそれを示す．任意の$\sigma \in C_j[\alpha]$に対して，$\sigma \in \bigcup_{i=1}^{j} I(w_1 \cdots w_i)$であるので，$C_j[\alpha] \subseteq \bigcup_{i=1}^{j} I(w_1 \cdots w_i)$である．$\bigcup_{i=1}^{j} I(w_1 \cdots w_i)$は有限集合であるので，$C_j[\alpha]$も有限集合である．

以上より，図3.5の項の正当性判定アルゴリズムは停止することが示された．また，以上の議論から，式(3.4)が後続の系列$w_{j+1} \cdots w_n$とは無関係に求められることは明らかである．

3.4 実験と結果の検討

前節までに，文の解析途中の段階において，後続がどのようなものであっても妥当である項，すなわち，正当な項を判定する手法を示した．

そのような項が漸進的構文解析の途中で出現するかどうかは，解析に使用する文法，及び，対象となる文に依存するが，もし，そのような項が現れることがほとんどないのであれば，提案手法の効果は低いといわざるを得ない．そこで，本手法の利用可能性を調査するために，それをGNU Common Lispを用いて実装し，実際の音声対話文とその解析に使用される文法を用いて，解析実験を行った．

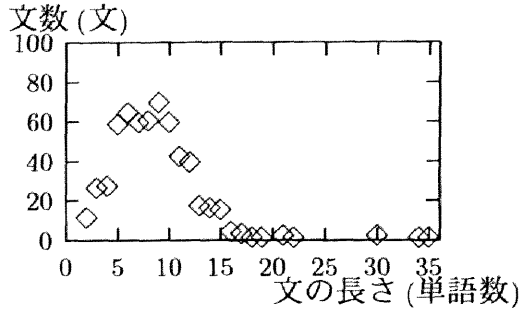

図 3.6: 文の長さの分布（ATIS コーパス）

3.4.1 構造確定タイミングの実験

まず，漸進的構文解析の途中で正しい構文構造を確定できる文が存在するかどうかについて，Penn Treebank[Marcus et al. 1993] に収録されている構文木付き ATIS コーパスを対象に調査した．図 3.6 に実験に用いたコーパスに含まれる文の長さの分布を示す．実験に用いた文法規則の数は 905 規則である[2]．1 文あたり平均で 166.8 個の項が生成されたが，そのうち 13.8%が妥当な項であった．

文の断片の構文構造の確定タイミングの例を章末の**付録 (A)** に挙げる．これらの例から，提案手法が文の入力途中の時点で文の断片の正しい構造を確定できることがわかる．578 文のうちで入力途中の段階で正しい項が確定した文の数は，全体の 60.7%にあたる 351 文であり，提案手法が有用である場合は必ずしも少なくないことを確認した．

付録の例 1 のようにほぼ同時に構造が確定する場合もあれば，例 4 のように入力が完了するまでほとんど構造が確定できない場合もあり，そのタイミングはまちまちである．そこで，文の断片の正しい構造が確定するタイミングをを定量的に調べるために，次の尺度を定義し，これを測定した．$s = w_1 \cdots w_n$ を入力文とし，$O_j(s)$

[2] 文法規則は，578 文に付与されている構文木データから規則を読み出すことにより獲得した．ただし，その規則をそのまま用いると解析処理時間が非常にかかるので，解析速度向上のために，文献 [松原ら 1999], 及び [Roark & Johnson 1999] の手法に従って文法規則を変換したものを用いた．

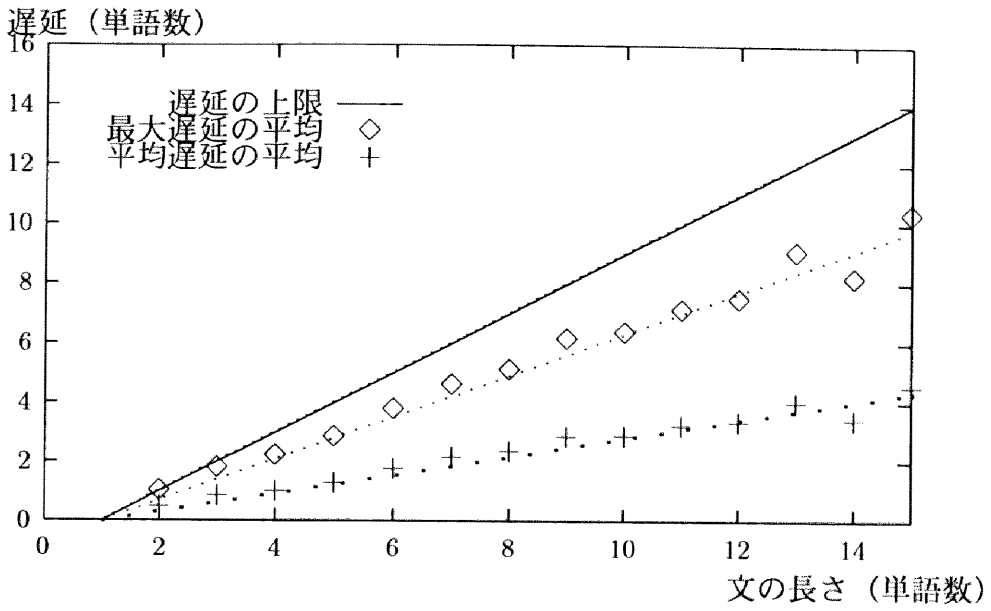

図 3.7: 文の長さと遅延の関係（ATIS コーパス）

を s の j 番目の単語 w_j が入力された時点で正しい項として出力された項の集合とする．また σ を $w_1 \cdots w_i$ に対する項としたとき，$f(\sigma) = i$ とし，j 番目の単語が入力された時点での遅延を

$$D(j,s) = j - \max_{\sigma \in O_j(s)} f(\sigma) \tag{3.5}$$

と定める．s に対する最大遅延，及び平均遅延を次のように定義する．

$$\text{最大遅延} : D_{max}(s) = \max_{1 \le j \le n} D(j,s) \tag{3.6}$$

$$\text{平均遅延} : D_{ave}(s) = \frac{1}{n} \sum_{j=1}^{n} D(j,s) \tag{3.7}$$

図 3.7 に，文の長さと遅延の関係を示す．遅延は文の長さを n とすると，$n-1$ を越えることはなく，これが図中に実線で示されている．細い点線と太い点線はそれぞれ，最小 2 乗法により求めた最大遅延，及び平均遅延に対する回帰直線である．この結果を見ると，文が長くなるにつれて遅延が大きくなる傾向がわかる．回帰直線

の勾配は約0.70である．最大遅延は文の長さのほぼ70%であることを示している．最大遅延と同様，平均遅延も文が長くなるにつれて大きくなる傾向がわかる．平均遅延の回帰直線の勾配は，約0.31であり，平均遅延は文の長さのほぼ31%である．

3.4.2 遅延の度合と文法のカバー率の関係の実験

遅延の度合は文法に依存していると考えられ，より多くの文を受理する文法，すなわちカバー率の高い文法を用いると，遅延の度合は大きくなると予想される．そこで，文法のカバー率と遅延の関係を調べるため，文法の規模を変えてカバー率と遅延を測定する実験を行った．ただし，実験対象としてATISコーパスの規模は小さいため，代わりに，日本語に対するコーパスではあるがATR音声言語データベース [浦谷ら 1994] を用いた．

文法規則は，前節の実験と同様，構文木データから規則を読み出すことにより獲得した[3]．文法規則の獲得に用いる対話を，40対話から40対話づづ200対話まで増やし，カバー率の異なる文法を獲得した．文法のカバー率は，規則の獲得に用いたものとは別の50対話1134文のうちの解析できた文の割合で計算した．遅延の度合として，回帰直線の勾配を用いた．これは，上記の1134文のうちの，40対話から獲得した文法で解析できた695文に対するものである．

参考として，40対話から獲得した文法を用いたときの正しい項の出力タイミングの例を，章末の**付録 (B)** に，文の長さと遅延の関係を図3.8に示す．最大遅延の回帰直線の勾配は約0.71，平均遅延のそれは0.33とATISコーパスの場合と比較的似た値をとっている．遅延に対する回帰直線の勾配と文法のカバー率との関係を図3.9

[3] 東京工業大学田中・徳永研究室において作成された構文木データ [1] を用いた．この構文木データは，形態素単位で解析された構文木が付与されているが，文献 [植木ら 1998] の方法に従って，文節間の係り受けを表現する構文木に変換し，それらから文法規則を獲得した．ATISコーパスの実験と同様に，文献 [松原ら 1999] 及び [Roark & Johnson 1999] の手法に従って，文法規則を変換したものを用いた．

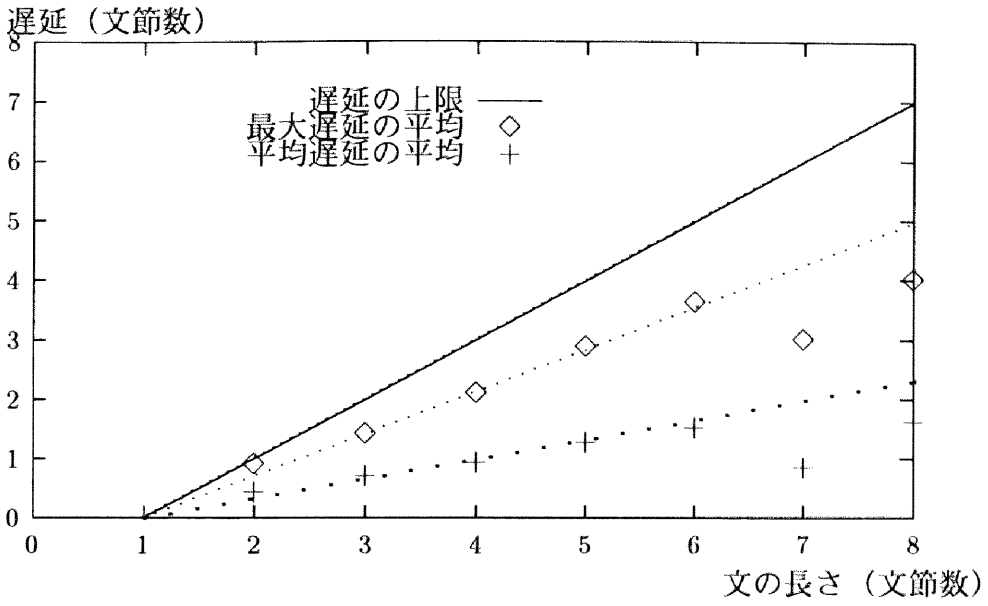

図 3.8: 文の長さと遅延の関係（ATR 音声言語データベース）

に示す．文法のカバー率の増大にともなって，回帰直線の勾配の値も大きくなっており，遅延の度合は文法のカバー率に依存していることがわかる．しかしながら，文法規則の獲得に用いた対話を 40 対話から 200 対話に増やすことにより，カバー率は 17.4%増加したが，最大遅延の回帰直線の勾配の増加は 0.12，平均遅延のそれは 0.06，と緩やかであった．

3.5 3章のまとめ

単語入力毎に逐次的実行する漸進的構文解析では一般に，語が入力されるごとに文の断片に対する構文構造を生成するが，文の断片の部分的な情報しか用いていないため，誤った構文構造を生成する可能性がある．本章では，構文構造を表す項の正誤を後続の入力の情報を活用して判定する方法を示し，実際の対話文コーパスを用いた解析実験により，構文構造確定タイミングの遅延の度合について調べた．本

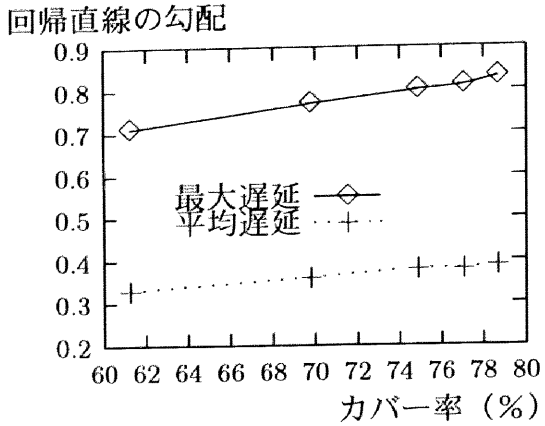

図 3.9: 文法のカバー率と遅延に対する回帰直線の勾配の関係

方法にしたがって構文構造の正しさが保証された時点でそれを出力すれば，場合によっては文末まで全く構文構造を出力しない場合もあるが，一方で，誤った解析結果を出力しなくなる．このようなアプローチは，必ずしも実時間音声言語処理システムの言語理解部としての実時間性を十分に満たすものではないが，漸進的構文解析の正確さと出力タイミングのトレードオフの問題における出力タイミングの上限を定めているとみなすことができ，この問題を検討する上での重要な知見を提供する．

付録:出力タイミングの例

以下に，実験で用いた文に対する出力タイミングの例を挙げる．なお，入力文 $w_1 \cdots w_n$ の断片 $w_1 \cdots w_i (i \leq n)$ に対する項で，妥当なものを $\sigma(w_i)$ と書き，妥当でないものを $\bar{\sigma}(w_i)$ と書く．$w_1 \cdots w_i$ に対する項が複数個ある場合については，最も早いタイミングで出力された項のみを示す．

(A) ATIS コーパスの例

例 1 I want to leave Monday morning .

入力単語	出力された項
I	
want	σ(I), σ(want)
to	σ(to)
leave	
Monday	σ(leave)
morning	
.	σ(Monday), σ(morning), σ(.)

例 2 Does that flight serve dinner ?

入力単語	出力された項
does	σ(does)
that	
flight	
serve	σ(that), σ(flight)
dinner	σ(serve)
?	σ(dinner), σ(?)

例 3 What kind of airplane do they use ?

入力単語	出力された項
what	
kind	
of	
airplane	
do	σ(what), σ(kind), σ(of), σ(airplane)
they	σ(do), σ(they)
use	
?	σ(use), σ(?)

例 4 Show me all the flights that go from Baltimore to Seattle .

入力単語	出力された項
show	σ(show)
me	
all	σ(me)
the	
flights	
that	
go	
from	
Baltimore	
to	
Seattle	
.	σ(all), σ(the), σ(flights), σ(that),
	σ(go), σ(from), σ(Baltimore)
	σ(to), σ(Seattle), σ(.)

例 5 Please list the flights from Charlotte to Long Beach after lunch time.

入力単語	出力された項
please	
list	σ(please), σ(list)
the	
flights	
from	σ(the), σ(flights)
Charlotte	
to	σ(from), σ(Charlotte)
long	
beach	
arriving	σ(to), σ(long), σ(beach)
after	σ(arriving)
lunch	
time	
.	σ(after), σ(lunch), σ(time), σ(.)

(B) ATR 音声言語データベースの例

例 1 はい では お名前と 日付便名を お願いいたします .

入力文節	出力された項
はい	
では	σ(はい), σ(では)
お名前と	
日付便名を	
お願いいたします	
.	σ(お名前と), σ(日付便名を), σ(お願いいたします), σ(.)

例2 情報を 探してみますので 少々 お待ちください．

入力文節	出力された項
情報を	
探してみますので	
少々	σ(情報を),
	σ(探してみますので)
お待ちください	
.	σ(少々),
	σ(お待ちください),
	σ(.)

例3 また なにか ございましたら お電話ください．

入力文節	出力された項
また	
なにか	
ございましたら	σ(また), σ(なにか)
	σ(ございましたら)
お電話ください	σ(お電話ください)
.	σ(.)

第4章　確率文脈自由文法に基づく漸進的構文解析

4.1　はじめに

3章では，漸進的構文解析モジュールの出力タイミングを遅らせ，後続の情報を待ち，その情報を用いて正しい構文構造を確定する手法を提案した．3章の手法のように，解析結果が正確であることが望ましいのはいうまでもない．しかし，たとえ誤った構文構造が解析結果から完全に取り除かれていなくても，構文解析に続くモジュール（意味解析モジュールや文脈解析モジュールなど）において誤った構文構造を取り除くことができる可能性や，誤った構文構造をもとにシステムが動作を決定しても，後の時点でそれを修復できる可能性[1]がある．誤った構文構造を出力することをある程度許すことにより，出力の同時性を高めることができると期待される．

本章では，そのような試みの一つとして，漸進的構文解析において，構文構造が正しいことが確定するまで出力を待つのではなく，正しい構文構造となる可能性を評価し，その可能性がある程度高くなった段階でそれを出力する手法を提案する[加藤ら 2002]．入力途中の段階で生成された構文構造の正しさは，残りの入力に依存して定まるが，この手法では，各構文構造に対して，残りの入力がその構文構造を正しくするような単語列である確率を確率文脈自由文法 (probabilistic context free

[1] 例えば，同時通訳システムにおいて，原言語文の入力途中の段階で生成した翻訳文が誤っている場合があるが，それに続く部分を翻訳するときに，その誤りを言い直し，全体として意味の取れる翻訳文を生成する手法が提案されている [松原ら 1998].

grammar, PCFG) に基づいて計算する．この確率を単語が入力されるごとに計算し，それが予め定めておいた閾値を超えた時点で，構文構造を出力する．これにより，ある程度の解析の正確さを保ちつつ，正確さを優先した3章の手法に比べて，より早いタイミングで構文構造を出力できる．Penn Treebank[Marcus et al. 1993] に収録されているATISコーパス，及びATR音声言語データベース [浦谷ら 1994] を用いた解析実験を行った．ATISコーパスを用いた実験では，正解率98.3%で，出力の遅延は最大でも文の長さの約44%，平均で約17%であった．ATR音声言語データベースを用いた実験では，正解率98.4%で，出力の遅延は最大でも文の長さの約48%，平均で約18%であった．

本章の構成は以下の通りである．まず，4.2節では，確率文脈自由文法に基づく文の断片に対する構文構造の評価手法を提案する．4.3節では，実験の概要を述べ，その結果に基づいて提案手法を評価する．

4.2 確率文脈自由文法に基づく構文構造の評価

3章で述べたように，文の断片に対する項の妥当性は，その断片に後続する入力に依存している．したがって，文の断片に後続する可能性の高い単語列がわかれば，妥当である可能性の高い項もわかる．例えば，図4.1に示す文法と辞書を用いて，英語文

I made the reservation for the room. (4.1)

を漸進的に解析することを考える．漸進的チャート解析における解析処理の過程を表4.1に示す．文の断片 "I made" を考えると，この断片に対しては，項#3〜#5が生成され，#3の未決定範疇列は *np* $, #4のそれは *np advp* $, #5のそれは *pp* $ である．すなわち，"made" に後続して名詞句のみが入力されたときには，#3が妥当な項であり，名詞句，及び形容詞句がこの順で入力されたときには，#4が妥当な

表 4.1: "I made the reservation for the room." に対する漸進的な解析過程

入力語	チャート		
	#	位置	項
	1	0-0	$[?]_s$
I	2	0-1	$[[[I]_{pron}]_{np}[?]_{vp}[?]_{\$}]_s$
made	3	0-2	$[[[I]_{pron}]_{np}[[made]_v[?]_{np}]_{vp}[?]_{\$}]_s$
	4	0-2	$[[[I]_{pron}]_{np}[[made]_v[?]_{np}[?]_{adjp}]_{vp}[?]_{\$}]_s$
	5	0-2	$[[[I]_{pron}]_{np}[[made]_v[?]_{pp}]_{vp}[?]_{\$}]_s$
the	6	0-3	$[[[I]_{pron}]_{np}[[made]_v[[the]_{det}[?]_n]_{np}]_{vp}[?]_{\$}]_s$
	7	0-3	$[[[I]_{pron}]_{np}[[made]_v[[the]_{det}[?]_n[?]_{pp}]_{np}]_{vp}[?]_{\$}]_s$
	8	0-3	$[[[I]_{pron}]_{np}[[made]_v[[the]_{det}[?]_n]_{np}[?]_{adjp}]_{vp}[?]_{\$}]_s$
	9	0-3	$[[[I]_{pron}]_{np}[[made]_v[[the]_{det}[?]_n[?]_{pp}]_{np}[?]_{adjp}]_{vp}[?]_{\$}]_s$
reservation	10	0-4	$[[[I]_{pron}]_{np}[[made]_v[[the]_{det}[reservation]_n]_{np}]_{vp}[?]_{\$}]_s$
	11	0-4	$[[[I]_{pron}]_{np}[[made]_v[[the]_{det}[reservation]_n[?]_{pp}]_{np}]_{vp}[?]_{\$}]_s$
	12	0-4	$[[[I]_{pron}]_{np}[[made]_v[[the]_{det}[reservation]_n]_{np}[?]_{adjp}]_{vp}[?]_{\$}]_s$
	13	0-4	$[[[I]_{pron}]_{np}[[made]_v[[the]_{det}[reservation]_n[?]_{pp}]_{np}[?]_{adjp}]_{vp}[?]_{\$}]_s$
for	14	0-5	$[[[I]_{pron}]_{np}[[made]_v[[the]_{det}[reservation]_n[[for]_p[?]_{np}]_{pp}]_{np}]_{vp}[?]_{\$}]_s$
	15	0-5	$[[[I]_{pron}]_{np}[[made]_v[[the]_{det}[reservation]_n[[for]_p[?]_{np}]_{pp}]_{np}[?]_{adjp}]_{vp}[?]_{\$}]_s$
the	16	0-6	$[[[I]_{pron}]_{np}[[made]_v[[the]_{det}[reservation]_n[[for]_p[[the]_{det}[?]_n]_{np}]_{pp}]_{np}]_{vp}[?]_{\$}]_s$
	17	0-6	$[[[I]_{pron}]_{np}[[made]_v[[the]_{det}[reservation]_n[[for]_p[[the]_{det}[?]_n[?]_{pp}]_{np}]_{pp}]_{np}]_{vp}[?]_{\$}]_s$
	18	0-6	$[[[I]_{pron}]_{np}[[made]_v[[the]_{det}[reservation]_n[[for]_p[[the]_{det}[?]_n]_{np}]_{pp}]_{np}[?]_{adjp}]_{vp}[?]_{\$}]_s$
	19	0-6	$[[[I]_{pron}]_{np}[[made]_v[[the]_{det}[reservation]_n[[for]_p[[the]_{det}[?]_n[?]_{pp}]_{np}]_{pp}]_{np}[?]_{adjp}]_{vp}[?]_{\$}]_s$
room	20	0-7	$[[[I]_{pron}]_{np}[[made]_v[[the]_{det}[reservation]_n[[for]_p[[the]_{det}[room]_n]_{np}]_{pp}]_{np}]_{vp}[?]_{\$}]_s$
	21	0-7	$[[[I]_{pron}]_{np}[[made]_v[[the]_{det}[reservation]_n[[for]_p[[the]_{det}[room]_n[?]_{pp}]_{np}]_{pp}]_{np}]_{vp}[?]_{\$}]_s$
	22	0-7	$[[[I]_{pron}]_{np}[[made]_v[[the]_{det}[reservation]_n[[for]_p[[the]_{det}[room]_n]_{np}]_{pp}]_{np}[?]_{adjp}]_{vp}[?]_{\$}]_s$
	23	0-7	$[[[I]_{pron}]_{np}[[made]_v[[the]_{det}[reservation]_n[[for]_p[[the]_{det}[room]_n[?]_{pp}]_{np}]_{pp}]_{np}[?]_{adjp}]_{vp}[?]_{\$}]_s$
.	24	0-8	$[[[I]_{pron}]_{np}[[made]_v[[the]_{det}[reservation]_n[[for]_p[[the]_{det}[room]_n]_{np}]_{pp}]_{np}]_{vp}[.]_{\$}]_s$

文法		辞書	
(r1)	s → np vp $	(l1)	pron → I
(r2)	np → pron	(l2)	det → the
(r3)	np → det n	(l3)	n → reservation
(r4)	np → det n pp	(l4)	n → room
(r5)	pp → p np	(l5)	p → for
(r6)	vp → v np	(l6)	v → made
(r7)	vp → v np adjp	(l7)	$ → .
(r8)	vp → v pp		

s : 文	np: 名詞句
vp : 動詞句	pp:前置詞句
adjp:形容詞句	
det : 冠詞	n : 名詞
pron: 代名詞	p : 前置詞
v : 動詞	$:文末記号

図 4.1: 構文解析のための文法と辞書

項である．前置詞句が入力されれば，#5 が妥当な項である．仮に，“I made” に続いて名詞句である単語列が入力される可能性が高いということがわかったとすると，#3 が妥当である可能性は高いということができる．

そこで本節では，各項に対して，残りの入力がその項を妥当にする単語列である確率を計算する一つの方法として，PCFG に基づく手法を提案する．まず，PCFG について簡単に説明し，次に，提案手法について述べる．

表 4.2: 文法規則の確率

文法規則	確率
(r1), (r5), (l1), (l2), (l5)〜(l7)	1
(r6)	0.7
(r4), (l3), (l4)	0.5
(r3)	0.3
(r2), (r8)	0.2
(r7)	0.1

4.2.1 確率文脈自由文法

PCFG は構文木，及び文の生起確率を計算するためのモデルである．構文木の生起確率は，その木の構成に用いられた規則の確率の積で計算する．文の生起確率は，その文に対する構文木の生起確率の和で計算する．文の断片に対する項，及び文の断片の生起確率も同様にして計算できる．

例えば，図 4.1 の文法と辞書に対して，表 4.2 のように確率を与えるとする．このとき，項#3 の構成に用いられた文法規則は，(r1), (r2), (r6), (l1), (l6) であるので，その生起確率は，$1 \times 0.2 \times 0.7 \times 1 \times 1 = 0.14$ である．“I made” に対する項#4, 及び#5 の生起確率を同様にして計算すると，それぞれ 0.02, 0.04 である．文の断片 “I made” の生起確率は，それに対する項#3〜#5 の生起確率の和，すなわち $0.14 + 0.02 + 0.04 = 0.2$ である．

以下では，項 σ の生起確率を $P(\sigma)$，文の断片 $\mathbf{w}$ の生起確率を $P(\mathbf{w})$ と表記する．

4.2.2 項を妥当にする単語列が入力される確率

まず，j番目の単語w_jまで入力されたとき，それまでに生成された項のそれぞれに関して，それを妥当にする残りの入力$\mathbf{w}$がどのような単語列であるかについて考える．

j番目の単語が入力された時点で生成された項，すなわち，$w_1 \cdots w_j$に対する項σが，$w_1 \cdots w_j\mathbf{w}$に対して妥当であるのは，3章でも述べたように，$\mathbf{w}$がσの未決定範疇列から導出されるときである．

j番目の単語が入力される以前に生成された項，すなわち，$w_1 \cdots w_i (i < j)$に対する項σが，$w_1 \cdots w_j\mathbf{w}$に対して妥当であるのは，σに包含されるような$w_1 \cdots w_j$に対する項の中に，妥当な項が存在するときである．これは，定義3.2と包含関係の推移性から成り立つ．例えば，“I made”に対する項#3は，“I made the”に対する項#6と#7を包含するが，“the”に続く入力単語列が“reservation for the room.”であるとき，#7は妥当な項であり，#7を包含する#3も妥当な項である．

以上のことから，j番目の単語w_jが入力された段階での，σを妥当にするw_jに続く入力単語列の集合$\Omega(\sigma, j)$を，次のように定義できる．なお，以下では，$I(w_1 \cdots w_j)$を文の断片$w_1 \cdots w_j$に対する項からなる集合とする．

$$\begin{aligned}\Omega(\sigma, j) \;=\; \{\mathbf{w} \mid &\sigma\text{に包含される}\tau \in I(w_1 \cdots w_j)\text{が存在し，} \\ &\tau\text{の未決定範疇列から}\mathbf{w}\text{が導出される}\}\end{aligned}$$

$w_1 \cdots w_j$に続く入力が，σを妥当にする単語列となる，すなわち，$\Omega(\sigma, j)$の要素となる確率は，次式のようになる．

$$\sum_{\mathbf{w} \in \Omega(\sigma,j)} P(\mathbf{w} \mid w_1 \cdots w_j) \;=\; \frac{\sum_{\mathbf{w} \in \Omega(\sigma,j)} P(w_1 \cdots w_j\mathbf{w})}{P(w_1 \cdots w_j)} \qquad (4.2)$$

この確率が大きい項σを出力すれば，構文解析結果の正確さを高めることができる．本手法では，この確率が，予め定めておいた閾値を超える項σを出力する．

ところで，(4.2) の右辺の分母は，次のように $w_1 \cdots w_j$ に対する項の生起確率から計算できる．

$$P(w_1 \cdots w_j) = \sum_{\sigma' \in I(w_1 \cdots w_j)} P(\sigma') \tag{4.3}$$

一方，(4.2) の右辺の分子は，$\Omega(\sigma, j)$ の各要素 $\mathbf{w}$ を列挙し，$P(w_1 \cdots w_j \mathbf{w})$ を計算し，その和を求めればよい．しかし，このようにして (4.2) の右辺の分子を計算することは，一般に，$\Omega(\sigma, j)$ は無限集合であるため，すべての $\Omega(\sigma, j)$ の要素を列挙してそれを計算することになり，実際上困難である．そこで，本手法では，$\Omega(\sigma, j)$ の要素を列挙するのではなく，$I(w_1 \cdots w_j)$ 中の項の生起確率から式 (4.2) 右辺の分子を近似的に計算する方法を採用する．

まず，$P(w_1 \cdots w_j \mathbf{w})$ について考える．$P(w_1 \cdots w_j \mathbf{w})$ は，その定義より，$w_1 \cdots w_j \mathbf{w}$ に対する未決定項を含まない項（以下では，**不活性項**と呼ぶ）の生起確率の和である．したがって，$T(\sigma, j)$ を $w_1 \cdots w_j \mathbf{w} (\mathbf{w} \in \Omega(\sigma, j))$ に対する不活性項の集合とすると，式 (4.2) 右辺の分子は次のようになる．

$$\sum_{\mathbf{w} \in \Omega(\sigma, j)} P(w_1 \cdots w_j \mathbf{w}) = \sum_{\sigma' \in T(\sigma, j)} P(\sigma') \tag{4.4}$$

ここで，項 $\tau \in I(w_1 \cdots w_j)$ の生起確率は，それが包含する不活性項の生起確率の和に等しいことに注意する．すなわち，$Sub(\tau)$ を τ が包含する不活性項の集合とすると，

$$P(\tau) = \sum_{\sigma' \in Sub(\tau)} P(\sigma')$$

である．したがって，$Sub(\tau) \subseteq T(\sigma, j) (\tau \in I(w_1 \cdots w_j))$ 中の不活性項については，その生起確率をそれぞれ求めなくても，$P(\tau)$ でそれらの生起確率の和を計算できる．本手法では，そのような不活性項の生起確率の和によって，式 (4.4) を近似する．すなわち，$Sub(\tau) \subseteq T(\sigma, j) (\tau \in I(w_1 \cdots w_j))$ の和集合を $V(\sigma, j)$ とし，$Sub(\tau) \subseteq T(\sigma, j)$ であるような $\tau \in I(w_1 \cdots w_j)$ である項の集合を $U(\sigma, w_1 \cdots w_j)$ としたとき（図 4.2 参照），次のように近似する．

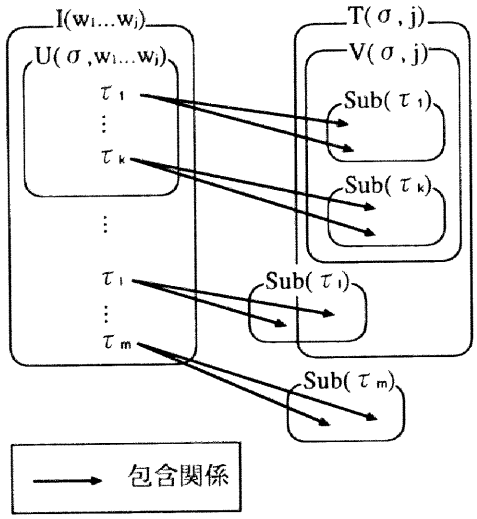

図 4.2: 集合 $V(\sigma, j)$ と $U(\sigma, w_1 \cdots w_j)$

$$\begin{aligned}\sum_{\sigma' \in T(\sigma,j)} P(\sigma') &= \sum_{\sigma' \in V(\sigma,j)} P(\sigma') + \sum_{\sigma' \in T(\sigma,j)-V(\sigma,j)} P(\sigma') \\ &= \sum_{\tau \in U(\sigma,w_1 \cdots w_j)} P(\tau) + \sum_{\sigma' \in T(\sigma,j)-V(\sigma,j)} P(\sigma') \\ &\approx \sum_{\tau \in U(\sigma,w_1 \cdots w_j)} P(\tau) \end{aligned} \tag{4.5}$$

この近似では，$T(\sigma, j)$ を，その生起確率の和が $I(w_1 \cdots w_j)$ 中の項の生起確率から計算できる部分 $V(\sigma, j)$ とそうでない部分 $T(\sigma, j) - V(\sigma, j)$ に分割し，前者の項の生起確率の和を式 (4.4) の近似とする．ここで注意すべきことは，$U(\sigma, w_1 \cdots w_j)$ 中の項に包含されない $T(\sigma, j)$ の不活性項の生起確率が加算されないので，式 (4.5) の値は，常に式 (4.4) の値以下であるということである（$T(\sigma, j)$ のすべての不活性項が $U(\sigma, w_1 \cdots w_j)$ 中の項に包含されるとき，それらの値は等しい.）．したがって，式 (4.2) 右辺の分子を式 (4.5) で近似した次の式

$$\sum_{\mathbf{w} \in \Omega(\sigma,j)} P(\mathbf{w} \mid w_1 \cdots w_j) \approx \frac{\sum_{\tau \in U(\sigma,w_1 \cdots w_j)} P(\tau)}{\sum_{\sigma' \in I(w_1 \cdots w_j)} P(\sigma')} \tag{4.6}$$

は，(4.2) の値以下であり，(4.6) の値が閾値を超える項を出力すれば，(4.2) の値が閾値を超える項だけが出力される．このような意味において，式 (4.6) は健全な近似

である.

具体的に，式 (4.6) を計算するために，$U(\sigma, w_1 \cdots w_j)$ の要素 τ が，どのような項であるかについて考える．そのような τ には次の 2 つの場合がある.

1. τ が σ に包含される場合.

2. τ は σ に包含されないが，σ に包含される $w_1 \cdots w_j$ に対する項と未決定範疇列が一致する場合.

（1）と（2）のいずれの場合でも，τ の未決定範疇列から導出される単語列はいずれも，Ω の定義から当然，$\Omega(\sigma, j)$ の要素である．したがって，τ に包含される任意の不活性項 σ' に対して，ある $\mathbf{w} \in \Omega(\sigma, j)$ が存在して，σ' は $w_1 \cdots w_j \mathbf{w}$ に対する項である，すなわち $\sigma' \in T(\sigma, j)$ である．したがって，$Sub(\tau) \subseteq T(\sigma, j)$ である．一方，（1）と（2）のいずれでもない場合には，一般に $Sub(\tau) \subseteq T(\sigma, j)$ は成り立たない．以上から，$U(\sigma, w_1 \cdots w_j)$ は σ に包含される $w_1 \cdots w_j$ に対する項，及び，それらと未決定範疇列が一致する $w_1 \cdots w_j$ に対する項からなる集合である，すなわち，(1) と（2）の場合の $\tau \in I(w_1 \cdots w_j)$ からなる集合である.

$I(w_1 \cdots w_j)$ と $U(\sigma, w_1 \cdots w_j)$ は j 番目の単語 w_j が入力された時点で計算できるので，式 (4.6) も w_j が入力された時点で計算できる.

以下では，式 (4.6) 右辺を，$S(\sigma, w_1 \cdots w_j)$ と書く.

4.2.3 閾値による出力タイミングの制御

前節で述べた方法により，j 番目の単語 w_j が入力された段階で，$w_1 \cdots w_j$ に続く入力が項 σ を妥当にする単語列である確率を近似的に計算することができる．提案手法では，各項 σ に対して，式 (4.6) の右辺を単語が入力されるごとに計算し，その値が予め定めておいた閾値を超えた段階で項を出力する．近似した値（式 (4.6) 右辺）よりも真の値（式 (4.6) 左辺）は常に大きい，もしくは等しいので，近似した値

が閾値を超えることは真の値も閾値を超えることを意味する．したがって，出力される項は，妥当である可能性が高い項であり，構文解析結果の正確さを高めることができる．閾値を大きくするほど解析の正確さは高まる．式 (4.2) の値，すなわち，$w_1 \cdots w_j$ に続く入力が項 σ を妥当にする単語列である確率が 1 であるとき，σ が妥当であることが確定する．したがって，閾値=1.0 の場合には，出力される項は，妥当であることが保証される．すなわち，3 章の手法のような正確さを優先した手法となる．

4.2.4 解析例

英語文 (4.1) に対して生成された表 4.1 の項について，前節で述べた確率を計算する例を以下に示す．出力タイミングの制御に用いる閾値は 0.8 とする．

単語 "made" が入力された段階では，項#3〜#5 が生成される．すなわち，$I(\text{I made}) = \{\#3, \#4, \#5\}$ である．生起確率はそれぞれ，$P(\#3) = 0.14$, $P(\#4) = 0.02$, $P(\#5) = 0.04$ である．#3 はそれ自身を包含し，それと未決定範疇列が一致する "I made" に対する項は#3 自身だけである．したがって，$U(\#3, \text{I made}) = \{\#3\}$ である．よって，$S(\#3, \text{I made}) = 0.14/(0.14 + 0.02 + 0.04) = 0.7$ である．この段階では閾値を超えないため，項#3 は出力しない．

次の単語 "the" が入力されると，項#6〜#9 が生成され，それらの生起確率は，$P(\#6) = 0.042$, $P(\#7) = 0.07$, $P(\#8) = 0.006$, $P(\#9) = 0.01$ である．#3 は，#6 と#7 を包含し，それらと未決定範疇列が一致する "I made the" に対する項は，それら自身だけである，すなわち，$U(\#3, \text{I made the}) = \{\#6, \#7\}$ であるので，$S(\#3, \text{I made the}) = (0.042 + 0.07)/(0.042 + 0.07 + 0.006 + 0.01) = 0.875$ である．閾値を超えたので，この段階で "I made" に対する項である#3 を出力する．

このように，残りの入力が項を妥当にするような単語列である確率を，単語が入力されるごとに計算し，それが閾値を超えた段階で項を出力する．

4.3 実験と結果の検討

提案手法を評価するために，それをGNU Common Lispを用いて実装し，構文解析実験を行った．

4.3.1 ATISコーパスを用いた実験

Penn Treebank[Marcus et al. 1993]に収録されている構文木付きATISコーパス全578文を対象とした．実験に用いた文法規則は，3章の実験と同じであり，その確率はコーパスより最尤推定法により求めた．

出力タイミングの制御に用いる閾値が1.0の場合（解析の正確さを優先した場合）と，0.9の場合の，項の出力タイミングの例を**付録 (A)**に挙げる．提案手法は，例3や例4のように妥当でない項を出力する場合もあるが，正確さを優先した手法に比べて，より早いタイミングで妥当な項を出力していることがわかる．

閾値が1.0の場合と0.9の場合について遅延を測定した．図4.3に，文の長さと遅延の関係を示す．遅延の上限は，文の長さをnとすると$n-1$であり，これを実線で示している．破線は最小2乗法により求めた回帰直線である．最大遅延の回帰直線の勾配は，閾値=1.0のとき約0.70，閾値=0.9のとき約0.44である．これは，閾値=1.0のとき，最大遅延は文の長さの70%，閾値=0.9のとき，最大遅延は文の長さの44%であることを示している．平均遅延の回帰直線の勾配は，閾値=1.0のとき約0.31，閾値=0.9のとき約0.17である．このことから，提案手法は，正確さを優先した3章の手法に比べて，出力タイミングの同時性を高くすることができることがわかる．

次に，本手法の応用システムの一つである英日同時通訳システム[Matsubara et al. 1999b]への適用の観点から本実験結果について考察する．一般的な音声通訳システムと異なり，同時通訳では，話者の発声に追従して訳文を出力

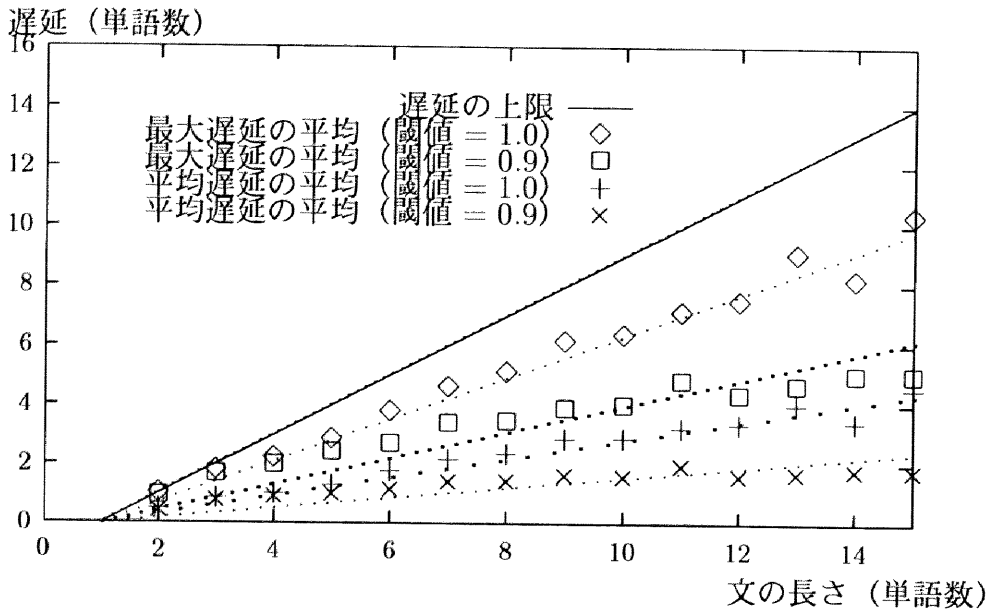

図 4.3: 文の長さと遅延の関係（ATIS コーパス）

することが求められ，訳文出力の遅れの程度は，できる限り小さいことが望ましい．同時通訳音声データの調査結果によれば，実際の同時通訳者による訳出の話者発話時間に対する遅れ時間はおよそ 1.6sec であり，単語に換算すると 6.6 単語程度である[2]．解析処理において最大遅延が 6.6 単語を超えるということは，同時通訳者よりも通訳タイミングが遅れることを意味しており，この数値は，同時通訳システムを実現する上で達成されるべき解析結果の出力遅れに関する一つの目安となる．実験に用いた 578 文のうちの最大遅延が 6.6 単語以下である文の数は，正確さを優先した 3 章の手法では 383 文と全体の 66.3%に留まるが，提案手法では 523 文と多く，その割合は全体の 90.5%を占める．このことから，提案手法に基づく同時通訳システムにより，より多くの文について実際の通訳者程度の同時性をもった通訳処理の実現が期待でき[3]，提案手法による解析結果出力の遅れの減少は，具体的な応用を考

[2] CIAIR 同時通訳コーパス [Matsubara et al. 2002] の旅行に関する対話において，英語話者発話の一単語当たりの平均発話時間は，0.24sec である．

[3] 解析結果を出力しても訳文を生成できるとは限らないため，実際に提案手法を同時通訳システム

表 4.3: 漸進的構文解析の正解率（ATIS コーパス）

閾値	正解率	妥当な項の数	出力項数
1.0	100.0%	3290	3290
0.9	98.3%	6092	6196
0.8	96.2%	6542	6802
0.7	92.7%	7104	7660
0.6	87.5%	7974	9110
0.5	78.6%	8865	11274

える上で意味を持つことを示唆している.

4.3.2 ATR 音声言語データベースを用いた実験

前節の実験は，文法規則の確率の獲得に用いたデータと実験対象が同一のクローズドテストである．オープンテストを行うには，ATIS コーパスは規模が小さいため，代わりに，日本語に対するコーパスであるが，ATR 音声言語データベース [浦谷ら 1994] を用いた実験を行った．120 対話 3051 文から文法規則 4784 規則を獲得し[4]，別の 50 対話 1134 文のうちの解析に成功した 851 文を対象とした．文の長さの平均は 3.5 文節である．1 文あたり平均で 744.2 個の項が生成されたが，そのうちの 0.7%が妥当な項であった．参考として，項の出力タイミングの例を**付録 (B)** に示す.

に適用し，翻訳実験を実施する必要があるが，それについては今後検討したい.

[4] 東京工業大学田中・徳永研究室において作成された構文木データ [1] を用いた．この構文木データは，形態素単位で解析された構文木が付与されているが，文献 [植木ら 1998] の方法に従って，文節間の係り受けを表現する構文木に変換し，それらから文法規則を獲得した．ATIS コーパスの実験と同様に，文献 [松原ら 1999] 及び [Roark & Johnson 1999] の手法に従って，文法規則を変換したものを用いた.

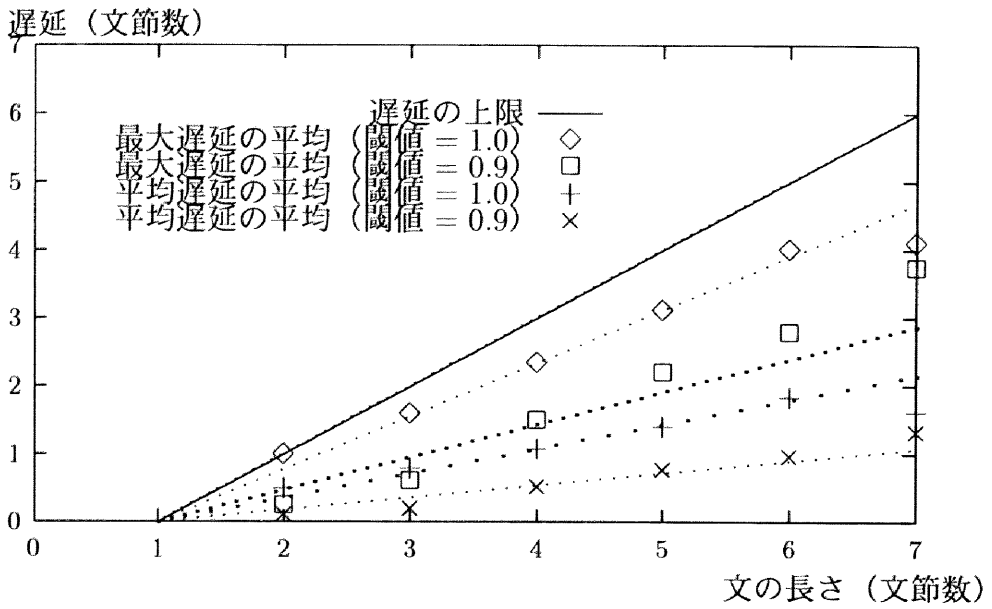

図 4.4: 文の長さと遅延の関係（ATR 音声言語データベース）

文の長さと遅延の関係を図 4.4 に示す．最大遅延の回帰直線の勾配は，閾値=1.0 のとき，約 0.78，閾値=0.9 のとき，約 0.48 である．平均遅延のそれは，閾値=1.0 のとき，約 0.36，閾値=0.9 のとき，約 0.18 である．このことから，ATIS コーパスの場合と同様に，提案手法が，正確さを優先した手法と比べて，出力の同時性が高いことがわかる．ATIS コーパスの実験と同様に，閾値を 0.5 から 1.0 まで 0.1 ずつ増加させた場合の正解率を表 4.4 に示す． 図 4.5 に示す正解率と遅延の関係を見ると，正確さを優先した手法に比べて，遅延の程度は小さくなっているが，正確さはそれほど損なわれていない．閾値=0.9 のとき，正解率は 98.4%であり，閾値=0.5 のときでも，正解率は 93.8%である．

表 4.4: 漸進的構文解析の正解率（ATR 音声言語データベース）

閾値	正解率	妥当な項の数	出力項数
1.0	100.0%	1560	1560
0.9	98.4%	3052	3103
0.8	98.3%	3158	3213
0.7	97.2%	3231	3324
0.6	95.7%	3244	3391
0.5	93.8%	3259	3473

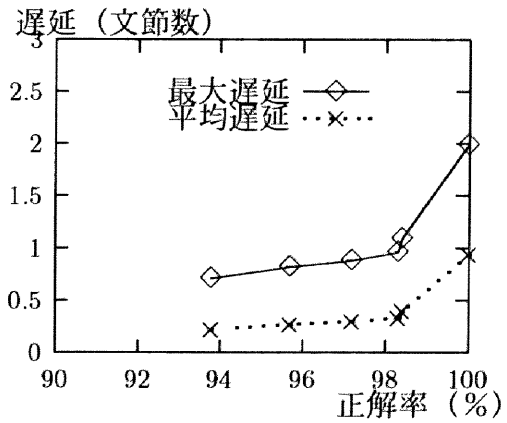

図 4.5: 正確さと遅延の関係（ATR 音声言語データベース）

4.4 4章のまとめ

本章では，漸進的構文解析において，構文構造が正しいものとなる確率をPCFGに基づいて計算する手法を提案し，その有効性をATISコーパス並びにATR音声言語データベースを用いて実験的に検証した．各構文構造に対して，残りの入力がその構文構造を妥当にする単語列である確率を，単語が入力されるごとに計算し，その確率が予め定めた閾値を超えた段階でその構文木を出力する．本手法は，漸進的構文解析において出力タイミングを制御する一つの方法として位置付けられる．正確さを優先した出力タイミング決定法に比べて，正確さをそれほど損なうことなく，より早いタイミングで構文木を出力できることが実験により示された．

付録:出力タイミングの例

以下に，実験で用いた文に対する出力タイミングの例を挙げる．なお，入力文$w_1 \cdots w_n$の断片$w_1 \cdots w_i (i \leq n)$に対する項で，妥当なものを$\sigma(w_i)$と書き，妥当でないものを$\bar{\sigma}(w_i)$と書く．$w_1 \cdots w_i$に対する項が複数個ある場合については，最も早いタイミングで出力された項のみを示す．

(A) ATIS コーパスの例

例1 I want to leave Monday morning .

入力単語	出力された項	
	閾値=1.0	閾値=0.9
I		σ(I)
want	σ(I), σ(want)	σ(want)
to	σ(to)	σ(to)
leave		σ(leave)
Monday	σ(leave)	
morning		σ(Monday),σ(morning)
.	σ(Monday),	σ(.)
	σ(morning), σ(.)	

例 2 Show me all the flights that go from Baltimore to Seattle .

入力単語	出力された項	
	閾値=1.0	閾値=0.9
show	σ(show)	σ(show)
me		σ(me)
all	σ(me)	
the		σ(all), σ(the)
flights		σ(flights)
that		σ(that)
go		
from		
Baltimore		
to		σ(go), σ(from)
		σ(Baltimore), σ(to)
Seattle		
.	σ(all), σ(the),	σ(Seattle), σ(.)
	σ(flights), σ(that),	
	σ(go), σ(from),	
	σ(Baltimore), σ(to),	
	σ(Seattle), σ(.)	

例3 Show me the last flight to leave.

入力単語	出力された項	
	閾値=1.0	閾値=0.9
show	σ(show)	σ(show)
me		σ(me)
the	σ(me)	
last		
flight		
to		σ(the), σ(last),
		σ(flight), $\overline{\sigma}$(to)
leave		σ(to) $\overline{\sigma}$(leave)
.	σ(the), σ(last),	σ(leave), σ(.)
	σ(flight), σ(to)	
	σ(leave), σ(.)	

例4 What flights do you have from Milwaukee to Tampa ?

入力単語	出力された項	
	閾値=1.0	閾値=0.9
what		
flights		
do		σ(what), $\overline{\sigma}$(what),
		σ(flights), $\overline{\sigma}$(flights),
		$\overline{\sigma}$(do)
you	σ(what), σ(flights)	σ(do), σ(you)
	σ(do), σ(you)	
have	σ(have)	σ(have)
from		
Milwaukee		
to		
Tempa	σ(from),	σ(from),
	σ(Milwaukee), σ(to)	σ(Milwaukee), σ(to)
?	σ(Tampa), σ(?)	σ(Tampa), σ(?)

(B) ATR音声言語データベースの例

例1 はい では お名前と 日付便名を お願いいたします .

入力文節	出力された項	
	閾値=1.0	閾値=0.9
はい		σ(はい)
では	σ(はい), σ(では)	σ(では)
お名前と		
日付便名を		
お願いいたします		σ(お名前と),
		σ(日付便名を),
		σ(お願いいたします)
.	σ(お名前と),	σ(.)
	σ(日付便名を),	
	σ(お願いいたします),	
	σ(.)	

例2 情報を 探してみますので 少々 お待ちください .

入力文節	出力された項	
	閾値=1.0	閾値=0.9
情報を		
探してみますので		σ(情報を),
		σ(探してみますので)
少々		
お待ちください		σ(少々),
		σ(お待ちください)
.	σ(情報を),	σ(.)
	σ(探してみますので),	
	σ(少々),	
	σ(お待ちください),	
	σ(.)	

例3 それで お泊まりになる 順番は どう いたしますか .

入力文節	出力された項	
	閾値=1.0	閾値=0.9
それで		$\overline{\sigma}$(それで)
お泊まりになる		
順番は		σ(それで),
		σ(お泊まりになる)
どう		
いたしますか		σ(順番は), σ(どう),
		σ(いたしますか)
.	σ(それで),	σ(.)
	σ(お泊まりになる),	
	σ(順番は), σ(どう),	
	σ(いたしますか)	
	σ(.)	

第5章　漸進的依存構造解析

5.1　はじめに

本章では，依存関係情報を付与した文脈自由文法に基づく漸進的な依存構造解析手法を提案する [Kato et al. 2001, 加藤ら 2003]．本手法では，範疇間の関係である到達可能性を用いて，文の入力と同時進行的に，修飾する単語と修飾される単語の間の関係である依存関係を計算する．本手法は，入力された文の断片を覆う構文木を必要としないため，構文木から依存関係を計算する手法に比べて，より効率的に依存関係を計算することができる．ATIS コーパスを用いた解析実験を行った結果，解析処理時間の短縮に，到達可能性を活用する本手法が有効であることを確認した．

本章の構成は以下の通りである．まず，次の 5.2 節で，従来の依存構造解析について説明し，それらが漸進的な依存構造解析を実現するうえでの問題について議論する．続く 5.3 節で，漸進的チャート解析に基づく依存構造解析手法について説明する．5.4 節で，到達可能性に基づく漸進的依存構造解析手法を提案する．この手法の正当性を主張する定理 5.8 の証明は付録に詳述する．5.5 節では，実験の概要を述べ，その結果に基づいて提案手法の評価を述べる．

5.2　依存構造解析

依存構造は，自然言語解析によって生成される解析結果の表現方法の一つであり，単語間の修飾・被修飾関係（依存関係）を表現する．近年，その重要性が広く認めら

れるようになった．実際，その解析手法に関する研究がなされている [Bröker 1998, Eisner 1996, Lombardo & Lesmo 1996, Sleator & Temperley 1991] とともに，依存構造を利用して構文的曖昧性を解消する方法 [Collins 1996] や，依存構造に基づいて翻訳文を生成する方法 [Alshawi et al 2000, 松原ら 1999] などが提案されている．

5.2.1 従来の依存構造解析手法

これまでに依存構造解析手法はいくつか提案されているが，それらは，

1. 依存関係の制約に従って依存構造を生成する手法
   [Eisner 1996, Sleator & Temperley 1991]

2. 文法規則に主辞情報を付与し，その情報にしたがって構文木を依存構造に変換する手法 [Bröker 1998, Collins 1996, Lombardo & Lesmo 1996]

の2つに大別できる．しかし，いずれの手法も漸進的な依存構造解析を実現する場合には問題が生じる．1. の手法を，単に，入力途中の文の断片に適用しても，漸進的依存構造解析は実現できない．その理由は，1. の方法で用いる制約が，文の依存構造の制約として妥当であっても，文の断片の依存構造の制約としては必ずしも妥当でないためである．すなわち，1. の方法では，「文全体の意味を代表する単語を除き，その他の単語は，必ず別の単語を修飾し，修飾する単語はただ一つである」という制約（唯一性の制約）を利用するが，入力途中の文の断片の中に出現する単語のいくつかについては，それが修飾する単語がまだ入力されていない場合があるため，文の断片の依存構造は唯一性の制約を満たすとは限らない．このような状況において，1. のアプローチが，どのような処理を行えばよいのかは明らかではない．一方，2. の方法では，依存構造を計算するためには，まず文全体の構文木を生成しなければならない．そのような構文木を生成してから依存構造を計算するのでは，解析の漸進性が大きく損なわれてしまう．

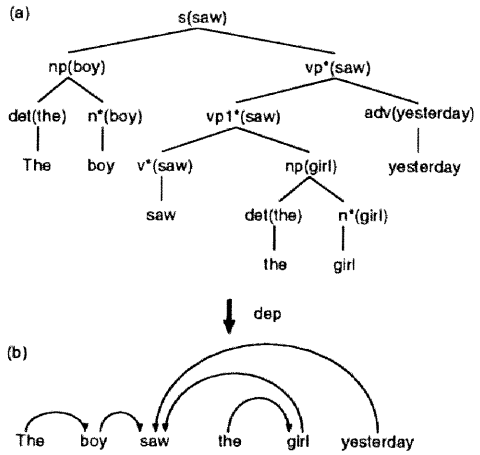

図 5.1: 構文木からの依存構造計算例 (a) 構文木，(b) (a) から計算される依存構造

5.2.2 構文木からの依存構造抽出

本章では，先頭から一単語ずつ順に入力されるたびに，それまでに入力された文の断片の依存構造を計算する手法を提案するが，まず，本手法の基礎となる依存構造解析手法である，文献 [Collins 1996] の手法について説明する．

依存関係では，修飾される単語は**主辞** (head) と呼ばれ，修飾する単語は**従属語** (dependent) と呼ばれる．以下では，主辞が w_h で従属語が w_d であるような依存関係を $\langle w_d \rightarrow w_h \rangle$ と書き，入力文中の単語間の依存関係の集合を，その文に対する依存構造と呼ぶことにする．図 5.1(b) に依存構造の例を示す．

文に対する依存構造は，文脈自由文法 (CFG) の各文法規則に対して head child と呼ばれる範疇を定めることにより，文に対する構文木から計算できる．各文法規則の右辺のただ一つの範疇が head child と呼ばれ，範疇が head child である句を，その他の句が修飾することを意味する．以下では，head child を記号 * で印付けることにする．例えば，図 5.2 の文法において，文法規則 $s \rightarrow np\ vp^*$ の head child は vp である．

各文法規則の head child を定めることにより，構文木全体の意味を代表する単語

s	$\rightarrow np\ vp^*$	det	$\rightarrow$ the
np	$\rightarrow det\ n^*$	n	$\rightarrow$ girl /boy
vp	$\rightarrow vp1^*\ adv\ /\ vp1^*\ pp$	v	$\rightarrow$ saw
$vp1$	$\rightarrow v^*\ np\ /\ v^*\ np\ np$	adv	$\rightarrow$ yesterday

図 5.2: 文法と辞書

である head word が，次のように定義される．

定義 5.1 (構文木の head word) σ を構文木とする．

1. $\sigma = [w]_X$ ならば，σ の *head word* は w である．

2. $\sigma = [[\cdots]_{X_1} \cdots [\cdots]_{X_h} \cdots [\cdots]_{X_m}]_A$ で，σ の構成に用いられた文法規則 $A \rightarrow X_1 \cdots X_h \cdots X_m$ の *head child* が X_h である，すなわち，$A \rightarrow X_1 \cdots X_h^* \cdots X_m$ ならば，σ の *head word* は $[\cdots]_{X_h}$ の *head word* である． □

以下では，構文木 σ の head word を $head(\sigma)$ と書く．構文木の head word が定まると，次に与える定義にもとづいて，構文木から依存構造を計算できる．

定義 5.2 (構文木の依存構造) σ を構文木とする．関数 dep を次のように定義する．

1. $\sigma = [w]_X$ ならば，$dep(\sigma) = \phi$ である．

2. $\sigma = [[\cdots]_{X_1} \cdots [\cdots]_{X_h} \cdots [\cdots]_{X_m}]_A$ とする．このとき，文法規則 $A \rightarrow X_1 \cdots X_h \cdots X_m$ の *head child* が X_h である，すなわち，$A \rightarrow X_1 \cdots X_h^* \cdots X_m$ ならば，

$$dep(\sigma) = d(\sigma) \cup \bigcup_{i=1}^{m} dep([\cdots]_{X_i})$$

である．ただし，$d(\sigma)$ は次のように定義する．

$$d(\sigma) = \{\langle w_d \rightarrow head([\cdots]_{X_h})\rangle \mid w_d = head([\cdots]_{X_j})(1 \leq j \leq m, j \neq h)\}$$

□

図5.1は，(a)の構文木から定義5.2に従って計算すると(b)のように依存構造が得られることを示している．範疇の隣の括弧で囲まれた単語は，その範疇を根とする構文木のhead wordである．

5.3 漸進的チャート解析と依存構造計算

単語の出現順序に従って，漸進的に単語間の依存関係を計算するには，次の手続きを実行すればよい．

1. 漸進的チャート解析のような，CFGベースの漸進的構文解析手法により，入力文の断片に対する部分構文木を生成する．
2. 前節で述べた[Collins 1996]の方法にしたがって，1.により生成された部分構文木から単語間の依存関係を計算する．

本節では，漸進的チャート解析に基づいて依存構造を計算する方法について述べる．

文の断片に対する依存構造は，漸進的チャート解析によって生成された項に，定義5.2で定義した依存構造を計算する関数depを適用することにより計算できる．例として，図5.2の文法を用いた文の断片 "The boy saw" の解析を考える．漸進的チャート解析では，"The boy saw" に対する項の一つとして，例えば，

$$[[[the]_{det}[boy]_{n}]_{np}[[[saw]_{v}[?]_{np}]_{vp1}[?]_{adv}]_{vp}]_{s} \tag{5.1}$$

を生成する．項(5.1)に含まれる項のhead wordは図5.3(a)のように得られ，文の断片 "The boy saw" に対する依存構造が，図5.3(b)のように計算できる．

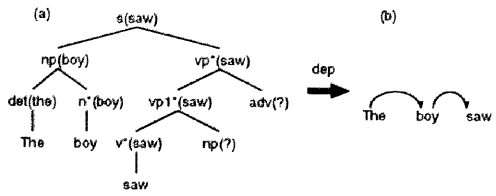

図 5.3: 文の断片に対する依存構造計算例

5.4 漸進的依存構造解析

前節で述べた手法により，漸進的に文の断片に対する依存構造を計算することができる．しかしながら，この方法は効率的ではなく，同じ依存構造をもつ項を多く作ってしまう可能性がある．例えば，漸進的チャート解析は “The boy saw” に対して式 (5.1) の項だけでなく，次のような項も生成する:

$$[[[the]_{det}[boy]_n]_{np}[[[saw]_v[?]_{np}]_{vp1}[?]_{pp}]_{vp}]_s \tag{5.2}$$

$$[[[the]_{det}[boy]_n]_{np}[[[saw]_v[?]_{np}[?]_{np}]_{vp1}[?]_{adv}]_{vp}]_s \tag{5.3}$$

$$[[[the]_{det}[boy]_n]_{np}[[[saw]_v[?]_{np}[?]_{np}]_{vp1}[?]_{pp}]_{vp}]_s \tag{5.4}$$

項 (5.1)〜(5.4) からは，同じ依存構造が得られる．単に依存構造を求めるだけであれば，このような同じ依存構造をもつ項をいくつも生成することは，解析効率の観点から望ましくない．同じ依存構造をもつ項を生成してしまう原因の一つとして，文法規則適用操作が挙げられる．すなわち，ある項に対する文法規則の適用の仕方が数通り存在したとしても，規則適用の結果得られる項のいくつかは同じ依存構造をもつ場合がある．漸進的チャート解析では，不活性弧のみならず活性弧に対しても文法規則を適用するため，その影響は大きい．そこで本節では，活性弧に対する文法規則の適用に代わる操作として，到達可能性による項の結合操作を提案し，これによって漸進的に依存構造が計算できることを示す．

5.4.1 到達可能性

到達可能性は範疇間の関係であり，次のように定義される．

定義 5.3 (到達可能性) 文法規則 $A \rightarrow XY \cdots Z$ が存在するならば，$X \rightsquigarrow A$ と書く．$\rightsquigarrow^*$ を $\rightsquigarrow$ の反射推移閉包とする．$X \rightsquigarrow^* Y$ ならば，X が Y に**到達可能** *(reachable)* であるという． □

X が Y に到達可能であるとは，Y を根とする構文木が X を左端の子孫としてもつことが可能であることを意味する．

依存構造を計算するためには，構文木の head word を定める必要があるが，文法規則を実際に適用することなく head word を定めるために，本手法では，到達可能性を分類する．すなわち，X が Y に到達可能であるということは，項 $[\cdots]_X$ に対していくつかの文法規則を適用すると，範疇が Y である項が得られることを意味するが，そのようにして得られた項の head word ともとの項 $[\cdots]_X$ の head word とがどのような関係にあるかという観点から分類する．なお，以下では，文法規則 r の head child の位置を $hc(r)$ と書く．

定義 5.4 $\overset{h}{\rightsquigarrow}$，及び $\overset{d}{\rightsquigarrow}$ を範疇間の関係とし，次のように定義する．$hc(r) = 1$ である文法規則 $r = A \rightarrow XY \cdots Z$ が存在するならば，$X \overset{h}{\rightsquigarrow} A$ である．$hc(r) \neq 1$ である文法規則 $r = A \rightarrow XY \cdots Z$ が存在するならば，$X \overset{d}{\rightsquigarrow} A$ である． □

$X \overset{h}{\rightsquigarrow}^* Y$ は，項 $[...]_X$ にいくつか文法規則を適用すると，範疇が Y である項が得られ，その項に $[...]_X$ の head word が伝搬することを意味する．一方，$X \rightsquigarrow^* \overset{d}{\rightsquigarrow} \rightsquigarrow^* Y$ は，項 $[...]_X$ にいくつか文法規則を適用すると，範疇が Y である項が得られ，その項に $[...]_X$ の head word が伝搬しないことを意味する．

5.4.2 到達可能性に基づく漸進的な依存構造解析

以上の準備のもとに，到達可能性に基づく漸進的依存構造解析手法を提案する．この手法は，次の3つの手順からなる．

（手順1） 通常の上昇型チャート解析によって項を生成し，

（手順2） 生成されたこれらの項を到達可能性を用いて結合し，

（手順3） 結合された項から依存構造を計算する．

本提案手法は，それによって求められる依存構造が，5.3節で述べた手法で求められるものと一致することが理論的に保証されるばかりでなく，それより格段に効率的な漸進的依存構造解析を実現するものである．以下，これらの事実を明らかにする．

上の手順2において到達可能性により結合される項を**結合項**と呼ぶことにし，これを以下のように定義する．

定義 5.5 (結合項) $\sigma_i (i = 1, \ldots, n)$ を通常の上昇型チャート解析により生成される項とし，各 i について σ_i の範疇を X_i，σ_i の最左未決定項の範疇を Y_i とする．$R_i \in \{\&_h, \&_d\}$ （$\&_h$ は項が関係 $\overset{h}{\rightsquigarrow}^*$ により結合されたことを，$\&_d$ は項が関係 $\rightsquigarrow^* \overset{d}{\rightsquigarrow} \rightsquigarrow^*$ により結合されたことを表す記号である）として，次の *(i)* と *(ii)* を満たすような並び $\sigma_1 R_1 \cdots R_{i-1} \sigma_i R_i \cdots R_{n-1} \sigma_n$ を結合項という．

(i) $R_i = \&_h$ ならば，$X_{i+1} \overset{h}{\rightsquigarrow}^* Y_i$.

(ii) $R_i = \&_d$ ならば，$X_{i+1} \rightsquigarrow^* \overset{d}{\rightsquigarrow} \rightsquigarrow^* Y_i$. □

手順2では，通常の上昇型チャート解析により生成された項を結合する．その操作は次の通りである．

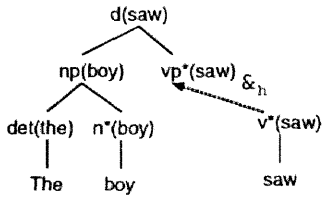

図 5.4: 結合項

(手順 2) (i,j,σ) をチャート中の活性弧とし，項 σ の最左未決定項を $[?]_Y$ とする．また，$(j,k,\sigma_1 R_1 \cdots R_{n-1}\sigma_n)$ をチャート中の結合項をラベルにもつ弧とし，σ_1 の範疇を X とする．このとき，$X \overset{h}{\rightsquigarrow}{}^{*} Y$ ならば，弧 $(i,k,\sigma\&_h\sigma_1 R_1 \cdots R_{n-1}\sigma_n)$ をチャートに追加する．$X \rightsquigarrow^{*}\overset{d}{\rightsquigarrow}\rightsquigarrow^{*} Y$ ならば，弧 $(i,k,\sigma\&_d\sigma_1 R_1 \cdots R_{n-1}\sigma_n)$ をチャートに追加する．

例えば，上昇型チャート解析により生成される項

$$[[[the]_{det}[boy]_n]_{np}[?]_{vp}]_s \tag{5.5}$$

$$[saw]_v \tag{5.6}$$

は図 5.4 のように結合される．

手順 3 では結合項から依存構造を計算する．その計算を定義するために，まず結合項の head word を定義する．

定義 5.6 (結合項の head word) τ を結合項とする．

1. τ が通常の上昇型チャート解析で生成される項ならば，τ の *head word* は $head(\tau)$.

2. $\tau = \sigma_1 R_1 \sigma_2 R_2 \cdots R_{n-1}\sigma_n$ とし，$\sigma_1 = [[\cdots]_{X_1} \cdots [\cdots]_{X_{k-1}}[?]_{X_k} \cdots [?]_{X_m}]_A$ とする．このとき，$R_1 = \&_h$ かつ，X_k が *head child* ならば，τ の *head word* は，$\sigma_2 R_2 \cdots R_{n-1}\sigma_n$ の *head word* である．そうでないとき，τ の *head word* は $head(\sigma_1)$ である． □

以下では，結合項τのhead wordを$head_C(\tau)$と書く．手順3では，結合項から次に与える定義にもとづいて依存構造を計算する．

定義 5.7 (結合項の依存構造) τを結合項とする．関数dep_Cを次のように定義する．

1. τが通常の上昇型チャート解析により生成される項ならば，$dep_C(\tau) = dep(\tau)$.

2. $\tau = \sigma_1 R_1 \sigma_2 R_2 \cdots R_{n-1}\sigma_n$とし，$\sigma_1 = [[\cdots]_{X_1} \cdots [\cdots]_{X_{k-1}} [?]_{X_k} \cdots [?]_{X_m}]_A$とする．このとき，

$$dep_C(\tau) = d_C(\tau) \cup dep_C(\sigma_2 R_2 \cdots R_{n-1}\sigma_n) \cup \bigcup_{i=1}^{k-1} dep([\cdots]_{X_i})$$

ただし，$d_C(\tau)$を以下のように定義する．また，$h = hc(A \rightarrow X_1 \cdots X_m)$とおく．

(a1) $R_1 = \&_h$で$h = k$ならば，

$$d_C(\tau) = \{\langle w_d \rightarrow head_C(\sigma_2 R_2 \cdots R_{n-1}\sigma_n)\rangle \mid w_d = head([\cdots]_{X_j})(1 \leq j \leq m, j \neq h)\}$$

(a2) $R_1 = \&_h$で$h \neq k$ならば，

$$\begin{aligned} &d_C(\tau) \\ &= \{\langle w_d \rightarrow head([\cdots]_{X_h})\rangle \mid \\ &\quad w_d = head([\cdots]_{X_j})(1 \leq j \leq m, j \neq h, j \neq k) \\ &\quad \text{または } w_d = head_C(\sigma_2 R_2 \cdots R_{n-1}\sigma_n)\} \end{aligned}$$

(b) $R_1 = \&_d$ならば，$d_C(\tau) = d(\sigma_1)$. □

提案手法では，項が$\&_h$で結合された場合には，結合項中の項のhead wordを伝搬させる（図5.5(a)）．項が$\&_d$で結合される場合には，head wordは伝搬されない（図5.5(b)）．このように，到達可能性を活用することにより，活性弧に文法規則を適用

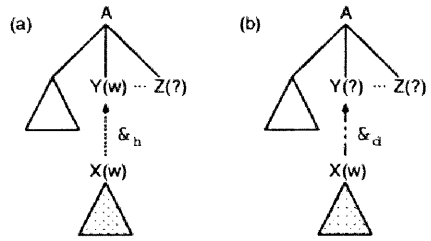

図 5.5: 到達可能性に基づく head word の伝搬

することなく，文の断片に対する依存構造を漸進的に計算することができる．一般に，ある項への文法規則の適用の仕方は何通りもあるのに対して，ここで定義した到達可能性の場合の数は高々2通りであるので，より効率的に依存構造を計算できる．さらに，提案手法が計算する依存構造は，5.3節で述べた手法のそれと等価である．すなわち次の定理が成り立つ．

定理 5.8 $\mathbf{w}$ を単語列とし，$I(\mathbf{w})$ を漸進的チャート解析により生成された $\mathbf{w}$ に対する項からなる集合とし，$C(\mathbf{w})$ を $\mathbf{w}$ に対する結合項からなる集合とする．このとき，$dep(I(\mathbf{w})) = dep_C(C(\mathbf{w}))$. □

以下では，定理5.8の証明を行う．

まず，漸進的チャート解析により生成された任意の項に対して，それと同じ依存構造をもつ結合項が存在することを示し，次に，任意の結合項に対して，それと同じ依存構造をもつ項が漸進的チャート解析により生成されることを示す．

はじめに，証明に用いるいくつかの定義を与える．$T(\mathbf{w})$ を通常のチャート解析により生成される，単語列 $\mathbf{w}$ に対する項を表す集合とする．σ を項としたとき，$cat(\sigma)$ で σ の範疇を表す．$\tau = \sigma_1 R_1 \cdots R_{n-1}\sigma_n$ を結合項としたとき，$cat_C(\tau) = cat(\sigma_1)$ とする．項 σ に対して文法規則 $r_p(p = 1, \ldots, q)$ を順に (操作 4) に従って適用した結果得られる項を $apply(\sigma, r_1 \cdots r_q)$ と書く．このように表記するときには，暗黙の了解として，σ に対して r_1, r_2, ..., r_q が次々と適用できるものとする．項 σ の最左未決

定項を項σ'で置き換えた結果得られる項を$rep(\sigma, \sigma')$と書く.

漸進的チャート解析により生成される，単語列$\mathbf{w}$に対する項の集合$I(\mathbf{w})$は，次のように定義できる.

定義 5.9 (漸進的チャート解析により生成される項) 単語列に対して項の集合を返す関数I_nを次のように定義する.

1. 任意の単語列$\mathbf{w}$に対して，$I_0(\mathbf{w}) = T(\mathbf{w})$.

2. 任意の単語列$\mathbf{w}$に対して，$I_{i+1}(\mathbf{w}) = I_{op4,i+1}(\mathbf{w}) \cup I_{op5,i+1}(\mathbf{w})$. ただし，$I_{op4,i+1}(\mathbf{w})$，及び$I_{op5,i+1}(\mathbf{w})$をそれぞれ次のように定義する.

$$I_{op4,i+1}(\mathbf{w}) = \{\sigma \mid \sigma' \in I_i(\mathbf{w}),\ \text{及び文法規則}\ r\ \text{が存在して}, \sigma = apply(\sigma', r)\}$$

$$I_{op5,i+1}(\mathbf{w}) = \{\sigma \mid \sigma_1 \in T(\mathbf{w}_1), \sigma_2 \in I_i(\mathbf{w}_2)(\mathbf{w} = \mathbf{w}_1\mathbf{w}_2)\ \text{が存在し},\ \sigma = rep(\sigma_1, \sigma_2)\}$$

更に，I_nを用いて，$I(\mathbf{w})$を次のように定義する.

$$I(\mathbf{w}) = \bigcup_{n=0}^{\infty} I_n(\mathbf{w})$$

□

$I_n(\mathbf{w})$は，$\mathbf{w}$に通常のチャート解析を施して生成される項に対して，(操作 4)，(操作 5) をn回適用した結果得られる項の集合である.

提案手法により生成される，単語列$\mathbf{w}$に対する結合項の集合$C(\mathbf{w})$は次のように定義できる.

定義 5.10 (結合項の集合) 単語列に対して結合項の集合を返す関数C_nを次のように定義する.

1. 任意の単語列$\mathbf{w}$に対して，$C_1(\mathbf{w}) = T(\mathbf{w})$.

2. 任意の単語列 $\mathbf{w}$ に対して，

$$C_{i+1}(\mathbf{w}) = \{\tau \mid \sigma_1 \in T(\mathbf{w}_1), \tau_2 \in C_i(\mathbf{w}_2)(\mathbf{w} = \mathbf{w}_1\mathbf{w}_2) \text{が存在して，} \\ \tau\text{は結合項}\sigma_1\&_h\tau_2\text{または}\sigma_1\&_d\tau_2\text{である}\}$$

$C(\mathbf{w})$ を次のように定義する．

$$C(\mathbf{w}) = \bigcup_{n=1}^{\infty} C_n(\mathbf{w})$$

□

$C_n(\mathbf{w})$ は n 個の項を結合して得られる結合項の集合である．

以下，定理 5.8 の証明に必要な補題を順に証明する．

補題 5.11 任意の n に対して，$\sigma \in I_n(\mathbf{w})$ ならば，$\sigma' \in I_{op5,j}(\mathbf{w}) \cup T(\mathbf{w})(j \leq n)$，及び文法規則 $r_p(p = 1, \ldots, q = n - j)$ が存在し，$\sigma = apply(\sigma', r_1 \cdots r_q)$ である．

証明 $I_n(\mathbf{w})$ の定義より明らか． □

補題 5.12 $\sigma \in I(\mathbf{w})$ とし，$r_1 r_2 \cdots r_q$ を q 個の文法規則の列とするとき，$dep(apply(\sigma, r_1 \cdots r_q)) = dep(\sigma)$ である．

証明 q に関する帰納法により証明する．

$q = 0$ のとき，$apply(\sigma, \varepsilon) = \sigma$ であり，補題が成り立つのは明らか．

$q = n$ のとき，補題が成り立つと仮定する．$q = n + 1$ とする．$r_1 = A \rightarrow XY \cdots Z$，$\mathbf{r} = r_2 \cdots r_{n+1}$ とおくと，

$$apply(\sigma, r_1 r_2 \cdots r_{n+1}) = apply([\sigma[?]_Y \cdots [?]_Z]_A, \mathbf{r}) \tag{5.7}$$

である．このとき，

$$\begin{aligned} dep(apply(\sigma, r_1 r_2 \cdots r_{n+1})) &= dep(apply([\sigma[?]_Y \cdots [?]_Z]_A, \mathbf{r})) && ((5.7)) \\ &= dep([\sigma[?]_Y \cdots [?]_Z]_A) && \text{(帰納法の仮定)} \\ &= dep(\sigma) && \text{(定義 5.2)} \end{aligned}$$

以上から，帰納法により補題 5.12 が証明された． □

補題 5.13 $\sigma \in I(\mathbf{w})$ とし，$r_p(p = 1, \ldots, q)$ を $hc(r_p) = 1$ であるような文法規則とする．このとき，$head(apply(\sigma, r_1 \cdots r_q)) = head(\sigma)$ である．

証明 q に関する帰納法で証明する．

$q = 0$ のとき補題が成り立つのは明らか．

$q = n$ のとき補題が成り立つと仮定する．そこで，$q = n+1$ とする．$r_1 = A \rightarrow XY \cdots Z$，$\mathbf{r} = r_2 \cdots r_{n+1}$ とおく．このとき，(5.7) である．一方，$hc(r_1) = 1$ であるので，定義 5.1 より，

$$head([\sigma[?]_Y \cdots [?]_Z]_A) = head(\sigma) \tag{5.8}$$

である．したがって，

$$\begin{aligned} head(apply(\sigma, r_1 r_2 \cdots r_{n+1})) &= head(apply([\sigma[?]_Y \cdots [?]_Z]_A, \mathbf{r})) \quad ((5.7)) \\ &= head([\sigma[?]_Y \cdots [?]_Z]_A) \quad (\text{帰納法の仮定}) \\ &= head(\sigma) \quad ((5.8)) \end{aligned}$$

以上から，帰納法により補題 5.13 が証明された． □

補題 5.14 $\sigma \in I(\mathbf{w})$ とし，$head(\sigma) =?$ とする．$r_p(p = 1, \ldots, q)$ を文法規則とする．このとき，$head(apply(\sigma, r_1 \cdots r_q)) =?$ である．

証明 q に関する帰納法により証明する．

$q = 0$ のとき，補題が成り立つのは明らか．

$q = n$ のとき補題が成り立つと仮定する．$q = n+1$ とする．$r_1 = A \rightarrow XY \cdots Z$，$\mathbf{r} = r_2 \cdots r_{n+1}$ とおく．

まず，$apply(\sigma, r_1) = [\sigma[?]_Y \cdots [?]_Z]_A$ について，

$$head([\sigma[?]_Y \cdots [?]_Z]_A) =? \tag{5.9}$$

が成り立つことを，$hc(r_1) = 1$ のときとそうでないときに場合を分けて示す．

$hc(r_1) = 1$ のとき：

$$\begin{aligned} head([\sigma[?]_Y \cdots [?]_Z]_A) &= head(\sigma) \text{ (定義 5.1)} \\ &= ? \qquad \text{(補題の仮定)} \end{aligned}$$

$hc(r_1) \neq 1$ のとき：H を r_1 の head child とすると，

$$\begin{aligned} head([\sigma[?]_Y \cdots [?]_H \cdots [?]_Z]_A) &= head([?]_H) \quad \text{(定義 5.1)} \\ &= ? \qquad\qquad \text{(定義 5.1)} \end{aligned}$$

である．

以上より，(5.9) が成り立つ．

$r_1\mathbf{r} = r_1 r_2 \cdots r_{n+1}$ に対して，$\mathbf{r}$ の長さが n であり，(5.9) が成り立つことから，帰納法の仮定より，

$$head(apply([\sigma[?]_Y \cdots [?]_Z]_A, \mathbf{r})) = ? \tag{5.10}$$

である．したがって，

$$\begin{aligned} head(apply(\sigma, r_1 r_2 \cdots r_{n+1})) &= head(apply([\sigma[?]_Y \cdots [?]_Z]_A, \mathbf{r})) \quad ((5.7)) \\ &= ? \qquad\qquad ((5.10)) \end{aligned}$$

以上から，帰納法により補題 5.14 が証明された． □

補題 5.15 $\sigma \in I(\mathbf{w})$ とし，$r_p (p = 1, \ldots, q)$ を文法規則とする．さらに，ある $r_i (1 \leq i \leq q)$ が存在して，$hc(r_i) \neq 1$ とする．このとき，$head(apply(\sigma, r_1 \cdots r_q)) = ?$ である．

証明 $\sigma' = apply(\sigma, r_1 \cdots r_{i-1})$ とおく．$r_i = A \rightarrow XY \cdots Z$ とする．$hc(r_i) \neq 1$ であ

るので，r_i の head child を H とすると，

$$\begin{aligned}
&head(apply(\sigma', r_i)) \\
=&head([\sigma'[?]_Y \cdots [?]_H \cdots [?]_Z]_A) && (\text{applyの定義}) \\
=&head([?]_H) && (\text{定義 } 5.1) \\
=&? && (\text{定義 } 5.1)
\end{aligned}$$

したがって，補題 5.14 より，$head(apply(apply(\sigma', r_i), r_{i+1} \cdots r_q)) = ?$ である．すなわち $head(apply(\sigma, r_1 \cdots r_{i-1} r_i r_{i+1} \cdots r_q)) = ?$ であり，補題 5.15 が証明された． □

補題 5.16 $\sigma_1 \in T(\mathbf{w}_1)$，かつ σ_1 の最左未決定項を $[?]_{X_k}$ とする．また，$\sigma_2 \in I(\mathbf{w}_2)$，$cat(\sigma_2) = X_k$ とする．そこで，$\sigma = rep(\sigma_1, \sigma_2)$ とする．また，$\tau_2 \in C(\mathbf{w}_2)$ とし，$\tau = \sigma_1 \&_h \tau_2$ を結合項とする．さらに，$head(\sigma_2) = head_C(\tau_2)$，$dep(\sigma_2) = dep_C(\tau_2)$ とする．このとき，

$$head(\sigma) = head_C(\tau) \text{ かつ } dep(\sigma) = dep_C(\tau)$$

である．

証明 $\sigma_1 = [[\cdots]_{X_1} \cdots [\cdots]_{X_{k-1}} [?]_{X_k} \cdots [?]_{X_m}]_A$ とおき，

$h = hc(A \to X_1 \cdots X_{k-1} X_k \cdots X_m)$ とおく．

まず，

$$head(\sigma) = head_C(\tau)$$

を $h = k$ のときと $h \neq k$ のときに場合分けして証明する．

$h = k$ のとき：

$$\begin{aligned}
head(\sigma) =& head(\sigma_2) && (\text{定義 } 5.1) \\
=& head_C(\tau_2) && (\text{補題の仮定}) \\
=& head_C(\tau) && (\text{定義 } 5.6)
\end{aligned}$$

$h \neq k$ のとき：

$$\begin{aligned} head(\sigma) &= head([\cdots]_{X_h}) && \text{(定義 5.1)} \\ &= head_C(\tau) && \text{(定義 5.6)} \end{aligned}$$

以上より，$head(\sigma) = head_C(\tau)$ である．

次に，$dep(\sigma) = dep_C(\tau)$ を証明する．そのためには，定義 5.2，定義 5.7，及び補題の仮定 $dep(\sigma_2) = dep_C(\tau_2)$ から，$d(\sigma) = d_C(\tau)$ を示せば十分である．$h = k$ と $h \neq k$ の場合に分けて証明する．

$h = k$ のとき：

$$\begin{aligned} d(\sigma) &= \{\langle w_d \rightarrow head(\sigma_2)\rangle \mid w_d = head([\cdots]_{X_i})(1 \leq i \leq m, i \neq h)\} && \text{(定義 5.2)} \\ &= \{\langle w_d \rightarrow head_C(\tau_2)\rangle \mid w_d = head([\cdots]_{X_i})(1 \leq i \leq m, i \neq h)\} && \text{(補題の仮定)} \\ &= d_C(\tau) && \text{(定義 5.7)} \end{aligned}$$

$h \neq k$ のとき：

$$\begin{aligned} d(\sigma) = \; & \{\langle w_d \rightarrow head([\cdots]_{X_h})\rangle \mid w_d = head([\cdots]_{X_i})(1 \leq i \leq m, i \neq h, i \neq k) \\ & \text{または } w_d = head(\sigma_2)\} && \text{(定義 5.2)} \\ = \; & \{\langle w_d \rightarrow head([\cdots]_{X_h})\rangle \mid w_d = head([\cdots]_{X_i})(1 \leq i \leq m, i \neq h, i \neq k) \\ & \text{または } w_d = head_C(\tau_2)\} && \text{(補題の仮定)} \\ = \; & d_C(\tau) && \text{(定義 5.7)} \end{aligned}$$

よって，$d(\sigma) = d_C(\tau)$ となり，定義 5.2，定義 5.7，及び補題の仮定により $dep(\sigma) = dep_C(\tau)$ である．

以上から，補題 5.16 は証明された． □

補題 5.17 $\sigma_1 \in T(\mathbf{w}_1)$，かつ σ_1 の最左未決定項を $[?]_{X_k}$ とする．また，$\sigma_2 \in I(\mathbf{w}_2)$，$cat(\sigma_2) = X_k$ とし，$\sigma = rep(\sigma_1, \sigma_2)$ とする．また，$\tau_2 \in C(\mathbf{w}_2)$ とし，$\tau = \sigma_1 \&_d \tau_2$ を結合項とする．さらに，$head(\sigma_2) =?$，$dep(\sigma_2) = dep_C(\tau_2)$ とする．このとき，

$$head(\sigma) = head_C(\tau) \text{ かつ } dep(\sigma) = dep_C(\tau)$$

である.

証明 この証明はほとんど補題5.16の証明と並行に進められる.

$\sigma_1 = [[\cdots]_{X_1} \cdots [\cdots]_{X_{k-1}} [?]_{X_k} \cdots [?]_{X_m}]_A$ とおき,

$h = hc(A \rightarrow X_1 \cdots X_{k-1} X_k \cdots X_m)$ とおく.

まず,

$$head(\sigma) = head_C(\tau)$$

を $h = k$ と $h \neq k$ の場合に分けて証明する.

$h = k$ のとき : $head(\sigma) = head(\sigma_2) =?$, $head_C(\tau) = head([?]_{X_k}) =?$ である.

$h \neq k$ のとき : $head(\sigma) = head([\cdots]_{X_h}) = head_C(\tau)$ である.

以上より, $head(\sigma) = head_C(\tau)$ である.

次に, $dep(\sigma) = dep_C(\tau)$ を証明するが, これは $d(\sigma) = d_C(\tau)$ を示せば十分である.

$h = k$ のとき : $d(\sigma)$, $d_C(\tau)$ はともに ϕ である. $h \neq k$ のとき :

$$\begin{aligned} d(\sigma) &= \{\langle w_d \rightarrow head([\cdots]_{X_h})\rangle \mid w_d = head([\cdots]_{X_i})(1 \leq i \leq m, i \neq h)\} \\ &= d(\sigma_1) \\ &= d_C(\tau) \end{aligned}$$

よって, $d(\sigma) = d_C(\tau)$ となり, $dep(\sigma) = dep_C(\tau)$ である.

以上より, 補題5.17が証明された. □

補題 5.18 任意の n と単語列 $\mathbf{w}$ に対して, $\sigma \in I_n(\mathbf{w})$ ならば, 次の *1.* と *2.* を満たす $\tau \in C(\mathbf{w})$ が存在する.

1. $dep(\sigma) = dep_C(\tau)$.

2. $\sigma \in I_{op5,n}(\mathbf{w}) \cup T(\mathbf{w})$ のときには，$cat(\sigma) = cat_C(\tau)$ かつ $head(\sigma) = head_C(\tau)$.

証明 n に関する帰納法により証明する.

$n = 0$ のとき，$\sigma \in I_0(\mathbf{w}) = T(\mathbf{w}) \subseteq C(\mathbf{w})$ であり，$\tau = \sigma$ とおけば補題が成り立つのは明らかである.

$n = l$ のとき補題が成り立つと仮定する．$\sigma \in I_{l+1}(\mathbf{w})$ とすると，$\sigma \in I_{op4,l+1}(\mathbf{w})$ または $\sigma \in I_{op5,l+1}(\mathbf{w})$ である.

$\sigma \in I_{op4,l+1}(\mathbf{w})$ のときには，1. の条件 $dep(\sigma) = dep_C(\tau)$ を満たす $\tau \in C(\mathbf{w})$ が存在することを示せばよい．$\sigma \in I_{op4,l+1}(\mathbf{w})$ であるから，文法規則 r，及び $\sigma' \in I_l(\mathbf{w})$ が存在して，$\sigma = apply(\sigma', r)$ である．補題 5.12 より，$dep(\sigma) = dep(\sigma')$ である．$\sigma' \in I_l(\mathbf{w})$ であるので，帰納法の仮定より，$dep(\sigma') = dep_C(\tau)$ である $\tau \in C(\mathbf{w})$ が存在し，$dep(\sigma) = dep(\sigma') = dep_C(\tau)$ である.

$\sigma \in I_{op5,l+1}(\mathbf{w})$ のときには，$\sigma_1 \in T(\mathbf{w}_1)$，$\sigma_2 \in I_l(\mathbf{w}_2)$, $\mathbf{w} = \mathbf{w}_1\mathbf{w}_2$ である σ_1, σ_2 が存在して，$\sigma = rep(\sigma_1, \sigma_2)$ である．$\sigma_2 \in I_l(\mathbf{w}_2)$ であるので，補題 5.11 より，$\sigma_2' \in I_{op5,j}(\mathbf{w}_2) \cup T(\mathbf{w}_2)(j \leq l)$ 及び文法規則 $r_p(p = 1, \ldots, q = l - j)$ が存在して，$\sigma_2 = apply(\sigma_2', r_1 \cdots r_q)$ である．このとき，帰納法の仮定より，

$$dep(\sigma_2') = dep_C(\tau_2) \tag{5.11}$$

$$cat(\sigma_2') = cat_C(\tau_2) \tag{5.12}$$

$$head(\sigma_2') = head_C(\tau_2) \tag{5.13}$$

である $\tau_2 \in C(\mathbf{w}_2)$ が存在する.

そこで，σ_1 の最左未決定項の範疇を X とする.

すべての $r_p(1 \leq p \leq q)$ について $hc(r_p) = 1$ のときには，$cat(\sigma_2') \overset{h}{\rightsquigarrow}^* X$ である，すなわち，$cat_C(\tau_2) \overset{h}{\rightsquigarrow}^* X$ であるので，定義 5.10 より $\sigma_1 \&_h \tau_2 \in C(\mathbf{w})$ で

ある．$\sigma_1 \&_h \tau_2$ に対して，

$$dep(\sigma_2) = dep(\sigma_2') \quad (\text{補題 } 5.12)$$
$$= dep(\tau_2) \quad ((5.11))$$

$$head(\sigma_2) = head(\sigma_2') \quad (\text{補題 } 5.13)$$
$$= head(\tau_2) \quad ((5.13))$$

であるので，補題 5.16 より，$dep(\sigma) = dep_C(\sigma_1 \&_h \tau_2)$，$head(\sigma) = head_C(\sigma_1 \&_h \tau_2)$ である．また，$cat(\sigma) = cat(\sigma_1) = cat_C(\sigma_1 \&_h \tau_2)$ である．

ある $r_p (1 \leq p \leq q)$ について，$hc(r_p) \neq 1$ のときには，$cat(\sigma_2') \rightsquigarrow^* \overset{d}{\rightsquigarrow} \rightsquigarrow^* X$，すなわち $cat_C(\tau_2) \rightsquigarrow^* \overset{d}{\rightsquigarrow} \rightsquigarrow^* X$ であるので，定義 5.10 より $\sigma_1 \&_d \tau_2 \in C(\mathbf{w})$ である．補題 5.12 より，$dep(\sigma_2) = dep(\sigma_2')$ であるので，$dep(\sigma_2) = dep_C(\tau_2)$ である．他方，補題 5.15 より，$head(\sigma_2) =?$ である．したがって，補題 5.17 より，$dep(\sigma) = dep_C(\sigma_1 \&_d \tau_2)$，$head(\sigma) = head_C(\sigma_1 \&_d \tau_2)$ である．さらに，$cat(\sigma) = cat(\sigma_1) = cat_C(\sigma_1 \&_d \tau_2)$ である．

以上から，$n = l + 1$ のとき補題は成り立つ．すなわち，帰納法により補題 5.18 が証明された． □

補題 5.19 任意の n，及び単語列 $\mathbf{w}$ に対して，$\tau \in C_n(\mathbf{w})$ ならば，$dep(\sigma) = dep_C(\tau)$，$head(\sigma) = head_C(\tau)$ かつ $cat(\sigma) = cat_C(\tau)$ を満たす $\sigma \in I(\mathbf{w})$ が存在する．

証明 n に関する帰納法で証明する．

$n = 1$ のとき，$\tau \in C_1(\mathbf{w}) = T(\mathbf{w}) \subseteq I(\mathbf{w})$ であるので，$\sigma = \tau$ とおけば明らか．

$n = l$ のとき補題が成り立つと仮定する．$\tau \in C_{l+1}(\mathbf{w})$ とすると，ある $\sigma_1 \in T(\mathbf{w}_1)$，$\tau_2 \in C_l(\mathbf{w}_2)$ $(\mathbf{w} = \mathbf{w}_1\mathbf{w}_2)$，$R \in \{\&_h, \&_d\}$ が存在して，$\tau = \sigma_1 R \tau_2$ とおける．帰納法の仮定より，$dep(\sigma_2) = dep_C(\tau_2)$ $head(\sigma_2) = head_C(\tau_2)$ かつ $cat(\sigma_2) = cat_C(\tau_2)$ を満たす $\sigma_2 \in I(\mathbf{w}_2)$ が存在する．

そこで，σ_1 の最左未決定項の範疇を X とする．

$R = \&_h$ のときには，$cat_C(\tau_2) \overset{h}{\rightsquigarrow}^* X$ であるので，文法規則 $r_p = A_p \rightarrow Y_p\alpha_p (p = 1, \ldots, q)$ が存在して，$Y_1 = cat_C(\tau_2)$，$Y_p = A_{p-1} (1 < p \leq q)$，$A_q = X$，$hc(r_p) = 1 (p = 1, \ldots, q)$ である（α_p は範疇の並びとする）．そこで，帰納法の仮定の σ_2 に対して，$\sigma_2' = apply(\sigma_2, r_1 \cdots r_q)$ とおく．このとき，定義 5.9 より，$\sigma_2' \in I(\mathbf{w}_2)$ である．$cat(\sigma_2') = X$ であるので，$rep(\sigma_1, \sigma_2')$ が存在し，それは $I(\mathbf{w})$ の要素である．これを σ とおくと，

$$\begin{aligned} dep(\sigma_2') &= dep(\sigma_2) \quad \text{(補題 5.12)} \\ &= dep(\tau_2) \quad \text{(帰納法の仮定)} \end{aligned}$$

$$\begin{aligned} head(\sigma_2') &= head(\sigma_2) \quad \text{(補題 5.13)} \\ &= head(\tau_2) \quad \text{(帰納法の仮定)} \end{aligned}$$

であるので，補題 5.16 より，$dep(\sigma) = dep_C(\tau)$ かつ $head(\sigma) = head_C(\tau)$ である．さらに $cat(\sigma) = cat(\sigma_1) = cat_C(\tau)$ である．

$R = \&_d$ のときには，$cat_C(\tau_2) \rightsquigarrow^* \overset{d}{\rightsquigarrow}^* X$ であるので，文法規則 $r_p = A_p \rightarrow Y_p\alpha_p (1 \leq p \leq q)$ が存在して，$Y_1 = cat_C(\tau')$，$Y_p = A_{p-1} (1 < p \leq q)$，$A_q = X$ であり，ある $r_p (1 \leq p \leq q)$ について，$hc(r_p) \neq 1$ である．$\sigma_2' = apply(\sigma_2, r_1 \cdots r_q)$ とおくと，$\sigma_2' \in I(\mathbf{w}_2)$ である．$cat(\sigma_2') = X$ であるので，$rep(\sigma_1, \sigma_2')$ は存在し，それは $I(\mathbf{w})$ の要素である．これを σ とおくと，補題 5.12 と帰納法の仮定より，$dep(\sigma_2') = dep_C(\tau_2)$ であり，補題 5.15 より，$head(\sigma_2') =?$ であるので，補題 5.17 より，$dep(\sigma) = dep_C(\tau)$ かつ $head(\sigma) = head_C(\tau)$ である．さらに $cat(\sigma) = cat(\sigma_1) = cat_C(\tau)$ である．

以上より，$n = k+1$ のとき補題が成り立つ，すなわち，帰納法により補題 5.19 が証明された． □

補題 5.18 より $dep(I(\mathbf{w})) \subseteq dep_C(C(\mathbf{w}))$ は明らか．

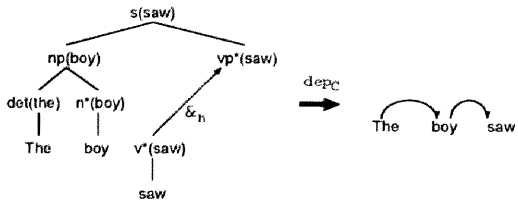

図 5.6: Head word 伝搬の例

補題 5.19 より $dep(I(\mathbf{w})) \supseteq dep_C(C(\mathbf{w}))$ は明らか.

したがって，これらから $dep(I(\mathbf{w})) = dep_C(C(\mathbf{w}))$ である．すなわち，定理 5.8 が証明された.

5.4.3 解析例

例として，文の断片 "The boy saw" の解析例を考える．この断片に対して，提案手法では，通常の上昇型チャート解析にしたがって，項 (5.5) と (5.6) を生成する．(5.6) の範疇は v で，(5.5) の最左未決定項の範疇は vp であり，$v \overset{h}{\rightsquigarrow}{}^{*} vp$ であるので，(5.5) と (5.6) は結合され，次の結合項を生成する.

$$[[[the]_{det}[boy]_n]_{np}[?]_{vp}]_s \&_h [saw]_v \tag{5.14}$$

この結合項 (5.14) の head word は，図 5.6 のように計算される．すなわち，$v \overset{h}{\rightsquigarrow}{}^{*} vp$ を用いて，(5.6) の head word である "saw" が (5.5) の最左未決定項に伝搬する．結合項 (5.14) に対して依存構造を求める関数 dep_C を適用することにより，"The boy saw" の依存構造 $\{\langle the \rightarrow boy \rangle, \langle boy \rightarrow saw \rangle\}$ が求められる.

これは，項 (5.1)〜(5.4) から 5.3 節で述べた方法によって得られる依存構造と同じものである.

5.5 実験と結果の検討

提案手法を評価するために，依存構造解析実験を行った．提案手法，5.3節で述べた手法，及び通常のチャート解析に基づく単純な方法（後述）をLinux PC（Pentium IV 2GHz，メインメモリ2GB）上にGNU Common Lispを用いて実装した．Penn Treebank[Marcus et al. 1993]に収録されている構文木付きATISコーパス全578文を対象とした．実験に用いた文法規則は，コーパスの構文木から取り出したもので，その数は509規則であった．文法規則のhead childの位置は，文献[Collins 1999]の方法に従って与えた．

5.5.1 解析処理時間の実験

提案手法の解析効率を評価するために，提案手法と5.3節で述べた手法の解析処理時間の比較を行った．図5.7は文の長さと一文あたりの平均解析処理時間の関係を示している[1]．提案手法の一文あたりの平均解析処理時間は0.017secであるのに対して，5.3節の手法のそれは11.081secであった．この結果からわかるように，到達可能性を用いた提案手法は，漸進的な依存構造解析の解析処理時間の短縮に効果的である．

入力された文の断片を覆う弧の数は，5.3節で述べた手法よりも提案手法の方が少なくなる．これによって解析処理時間が短縮される．例えば，図5.8は，英語文

I need to have dinner served. (5.15)

に対して生成された弧の数を示したものである．生成された結合項をラベルにもつ弧の数は，漸進的チャート解析により生成された弧に比べて明らかに少ない．5番

[1] 1単語当たりの解析処理時間が60secを超えた文については，解析を中断した．この結果は，5.3節の手法で解析を中断しなかった154文に対するものである．提案手法では，この154文を含む514文について解析を中断しなかった．

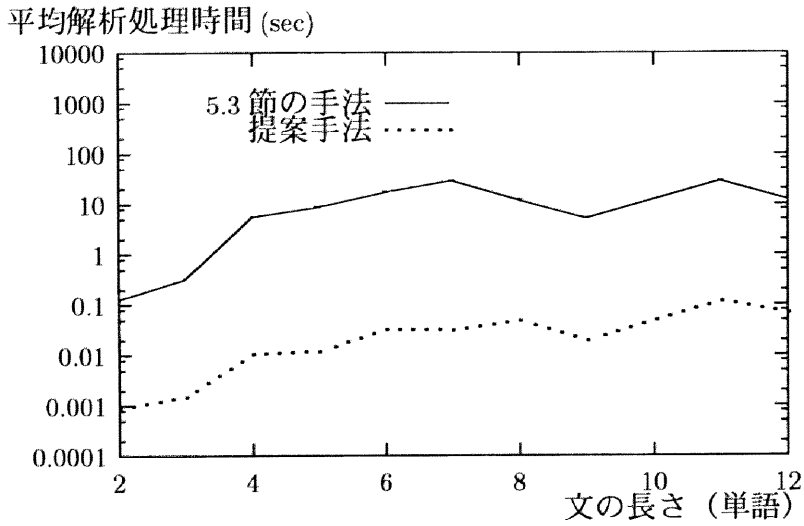

図 5.7: 文の解析処理時間

目の単語 "dinner" が入力されたときを見ると，漸進的チャート解析では，約 6×10^4 の弧が生成されるのに対して，提案手法では約 100 の弧が生成されているにすぎない．図 5.9 は同じ文の解析処理時間を示したものであるが，実際，"dinner" が入力された段階での解析処理時間は大幅に短縮されている．

5.5.2 時間制限下での解析精度

漸進的な依存構造解析手法を，実時間音声対話や同時通訳などの音声言語処理システムに導入する場合，ユーザ発話の解析処理時間がユーザの発話時間を下回ることが求められる．発話時間以上に解析処理に時間を要すれば，たとえ漸進的に依存構造解析を進めても，結果として文の入力との同時進行性が損なわれることになるためである．本節では，このような時間制限下での解析精度について検討する．まず，文 s の断片 x の正解依存構造を次のように定義する．

1. コーパスで付与された文 s の構文木を，文法の主辞情報にしたがって依存構造に変換する．(これを s の正解依存構造と呼ぶ)．

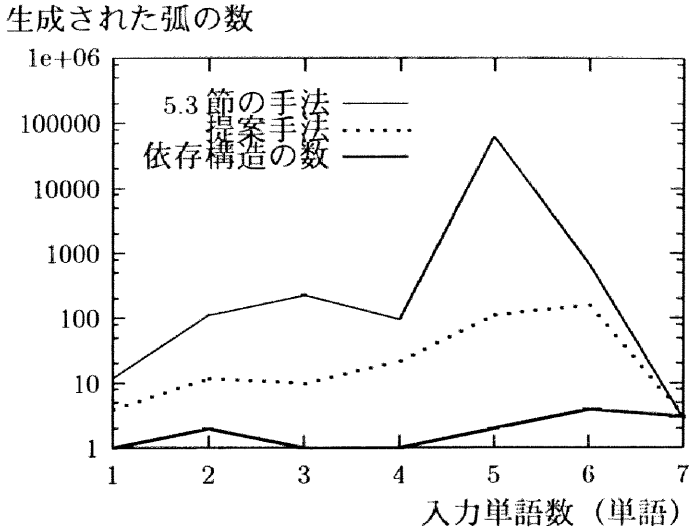

図 5.8: 英語文 "I need to have dinner served." に対して生成される弧の数

2. s の正解依存構造を構成する依存関係のうちで，x に出現する単語から構成されるものだけを取り出す（それらの依存関係からなる依存構造を x の正解依存構造と呼ぶ）．

例えば，英語文 "The boy saw the girl yesterday." に対する正解依存構造は $\{\langle the \rightarrow boy\rangle, \langle boy \rightarrow saw\rangle, \langle the \rightarrow girl\rangle, \langle girl \rightarrow saw\rangle, \langle yesterday \rightarrow saw\rangle\}$ であり，その断片 "The boy saw the" の正解依存構造は，$\{\langle the \rightarrow boy\rangle, \langle boy \rightarrow saw\rangle\}$ である．提案手法と 5.3 節の手法とは，定理 5.8 の意味で等価であり，したがって，本提案手法は，構文木の構成に必要な文法規則が与えられた場合には，正解依存構造を必ず計算できる．実際に，実験において解析結果を生成できたすべての文の断片に対して，提案手法は正解依存構造を計算することができた（表 5.1 参照）．しかし，このことが保証されるのは解析処理時間に制限がない場合である．解析処理時間が制限された状況では，正解依存構造を計算できるとは限らず，提案手法がどの程度の解析精度を達成できるかは明らかではない．そこで，解析処理に制限時間を設け，その制限下で正解依存構造を生成できた文の断片の割合を漸進的依存構造解析における解

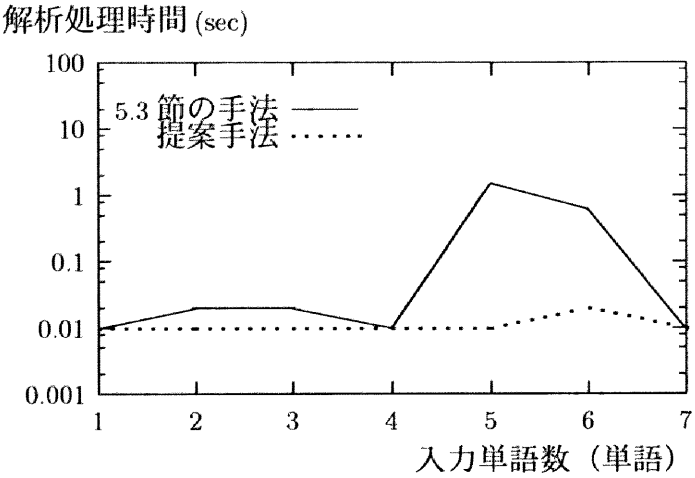

図 5.9: 英語文 "I need to have dinner served." に対する 1 単語当たりの解析処理時間

表 5.1: 漸進的依存構造解析の正解率

	正解率 (%)	解析結果を生成できた文の断片の割合 (%)
提案手法	77.0	77.0
5.3 節の手法	8.9	8.9
ベースライン	64.2	88.9

析精度と考え，実験を行った．本実験では，自然な英語発話における発話速度を考慮し，一単語あたりの解析処理の時間制限を 0.3sec と定めた[2]．すなわち，単語を先頭から順に 0.3sec 経過するごとに入力し，文の断片 $w_1 \cdots w_i$ の依存構造の計算を単語 w_i の入力がされた時点で開始した（ただし，$w_1 \cdots w_{i-1}$ の解析が完了していない場合は，その解析が完了してから計算を開始した）．正解率を，「時間内に正解依存構造を計算できた文の断片の割合」として評価した．実験の対象とした 578 文の

[2] CIAIR 同時通訳データベース [Matsubara et al. 2002] の講演に関する英語話者発話を調査したところ，一単語当たりの平均発話時間は約 0.32sec であった．

平均の長さは8.5単語であり，漸進的依存構造解析が対象とする文の断片は4931個存在する．

ところで，漸進的に依存構造を生成する方法として，通常のチャート解析により生成されたすべての項から依存構造を計算する単純な方法も考えられる．例えば，図5.2の文法のもとで，通常のチャート解析により，文の断片 “The boy” を解析すると，項 $[[the]_{det}[boy]_n]_{np}$ が生成される．この項の依存構造は $\{\langle the \rightarrow boy \rangle\}$ であり，これを “The boy” の依存構造として計算する方法である．ただし，この方法では，正解依存構造を生成できない場合もある．実際，同様の文法で文の断片 “The boy saw” を解析すると，項 $[[the]_{det}[boy]_n]_{np}$ や $[[saw]_v[?]_{np}]_{vp1}$ などが生成される．それらの項の依存構造は，それぞれ $\{\langle the \rightarrow boy \rangle\}$ や ϕ であり，正解依存構造 $\{\langle the \rightarrow boy \rangle, \langle boy \rightarrow saw \rangle\}$ を生成できない．チャート解析に基づく手法は，項の結合を行わないため，あらゆる文の断片に対して必ずしも正解依存構造を生成できるわけではないが，その分，解析を高速に行えるという特徴がある．以下では，この単純な方法をベースラインの方法と呼ぶことにし，これについても同様の実験を行い比較した．

提案手法，5.3節の手法，及びベースラインの方法のそれぞれの正解率を表5.1に示す．以下では，提案手法と5.3節の手法との比較，及び提案手法とベースラインの手法との比較について順に述べる．

提案手法と5.3節の手法との比較

5.3節の手法は，提案手法に比べて解析処理時間が非常に長くかかるため，かなりの数の文の断片について解析結果を生成できず，結果として正解率は8.9%と低い．この実験結果は，入力文を同時進行的に解析するには，5.3節の手法は解析速度が不十分であることを示している．一方，提案手法の正解率は77.0%であり，2章の手法にくらべて68.1%も高い．この結果は，解析の同時性を達成するために，提案手法による解析の効率化が有効であることを示している．

表 5.2: 正解を計算できた文の断片の数

正解依存構造生成の成否		文数（個）
提案手法	ベースライン	
成功	成功	2829
成功	失敗	968
失敗	成功	335
失敗	失敗	799
合計		4931

提案手法とベースラインの手法との比較

提案手法とベースラインの手法との違いは，後者では到達可能性に基づく項の結合を行わない点であるので，提案手法とベースラインの手法との比較は，到達可能性に基づく項の結合の効果の評価であると解釈できる．その評価のために，文の断片全4931個を，いずれの手法により正解依存構造を計算できたかという観点から分類した．結果を表5.2に示す．提案手法によってのみ正解依存構造が計算できた文の断片は，968個とかなりの数であるが，これらすべての文の断片に対して，ベースラインの手法は制限時間内に解析結果を計算した．このことは，少なくともこれらの968個の断片に対しては，結合項を生成しなければ，正解依存構造が計算できないことを意味している．一方，ベースラインの手法によってのみ正解依存構造が計算できた文の断片は335個存在したが，これらは，提案手法では制限時間内に依存構造を計算できなかったものである．これは，ベースラインの手法の方が，到達可能性に基づく項の結合を行わない分，解析処理時間が短いためである[3]．このように，一方の手法により正解依存構造を計算できる文の断片の集合が，他方のそれ

[3] ベースラインの手法の一文当たりの平均解析処理時間は，0.009sec である．

を含んでいるわけではないので，単純にその優劣を決めることはできないが，全体としては，表5.1に示すように提案手法の方が正解率が高くなっている．

5.6 5章のまとめ

本章では，単語が文の先頭から順に入力されるに従って，それまでに入力された文の断片に対して，文法上許されるすべての依存構造を漸進的に計算する手法を提案した．CFGベースの漸進的構文解析に対して，構文木から依存関係を計算する手法を適用することにより，漸進的な依存構造解析ができることを示し，より効率的な漸進的依存構造解析を実現する方法として，範疇間の関係である到達可能性に基づく手法を提案した．到達可能性に基づく手法が，漸進的構文解析に基づく手法と同等の依存構造を生成できることを理論的に示し，解析処理時間の短縮に効果があることを実験により示した．

第6章　おわりに

本論文では，実時間音声言語処理システムの実現を目指して，その言語理解部を担う漸進的構文解析手法を提案した．音声入力に対して同時的で正確さの高い漸進的構文解析の実現を目標に，漸進的構文解析の正確さを高めるための方法として，構文構造が正しいことが保証されるまでその出力を遅延する方法，及び，構文構造が正しい可能性が高くなった時点でそれを出力する方法を提案した．効率的かつ漸進的な解析を実現する方法として，漸進的依存構造解析アルゴリズムを提案した．

第1章では，漸進的構文解析に関する研究動向を概観し，これまでの研究が，文の断片に対する構文構造をどのように表現するかということを主目的としていることを指摘し，実時間音声言語処理システムの言語理解部として漸進的構文解析を用いる場合の課題について議論した．次の第2章では，本研究のベースとなる漸進的構文解析手法である漸進的チャート解析について説明した．

第3章では，漸進的構文解析において，正しい構文構造が確定するまで出力タイミングを遅らせる手法を提案した．文の入力途中の段階で，残りの入力単語列がどのようなものであるかを文法に基づいて予測し，どのような単語列が入力されても正しいことが保証された時点で，構文構造を出力する．この手法では，出力の即時性が損なわれ，文の入力が完了するまで解析結果を全く出力できない可能性もあるが，一方で，解析の正確さは保証されるという特徴がある．実際の対話文データに対して提案手法を適用する実験を実施し，データ全体の6割程度の数の文に対して，文の入力途中の段階で構文構造の正しさを判定でき，構文構造を出力できることを確認した．

さらに第4章では，確率文脈自由文法に基づいて漸進的構文解析の出力タイミングを定める方法を提案した．漸進的構文解析において生成される構文構造が正しいものである確率を単語が入力されるごとに計算し，その確率が予め定めた閾値を超えた段階でその構文構造を出力する．正確さを優先した3章の手法と比べて，正確さをそれほど損なうことなく，より早いタイミングで構文構造を出力することができることを構文解析実験により確認した．解析正解率は約98%で，出力の遅れは，最大でも文の長さの約4割，平均で約2割であり，出力をそれほど遅らせることなく正確さの高い漸進的構文解析を実現できた．

第5章では，効率的な漸進的構文解析を実現する漸進的依存構造解析手法を提案した．本手法では，範疇間の関係である到達可能性を活用することにより，文の断片に対する構文木を生成することなく，依存構造の生成が可能であり，効率的な解析が実現できる．提案手法により求められる依存構造が，漸進的チャート解析に基づく手法で求められるものと一致することを理論的に示すとともに，それに比べて解析処理時間が大幅に短縮されることを実験により確認した．漸進的チャート解析に基づく直接的な手法が，全体の約8.9%の文の断片しか同時的に処理できないのに対して，提案手法では，約77%の文の断片を同時的に処理することができた．この結果から，音声入力と同時進行的な漸進的構文解析の実現に，提案手法が有効であることが確認できた．

6.1 今後の課題と将来への展望

本研究が残した課題と将来への展望を述べる．

まず，漸進的構文解析の出力タイミングを制御し，正確さを向上する手法に関して，即時性の観点から出力タイミングを検討する必要がある．本論文では，漸進的構文解析の正確さを高めることを目標に，出力タイミングを制御する方法を提案した．漸進的構文解析の解析の正確さと出力タイミングの間にはトレードオフの関係

が存在するが，本論文では，解析の正確さを優先して出力タイミングを遅らせる方法，及び，構文構造が正しいものとなる可能性を評価し，それが高くなるまで出力を遅らせる方法を提案した．これらの方法は，解析の正確さをある程度達成するという条件のもとで，可能な限り出力タイミングを早める方法と位置付けることができる．このようなアプローチに従えば，解析の正確さは保たれるがその一方で，解析結果出力の即時性についてはどのような保証もない．出力の即時性も考慮し出力タイミングを制御する必要があるが，そのような方法を検討することは将来の課題である．

また，漸進的構文解析の高速化に関しては，効率的かつ漸進的な句構造解析手法の開発という研究課題がある．本論文では，効率的な漸進的構文解析手法として，漸進的に依存構造を生成する手法を提案したが，より詳細な言語解析が求められる場面では，句構造のような依存構造より複雑な構造を生成する枠組が必要になると考えられる．現在，漸進的かつ効率的に句構造を計算する手法として，依存関係に関する統計情報を用いて，意味的に誤っている句構造を解析途中の段階で枝刈りする漸進的構文解析手法を検討している [Murase et al. 2001]．

さらに，提案手法の利用可能性について，より詳細な調査をするために，大規模な対話文データを用いた実験が望まれる．今回の実験で用いたATISコーパス，及びATR音声言語データベースは旅行予約に関する比較的単純な対話を収録したコーパスであり，その規模も非常に小さい．より複雑な話し言葉（例えば，講演など）に対して，提案手法がどのような振舞いをするかを調査することは将来の課題である．現在，話し言葉に関する大規模コーパスの整備が進められており [古井ら 2002, Matsubara et al. 2002]，これらを利用した実験などが考えられる．

漸進的構文解析の解析精度，および解析速度を向上する方法として，本論文で提案した手法とは異なる次のアプローチが考えられる．これらの方法についても今後，検討したい．

- 解析処理単位の検討

  本論文で提案した手法では，単語が入力されるたびにそれまでに入力された文の断片に構文構造の候補を生成しているが，必ずしも単語単位で構文構造を生成する必要はない．ある程度まとまった単位で漸進的な解析処理を実行することにより，効率的な解析を実現できる可能性がある [森ら 2000]．しかし，そのような処理単位をどのように定めればよいかはほとんどわかっておらず，検討が必要である．本論文では，漸進的構文解析において正しい構文構造が確定するタイミングについて議論したが，この確定タイミングが処理単位に関する知見を与えてくれる可能性もあるため，このような観点から解析処理単位の問題について今後，検討したい．

- 有限状態変換器に基づく漸進的構文解析

  本論文では，文脈自由文法に基づく漸進的構文解析をベースとして議論を展開した．近年，有限状態変換器に基づく高速な構文解析手法が提案されているが [Kaiser 1999]，これを拡張し，高速な漸進的構文解析を実現する方法についても検討している [湊ら 2001]．

- 漸進的構文解析に適した文法の開発

  漸進的構文解析において生成される構文構造の数は，それが用いる文法の書き方に依存する．効率的で精度の高い漸進的構文解析を実現するために，これに適した文法規則を明らかにすることが必要である．また，一般の文脈自由文法をそのような文法に変換する手法を開発することが望まれる [松原ら 1999]．

なお，本論文では，解析の正確さ，および速度に焦点を当てて，漸進的構文解析について議論したが，ここでは取り上げなかった課題として，頑健な漸進的構文解析手法の開発がある．話し言葉を処理する上での問題の一つとして，文法的に不適格な現象が挙げられる．本論文で提案した手法は，入力文が文法的に適格であること

を前提としており，不適格な文を頑健に処理できるかどうかは明らかではない．不適格な文を漸進的かつ頑健に処理するためには，解析の早期に文に含まれる不適格表現を同定し，これを処理することが必要である [Kato et al. 1999]．話し言葉には，不適格な文が多数出現するため，音声言語処理システムの言語理解部の実現に向けて，このような枠組についても検討する必要がある．

謝辞

本研究を進め，まとめるにあたり，懇切丁寧な御指導と御鞭撻を頂いた名古屋大学大学院工学研究科教授の稲垣康善先生に心から感謝いたします．

本論文をまとめるにあたり，貴重な御示唆と御指導を頂いた名古屋大学大学院工学研究科教授の坂部俊樹先生ならびに名古屋大学情報連携基盤センター助教授の松原茂樹先生に深く感謝いたします．

日頃より有意義な御議論を頂き，また，本論文をまとめるにあたり適切な御示唆を頂いた名古屋大学大学院工学研究科助教授の外山勝彦先生に感謝の意を表します．

熱心に御討論くださった名古屋大学情報連携基盤センター河口信夫助教授，山口由紀子助手，名古屋大学大学院国際開発研究科 Muhtar Mahsut 助手，名古屋大学大学院工学研究科小川泰弘助手，杉野花津江先生（元名古屋大学助手）に感謝します．

名古屋大学稲垣研究室の多くの皆様から様々の教示と示唆を頂きました．記して感謝します．特に，漸進的構文解析手法について御討論いただいた浅井悟氏，渡邊善之氏，佐藤利光氏，西脇孝文氏，村瀬隆久氏，森大輔氏，湊恵一氏，自然言語処理について御議論頂いた杉本渉氏，岩島恵一氏，相澤靖之氏，木村直樹氏，高木亮氏，岩本雅文氏，大原誠氏，笠浩一郎氏，Nur Suraya 氏，大野誠寛氏，水野敦氏に感謝します．

研究活動を行うにあたりいろいろとお世話になった名古屋大学稲垣研究室秘書の伊藤千佳子さんに感謝します．

本研究では，ATIS コーパス，及び ATR 音声言語データベースとその構文木データを利用しました．これらの作成に携わった方々に感謝いたします．さらに，ATIS

コーパスの使用に関して御教示頂いた名古屋大学コーパスプロジェクトのメンバーの皆様に感謝いたします.

参考文献

[秋葉 & 田中 1992] 秋葉 友良, 田中 穂積 : 拡張部分木を用いた漸進的構文解析, 情報処理学会第 45 回全国大会 (3), pp.175–176 (1992).

[秋葉ら 1993] 秋葉 友良, 伊藤 克亘, 奥村 学, 田中 穂積 : 増進的曖昧性解消モデルに基づいた日本語解析, コンピュータソフトウェア, Vol.10, No.1, pp.29–40 (1993).

[Allen et al. 2001] James Allen, George Ferguson and Amanda Stent: An Architecture for More Realistic Conversational Systems, In *Proceedings of International Conference of Intelligent User Interfaces*, pp.1–8 (2001).

[Alshawi et al 2000] Hiyan Alshawi, Srinivas Bangalore and Shona Douglas, Learning Dependency Translation Models as Collections of Finite State Head Transducers, *Computational Linguistics*, Vol.26, No.1, pp.45–60 (2000).

[Bröker 1998] Norbert Bröker, Separating Surface Order and Syntactic Relations in a Dependency Grammar, In *Proceedings of 17th International Conference on Computational Linguistics and 36th Annual Meeting of the Association for Computational Linguistics*, pp.174–180 (1998).

[Collins 1996] Michael Collins: A New Statistical Parser Based on Bigram Lexical Dependencies, In *Proceedings of 34th Annual Meeting of the Association for Computational Linguistics (ACL '96)*, pp.184–191 (1996).

[Collins 1999] Michael Collins: Head-Driven Statistical Models for Natural Language Parsing, PhD Dissertation, University of Pennsylvania (1999).

[Core and Schubert 1999] Mark G. Core and Lenhart K. Schubert: A Syntactic Framework for Speech Repairs and Other Disruptions, In *Proceedings of 37th Annual Meeting of the Association for Computational Linguistics (ACL '99)*, pp.413–420 (1999).

[Eisner 1996] Jason M. Eisner, Three New Probabilistic Models for Dependency Parsing: An Exploration, In *Proceedings of 16th International Conference on Computational Linguistics*, pp.340–345 (1996).

[Ehsani et al. 1994] Farzad Ehsani, Kaichiro Hatazaki, Jun Noguchi and Takao Watanabe: Interactive Speech Dialogue System Using Simultaneous Understanding, In *Proceedings of 3rd Internatinal Conference on Spoken Language Processing (ICSLP '94)*, pp.879–882 (1994).

[Funakoshi et al. 2002] Kotaro Funakoshi, Takenobu Tokunaga and Hozumi Tanaka: Processing Japanese Self-correction in Speech Dialog Systems, In *Proceedings of 19th International Conference on Computational Linguistics (COLING 2002)*, pp.287–293 (2002).

[古井ら 2002] 古井 貞熙，前川 喜久雄，井佐原 均：『話し言葉工学』プロジェクトのこれまでの成果と展望，第2回話し言葉の科学と工学ワークショップ講演予稿集，pp.1–5 (2002).

[Haddock 1987] Nicholas J. Haddock: Incremental Interpretation and Combinatory Categorial Grammar, In *Proceedings of the 10th International Joint Conference on Artificial Intelligence (IJCAI '87)*, pp.661–663 (1987).

[羽尻ら 1998] 羽尻 公一郎，岡田 美智男，小川 均：日本語漸次的発話産出における格助詞補完と文法的不整合の解消，認知科学, Vol.5, No.3, pp.87–99 (1998).

[Hirst 1988] Graeme Hirst: Semantic Interpretation and Ambiguity, *Artificial Intelligence*, Vol.34, pp.131–177 (1998).

[Inagaki & Matsubara 1995] Yasuyoshi Inagaki and Shigeki Matsubara: Models for Incremental Interpretation of Natural Language, In *Proceedings of the 2nd Symposium on Natural Language Processing (SNLP '95)*, pp.51–60 (1995).

[Kato et al. 1999] Yoshihide Kato, Shigeki Matsubara, Katsuhiko Toyama and Yasuyoshi Inagaki: Spoken Language Parsing with Robustness and Incrementality, In *Proceedings of the 5th Natural Language Processing Pacific Rim Symposium (NLPRS-99)* pp.132–137 (1999).

[Kato et al. 2000] Yoshihide Kato, Shigeki Matsubara, Katsuhiko Toyama and Yasuyoshi Inagaki : Spoken Language Parsing Based on Incremental Disambiguation, In *Proceedings of 6th International Conference on Spoken Language Processing (ICSLP-2000)*, Vol.2, pp.999–1002 (2000).

[Kato et al. 2001] Yoshihide Kato, Shigeki Matsubara, Katsuhiko Toyama and Yasuyoshi Inagaki : Efficient Incremental Dependency Parsing, In *Proceedings of 7th International Workshop on Parsing Technologies (IWPT-2001)*, pp.225–228 (2001).

[加藤ら 2002] 加藤 芳秀, 松原 茂樹, 外山 勝彦, 稲垣 康善：確率文脈自由文法に基づく漸進的構文解析，電気学会論文誌, Vol.122-C, No.12, pp.2109–2119 (2002).

[加藤ら 2003] 加藤 芳秀, 松原 茂樹, 外山 勝彦, 稲垣 康善 : 主辞情報付き文脈自由文法に基づく漸進的な依存構造解析, 電子情報通信学会論文誌, Vol.J86-D-II, No.1, pp.84–97 (2003).

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

[Kaiser 1999] Edward C. Kaiser: Robust, Finite-State Parsing for Spoken Language Understanding, In *Proceedings of 37th Annual Meeting of the Association for Computational Linguistics (ACL '99)*, pp.573–578 (1999).

[Levelt 1989] W. J. Levelt: *Speaking: From Intention to Articulation*, MIT Press (1989).

[Lombardo & Lesmo 1996] Vincenzo Lombardo and Leonardo Lesmo, An Earley-type Recognizer for Dependency Grammar, In *Proceedings of 16th International Conference on Computational Linguistics*, pp.723–728 (1996).

[Marcus et al. 1993] Mitchell P. Marcus, Beatrice Santorini and Mary Ann Marcinkiewicz: Building a Large Annotated Corpus of English: the Penn Treebank, *Computational Linguistics*, Vol.19, No.2, pp.310–330, (1993).

[Marslen-Wilson 1973] William Marslen-Wilson: Linguistic structure and speech shadowing at very short latencies, *Nature*, Vol.244, pp.522-533 (1973).

[Matsubara et al. 1997] Shigeki Matsubara, Satoru Asai, Katsuhiko Toyama and Yasuyoshi Inagaki: Chart-based Parsing and Transfer in Incremental Spoken Language Translation, In *Proceedings of 4th Natural Language Processing Pacific Rim Symposium (NLPRS '97)*, pp.521–524 (1997).

[松原ら 1998] 松原 茂樹，浅井 悟，外山 勝彦，稲垣 康善 ：不適格表現を活用する漸進的な英日話し言葉翻訳手法，電気学会論文誌, Vol.118-C, No.1, pp.71-78 (1998).

[松原ら 1999] 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講演論文集，635, (1999).

[Matsubara et al. 1999b] Shigeki Matsubara, Katsuhiko Toyama and Yasuyoshi Inagaki: Sync/Trans: Simultaneous Machine Interpretation between English and Japanese, N. Foo (Ed.): *Advanced Topics in Artificial Intelligence (AI'99), Lecture Note in Artificial Intelligence 1747*, pp.134–143 (1999).

[松原ら 1999] 松原 茂樹，渡邊 善之，外山 勝彦，稲垣 康善，英日話し言葉翻訳のための漸進的文生成手法，情報処理学会研究報告，NL-132，pp.95–100, (1999).

[Matsubara et al. 2002] Shigeki Matsubara, Akira Takagi, Nobuo Kawaguchi and Yasuyoshi Inagaki: Bilingual Spoken Monologue Corpus for Simultaneous Machine Interpretation Research, In *Proceedings of 3rd International Conference on Language Resources and Evaluation (LREC-2002)* Vol.I, pp.153-159 (2002).

[Mellish 1985] Chris S. Mellish: Computer Interpretation of Natural Language Descriptions, Ellis Horwood Limited (1985). 田中 穂積 (訳) : 自然言語意味理解の基礎，サイエンス社 (1987).

[Milward & Cooper 1994] David Milward and Robin Cooper: Incremental Interpretation: Applications, Theory, and Relationship to Dynamic Semantics, In *Proceedings of 15th International Conference on Computational Linguistics (COLING '94)*, pp.748–754 (1994).

[Milward 1995] David Milward: Incremental Interpretation of Categorial Grammar, In *Proceedings of 7th Conference of European Chapter of the Association for Computational Linguistics (EACL '95)*, pp.119–126 (1995).

[Mima et al. 1998] Hideki Mima, Hitoshi Iida and Osamu Furuse, Simultaneous Interpretation Utilizing Example-based Incremental Transfer, In *Proceedings of 17th of International Conference on Computational Linguistics and 36th Annual Meeting of the Association for Computational Linguistics (COLING-ACL '98)*, pp.855–861 (1998).

[湊ら 2001] 湊 恵一，加藤 芳秀，松原 茂樹，稲垣 康善：漸進的構文解析のための統計情報を用いた有限状態変換器作成手法，電気関係学会東海支部連合大会講演論文集，494 (2002).

[森ら 2000] 森 大輔，加藤 芳秀，松原 茂樹，稲垣 康善：複数語ごとの入力に対する漸進的構文解析，情報処理第61回全国大会講演論文集，pp. 347–348, (2000).

[Murase et al. 2001] Takahisa Murase, Shigeki Matsubara, Yoshihide Kato and Yasuyoshi Inagaki, Incremental CFG Parsing with Statistical Lexical Dependencies, In *Proceedings of 6th Natural Language Processing Pacific Rim Symposium (NLPRS-2001)*, pp.353–358 (2001).

[Nakano et al. 1999] Mikio Nakano, Noboru Miyazaki, Jun-ichi Hirasawa, Kohji Dohsaka and Takeshi Kawabata: Understanding Unsegmented User Utterances in Real-Time Spoken Dialogue System, In *Proceedings of 37th Annual Meeting of the Association for Computational Linguistics (ACL '99)*, pp.200–207 (1999).

[Okumura & Tanaka 1990] Manabu Okumura and Hozumi Tanaka: Towards Incremental Disambiguation with Generalized Discrimination Network, In *Proceed-*

ings of 11th Annual National Conference on Artificial Intelligence (AAAI-90), pp.990–995 (1990).

[Roark & Johnson 1999] Brian Roark and Mark Johnson: Efficient Probabilistic Top-Down and Left-Corner Parsing, In *Proceedings of 37th Annual Meeting of the Association for Computational Linguistics*, pp.421–428 (1999).

[Sleator & Temperley 1991] Daniel Sleator and Davy Temperley, Parsing English with a Link Grammar, *Carnegie Mellon University Computer Science technical report CMU-CS-91-196* (1991).

[Stabler 1994] Edward P. Stabler: The Finite Connectivity of Linguistic Structure, C. Clifton, L. Frazier and K. Rayner(Eds.): *Perspectives on Sentence Processing*, pp.303–336, Lawrence Erlbaum (1994).

[Sturt and Crocker 1996] Patrick Sturt and Matthew W. Crocker: Monotonic Syntactic Processing: A Cross-lingusitic Study of Attachment and Reanalysis, *Language and Cognitive Processes Vol.11. No.5*, pp.449–494 (1996).

[植木ら 1998] 植木 正裕，白井 清昭，徳永 健伸，田中 穂積: 構造つきコーパスの共有化に関する一考察，情報処理学会研究報告，NL-128，pp.61–66 (1998).

[浦谷ら 1994] 浦谷 則好，竹沢 寿幸，松尾 秀彦，森田 千帆：音声言語データベースの構成，*Technical Report TR-IT-0056*, ATR(1994).

[山本ら 1992] 山本 幹夫，小林 聡，中川 聖一：音声対話文における助詞落ち・倒置の分析と解析手法，情報処理学会論文誌，Vol.33, No.11, pp.1322–1330 (1992).

[1] http://tanaka-www.cs.titech.ac.jp/pub/sldb-tree/