論　文

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

加藤　芳秀[†]　　松原　茂樹[††,†††]　　外山　勝彦[†,†††]　　稲垣　康善[†]

Incremental Dependency Parsing Based on Headed Context Free Grammar

Yoshihide KATO[†], Shigeki MATSUBARA[††,†††], Katsuhiko TOYAMA[†,†††], and Yasuyoshi INAGAKI[†]

あらまし　本論文では，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．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．

キーワード　漸進的構文解析，依存関係，依存構造解析，到達可能性，漸進的解釈

1. まえがき

自然言語解析とは，文の構成要素間の構文的関係や意味的関係を明らかにすることであり，これまで提案された手法の多くは，1 文単位での解析処理を前提とする．しかし，話し言葉の解析の場合には，必ずしもそのような前提が適切であるとは限らない．というのも，音声は音素の時間的連続として現れるため，音声言語処理システムによっては，音声入力と同時進行的に解析を進めなければならない場合があるからである．例えば，ユーザの発話途中での相づちや割込みを適切なタイミングで生成する実時間音声対話システム [8], [19] では，ユーザが発話している途中でもシステムは発話理解を行う必要がある [1], [16], [20]．また，話者発声と同時進行的に訳文を出力する同時通訳システム [13], [17] では，原言語の発声途中における目標言語への構文変換が求められる [3], [12]．このように，音声入力と同時的に言語処理を実行する場面においては，「入力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に対する解析結果を生成する手法」，すなわち，自然言語の漸進的解析手法が必要である．

ところで，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に依存構造がある．依存構造は単語間の修飾・被修飾関係に着目した解析方法であり，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 [4], [7], [9], [21] とともに，依存構造を利用して構文的あいまい性を解消する方法 [5] や，依存構造に基づいて翻訳文を生成する方法 [2], [14] などが提案されている．しかしながら，漸進的に依存構造を解析する方法についてはこれまでほとんど検討されていない．

一般に，依存構造解析手法は，

（1）依存関係の制約に従って依存構造を生成する手法 [7], [21]

（2）文法規則に主辞情報を付与し，その情報に従って構文木を依存構造に変換する手法 [4], [5], [9]

の二つに大別できるが，いずれの手法も漸進的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

† 名古屋大学大学院工学研究科，名古屋市
Graduate School of Engineering, Nagoya University, Furo-cho, Chikusa-ku, Nagoya-shi, 464-8603 Japan
†† 名古屋大学情報連携基盤センター，名古屋市
Information Technology Center, Nagoya University, Furo-cho, Chikusa-ku, Nagoya-shi, 464-8601 Japan
††† 名古屋大学統合音響情報研究拠点，名古屋市
Center for Integrated Acoustic Information Research, Nagoya University, Furo-cho, Chikusa-ku, Nagoya-shi, 464-8603 Japan

(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．

そこで，本論文では，先頭から 1 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．本論文で提案する手法は，依存関係情報を付与した文脈自由文法に基づいて，入力途中の文の断片に対して，文法上可能な依存構造をすべて生成する．まず，文脈自由文法に基づく漸進的構文解析と，構文木からの依存構造計算を組み合わせることにより，このような漸進的な依存構造解析が実現できることを示す．次に，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構文木を生成する必要がなく，効率的な解析が可能であり，漸進的な話し言葉処理に適している．提案手法の正当性を理論的に証明するとともに，Penn TreeBank [11] に収録されている ATIS コーパスを用いた解析実験を行い，提案手法が解析処理時間の短縮に有効であることを確認した．**2.** で述べる漸進的チャート解析に基づく直接的な依存構造解析手法に比べると，平均して約 1000 分の 1 程度の依存構造解析時間の短縮を示した．

本論文の構成は以下のとおりである．まず，次の **2.** 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．**3.** 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 8 の証明は付録に詳述する．**4.** 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．

2. 漸進的な依存構造解析

依存関係は，修飾する単語と修飾される単語との間の関係であるが，これは，句構造を定める文脈自由文法（CFG）の各文法規則に句の間の依存関係情報を付与することにより，構文木から計算することができる [5]．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．

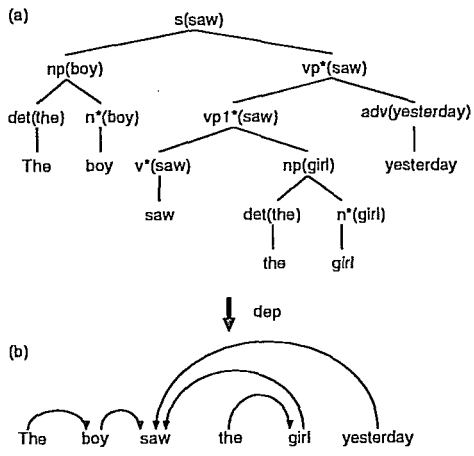

図 1 構文木からの依存構造計算例
(a) 構文木
(b) (a) から計算される依存構造

Fig. 1 An example of computing dependencies form parse tree.

（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） 文献 [5] の方法に従って,(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．

本章では，まず，構文木から依存関係を計算する方法について説明し，次に，漸進的構文解析手法の一つである漸進的チャート法，及びそれに基づく依存構造計算手法について述べる．

2.1 依存構造

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 1 (b) に依存構造の例を示す．

2.2 構文木からの依存構造計算

文に対する依存構造は，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 [5]．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

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 2 文法と辞書
Fig. 2 Grammar and lexicon.

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 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である head word が，次のように定義される．

[定義 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，次に与える定義に基づいて，構文木から依存構造を計算できる．

[定義 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．

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

□

$dep(\sigma)$ は σ から計算される依存構造である．図 1 は，(a) の構文木から定義 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．

2.3 漸進的チャート解析と依存構造計算

前節で述べた構文木からの依存構造計算を，CFG に基づく漸進的構文解析により生成される部分構文木に対して適用することにより，漸進的な依存構造解析を実現できる．本節では，まず，漸進的に文の断片に対する部分構文木を生成する漸進的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基づいて依存構造を計算する方法について述べる．

2.3.1 漸進的チャート解析

漸進的チャート解析は，従来のチャート解析 [10] と同様に，チャート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，入力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，最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．

漸進的チャート解析は通常の上昇型チャート解析に新たな操作を導入した手法である．通常の上昇型チャート解析は，次の三つの操作を実行する．なお，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．

これに対して，漸進的チャート解析では，活性弧への文法規則の適用と活性弧による項の置換えができるように，更に次の二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．

すなわち，漸進的チャート解析は活性弧への文法規則の適用，及び，活性弧にラベル付けされた項の最左未決定項を別の活性弧にラベル付けされた項で置き換える操作を導入している．

例として，文の断片 “The boy saw” の解析を考える．通常の上昇型チャート解析では，次のような項が生成される．

$$[[[the]_{det}[boy]_n]_{np}[?]_{vp}]_s \quad (1)$$

$$[saw]_v \quad (2)$$

$$[[saw]_v[?]_{np}]_{vp1} \quad (3)$$

しかし，“The boy saw” に対する項は生成されない．一方，漸進的チャート解析では，項 (3) に文法規則 $vp \rightarrow vp1\ adv$ を適用することにより，項

$$[[[saw]_v[?]_{np}]_{vp1}[?]_{adv}]_{vp} \quad (4)$$

を生成し，(1) の最左未決定項を項 (4) で置き換えることにより，“The boy saw” に対して項

$$[[[the]_{det}[boy]_n]_{np}[[[saw]_v[?]_{np}]_{vp1}[?]_{adv}]_{vp}]_s \quad (5)$$

を生成する．

2.3.2 依存構造計算

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 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．例えば，項 (5) に含まれる項の head word は図 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 3 (b) のように計算できる．

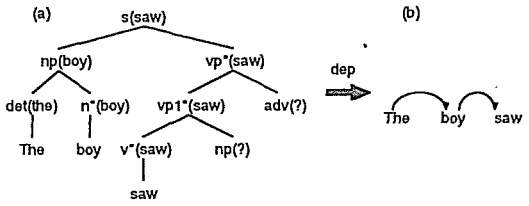

図 3 文の断片に対する依存構造計算例
Fig. 3 An example of computing dependencies for initial fragment.

3. 漸進的依存構造解析

前章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) の項だけでなく，次のような項も生成する：

$$[[[the]_{det}[boy]_n]_{np}[[[saw]_v[?]_{np}]_{vp1}[?]_{pp}]_{vp}]_s \quad (6)$$

$$[[[the]_{det}[boy]_n]_{np}[[[saw]_v[?]_{np}[?]_{np}]_{vp1}[?]_{adv}]_{vp}]_s \quad (7)$$

$$[[[the]_{det}[boy]_n]_{np}[[[saw]_v[?]_{np}[?]_{np}]_{vp1}[?]_{pp}]_{vp}]_s \quad (8)$$

項 (5)〜(8) からは，同じ依存構造が得られる．単に依存構造を求めることだけを考えると，このような同じ依存構造をもつ項をいくつも生成しないで，正しく依存構造が計算できるのであれば，その方が解析効率の観点から望ましい．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．そこで本章では，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．

3.1 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 3]（到達可能性） 文法規則 $A \to XY\cdots Z$ 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 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 $r = A \to XY\cdots Z$ 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 $r = A \to XY\cdots Z$ 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．

3.2 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，次の三つの手順からなる．

（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，2. 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，それより大幅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]（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．その操作は次のとおりである．

（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，項 (1) と (2) は図 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 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 $head(\tau)$.

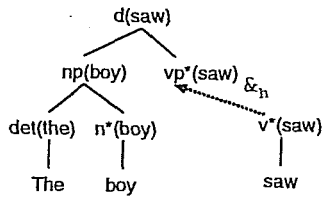

図 4 結 合 項
Fig. 4 Connecting terms.

（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基づいて依存構造を計算する．

[定義 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，$h = hc(A \to X_1 \cdots X_m)$ とおく．

(a1) $R_1 = \&_h$ で $h = k$ ならば，

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

(a2) $R_1 = \&_h$ で $h \ne k$ ならば，

$$d_C(\tau) = \{\langle w_d \to head([\cdots]_{X_h})\rangle \mid w_d = head([\cdots]_{X_j})(1 \le j \le m, j \ne h, j \ne k) \text{ または } w_d = head_C(\sigma_2 R_2 \cdots R_{n-1}\sigma_n)\}$$

(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 (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 (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．文法規則の適用の仕方が何通りもあるのに対して，分類した到達可能性の場合の数はたかだか 2 通りであるので，より効率的に依存構造を計算できる．更に，提案手法が計算する依存構造は，**2.** 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．

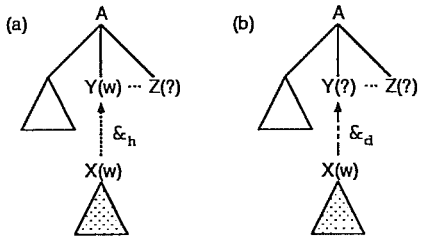

図 5 到達可能性に基づく head word の伝搬
Fig. 5 Head word propagation based on reachability relation.

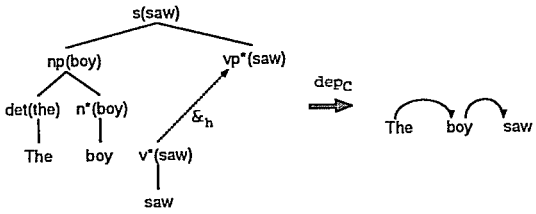

図 6 Head word 伝搬の例
Fig. 6 An example of head word propagation.

[定理 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，$dep(I(\mathbf{w})) = dep_C(C(\mathbf{w}))$． □

証明は付録に示す．

3.3 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従って，項 (1) と (2) を生成する．(2) の範疇は v で，(1) 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，(1) と (2) は結合され，次の結合項を生成する．

$$[[[the]_{det}[boy]_n]_{np}[?]_{vp}]_s \&_h [saw]_v \qquad (9)$$

この結合項 (9) の head word は，図 6 のように計算される．すなわち，$v \overset{h}{\rightsquigarrow}{}^{*} vp$ を用いて，(2) の head word である “saw” が (1) の最左未決定項に伝搬する．結合項 (9) 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 $\{\langle the \to boy\rangle, \langle boy \to saw\rangle\}$ が求められる．

これは，項 (5)〜(8) から **2.** 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．

4. 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，**2.** で述べた手法，及び通常の上昇型チャート解析に基づく単純な方法（後述）を Linux PC（Pentium IV 2 GHz，メインメモリ 2 GB）上に GNU Common Lisp を用いて実装した．Penn Treebank [11] 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 [6] の方法に従って与えた．

4.1 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 **2.** で述べた手法の解析処理時間の比較を行った．図 7 は文の長さと 1 文当りの平均解析処理時間の関係を示している(注1)．提案手法の 1 文当りの平均解析処理時間は 0.017s であるのに対して，**2.** の手法のそれは 11.081s 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，**2.** 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 8 は，英語文

I need to have dinner served. (10)

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 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．

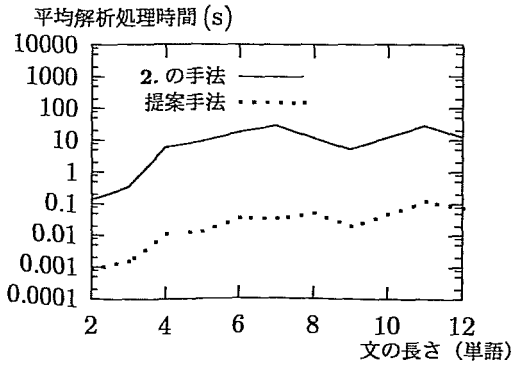

図 7 文の解析処理時間
Fig. 7 Parsing time per sentence.

4.2 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

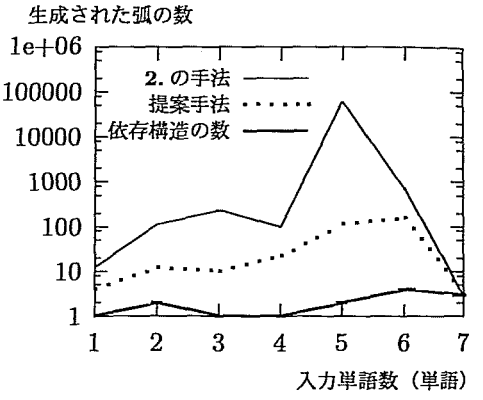

図 8 英語文 "I need to have dinner served." に対して生成される弧の数
Fig. 8 The number of edges for the sentence, "I need to have dinner served."

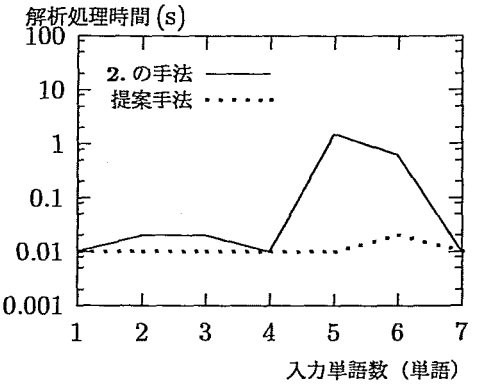

図 9 英語文 "I need to have dinner served." に対する 1 単語当りの解析処理時間
Fig. 9 The parsing time per word for the sentence, "I need to have dinner served."

(注1)：1 単語当りの解析処理時間が 60s を超えた文については，解析を中断した．この結果は，**2.** 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．

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，文法の主辞情報に従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）．

（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 **2.** の手法とは，定理 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 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，1 単語当りの解析処理の時間制限を 0.3s と定めた(注2)．すなわち，単語を先頭から順に 0.3s 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，通常の上昇型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 2 の文法のもとで，通常の上昇型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．上昇型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．

表 1　漸進的依存構造解析の正解率
Table 1　Precision of incremental dependency parsing.

	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
2. の手法	8.9	8.9
ベースライン	64.2	88.9

提案手法，**2.** 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 1 に示す．以下では，提案手法と **2.** 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．

4.2.1　提案手法と **2.** の手法との比較

2. 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，**2.** 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.** の手法に比べて 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．

4.2.2　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

(注2)：CIAIR 同時通訳データベース [15] の英語話者発話を調査したところ，1 単語当りの平均発話時間は約 0.32s であった．

表 2 正解を計算できた文の断片の数
Table 2 The number of correctly parsed fragments.

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

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 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(注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 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. むすび

本論文では，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．

また，これまでに，依存関係を利用し漸進的構文解析を効率化する手法が提案されているが [18]，この手法では部分構文木から依存構造を計算している．本論文で提案した依存構造計算法とこの手法を組み合わせることにより，漸進的構文解析の効率化が期待できるが，そのような手法についても今後検討したい．

謝辞 末筆ながら，本研究を進めるにあたり御協力して頂いた研究室の皆様に感謝致します．また，本論文を改善する上で，査読者の方々には貴重な御意見を頂きました．ここに深く感謝致します．

(注3)：ベースラインの手法の 1 文当りの平均解析処理時間は，0.009s である．

文献

[1] J. Allen, G. Ferguson, and A. Stent, "An architecture for more realistic conversational systems," Proc. Intelligent User Interfaces, pp.1–8, Santa Fe, NM, Jan. 2001.

[2] H. Alshawi, S. Bangalore, and S. Douglas, "Learning dependency translation models as collections of finite state head transducers," Computational Linguistics, vol.26, no.1, pp.45–60, 2000.

[3] J.W. Amtrup, "Incremental speech translation," Number 1735 in Lecture Notes in Artificial Intelligence, Springer Verlag, Berlin, Heidelberg, New York, 1999.

[4] N. Bröker, "Separating surface order and syntactic relations in a dependency grammar," Proc. 17th International Conf. on Computational Linguistics and 36th Annu. Meeting of the Association for Computational Linguistics, pp.174–180, Montreal, Canada, Aug. 1998.

[5] M.J. Collins, "A new statistical parser based on bigram lexical dependencies," Proc. 34th Annu. Meeting of the Association for Computational Linguistics, pp.184–191, Santa Cruz, USA, June 1996.

[6] M. Collins, Head-driven statistical models for natural language parsing, PhD Dissertation, University of Pennsylvania, 1999.

[7] J.M. Eisner, "Three new probabilistic models for dependency parsing: An exploration," Proc. 16th International Conf. on Computational Linguistics, pp.340–345, Copenhagen, Denmark, Aug. 1996.

[8] F. Ehsani, K. Hatazaki, J. Noguchi, and T. Watanabe, "Interactive speech dialogue system using simultaneous understanding," Proc. 3rd Internatinal Conf on Spoken Language Processing, pp.879–882, Yokohama, Japan, Sept. 1994.

[9] V. Lombardo and L. Lesmo, "An Earley-type recognizer for dependency grammar," Proc. 16th International Conf. on Computational Linguistics, pp.723–728, Copenhagen, Denmark, Aug. 1996.

[10] M. Kay, "Algorithm schemata and data structures in syntactic processing," Technical Report, CSL-80-12

Xerox PARC, 1980.

[11] M.P. Marcus, B. Santorini, and M.A. Marcinkiewicz, "Building a large annotated corpus of English: The Penn Treebank," Computational Linguistics, vol.19, no.2, pp.310–330, 1993.

[12] S. Matsubara, S. Asai, K. Toyama, and Y. Inagaki, "Chart-based parsing and transfer in incremental spoken language translation," Proc. 4th Natural Language Processing Pacific Rim Symposium, pp.521–524, Phuket, Thailand, Dec. 1997.

[13] S. Matsubara, K. Toyama, and Y. Inagaki, "Sync/Trans: Simultaneous machine interpretation between English and Japanese," in Advanced Topics in Artificial Intelligence(AI'99), Lecture Note in Artificial Intelligence 1747, ed. N. Foo, pp.134–143, 1999.

[14] 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，" 情処学 NL 研報，NL-132, pp.95–100, 1999.

[15] S. Matsubara, A. Takagi, N.Kawaguchi, and Y. Inagaki, "Bilingual spoken monologue corpus for simultaneous machine interpretation research," Proc. 3rd International Conf. on Language Resources and Evaluation, vol.I, pp.153–159, Canary Islands, Spain, May 2002.

[16] D. Milward and R. Cooper, "Incremental interpretation: Applications, theory, and relationship to dynamic semantics," Proc. 15th International Conf. on Computational Linguistics, pp.748–754, Kyoto, Japan, Aug. 1994.

[17] H. Mima, H. Iidam, and O. Furuse, "Simultaneous interpretation utilizing example-based incremental transfer," Proc. 17th International Conf. on Computational Linguistics and 36th Annu. Meeting of the Association for Computational Linguistics, pp.855–861, Montreal, Canada, Aug. 1998.

[18] T. Murase, S. Matsubara, Y. Kato, and Y. Inagaki, "Incremental CFG parsing with statistical lexical dependencies," Proc. 6th Natural Language Processing Pacific Rim Symposium, pp.353–358, Tokyo, Nov. 2001.

[19] M. Nakano, N. Miyazaki, J. Hirasawa, K. Dohsaka, and T. Kawabata, "Understanding unsegmented user utterances in real-time spoken dialogue systems," Proc. 37th Annu. Meeting of the Association for Computational Linguistics, pp.200–207, College Park, USA, June 1999.

[20] 中野幹生，堂坂浩二，"音声対話システムの言語・対話処理，" 人工知能誌，vol.17, no.3, pp.271–278, 2002.

[21] D. Sleator and D.Temperley, "Parsing English with a link grammar," Carnegie Mellon University Computer Science technical report, CMU-CS-91-196, Oct. 1991.

付　　録

定理 8 の証明

はじめに，証明に用いるいくつかの定義を与える．$T(\mathbf{w})$ を通常の上昇型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 $r_1, r_2, \ldots, r_q$ 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 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，$\mathbf{w}$ に通常の上昇型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 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 8 の証明に必要な補題を順に証明する．

[補題 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 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．$r_1 = A \to XY \cdots Z$，$\mathbf{r} = r_2 \cdots r_{n+1}$ とおくと，

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

である．このとき，

$$\begin{aligned} &dep(apply(\sigma, r_1 r_2 \cdots r_{n+1})) \\ &= dep(apply([\sigma[?]_Y \cdots [?]_Z]_A, \mathbf{r})) \quad ((\text{A·1})) \\ &= dep([\sigma[?]_Y \cdots [?]_Z]_A) \quad (\text{帰納法の仮定}) \\ &= dep(\sigma) \quad (\text{定義 2}) \end{aligned}$$

以上から，帰納法により補題 12 が証明された． □

[補題 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．$r_1 = A \to XY \cdots Z$，$\mathbf{r} = r_2 \cdots r_{n+1}$ とおく．このとき，(A·1) である．一方，$hc(r_1) = 1$ であるので，定義 1 より，

$$head([\sigma[?]_Y \cdots [?]_Z]_A) = head(\sigma) \quad \text{(A·2)}$$

である．したがって，

$$\begin{aligned} &head(apply(\sigma, r_1 r_2 \cdots r_{n+1})) \\ &= head(apply([\sigma[?]_Y \cdots [?]_Z]_A, \mathbf{r})) \quad ((\text{A·1})) \\ &= head([\sigma[?]_Y \cdots [?]_Z]_A) \quad (\text{帰納法の仮定}) \\ &= head(\sigma) \quad ((\text{A·2})) \end{aligned}$$

以上から，帰納法により補題 13 が証明された． □

[補題 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．$r_1 = A \to XY \cdots Z$，$\mathbf{r} = r_2 \cdots r_{n+1}$ とおく．

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

$$head([\sigma[?]_Y \cdots [?]_Z]_A) =? \quad \text{(A·3)}$$

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：

$$\begin{aligned} head([\sigma[?]_Y \cdots [?]_Z]_A) &= head(\sigma) \quad (\text{定義 1}) \\ &= ? \quad (\text{補題の仮定}) \end{aligned}$$

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

$$\begin{aligned} &head([\sigma[?]_Y \cdots [?]_H \cdots [?]_Z]_A) \\ &= head([?]_H) \quad (\text{定義 1}) \\ &= ? \quad (\text{定義 1}) \end{aligned}$$

である．

以上より，(A·3) 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，(A·3) が成り立つことから，帰納法の仮定より，

$$head(apply([\sigma[?]_Y \cdots [?]_Z]_A, \mathbf{r})) =? \quad \text{(A·4)}$$

である．したがって，

$$
\begin{aligned}
&head(apply(\sigma, r_1 r_2 \cdots r_{n+1})) \\
&\quad = head(apply([\sigma[?]_Y \cdots [?]_Z]_A, \mathbf{r})) \quad ((A\cdot1)) \\
&\quad = ? \quad ((A\cdot4))
\end{aligned}
$$

以上から，帰納法により補題 14 が証明された．□

[補題 15] $\sigma \in I(\mathbf{w})$ とし，$r_p(p = 1, \ldots, q)$ を文法規則とする．更に，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，

$$
\begin{aligned}
&head(apply(\sigma', r_i)) \\
&\quad = head([\sigma'[?]_Y \cdots [?]_H \cdots [?]_Z]_A) \text{(applyの定義)} \\
&\quad = head([?]_H) \quad \text{(定義 1)} \\
&\quad = ? \quad \text{(定義 1)}
\end{aligned}
$$

したがって，補題 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 15 が証明された．□

[補題 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．更に，$head(\sigma_2) = head_C(\tau_2)$，$dep(\sigma_2) = dep_C(\tau_2)$ 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，$h = hc(A \rightarrow X_1 \cdots X_{k-1} X_k \cdots X_m)$ とおく．

まず，

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

を $h = k$ 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：

$$
\begin{aligned}
head(\sigma) &= head(\sigma_2) \quad \text{(定義 1)} \\
&= head_C(\tau_2) \quad \text{(補題の仮定)} \\
&= head_C(\tau) \quad \text{(定義 6)}
\end{aligned}
$$

$h \neq k$ のとき：

$$
\begin{aligned}
head(\sigma) &= head([\cdots]_{X_h}) \quad \text{(定義 1)} \\
&= head_C(\tau) \quad \text{(定義 6)}
\end{aligned}
$$

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 2，定義 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：

$$
\begin{aligned}
d(\sigma) &= \{\langle w_d \rightarrow head(\sigma_2)\rangle \mid w_d = head([\cdots]_{X_i}) \\
&\qquad (1 \leq i \leq m, i \neq h)\} \quad \text{(定義 2)} \\
&= \{\langle w_d \rightarrow head_C(\tau_2)\rangle \mid w_d = head([\cdots]_{X_i}) \\
&\qquad (1 \leq i \leq m, i \neq h)\} \quad \text{(補題の仮定)} \\
&= d_C(\tau) \quad \text{(定義 7)}
\end{aligned}
$$

$h \neq k$ のとき：

$$
\begin{aligned}
&d(\sigma) \\
&= \{\langle w_d \rightarrow head([\cdots]_{X_h})\rangle \mid \\
&\qquad w_d = head([\cdots]_{X_i})(1 \leq i \leq m, i \neq h, i \neq k) \\
&\qquad \text{または } w_d = head(\sigma_2)\} \quad \text{(定義 2)} \\
&= \{\langle w_d \rightarrow head([\cdots]_{X_h})\rangle \mid \\
&\qquad w_d = head([\cdots]_{X_i})(1 \leq i \leq m, i \neq h, i \neq k) \\
&\qquad \text{または } w_d = head_C(\tau_2)\} \quad \text{(補題の仮定)} \\
&= d_C(\tau) \quad \text{(定義 7)}
\end{aligned}
$$

よって，$d(\sigma) = d_C(\tau)$ となり，定義 2，定義 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 16 は証明された．□

[補題 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．更に，$head(\sigma_2) =?$，$dep(\sigma_2) = dep_C(\tau_2)$ 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 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：

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

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 17 が証明された． □

[補題 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 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 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，

$$dep(\sigma_2') = dep_C(\tau_2) \qquad \text{(A·5)}$$

$$cat(\sigma_2') = cat_C(\tau_2) \qquad \text{(A·6)}$$

$$head(\sigma_2') = head_C(\tau_2) \qquad \text{(A·7)}$$

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 10 より $\sigma_1 \&_h \tau_2 \in C(\mathbf{w})$ である．$\sigma_1 \&_h \tau_2$ に対して，

$$\begin{aligned} dep(\sigma_2) &= dep(\sigma_2') \quad \text{(補題 12)} \\ &= dep(\tau_2) \quad \text{((A·5))} \end{aligned}$$

$$\begin{aligned} head(\sigma_2) &= head(\sigma_2') \quad \text{(補題 13)} \\ &= head(\tau_2) \quad \text{((A·7))} \end{aligned}$$

であるので，補題 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，$cat(\sigma_2') \rightsquigarrow^* \overset{d}{\rightsquigarrow} \rightarrow^* X$，すなわち $cat_C(\tau_2) \rightsquigarrow^* \overset{d}{\rightsquigarrow} \rightarrow^* X$ であるので，定義 10 より $\sigma_1 \&_d \tau_2 \in C(\mathbf{w})$ である．補題 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 15 より，$head(\sigma_2) =?$ である．したがって，補題 17 より，$dep(\sigma) = dep_C(\sigma_1 \&_d \tau_2)$，$head(\sigma) = head_C(\sigma_1 \&_d \tau_2)$ である．更に，$cat(\sigma) = cat(\sigma_1) = cat_C(\sigma_1 \&_d \tau_2)$ 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 18 が証明された． □

[補題 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 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，

$$\begin{aligned} dep(\sigma_2') &= dep(\sigma_2) \quad \text{(補題 12)} \\ &= dep(\tau_2) \quad \text{(帰納法の仮定)} \end{aligned}$$

$$\begin{aligned} head(\sigma_2') &= head(\sigma_2) \quad \text{(補題 13)} \\ &= head(\tau_2) \quad \text{(帰納法の仮定)} \end{aligned}$$

であるので，補題 16 より，$dep(\sigma) = dep_C(\tau)$ かつ $head(\sigma) = head_C(\tau)$ である．更に $cat(\sigma) = cat(\sigma_1) = cat_C(\tau)$ 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 12 と帰納法の仮定より，$dep(\sigma_2') = dep_C(\tau_2)$ であり，補題 15 より，$head(\sigma_2') =?$ であるので，補題 17 より，$dep(\sigma) = dep_C(\tau)$ かつ ~~$head$~~$(\sigma) = head_C(\tau)$ である．更に $cat(\sigma) = cat(\sigma_1) = cat_C(\tau)$ 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 19 が証明された． □

補題 18 より $dep(I(\mathbf{w})) \subseteq dep_C(C(\mathbf{w}))$ は明らか．

補題 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 8 が証明された．

（定理 8 の証明終り）

（平成 14 年 1 月 31 日受付，7 月 15 日再受付）

加藤　芳秀　（学生員）

1997 名大・工・情報卒．1999 同大大学院博士前期課程了．現在，同博士後期課程在学中．自然言語処理に関する研究に従事．情報処理学会，言語処理学会各会員．

松原　茂樹　（正員）

1993 名工大・工・電気情報卒．1998 名大大学院博士課程了．同大言語文化部助手を経て，2002 同大情報連携基盤センター助教授．工博．1996～1998 日本学術振興会特別研究員．自然言語処理，音声言語処理，機械翻訳の研究に従事．情報処理学会，人工知能学会，言語処理学会，日本通訳学会，ACL 各会員．

外山　勝彦　（正員）

1984 名大・工・電気卒．1989 同大大学院博士課程了．同大助手，中京大学講師，助教授を経て，1997 名大・工・助教授．工博．論理に基づく知識表現と推論，自然言語理解に関する研究に従事．情報処理学会，言語処理学会，人工知能学会，日本認知科学会各会員．

稲垣　康善　（正員）

1962 名大・工・電子卒．1967 同大大学院博士課程了．同大助教授，三重大学教授を経て，1981 名大・工・教授．工博．この間，スイッチング回路理論，オートマトン・言語理論，計算論，ソフトウェア基礎論，並列処理論，代数的仕様記述法，人工知能基礎論，自然言語処理などの研究に従事．本会副会長並びにフェロー．情報処理学会，言語処理学会，人工知能学会，電気学会，日本ソフトウェア科学会，日本 OR 学会，IEEE，ACM，EATCS 各会員．