社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2005－NL－167 (10)
2005－SLP－ 56 (10)
2005／5／27

選好依存文法 (PDG) における文解析能力の評価方式について

平川 秀樹
(株) 東芝 研究開発センター 知識メディアラボラトリ
email: hideki.hirakawa@toshiba.co.jp

選好依存文法 (PDG) は，自然言語の形態素，構文，意味に渡る複数の圧縮共有データ構造を有し，各レベルの選好知識を活用して文解析を統合的に行う枠組みである．従来の文解析性能評価法は，構文木に基づく方式が主であり，依存構造を最終出力とする PDG への直接的適用は困難である．また，PDG では，最終出力結果に対する評価だけでなく，システムの選好能力の評価も重要であると考える．本稿では，PDG の出力結果である依存構造をベースに総合解析精度，正解候補生成能力，曖昧性解消能力を測定する各種指標を提案し，実際の解析実験により提案指標の振る舞いを分析する．

Evaluation Measures for Natural Language Analyser based on Preference Dependency Grammar

Hideki Hirakawa
Knowledge Media Laboratory
TOSHIBA Corporate Research & Development Center
1, Komukai-Toshiba-cho, Saiwaiku, Kawasaki, 212-8582, Japan

Preference Dependency Grammar (PDG) is a new framework for the morphological, syntactic and semantic analysis using multiple kinds of paked shared data structures to utilize multi-level preference knowledge. Most of traditional sentence analysis measures utilize phrase structures and are not directly applicable to PDG-based systems whose final outputs are depencency structures. It is important to measure not only the total ability but also the preference ability of PDG-based systems. This paper proposes four evaluation measures for the total analysis accuracy, the hypothesis generation ability and the disambiguation ability of PDG-based system. This paper also gives an experiment for analyzing these measures.

1 はじめに

自然言語解析システムの評価は，できるだけ標準的な手法を利用することが望ましいが，そのシステムが目標とする能力を測定することが前提となる．現在提案中の選好依存文法 (PDG:Preference Dependency Grammar) は，選好知識と制約知識を統合して扱い，最終出力として依存構造を生成するなどの特徴を持っており，必ずしも従来の評価方式が適用できるというわけではない．このため本稿では，PDG に即した依存構造をベースにした新しい性能評価方式を提案する．

自然言語解析システムの評価方式については，文献 1) にサーベイされているように，評価対象機能や評価項目に応じて様々な提案がなされている．構文木の句境界をベースに適合率と再現率を計算する GEIG(Grammar Evaluation Interest Group Scheme) の手法[2] は，複数の構文解析システムの比較が容易という特長を持つ反面，評価結果が実際の感覚にマッチしないなどの問題が指摘されている[1]．文献 3) は，文の木構造中の文法カテゴリ情報を反映した評価方式を提案し，句の境界をベースにした方式よりも正解度が直感に合うと主張している．ただし，この手法では構文カテゴリを統一しないと複数のシステム間の比較ができないという課題がある．構文木は，構文解析文法と密接に関係し，一般にそれぞれのシステムが異なった文法カテゴリの集合を持つのは避けられない．このため，構造のディテールに影響されないようにため，構文木そのもので精度を測定するのではなく，構文木から導き出される語の間の文法的・論理構造的な依存関係をベースに性能を測定するという関係スキーマ (relational schema) と呼ばれる方法が提案されるようになってきている[4),5),6]が，標準的な手法は確立されてはいない．一方，依存構造の精度評価については，文献 3) で提案が見られるが，標準的な方法は確立されていない．

本稿の評価対象である PDG は，後述するように句構造と依存構造の両者の解析形態を持つが，PDG の最終解析結果が依存構造であることや解析評価の関係スキーマへの流れから依存構造に基づく評価を採用し，総合解析精度の指標として「アーク正解率」(APR:Arc Precision Ratio) ならびに「単語係り受け正解率」(WDPR:Word Dependency Precision Ratio) を提案する．また，PDG では，最終的な解析結果だけでなく，正解解釈を候補として生成する能力や可能な複数解釈に対する正解の選択能力も評価ポイントであるため，正解候補生成能力の指標として「正解可能文生成率」(PCSR: Possibly Correct Sentence Ratio) を，曖昧性解消能力の指標として「アーク曖昧性解消率」(ADPR: Arc Disambiguation Precision Ratio) を提案し，実際の解析実験によりこれら指標の振る舞いについて分析する．

2 選好依存文法 (PDG) の概要

2.1 選好依存文法の処理フロー

PDG は，形態素，構文，意味の 3 レベルの解析を行う枠組みであり，各レベルの知識の選好的適用[*1] と総合解釈を可能とすることにより，潜在的な解釈を切り捨てることなく全体の可能性の中から効率的に最適な解釈を取り出すことを目標に設計されている[9]．こ

[*1] 言語解析における知識の選好的適用とは，生成された解釈に優先順位をつけるような適用の仕方である．一方，制約的適用は解釈を棄却するような適用の仕方である．

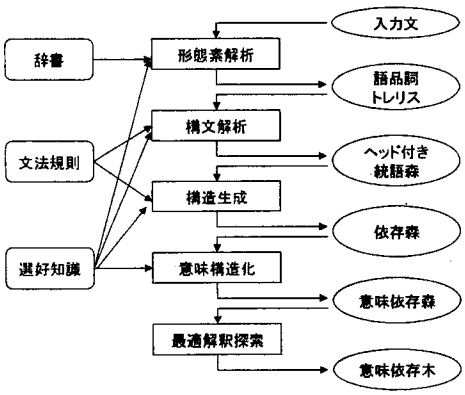

図 1: PDG の解析フロー図

のため，「共有構造」「優先度設定」「最適解導出」の枠組みを形態素処理から意味処理に渡って提供する．図 1 が PDG の全体フローを示している．形態素，構文，意味の解析処理コンポーネントが存在し，それらは，基本的に自然言語処理の各レベルの全ての解釈を内包する PDG の共有データ構造で結ばれている．詳細は省略するが，図のヘッド付き統語森は圧縮共有統語森[7] の 1 種であり可能な句構造解釈全体を保持する．また，依存森と意味依存森は，可能な依存構造解釈全体（依存木の集合）を保持する．これらデータ構造に対して知識の選好的適用による優先度を与えることが可能であり，最終的には，最適解釈探索部により統合評価され，最も確からしい解釈が計算される．

2.2 依存森

依存森は，文の可能な解釈である依存木の集合を圧縮共有するデータ構造であり，依存グラフ (Dependency Graph) と共起マトリックス (Co-occurence Matrix) より成る．依存森には大きく機能依存森 (Functional Dependency Forest) と意味依存森 (Semantic Dependency Forest) の 2 種類がある．単に依存森と記述した場合は，前者を対象とする．図 2 は，"Time flies like an arrow" に対するスコア付きの依存森の例である．依存グラフは，ノードと有向アークより成り，それぞれ語品詞[*2]，依存関係を表している．アークは，ID と選好スコアを有している．選好スコアは，選好知識を元に付与されるスコアであり，大きいほどそのアークの優先度が高くなる．共起マトリックスは，アーク集合を行と列に取り，アーク間の共起関係を規定する．共起マトリックス $CM(i,j)$ が○の場合に限り，アーク i と j は 1 つの依存木（解釈）において共起可能である．文の解釈となる依存木は，アーク間の共起関係が成立する，入力文全体を被覆するなどの整依存木条件を満足する整依存木 (Well-formed Dependency Tree) である[8]．

2.3 選好スコアと最適依存木

依存森の各アークならびにノードには，選好知識から計算される選好スコアと呼ばれる得点が付与され，解釈の優先度が表現される．例えば，語品詞のコーパスの頻度や語品詞間の係り受けの頻度などによるスコア付けが行なわれる．文に対する各解釈，すなわち，整依存木のスコアは，ノードやアークのスコアの総計として計算される[*3]．最大のスコアを持つ整依存木を最適解あるいは最適依存木と呼び，PDG の解析結果となる．最適依存木は，複数個存在する場合もある．

2.4 PDG における評価のポイント

PDG は，多レベルのデータを扱う枠組みであり，形態素構造，句構造，機能依存構造，意味依存構造を含んでいる．最終的な出力としては意味依存構造を想定しているが，本稿では，正解データや選好知識の準備などの観点から機能依存構造を評価対象とし，意味依存構造での評価は今後の課題とする．但し，評価手法自体は，意味依存構造に対しても適用可能であり，一般性を持っている．また，PDG の出力である依存木の精度評価，すなわち，解析精度はシステム全体の能力の評価の基本であるが，正解解釈を候補として生成する能力や可能な複数解釈に対する正解の選択能力も評価ポイントであり，この観点での評価指標を提案する．

3 依存構造の評価方式

3.1 解析精度の総合評価指標

解析精度の指標として，入力文に対する正解解釈 (正解依存木) と PDG の出力 (出力依存木) を元に計算する「アーク正解率」ならびに「単語係り受け正解率」を用いる．PDG では，選好スコアにより最適な依存木を算出するが，一般に 1 文に対して複数の最適解が存在しうる．このため，これら指標は，複数の解析結果を出力として許す方式になっている．

3.1.1 アーク正解率

アーク正解率は，出力依存木のアークの正解率であり，次の式で定義する．

$$\text{アーク正解率} = \frac{\text{出力依存木中の正解アーク数}}{\text{出力依存木の全アーク数}}$$

1 文に対して最適な正解候補が複数個存在する場合には，それら複数の解のアークを全て加算して計算する．以下具体例を示して説明する．

図 3 に示すように，例文"Time flies like an arrow" に対して，正解依存木が CDT であり，最適解として ODT_1,ODT_2 の 2 つの出力依存木が得られたとす

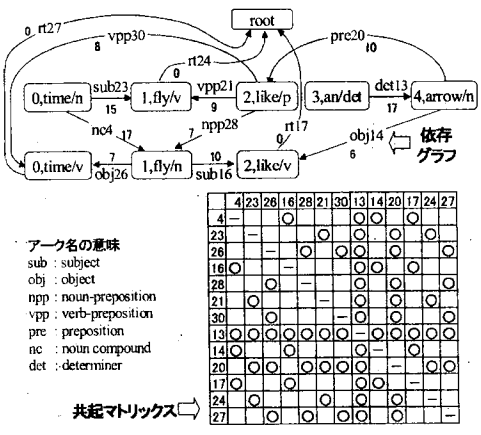

図 2: 例文に対するスコア付き依存森

[*2] 単語と品詞の組を語品詞 (WPP:Word POS Pair) と呼ぶ．単語 time は"time /v","time/n" 等の語品詞を持つ．

[*3] 図 2 は，アークのスコアのみの依存森である．

正解依存木:

```
CDT = { arc(sub,[time]-n-0,[flies]-v-1)},    /* ca1 */
      arc(root,[flies]-v-1),[])              /* ca2 */,
      arc(vpp,[like]-pre-2,[flies]-v-1)),    /* ca3 */
      arc(pre,[arrow]-n-4,[like]-pre-2),     /* ca4 */
      arc(det,[an]-det-3,[arrow]-n-4) }      /* ca5 */
```

出力依存木:

```
ODT1 ={ arc(sub,[time]-n-0,[flies]-v-1)},    /* oa1 */
      arc(root,[flies]-v-1),[]),             /* oa2 */
      arc(vpp,[like]-pre-2,[flies]-v-1)),    /* oa3 */
      arc(pre,[arrow]-n-4,[like]-pre-2),     /* oa4 */
      arc(det,[an]-det-3,[arrow]-n-4) }      /* oa5 */

ODT2= { arc(nc,[time]-n-0,[flies]-n-1),      /* oa6 */
      arc(sub,[flies]-n-1,[like]-v-2),       /* oa7 */
      arc(root,[like]-v-2),[]),              /* oa8 */
      arc(det,[an]-det-3,[arrow]-n-4),       /* oa9 */
      arc(obj,[arrow]-n-4,[like]-v-2) }      /* oa10 */
```

図 3: 例文に対する正解依存木と出力依存木

る[*4]. ODT_1,ODT_2 全体のアーク 10 個のうち，正解アークは，oa1～oa5,oa9 の 6 個であるので，この文に対するアーク正解率は，6/10=0.6 となる．文集合に対しては，それぞれの文の出力依存木の全アーク数と出力依存木中の正解アーク数を合計して全体としての比率を計算することとなる．

3.1.2 単語係り受け正解率

アーク正解率は，語品詞の別と依存関係の別を考慮した精度であり文解釈能力を比較する上では適切であるが，品詞体系や依存関係体系などシステム固有の体系をベースとしており，複数のシステムの出力の比較などには必ずしも適しているとはいえない．このため，こうした部分を排除した評価の方法として単語係り受け正解率を用いる[*5]．単語係り受け正解率は，アーク正解率の計算において依存関係名の違いや品詞の違いを全て無視して正解を計算する方式である．前節の例に対する単語係り受け正解率は，oa6,oa10 の 2 つが正解として加算されるため 8/10=0.8 となる．

3.2 正解候補生成/選好能力の評価指標

3.2.1 PDG の枠組みと正解候補生成/選好能力評価

仮説 (可能な解釈) の扱いという観点からいうと PDG の枠組みは，次の 3 要素からなっている[9]．

(a) 入力文に対する仮説生成
(b) 制約知識による仮説の棄却
(c) 選好知識による仮説の優先度付けと最適解釈の選択

(a) は，その解析システムが正解解釈を候補として内部的に生成しうるか否かの能力である．(b) は，生成された仮説から不要な仮説を棄却する能力である．(c) は，生成された仮説に対して適切な優先度付けをする能力 (選好知識と利用の仕方の良さ) である．

(a)～(c) は，一般的な概念であるが，依存森を評価対象とした場合には，(a),(b) の総計として，正解依存木が依存森に存在するか否かが 1 つの能力の指標と考えられる．本稿では，依存森中に正解依存木を含む文を正解可能文 (Possibly Correct Sentence) と呼び，正解生成能力の指標として，評価対象文全体に対する正解可能文の割合を採用し，これを「正解可能文生成率」と呼ぶ．一方，(c) は，依存森中から正解アークを選択する選好能力であり，本稿では，以下で述べる「アーク曖昧性解消率」を選好能力指標として提案する．

3.2.2 アーク曖昧性解消率

選好能力は，単純に出力の正否だけでなくその選好タスクの困難度に応じた能力測定を行なう必要がある．同じ正解アークを選択する場合でも，選択対象アークの候補数が 2 個の場合と 10 個の場合では，その困難度は大きく異なる．アーク曖昧性解消率では，選択対象アークの候補数に比例した得点をアサインすることで正解タスクの困難度を組み込む．また，もともと正解が生成されていない仮説に対しては，いかなる選好知識も正解を生成できないため，能力測定の対象から外すべきである．また，逆に，解の候補が正解 1 つに絞られているような場合は，いかなる選好知識を適用しても正解になるため，同様に評価対象から外すべきであると考える．以上のような考察のもとに，対象アークの数で重み付けしたスコアに対してどの程度正解が得られたかを示す「アーク曖昧性解消率」(曖昧性解消率とも記述する) を提案する．

図 4 に曖昧性解消率を求めるアルゴリズムを示す．このアルゴリズムは入力文に対する正解依存木 CDT，出力依存木 ODT_1～ODT_n(n は最適解の数)，依存グラフ DG を入力とする．出力依存木のアークのコレクションを $ODTArcs$ と記述する．step1 で 1 つの正解アーク $onearc$ を取り出す．$onearc$ が DG に含まれていない場合，すなわち，もともと正解を生成できない場合は，そのアークは評価の対象としない (step2)．また，$onearc$ に対する曖昧性が存在しない場合にも，評価の対象をしない (step4)．$onearc$ に対して曖昧性が存在する場合には，step5 で DG 中にある位置 sp を始点とするアークの数を $onearc$ に対するスコアとして，$MaxArcScore$ に

```
position(a)  : アークaの依存ノードの開始位置
arc_num(a,S) : アークのコレクションSに存在するアークaの数
arc_num_at_position(p,S) : アークのコレクションSに存在する開始
                位置がpのアークの数

/* step0: 初期化  */
MaxArcScore :=0;  /* 最大可能得点数(アーク数) */
ArcScore = 0;     /* 得点数 (獲得アーク数) */

/* step1: 正解依存木CDTの各アークを取り出し処理する  */
foreach onearc in CDT {
{
 /* step2: 正解が依存グラフDGに存在しない→評価対象としない */
 if( onearc ∈ DG )  { next; }

 /* step3: sp(取出したアークの表層位置)ならびに              */
 /*        Num_in_DG(位置spにおける可能なアーク数)の算出  */
 sp := position(onearc);
 ArcNum_in_DG := arc_num_at_position(sp,DG);

 /* step4: 選択対象が1つ → 評価対象としない */
 if( ArcNum_in_DG == 1 ) { next; }

 /* step5: 対象となるアークの最大得点数(候補アーク数) */
 MaxArcScore := MaxArcScore + ArcNum_in_DG;

 /* step6: 表層位置spに対するアーク候補数を最大得点として、*/
 /*        それに対する正解得点数を計算し積算する          */
 CorrectArcRatio :=
   arc_num(onearc,ODTArcs)/arc_num_at_position(sp,ODTArcs);
 OneArcScore := ArcNum_in_DG * CorrecArcRatio;
 ArcScore := ArcScore + OneArcScore;
}

/* step7: 曖昧性解消能力指標「アーク曖昧性解消率」を計算 */
ArcSelectionAbilityRatio := ArcScore/MaxArcScore;
```

図 4: アーク曖昧性解消率計算アルゴリズム

[*4] 依存木のスコアは省略している．また，ODT_1,ODT_2 は図 2 のスコア付き依存森から導き出される解ではない．

[*5] ここでの「係り受け」は左の語から右の語へ係るという意味は含まず，「依存支配」の依存関係を表している．

依存グラフ:

```
DG = { arc(nc, [time]-n-0, [flies]-n-1),      /* pa1 */
       arc(sub, [time]-n-0, [flies]-v-1),     /* pa2 */
       arc(root, [time]-v-0, []),             /* pa3 */
       arc(obj, [flies]-n-1, [time]-v-0),     /* pa4 */
       arc(sub, [flies]-n-1, [like]-v-2),     /* pa5 */
       arc(root, [flies]-v-1, []),            /* pa6 */
       arc(root, [like]-v-2, []),             /* pa7 */
       arc(npp, [like]-pre-2, [flies]-n-1),   /* pa8 */
       arc(vpp, [like]-pre-2, [flies]-v-1),   /* pa9 */
       arc(vpp, [like]-pre-2, [time]-v-0),    /* pa10 */
       arc(det, [an]-det-3, [arrow]-n-4),     /* pa11 */
       arc(obj, [arrow]-n-4, [like]-v-2),     /* pa12 */
       arc(pre, [arrow]-n-4, [like]-pre-2) }  /* pa13 */
```

図 5: 例文に対する依存グラフ（アーク集合）

積算する．step6 では，*onearc* に対するアーク正解率 *CorrectArcRatio* を出力 *ODTArcs* 中にある表層位置 *sp* のアーク数に占める正解アーク (*onearc*) の割合として求め，最大得点と正解率の積として *onearc* に対するスコア *OneArcScore* を計算する．これを *ArcScore* に積算する．曖昧性解消率は，全ての正解アークに対して積算された *ArcScore* と *MaxArcScore* の比として step7 で求められる．曖昧性解消率は，0 以上 1 以下の値を取る．以下，簡単に例文に対する計算の例を示す．

今，例文に対する正解依存木 *CDT* と出力依存木 ODT_1,ODT_2 が図 3 であり，依存グラフ *DG* が図 5 であるとする．アルゴリズムの step1 では，正解依存木 *CDT* 中の 1 番目アーク ca1(図 3) が取り出され *onearc* に設定され，step2 で *sp* に依存ノード"[time]-n-0" の開始位置 0 がセットされる．*onearc* が *DG* に存在するため step3 の条件では排除されない．step4 では，$arc\_num\_at\_position(0, DG) = 3$ のため評価対象となり step5 に進む．step5 で *MaxArcScore* は 3 にセットされる．次に step6 で，$CorrectArcRatio = 1/2 = 0.5$, $OneArcScore = 3 * 0.5 = 1.5$ となり，*OneArcScore* は 1.5 となる．以下同様に各アークに対して計算が行なわれる．正解アーク ca11 は，*DG* に解釈が 1 つしかないため，step3 において評価対象外と判定され，計算結果は次のようになる．

Arc	sp	Arcnum	CrctArcRto	ArcScr
ca1	0	3	0.5	1.5
ca2	1	3	0.5	1.5
ca3	2	4	0.5	2.0
ca4	4	2	0.5	1.0
total		12		6.0

Arcnum, CrctArcRto, ArcScr は，それぞれ *Arcnum_in_DG*, *CorrectArcRatio*, *OneArcScore* である．*MaxArcScore* は，$3+3+4+2 = 12$ であり，*ArcScore* は，$1.5+1.5+2+1 = 6$ であるため，$ArcSelectionAbilityRatio = 6/12 = 0.5$ となる．この例の場合，対象となるアークの *CorrectArcRatio* が全て 0.5 であるため，全体として 0.5 になるが，例えば，3 番目のアークの *CorrectArcRatio* が 1 の場合は，$ArcSelectionAbilityRatio = 8/12 = 0.67$ となるのに対して，4 番目のアークの *CorrectArcRatio* が 1 の場合は，$ArcSelectionAbilityRatio = 7/12 = 0.58$ となり，曖昧性解消率は，アークの曖昧性解消の困難度 (候補アークの数) に応じた評価スコアになっている．

4 評価実験

4.1 実験環境

提案した評価方式の評価を行なうため，実験対象コーパス，正解データ，PDG の文法ならびに選好知識を用意した．正解データは，文に対する正解依存木である．選好知識としてはコーパスにおける語品詞頻度に基づく選好スコアを採用している．実験対象の素材は，ソフトウェアマニュアルなどからなる英語の技術文書約 62 万文 (463 万語[*6]) である．正解データならびに語品詞情報を大規模に用意するために，PDG とは異なる既存の文解析システム (これをオラクルシステムと呼ぶ) の出力結果を正解データや選好知識源として用いることとした．以下，正解データならびに語品詞の頻度情報の準備について説明する．

元のコーパスには，文抽出の誤りに起因する非文や技術文書に特徴的な表やインデックスなどの文要素以外の表現が多数含まれている．また，オラクルシステムで構文解析に失敗する文は，正解の生成ができない．このため，次のような文は，実験対象 62 万文から除外した．

(a) オラクルシステムで構文解析失敗の文 (7.1 万文)
(b) (a) 以外でピリオドで終わらない文 (20.4 万文)
(c) (b) 以外で大文字で始まらない文 (22 万文)

この結果，実験対象データとして 125320 文 (1844758 語,14.7 語/文) が得られた．これらの文に対しては，オラクルシステムにより，正解語品詞，正解依存木が付与されている．

15 文に 1 文の割合でオープンデータを抽出し，実験対象データをオープンデータ (8605 文,126684 語,14.7 語/文) とクローズデータ (116715 文,1718074 語,14.7 語/文) に分離した．オープンデータは評価実験に利用し，クローズデータは，語品詞頻度の算出に用い，その結果を選好知識のベースとして用いた．抽出語品詞数は，186.9 万件 (異なり数 44470) となった．語品詞は形態素解析を行なった結果の語数であるため，wc でカウントしている単語数とは一般に一致しない．オープンデータの語数分布を図 6 に示す．

文法に関しては，基本的な構文を解析する小規模な基本文文法 (文法 B) と，基本文文法を構築する途中段階のミニ文法 (文法 M) の 2 種類の PDG の解析文法を用意した．これらの文法は，オラクルシステムと同じ体系の依存構造を出力する．また，形態素解析

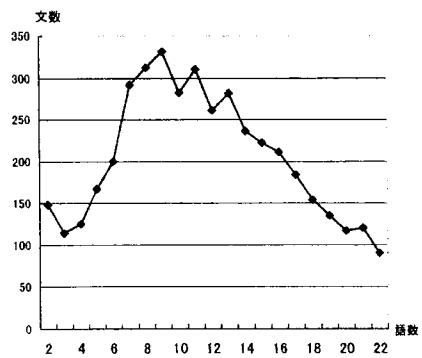

図 6: 対象文の語数分布

[*6] unix の wc コマンドで算出した値

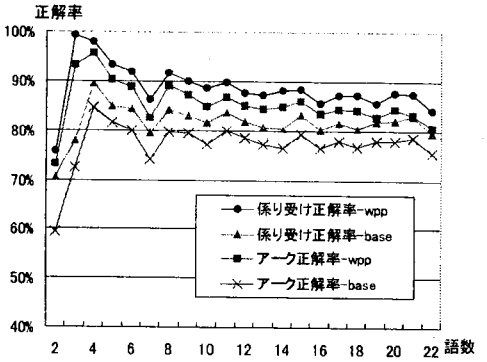

図 7: 単語係り受け正解率とアーク正解率

は，オラクルシステムと実験システムとで同一のデータを使用している．

4.2 評価実験

Prolog 上で実装した PDA のインタプリタを用い，上記オープンデータ 8605 文に対して，基本文文法を用いて評価実験を行なった．処理系のリソースの制約から単語数 23 以上の文は対象外とし，対象文数は 6882 文となった．このうち構文解析に成功した文は 4334 文であり，解析成功率は 63%である．解析失敗文については，通常部分解析を合成するなどの処理が行なわれるが，本稿では解析成功文を分析の対象としている．以下，各種条件の変化を交えながら評価指標の比較を行なう．

4.2.1 アーク正解率と単語係り受け正解率

図 7 にアーク正解率と単語係り受け正解率の入力文長さに対する値を，選好知識あり／なしの両方について示す．図の wpp は，語品詞頻度に基づくの選好知識の利用を，base は，選好知識の利用がないベースラインの値を示している．全体に対する平均正解率は，アーク正解率が 77.8%(選好知識なし)，85.1%(選好知識あり) であり，単語係り受け正解率が，81.8%(選好知識なし)，87.9%(選好知識あり) であった．品詞頻度という単純な選好知識であるが，知識の導入により全体にかなりの性能改善が得られている．図より，両正解率は単語長に対して類似した変化を示しており，正解率の指標として同等のものと言える．2〜3 単語長の文の正解率が低いのは，構文バリエーションが少なく，当たり外れが大きいことに起因すると考えられる．知識の有無による各正解率の差は，アーク正解率が 7.3%，単語係り受け正解率が 6.1%と，アーク正解率での改善度合いがやや高い．また，知識の有無による両正解率間の平均差は，選好知識なしが 4.0%，選好知識ありが 2.9%と多少の開きがある．これらは，通常，低精度での改善効果は大きいことを考慮すれば妥当な結果と考えられる．

今回の実験対象である英語は単語係り受け関係と構文的関係との対応度が，日本語などに比べて高い可能性がある．日本語でのアーク正解率と単語係り受け正解率の差との比較は今後の課題の 1 つと考えられる．

4.2.2 アーク正解率と曖昧性解消率の比較 (知識の有無)

図 8 にアーク正解率と曖昧性解消率の入力文長さに対する値を，選好知識あり／なしの両方について示す．両指標の数値的な比較は意味がないが，全体に対する平均値は，選好知識ありアーク正解率 (AK) が 85.1%，選好知識なしアーク正解率 (AB) が 77.8%，選好知識ありアーク曖昧性解消率 (DK) が 65.8%，選好知識なし曖昧性解消率 (DB) が 42.0%であった．

DB は，文の長さによらずほぼ一定であることから，曖昧性解消の難しさ (多義の多さ) は，文長に対してそれ程変化していないと言える．これに対し，DK は，全体に DN より高精度であるが，長さにより減少する傾向にある．これより，文が長くなると採用した選好知識の曖昧性解消能力がやや下がることが分かる．アーク正解率は，構文解析能力と曖昧性解消能力の両方を含んだ指標であり，DK の低下は，長文における AK と AB の差の縮小に反映されている．ただし，アーク曖昧性解消率とアーク正解率の増減は，必ずしも連動するわけではない．例えば，単語数 2 から 3 では，DK の低下に対して AK は向上し，単語数 4〜6 では，DK の増加に対して AK は減少している．単語数 4〜6 では，1 文あたりの単語数の増加によって構文解析誤りが増えていると考えられる．

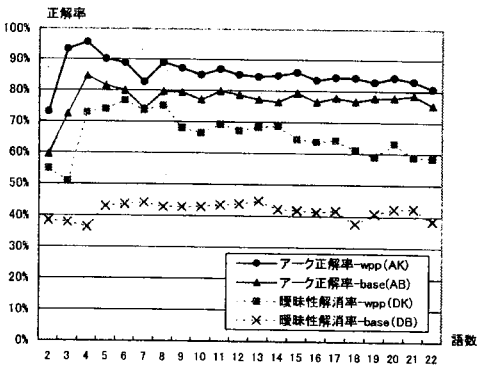

図 8: 単語係り受け正解率とアーク正解率

4.2.3 アーク正解率と曖昧性解消率の比較 (正解可能文生成率)

実験対象 6882 文のうち 3224 文 (74.4%) が正解可能文であった．正解可能文生成率の影響を見るために，正解可能文のみの文集合 (正解可能文生成率が 100%) に関してデータを集計した．図 9 にアーク正解率と曖昧性解消率の入力文長さに対する値を，正解可能文対象 (C:Correct Answer Contained) ／全文対象 (A:All Sentences) の両方について示す．平均値は，正解可能文対象のアーク正解率 (AC) が 90.4%，全文対象のアーク正解率 (AA) が 85.1%，正解可能文対象の曖昧性解消率 (DC) が 42.2%，全文対象の曖昧性解消率 (DA) が 42.0%であった．正解可能文生成率の向上により，アーク正解率ならびに曖昧性解消率の大幅な改善が見られた．

DC,DA の比較より，曖昧性解消率は，正解可能文生成率の差に関わらず，ほぼ同じ値となり，文が長くなるにつれて減少してゆく．このことから，アーク曖昧性解消率は，正解生成能力とは独立した性能指標になっているといえる．一方，アーク正解率に関しては，AC(正解可能文生成率 100%) の値は文長 7 以上の文では，ほぼ一定になっているのに対し，AA(正解可能文生成率 74.4%) では文の長さに対して減少傾向が見られる．これは，正解可能文生成率は，文長に対するアーク正解率の低下と関連があり，アーク正解率の低下を押さえるには正解生成能力を向上することが有用であるとの示唆を与える．

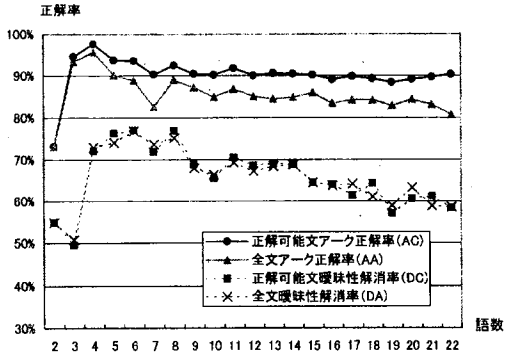

図 9: アーク正解率と曖昧性解消率比較 (正解可能文生成率 74.4%/100%)

4.2.4 アーク正解率と曖昧性解消率の比較 (文法カバレッジ)

基本文文法 (文法 B) とミニ文法 (文法 M) を用いて比較実験を行なった. 全対象文 6882 文に対する構文解析成功文 (率) は, 文法 B が 4334 文 (63.0%), 文法 M が 3139 文 (45.6%) であった. 各々の解析成功文のうち, 正解可能文 (率) は, 文法 B が 3224 文 (74.4%), 文法 M が 2135 文 (68.0%) であった. 図 10 にアーク正解率と曖昧性解消率の入力文長さに対する値を文法 B／文法 M の両方について示す. 平均値は, 文法 B のアーク正解率 (AB) が 85.1%, 文法 M のアーク正解率 (AM) が 83.4%, 文法 B のアーク曖昧性解消率 (DB) が 65.8%, 文法 M の曖昧性解消率 (DM) が 68.9%であった.

DB,DM は, 単語長 7〜16 の文頻度の高い範囲において, 多少の増減はあるがほぼ同じ値を取っている. これに対して, AB,AM は, 同じ範囲において常に AB が AM を上回る値となっている. これから, 曖昧性解消率は文法の違いによる差異が少ない指標となっていると考えられる. 曖昧性解消率において, 文法による差が無いので, 選好知識によるスコアリング方法が曖昧性解消率の低下の主な原因であることが推察される.

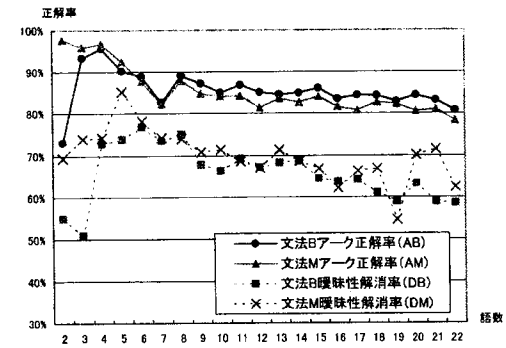

図 10: アーク正解率と曖昧性解消率比較 (基本文文法/ミニ文法)

5 おわりに

本稿では, PDG の枠組みに即した文解析性能について検討し, 最終出力の解析精度を測るアーク正解率と単語係り受け正解率, 選好能力を測るアーク曖昧性解消率, 正解生成能力を測る正解可能文生成率の 4 つの指標を提案し, 英語技術文書に対して実験を実施した. アーク正解率は, 選好能力と正解生成能力の両方を含んだ総合指標であり, 今回の実験環境では, 単語係り受け正解率とほぼ同様の振る舞いを示した. また, 曖昧性解消率は, 今回の実験によれば, 正解可能文生成率, 解析文法に対してほぼ独立であると考えられ, 選好能力の良し悪しを測定する指標として良好な性質をもっている. また, 曖昧性解消率の測定により, 実験で用いた選好知識とその適用方法が文長に対する能力劣化を持つという 1 つの課題が浮かび上がってきた.

今後, 文解析システムの解析精度を向上するためには, 解析文法を拡張して構文解析成功率を向上することは基本であるが, 長文に対する文正解率の低下を押さえるために正解可能文生成率の向上と, 長文に対する選好能力の劣化の抑制がポイントとして考えられる.

参考文献

[1] Carroll, J., Briscoe, E. and Sanfilippo, A.: *Parser evaluation: a survey and a new proposal* , In Proceedings of the 1st International Conference on Language Resources and Evaluation, (1998).

[2] Grishman, R., Macleod, C. and Sterling, J.: *Evaluating parsing strategies using standardized parse files*, In Proceedings of the 3rd conference on Applied Natural Language Processing, (1992).

[3] Sampson, G.: *Proposal for improving the measurement of parse accuracy* , International Journal of Corpus Linguistics, Vol.5, pp. 53-68, (2000).

[4] Lin, D.: *A dependency-based method for evaluating broad-coverage parsers*, Natural Language Engineering archive, Vol.4, Issue 2, pp. 97-114, (1998).

[5] Srinivas, B.: *A lightweight dependency analyzer for partial parsing*, Natural Language Engineering Vol.6, Issue 2, pp. 113-138, (2000).

[6] Briscoe, E., Carroll, J., Graham, J. and Copestake, A.: *Relational evaluation schemes*, In Proceedings of the Beyond PARSEVAL Workshop at the 3rd International Conference on Language Resources and Evaluation, Las Palmas, Gran Canaria., pp 4-8(2002).

[7] M. Tomita, "Generalized LR Parsing", Kluwer Academic Publishers, Boston, MA, (1991).

[8] 平川秀樹, "統語森に対応する圧縮共有型依存構造「依存森」について", 情報処理学会自然言語処理研究会,NL-167(予定), (2005).

[9] 平川秀樹, "最適解探索に基づく日本語意味係り受け解析", 情報処理学会論文誌,Vol.43,No.03, (2002).