確率木接合文法に基づく漸進的構文解析

加藤 芳秀† 松原 茂樹†† 稲垣 康善†††

† 名古屋大学大学院国際開発研究科 〒 464-8601 名古屋市千種区不老町
†† 名古屋大学情報連携基盤センター 〒 464-8601 名古屋市千種区不老町
††† 愛知県立大学情報科学部 〒 480-1198 愛知県愛知郡長久手町
E-Mail: yosihide@gsid.nagoya-u.ac.jp

概要 本稿では，漸進的構文解析により生成された部分構文木の妥当性を逐次的に評価する手法を提案する．本手法は，単語が入力されるたびに，それ以前の段階で生成されたすべての部分構文木についてその妥当性を確率木接合文法に基づき評価する．評価値は，単語入力のたびに更新され，それが閾値を超えた段階で，その部分構文木を出力する．構文解析実験により，提案手法が，高カバー率，高精度な漸進的構文解析の実現に有効であることを確認した．

Incremental Parsing based on Probabilistic Tree Adjoining Grammar

Yoshihide Kato† Shigeki Matsubara†† Yasuyoshi Inagaki†††

† Graduate School of International Development, Nagoya University
†† Information Technology Center, Nagoya University
Furo-cho, Chikusa-ku, Nagoya, 464-8603 Japan
††† Faculty of Information Science and Technology, Aichi Prefectural University
1522-3 Ibaragabasama, Kumabari, Nagakute-cho, Aichi-gun, 480-1198 Japan
E-Mail: yosihide@gsid.nagoya-u.ac.jp

Abstract This paper proposes a method for evaluating the validity of partial parse trees constructed in incremental parsing. Our method is based on probabilistic tree adjoining grammar, and it incrementally evaluates the validity for each partial parse tree on a word-by-word basis. In our method, incremental parser returns partial parse trees at the point where the validity for the partial parse tree becomes greater than a threshold. Experimental results demonstrate that incremental evaluation of partial parse trees realizes a broad-coverage accurate incremental parser.

1 はじめに

漸進的構文解析とは，自然言語文をその単語の出現順序に従って順次解析し，文の入力途中の段階でその構文構造を捉える枠組である．同時通訳システムやリアルタイム字幕生成システムなどの実時間音声言語処理システムの実現に必要不可欠な要素技術の一つである [1, 2, 3].

これまでに，漸進的構文解析手法がいくつか提案されている [4, 5, 6, 7, 8]．これらの手法では，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対する部分構文木の候補を生成できる．しかし，漸進的構文解析に特有な局所的曖昧性の問題をこれらの手法は全く扱っていない．局所的曖昧性とは，文

の断片に対する構文構造の曖昧性であり，その断片の情報のみでは解消できないが，後続の入力の情報がある程度わかれば解消できる曖昧性のことである．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言語処理モジュールとして組み込んだ場合，局所的曖昧性の解消は，システム全体の性能に影響を与えるため重要である．

一方，局所的曖昧性の問題に対して，Marcus は単語を数単語先読みすることにより，妥当な部分構文木を決定的に生成する手法を提案している [9]．また，Kato らは，漸進的構文解析の出力タイミングを遅らせ，生成された部分構文木の候補の中から，妥当な部分構文木が決定されてからそれを出力する手法を提案している [10]．しかしながら，これらの手法によりカバー率の高い漸進的構文解析が実現できるとは言い難い．Marcus の手法では，先読み規則を人手で作成しているが，カバー率の高い規則を得ることができるかどうかは明らかではない．Kato らの手法は，文脈自由文法をベースに可能なすべての部分構文木を生成することを前提としているため，大規模な文法を扱うことは難しい．

そこで本稿では，部分構文木の妥当性を逐次的に評価する漸進的構文解析手法を提案する．提案手法では，確率木接合文法を大規模構文木コーパスから抽出し，これをベースに漸進的構文解析を実現する．また，Kato らの手法のように，文法上可能なすべての部分構文木を生成するのではなく，ビームサーチにより確率の低い部分構文木を枝刈りする．これにより，カバー率の高い漸進的構文解析の実現が期待できる．局所的曖昧性を解消するために，本手法では，単語が入力されるたびに，それ以前の段階で生成されたすべての部分構文木についてその妥当性を評価する．評価値は，単語が入力されるたびに，その単語の情報を反映した値に更新され，それがあらかじめ定めた閾値を超えた段階で，その部分構文木を出力する．これにより部分構文木の出力タイミングは，単語の入力タイミングに対して遅れるものの，局所的曖昧性が解消できるため，精度の高い漸進的構文解析の実現が期待できる．

本稿の構成は以下の通りである．次の 2 節では，確率木接合文法に基づく漸進的構文解析を提案する．3 節では，漸進的構文解析において生成される部分構文木の妥当性について論じる．4 節では，部分構文木の妥当性を漸進的に評価する手法を提案する．5 節では，提案手法の実験による評価について報告する．

2 木接合文法に基づく漸進的構文解析

本節では，木接合文法 (tree adjoining grammar, TAG) [11] に基づく漸進的構文解析手法を提案する．

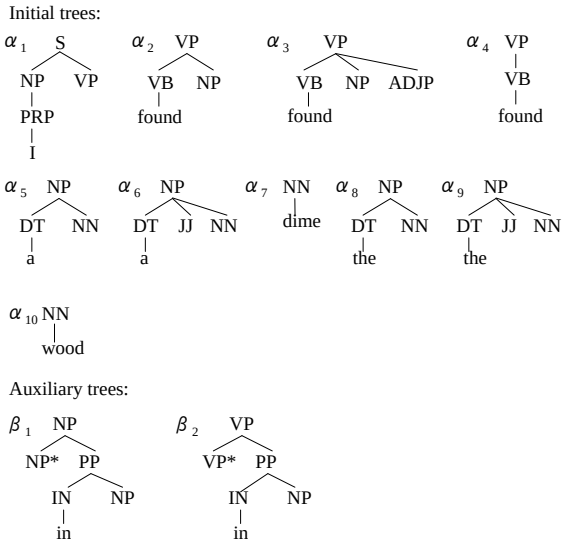

図 1: 漸進的構文解析のための木接合文法

2.1 漸進的構文解析のための木接合文法

木接合文法は**初期木** (initial tree) と**補助木** (auxiliary tree) の集合からなる．初期木と補助木をあわせて**基本木** (elementary tree) と呼ぶ．補助木には，**足** (foot) と呼ばれる特別な節点があり，*でマークされている．漸進的な解析処理を実現するために，提案手法では，基本木の形を次のように制限する．まず，**最左展開木**を定義する．

定義 1 (最左展開木) 最左展開木は，次の *1.* か *2.* のいずれかである．

1. t を終端記号，X を非終端記号とする．このとき，$[t]_X$ は最左展開木である．
2. σ を最左展開木とし，X, X_1, …, X_k を非終端記号とする．このとき，$[\sigma X_1 \dots X_k]_X$ は最左展開木である． □

基本木の形を次のように制限する．

- すべての初期木は，最左展開木である．
- すべての補助木は，次の形をした木である．ただし，$X, X_1, \dots, X_k$ を非終端記号とし，σ を最左展開木とする．

$$[X^* \sigma X_1 \dots X_k]_X$$

図 1 に上記の制限を満たした木接合文法の例を示す．これらの基本木は，**代入** (substitution)，及び**接合** (adjunction) と呼ばれる 2 つの操作により結合され，**部分構文木** (partial pase tree) を構成する．以下では，非終端記号をラベルにもつ節点を**非終端節点**と，非終端記号をラベルにもつ葉を**非終端葉**と呼ぶ．

表 1: “I found a dime in the wood.” に対する漸進的構文解析の処理過程

word	#	partial parse tree
	1	s
I	2	$[[[I]_{prp}]_{np} vp]_s$
found	3	$[[[I]_{prp}]_{np}[[found]_{vb} np]_{vp}]_s$
	4	$[[[I]_{prp}]_{np}[[found]_{vb} np\ adjp]_{vp}]_s$
	5	$[[[I]_{prp}]_{np}[[found]_{vb}]_{vp}]_s$
a	6	$[[[I]_{prp}]_{np}[[found]_{vb}[[a]_{dt} nn]_{np}]_{vp}]_s$
	7	$[[[I]_{prp}]_{np}[[found]_{vb}[[a]_{dt} jj\ nn]_{np}]_{vp}]_s$
	8	$[[[I]_{prp}]_{np}[[found]_{vb}[[a]_{dt} nn]_{np} adjp]_{vp}]_s$
	9	$[[[I]_{prp}]_{np}[[found]_{vb}[[a]_{dt} jj\ nn]_{np} adjp]_{vp}]_s$
dime	10	$[[[I]_{prp}]_{np}[[found]_{vb}[[a]_{dt}[dime]_{nn}]_{np}]_{vp}]_s$
	11	$[[[I]_{prp}]_{np}[[found]_{vb}[[a]_{dt}[dime]_{nn}]_{np} adjp]_{vp}]_s$
in	12	$[[[I]_{prp}]_{np}[[[found]_{vb}[[a]_{dt}[dime]_{nn}]_{np}]_{vp}[[in]_{in} np]_{pp}]_{vp}]_s$
	13	$[[[I]_{prp}]_{np}[[found]_{vb}[[[a]_{dt}[dime]_{nn}]_{np}[[in]_{in} np]_{pp}]_{np}]_{vp}]_s$
the	14	$[[[I]_{prp}]_{np}[[[found]_{vb}[[a]_{dt}[dime]_{nn}]_{np}]_{vp}[[in]_{in}[[the]_{dt} nn]_{np}]_{pp}]_{vp}]_s$
	15	$[[[I]_{prp}]_{np}[[[found]_{vb}[[a]_{dt}[dime]_{nn}]_{np}]_{vp}[[in]_{in}[[the]_{dt} jj\ nn]_{np}]_{pp}]_{vp}]_s$
	16	$[[[I]_{prp}]_{np}[[found]_{vb}[[[a]_{dt}[dime]_{nn}]_{np}[[in]_{in}[[the]_{dt} nn]_{np}]_{pp}]_{np}]_{vp}]_s$
	17	$[[[I]_{prp}]_{np}[[found]_{vb}[[[a]_{dt}[dime]_{nn}]_{np}[[in]_{in}[[the]_{dt} jj\ nn]_{np}]_{pp}]_{np}]_{vp}]_s$
wood	18	$[[[I]_{prp}]_{np}[[[found]_{vb}[[a]_{dt}[dime]_{nn}]_{np}]_{vp}[[in]_{in}[[the]_{dt}[wood]_{nn}]_{np}]_{pp}]_{vp}]_s$
	19	$[[[I]_{prp}]_{np}[[found]_{vb}[[[a]_{dt}[dime]_{nn}]_{np}[[in]_{in}[[the]_{dt}[wood]_{nn}]_{np}]_{pp}]_{np}]_{vp}]_s$

代入 σ を部分構文木とし，その最も左に位置する非終端葉を η とする．η のラベルを X とする．このとき，η を，根のラベルが X である初期木 α で置き換える．α を代入する操作を s_α と書き，σ に s_α を適用した結果を $s_\alpha(\sigma)$ と書く．

接合 木 σ を非終端節点 η で分割し，その節点と同じラベルを根にもつ補助木 β を挿入する．β を接合する操作を a_β と書き，σ に a_β を適用した結果を $a_\beta(\sigma)$ と書く．

句構造規則に基づく漸進的構文解析では，文法が左再帰規則を含む場合，解析が無限ループに陥ってしまうが，接合操作を用いて左再帰構造を処理することにより，この問題を回避できる [12].

提案手法では，i 番目の単語 w_i が入力されたとき，$w_1 \cdots w_{i-1}$ に対する部分構文木に対して，w_i に対する基本木を代入・接合することにより，$w_1 \cdots w_{i-1} w_i$ に対する部分構文木を生成する．

例として，図 1 の木接合文法を用いて，次の文を漸進的構文解析する場合を考える．

I found a dime in the wood. (1)

表 1 は文 (1) に対する部分構文木生成の過程を示している．単語 “found” が入力されたとき，“I” に対する部分構文木#2 に対して図 1 の初期木 $\alpha_2, \alpha_3, \alpha_4$ を代入することにより部分構文木#3, #4 および #5 がそれぞれ生成される．単語 “in” が入力されたとき，‘I found a dime” に対する部分構文木#10 に対して β_1 と β_2 を接合することにより，部分構文木#12 と #13 が生成される．この例が示すように，木接合文法に基づき，単語入力ごとに，それまでに入力された文の断片に対する部分構文木の候補を生成する漸進的構文解析が実現できる．

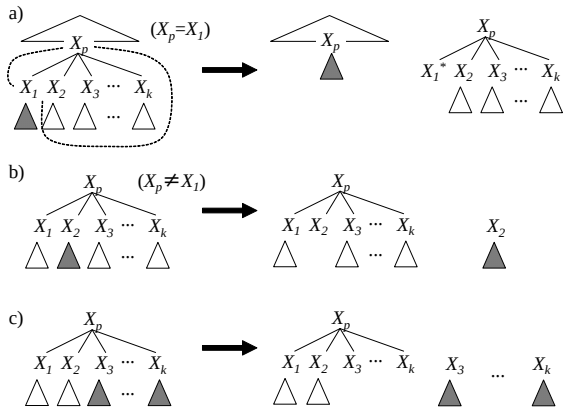

図 2: 構文木分割による基本木抽出

2.2 構文木コーパスからの文法抽出

カバー率の高い木接合文法を構築する方法の一つとして，文献 [13, 14, 15] の手法のように，構文木コーパス中の構文木を分割し，基本木を抽出することが考えられる．本節では，2.1 節で述べたような基本木を構文木コーパスから抽出する方法を提案する．

本手法では，次のようにして，コーパス中の構文木を分割する（図 2 参照.）.

a) 左端に位置する節点 η_1 がその親 η_p と同じ記号をラベルにもつならば，η_1 と η_p で構文木を分割し，上側の木と下側の木を結合する．中間の木の η_1 は足節点である．

b) 左から 2 番目に位置する節点 η_2 に対して，その左の節点 η_1 と，その親 η_p が同じ記号をラベ

ルに持たないならば，η_2 で構文木を分割する．

c) 左から 3 番目以降に位置する節点で構文木を分割する．

例えば，図 1 の初期木 α_1, α_2, α_5, α_7, α_8, α_{10} 及び，補助木 β_2 が表 1 の構文木#18 から抽出される．

提案する基本木抽出手法は，文献 [13, 14, 15] の手法と似ているが，分割する節点の選び方に違いがある．文献 [13, 14, 15] の手法では，統語範疇の主辞情報を用いて，主辞以外の節点で構文木を分割するのに対して，提案手法では，左再帰節点と左端に位置しない節点で構文木を分割する．本提案手法により抽出された基本木は，単語単位で左から順に結合することができるのに対して，従来の手法で得られる基本木は，このような性質を持っていない[1]．

2.3 確率木接合文法

本節では確率木接合文法について述べる．本手法では，基本木を部分構文木に代入・接合する事象に確率を割り当てる．この確率をコーパス中のデータから直接推定できればよいが，データスパースネスの問題が生じ，現実的でないので，本手法では，Chiang の手法 [14] のように，基本木が生成される過程を 2 つのステップにわける，すなわち，まず，基本木から単語を除いた**テンプレート** (template) を生成し，次に単語を生成する．代入操作 s_α（接合操作 a_β）が部分構文木 σ の節点 η に適用される確率を，次のように分解する．

$$P(s_\alpha \mid \sigma, \eta) = P(s_{\alpha'} \mid \sigma, \eta) P(w_\alpha \mid s_{\alpha'}, \sigma, \eta) \quad (2)$$

$$P(a_\beta \mid \sigma, \eta) = P(a_{\beta'} \mid \sigma, \eta) P(w_\beta \mid a_{\beta'}, \sigma, \eta) \quad (3)$$

α' は α のテンプレート，w_α は α の単語である．β についても同様である．ここで，単語が生成される確率がその品詞のみに依存すると仮定する．すなわち，単語の生成確率を次のように定義する．

$$P(w_\alpha \mid s_{\alpha'}, \sigma, \eta) = P(w_\alpha \mid t_\alpha) \quad (4)$$

$$P(w_\beta \mid a_{\beta'}, \sigma, \eta) = P(w_\beta \mid t_\beta) \quad (5)$$

t_α, t_β は，それぞれ，単語 w_α, w_β の品詞である．

次にテンプレートを代入，接合する確率について考える．本手法では，テンプレート α' を部分構文木 σ の節点 η に代入する確率が，η のラベル X，η の親のラベル X_p，η の祖父のラベル X_g，η の左隣の節点のラベル X_s，η の 2 つ左の節点のラベル X_{ss} に依存すると仮定する（図 3 参照）．テンプレートの代入確率は次のようになる．

[1]例えば，構文木#18 を分割する場合を考える．名詞句 $[[a]_{dt}[dime]_{nn}]_{np}$ の主辞は，nn である．したがって，従来の手法では，dt でラベル付けされた節点で分割され，“a” に対する初期木として $[a]_{dt}$ が得られる．しかし，この初期木 $[a]_{dt}$ は，“I found” に対する部分構文木と結合できない．この木を代入できる節点は，“dime” に対する初期木 $[dt[dime]_{nn}]_{np}$ に存在し，“I found” に対する部分構文木には存在しないからである．

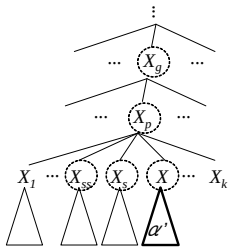

図 3: テンプレートの代入確率

$$P(s_{\alpha'} \mid \sigma, \eta) = P(s_{\alpha'} \mid X, X_p, X_g, X_s, X_{ss}) \quad (6)$$

データスパースネスの問題に対処するため，低次の言語モデルを用いて次のようにスムージングする．

$$\begin{aligned} & P(s_{\alpha'} \mid X, X_p, X_g, X_s, X_{ss}) \\ = \; & \lambda_1 P_{ML}(s_{\alpha'} \mid X, X_p, X_g, X_s, X_{ss}) \\ & + \lambda_2 P_{ML}(s_{\alpha'} \mid X, X_p, X_s, X_{ss}) \\ & + (1 - \lambda_1 - \lambda_2) P_{ML}(s_{\alpha'} \mid X, X_p, X_s) \end{aligned}$$

ここで，P_{ML} はコーパスから最尤推定により計算された確率を表す．λ_1 と λ_2 は補間係数である．

テンプレート β' を部分構文木 σ の節点 η に接合する確率は，代入確率と同じように定義する．

$$P(a_{\beta'} \mid \sigma, \eta) = P_{ML}(a_{\beta'} \mid X, X_p, X_g, X_s) \quad (7)$$

接合操作が節点 η に適用されない確率は次のように計算する．

$$\begin{aligned} & P(nil_X \mid X, X_p, X_g, X_s) \\ = \; & 1 - \sum_{\beta' \in A} P(a_{\beta'} \mid X, X_p, X_g, X_s) \end{aligned} \quad (8)$$

nil_X は X をラベルにもつ節点に接合操作が適用されないことを表す．A は補助木のテンプレートの集合である．

部分構文木の生起確率は，その生成に必要な操作の確率の積として計算する．

2.4 解析戦略

効率的な解析処理を実現するために，本手法では，解析戦略としてビームサーチを採用する．i 番目の単語 w_i が入力された時点で生成された $w_1 \cdots w_i$ に対する部分構文木のそれぞれについて，その生起確率を計算し，上位 N 個（N は固定）の部分構文木を残し，その他の部分構文木は削除する．

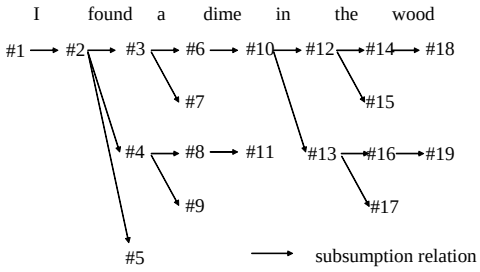

図 4: 部分構文木間の包含関係

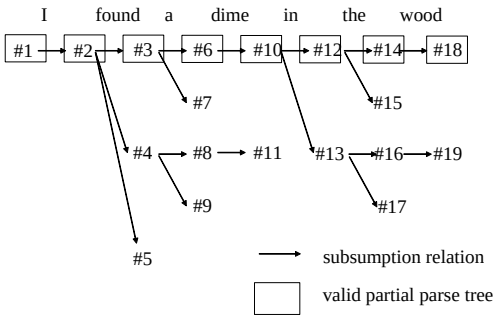

図 5: 妥当な部分構文木

3 部分構文木の妥当性

本節では，部分構文木の妥当性に関していくつか定義を与える．部分構文木の妥当性について述べる準備として，部分構文木間の包含関係を定義する．

定義 2 (包含関係) σ と τ を部分構文木とする．ある初期木 α に対して，$s_\alpha(\sigma) = \tau$，あるいは，ある補助木 β に対して，$a_\beta(\sigma) = \tau$ のいずれかが成り立つならば，$\sigma \triangleright \tau$ である．$\triangleright^*$ を $\triangleright$ の反射推移閉包とする．$\sigma \triangleright^* \tau$ ならば，σ は τ を**包含する**という．□

σ が τ を包含するとは，σ に対して代入や接合をいくつか適用すると τ が得られることを意味する．図 4 に，文 (1) に対して生成される部分構文木の間の包含関係を示す．

文の断片に対する部分構文木が構文的関係を正しく捉えているならば，その部分構文木は，文全体に対する正解構文木を包含している．そのような部分構文木を**妥当な**部分構文木と呼び，次のように定義する．

定義 3 (妥当な部分構文木) σ を部分構文木とし，$w_1 \cdots w_n$ を入力文とする．σ が $w_1 \cdots w_n$ に対する正解構文木を包含するならば，σ は $w_1 \cdots w_n$ に対して**妥当である**という．□

例えば，#18 を文 (1) に対する正解構文木であると仮定する．このとき，#3 $\triangleright^*$ #18 であるので，部分構文木#3 は (1) に対して妥当である．一方，部分構文木#4 と#5 は (1) に対して妥当ではない．図 5 に，文 (1) に対して妥当な部分構文木を示す．

4 部分構文木の妥当性の評価

文の断片に対する部分構文木の妥当性は，その断片に続く入力に依存している．例えば，部分構文木 #3 〜#5 の妥当性は，単語 “found” に続く入力に依存している．これは，単語入力ごとに部分構文木の妥当性が変わることを意味している．そこで，部分構文木の妥当性を次のように定義する．

$$V(\sigma \mid w_1 \cdots w_j) = \frac{\sum_{\tau \in Sub(\sigma, w_1 \cdots w_j)} P(\tau)}{\sum_{\tau \in T(w_1 \cdots w_j)} P(\tau)} \quad (9)$$

ここで，σ は文の断片 $w_1 \cdots w_i (i \leq j)$ に対する部分構文木であり，$T(w_1 \cdots w_j)$ は文の断片 $w_1 \cdots w_j$ に対して漸進的構文解析が生成した部分構文木の集合である．また，$Sub(\sigma, w_1 \cdots w_j)$ は $T(w_1 \cdots w_j)$ の部分集合で，σ に包含されるものの集合である．式 (9) は入力が $w_1 \cdots w_j$ までわかった段階での σ の妥当性を表現している．σ が入力文に対して妥当であるのは，σ に包含される $w_1 \cdots w_j$ に対する部分構文木のいずれかが妥当な場合である．式 (9) は，生成された部分構文木のうちのそのような部分構文木の割合を表している．

4.1 部分構文木の出力

局所的曖昧性を解消するため，提案手法では，部分構文木の妥当性が閾値を超えた段階で，それを出力する．

j 番目の単語 w_j が入力されたとき，つぎのような部分構文木を出力する．

$$argmax_{\{\sigma : V(\sigma, w_1 \cdots w_j) \geq \theta\}} l(\sigma) \quad (10)$$

ここで，θ は閾値で $[0, 1]$ の値をとる．$l(\sigma)$ は σ が含む単語の数である．出力される部分構文木は，式 (9) が閾値 θ を超える部分構文木のうち，最も多くの単語を含むものである．

4.2 解析例

文 (1) に対する解析例を考える．閾値 $\theta = 0.8$ とする．各操作の適用確率は，表 2 のとおりとする．文 (1) に対して妥当な部分構文木である#3 がどのタイミングで出力されるかについて考える．単語 “found” が

表 2: 操作の適用確率

operation	probability
s_{α_1}	1.0
s_{α_2}	0.7
s_{α_7}, $s_{\alpha_{10}}$	0.5
s_{α_5}, s_{α_8}	0.3
s_{α_4}, s_{α_6}, s_{α_9}	0.2
s_{α_3}	0.1
a_{β_1}	0.3
a_{β_2}	0.7
nil_{NP}	0.7
nil_{VP}	0.3

表 3: 出力される部分構文木

input word	output partial parse tree
I	#2
found	
a	#3
dime	#10
in	#12
the	
wood	#18

入力されたとき，部分構文木#3 〜#5 が生成される．すなわち，$T(\text{I found}) = \{\#3, \#4, \#5\}$ である．図 4 に示すように，$Sub(\#3, \text{I found}) = \{\#3\}$ である．さらに，$P(\#3) = 0.7$, $P(\#4) = 0.1$，$P(\#5) = 0.2$ である．したがって，$V(\#3, \text{I found}) = 0.7/(0.7 + 0.1 + 0.2) = 0.7$ である．$V(\#3, \text{I found}) < \theta$ であるので，部分構文木#3 はこの時点では出力されない．単に，候補として保持しておくだけである．

次の単語 “a” が入力されたとき，部分構文木#6 〜#9 が生成される．$P(\#6) = 0.21$，$P(\#7) = 0.14$，$P(\#8) = 0.03$，$P(\#9) = 0.02$ であり，$Sub(\#3, \text{I found a}) = \{\#6, \#7\}$ である．したがって，$V(\#3, \text{I found a}) = (0.21 + 0.14)/(0.21 + 0.14 + 0.03 + 0.02) = 0.875$ であるので，$V(\#3, \text{I found a}) \geq \theta$ であり，部分構文木#3 はこの時点で出力される．

表 3 に，各単語入力に対して出力される部分構文木を示す．

この例が示すように，提案する漸進的構文解析では，部分構文木の出力を遅らせる．一方で，この例において出力された部分構文木はすべて妥当なものとなっている．

5 実験による評価

提案手法の性能を評価するために，構文解析実験を行った．提案する漸進的構文解析を GNU Common Lisp で実装した．文法は，Penn Treebank[16] の WSJ コーパスのセクション 2-21 から抽出した（構文木は，Klein らの手法 [17] に従っていくつかの情報を付与している．さらに，Roark の手法 [18] に従って構文木を変換している．）．補間係数の λ_1，λ_2 は，EM アルゴリズムにより推定し，学習データとしてセクション 24 を用いた．ビームサーチの幅は 500 である，すなわち漸進的構文解析の過程で，確率値が上位 500 位以内の部分構文木だけを残し，それ以外は削除して解析を進めた．

セクション 23 の文に対して，ラベル正解率とラベル再現率はそれぞれ 83.1%，81.4%であった．実験の対象として，セクション 23 の文のうち，漸進的構文解析の最終的な解析結果が正解構文木と一致した 417 文を用いた．これらの文の平均文長は 13.5 単語である．閾値が，0.5, 0.6, 0.7, 0.8, 0.9 及び 1.0 に場合について，解析精度と遅延を測定した．

遅延は次のように定義する．$s = w_1 \cdots w_n$ を入力文とし，$o_j(s)$ を，j 番目の単語 w_j が入力されたときに出力された部分構文木とする．j 番目の単語が入力されたときの遅延を次のように定義する．

$$D(j, s) = j - l(o_j(s)) \tag{11}$$

最大遅延 $D_{max}(s)$ と平均遅延を次のように定義する．

$$D_{max}(s) = \max_{1 \leq j \leq n} D(j, s) \tag{12}$$

$$D_{ave}(s) = \frac{1}{n} \sum_{j=1}^{n} D(j, s) \tag{13}$$

解析精度は，出力された部分構文木のうちの妥当なものの割合と定める．

さらに，ベースラインとして，遅延が常に 0 である解析，すなわち，確率が最も大きい部分構文木を出力する方法についても，解析精度と遅延を測定した．

表 4 に解析精度と遅延を示し，図 6 にそれらの間の関係を示す．

この実験結果は，精度と遅延の間のトレードオフの関係を示している．ベースラインと比較して，出力は遅れるものの，解析精度は大幅に向上している．$\theta = 1$ のとき，出力される部分構文木の妥当性は保障され，この場合，加藤らの手法 [10] と似た手法とみなすことができる．この場合と比較すると，$\theta < 1$ のとき，大きく遅延が減少している．

この実験結果は必ずしも，提案手法が最善のものであることを示すわけではないが，高い解析精度と低い遅延の解析をある程度達成できていると言える．

表 4: 解析精度と遅延

	precision(%)	D_{max}	D_{ave}
$\theta = 1.0$	100.0	13.2	7.2
$\theta = 0.9$	99.3	10.2	4.9
$\theta = 0.8$	96.8	8.9	3.9
$\theta = 0.7$	95.3	7.7	3.2
$\theta = 0.6$	92.3	6.6	2.6
$\theta = 0.5$	88.6	5.4	1.9
baseline	74.3	0.0	0.0

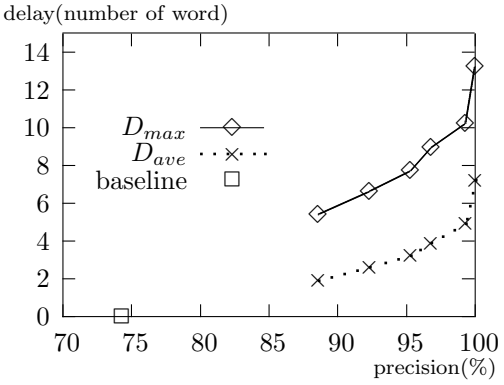

図 6: 解析精度と遅延の関係

6 おわりに

本稿では，漸進的構文解析において生成された部分構文木の妥当性を評価する手法を提案した．この手法は，確率木接合文法に基づいている．単語が入力されるたびに，各部分構文木に対してその妥当性を随時計算し，妥当性が閾値を超えた段階でそれを出力する．本手法では，出力すべき部分構文木の決定を遅らせる．

提案手法を評価するために，Penn Treebank を用いた構文解析実験を実施した．この実験により，提案手法が漸進的構文解析の精度向上に有効であることが確認できた．

実験結果は，解析精度と遅延のトレードオフを示している．漸進的構文解析の性能を総合的に評価するために，遅延と解析精度の両者を総合的に評価する尺度が必要であるが，これについては今後検討したい．

参考文献

[1] J. Allen, G. Ferguson, and A. Stent, “An Architecture for More Realistic Conversational Systems,” Proceedings of International Conference of Intelligent User Interfaces, pp.1–8, 2001.

[2] Y. Inagaki and S. Matsubara, “Models for Incremental Interpretation of Natural Language,” Proceedings of the 2nd Symposium on Natural Language Processing, pp.51–60, 1995.

[3] D. Milward and R. Cooper, “Incremental Interpretation: Applications, Theory, and Relationship to Dynamic Semantics,” Proceedings of the 15th International Conference on Computational Linguistics, pp.748–754, 1994.

[4] F. Costa, V. Lombardo, P. Frasconi, and S. G., “Wide Coverage Incremental Parsing by Learning Attachment Preferences,” Proceedings of the 7th Congress of the Italian Association for Artificial Intelligence, pp.297–307, 2001.

[5] N.J. Haddock, “Incremental Interpretation and Combinatory Categorial Grammar,” Proceedings of the 10th International Joint Conference on Artificial Intelligence, pp.661–663, 1987.

[6] S. Matsubara, S. Asai, K. Toyama, and Y. Inagaki, “Chart-based Parsing and Transfer in Incremental Spoken Language Translation,” Proceedings of the 4th Natural Language Processing Pacific Rim Symposium, pp.521–524, 1997.

[7] D. Milward, “Incremental Interpretation of Categorial Grammar,” Proceedings of the 7th Conference of European Chapter of the Association for Computational Linguistics, pp.119–126, 1995.

[8] B. Roark, “Probabilistic Top-Down Parsing and Language Modeling,” Computational Linguistics, vol.27, no.2, pp.249–276, 2001.

[9] M. Marcus, A Theory of Syntactic Recognition for Natural Language, MIT Press, Cambrige, MA, 1980.

[10] Y. Kato, S. Matsubara, K. Toyama, and Y. Inagaki, “Spoken Language Parsing based on Incremental Disambiguation,” Proceedings of the 6th International Conference on Spoken Language Processing, pp.999–1002, 2000.

[11] A.K. Joshi, “Tree Adjoining Grammar: How Much Context-Sensitivity is required to provide reasonable structural descriptions?,” in

Natural Language Parsing, ed. D.R. Dowty, L. Karttunen, and A. Zwicky, pp.206–250, Cambridge University Press, Cambridge, 1985.

[12] V. Lombardo and P. Sturt, "Incremental Processing and Infinite Local Ambiguity," Proceedings of the 19th Annual Conference of the Cognitive Science Siciety, pp.448–453, 1997.

[13] J. Chen and K. Vijay-Shanker, "Automated Extraction of TAGs from the Penn Treebank," Proceedings of the 6th International Workshop on Parsing Technologies, pp.65–76, 2000.

[14] D. Chiang, "Statistical Parsing with an Automatically Extracted Tree Adjoining Grammar," in Data-Oriented Parsing, ed. R. Bod, R. Scha, and K. Sima'an, pp.299–316, CSLI Publications, Stanford, 2003.

[15] F. Xia, "Extracting Tree Adjoining Grammars from Bracketed corpora," Proceedings of the 5th Natural Language Processing Pacific Rim Symposium, pp.398–403, 1999.

[16] 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.

[17] D. Klein and C.D. Manning, "Accurate Unlexicalized Parsing," Proceedings of the 41th Annual Meeting of the Ass0ciation for Computational Linguistics, pp.423–430, 2003.

[18] B. Roark, "Robust Garden Path Parsing," Natural Language Engineering, vol.10, no.1, pp.1–24, 2004.