実用的なHPSG文法のための二つの手法: 型のCombiningと選言的素性構造のPacking

宮尾 祐介　鳥澤 健太郎　建石 由佳　辻井 潤一
東京大学理学部情報科学科
{yusuke, torisawa, yucca, tsujii}@is.s.u-tokyo.ac.jp

1 はじめに

HPSG[5]は高い記述力と柔軟性を持った文法記述の枠組であり，自然言語における構文と意味とを統一的に記述することができると期待されている．そこで我々は，HPSGに基づく大規模な英文解析システムXHPSG[11]の開発を現在進めている．しかし，現時点でのXHPSGシステムは解析速度が遅くあまり実用的でない．

本稿では，XHPSGシステムの解析速度低下の要因として考えられる2つの問題点，(1)基本データ構造が大きいことと(2)大量の選言的なデータを扱うことに対し，それぞれに対処する2つの高速化手法(1)型のCombiningと(2)選言的素性構造のPackingを提案する．我々はそれぞれの手法をXHPSGシステムに実装し，解析時間の比較実験を行なった．これらの高速化手法はXHPSGシステムの解析速度を上昇させるために開発されたものであるが，HPSGに基づく同じ問題点を持った他の解析システムにも適用可能である．

2 背景

2.1 HPSG

我々の開発している文法が理論的基礎としているHPSGは，素性構造[1]およびその上に定義される単一化という操作を用いて文法を記述する枠組である．終端記号(語彙項目)，非終端記号，生成規則(スキーマ)はすべて素性構造を用いて記述される．素性構造は素性と型からなる再帰的なデータ構造で，ノードとエッジにラベルが付与された根付き有向グラフで表現される．ノードに付与されたラベルが型に当たり，エッジに付与されたラベルが素性に当たる．

型の間の関係は，型階層で表現される．例えば図1は，簡単な型階層を示している．この型階層では，四角形が最も一般的な型として定義され，それより特殊な型としてひしがたと長方形が定義され，更にそれらより特殊な型として正方形が定義されることを表現している．

2.2 XHPSG

XHPSGとは，我々が現在開発中の，HPSGに基づく大規模な英文解析システムである．XHPSGシステムは英文法とパーザから構成されている．文法部分は，すでに公開されているXTAG[3]システムの文法から変換することにより開発している[1]．パーザ部分は，現時点では単純なCKYベースのパーザである．

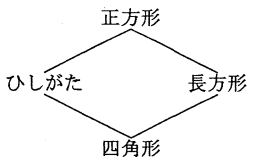

図 1: 様々な四角形の間の関係を表現している型階層.

verb	
VMODE	*vmode_ppart*
MAINV	*plus*
INV	*boolean*
EXTRACT	*boolean*
TRANS	*boolean*
CASE	*case*
HEADPHON	⟨"credit"⟩
MODL	*non_sign*
MODR	*non_sign*
PASS	*plus*
PERF	*boolean*
PRD	*boolean*
PROG	*boolean*
TENSE	*tense*
ASSIGN_CASE	*nom*

図 2: 動詞 "credited" に対して XHPSG 文法が割り当てる語彙項目の HEAD 素性の値.

XHPSGの文法は非常に大規模であり，そのために以下の2つの問題点を持っている．

1. 語の情報を非常に詳細に記述しているため，その情報を表現している素性構造のノード数が多い．

   XTAGで使用されている25個の素性がすべてXHPSGに取り入れられているため素性の数が非常に多くなり，その結果ノード数が多くなっている．例えば図2は，動詞 "credited" に対する語彙項目のHEAD素性の値を示している．このように，HEAD素性だけでも多くの素性が存在する．この例の語彙項目の全体のノード数は101であるが，150以上になる場合もある．

2. 語の多様な使われ方に対応しているため，1つの語に対応する語彙項目の数が多い．

[1] XTAGは語彙約317,000を有し，Wall Street Journalの語数10以下の文の約56%について正解を含む構文木セットを生成できる(1996年現在)[6]が，解析速度が遅く，あまり実用的ではない．

XHPSG では XTAG の各語彙項目の木に対して 1 対 1 に素性構造が対応する．XTAG は様々な語の使われ方に対し別々の木を語彙項目として割り当てるため，結果として XHPSG では大量の素性構造が語彙項目として 1 つの語に割り当てられる．例えば，名詞では 12 個，一般の動詞では 20 ～ 30 個，語彙項目が多い語では 100 以上の語彙項目が割り当てられる．

以上の問題点が，XHPSG システムの解析速度低下の要因となっている．

2.3 LiLFeS

XHPSG システムの文法およびパーザは，我々の研究室で開発中の素性構造記述言語 LiLFeS[7, 8] で記述され，LiLFeS システム上に実装されている．LiLFeS システムは，型と素性構造の操作 (単一化) について次のような 2 つの特徴を持っている．

- 型の単一化は，2 次元のテーブルを用いることにより，一定時間で計算される．
- 素性構造の単一化は，LiLFeS システムに組み込まれており高速に計算されるが，素性構造のノード数に比例した時間で計算される．

これらの特徴は，本稿で提案する 2 つの手法の有効性に大きく関係している．

3 2 つの手法の説明

本節では，2.2 節で述べた 2 つの問題点に対して，それぞれを解決する 2 つの手法を提案し，それらのアルゴリズムを例を用いて説明する．第 1 の問題点には型の Combining という手法を，第 2 の問題点には選言的素性構造の Packing という手法を提案する．

3.1 型の Combining

型の Combining は，素性構造の一部を型で表現することにより，素性構造のノード数を削減する手法である．2.2 節で XHPSG 文法の第 1 の問題点として述べたように，XHPSG 文法で扱う素性構造は非常にノード数が多い．2.3 節で述べたように LiLFeS システムでは素性構造の単一化の計算時間はノード数に比例するため，この問題点は解析速度低下の要因になっていると考えられる．従って，型の Combining を適用してノード数を削減することにより，XHPSG システムの解析時間を短縮できると期待される．

$$\left[\begin{array}{ll} \text{身長} & \text{高い} \\ \text{体重} & \text{軽い} \end{array}\right] \quad \left[\ \text{身長体重} \quad \text{高い軽い}\ \right]$$

図 3: 身長と体重の情報を表現している素性構造 (左) と，それに型の Combining を適用して得られる素性構造 (右)．

ここでは簡単のため，2 つの素性を持つ素性構造を，1 つの素性を持つ素性構造に変換することを考える．この手法を繰り返し適用することにより，3 つ以上の素性を持つ素性構造も 1 つの素性を持つ素性構造に変換できる．例えば，図 3 の左にある素性構造を，その右にある素性構造に変換するとする．この時，左の素性構造では図 4 の様な 2 つの型階層の中の型で表現されていた情報を，1 つの型で表現しなくてはならない．従って，元の 2 つの型の置かれている型階層を合併するという操作が必要になる．その結果が，図 5 の型階層である．この型階層が図 3 右の素性構造の素性に割り当てられていれば，左の素性構造と等価な情報を表現していることになる．この結果，図 3 の例では素性構造のノードの数を 3 分の 2 に削減することができる．型階層は複雑になっているが，型の単一化は LiLFeS システムでは一定時間で行なわれる (2.3 節参照) ため，素性構造の単一化の計算時間は 3 分の 2 になる．

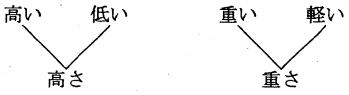

図 4: 図 3 左の素性構造の各素性に割り当てられた型階層．

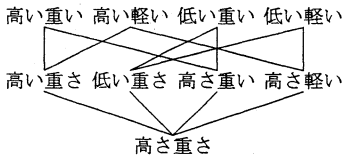

図 5: 図 4 の 2 つの型階層を合併した型階層．

この手法には，(1)Combining を適用する素性は他の素性と構造共有してはならない，そして (2) 空間計算量が指数関数的に増加する，という問題がある．しかし，図 2 に示される様な XTAG から XHPSG に取り込まれた素性は，他の素性と構造共有することが無く，またほとんどの素性が少数の型から構成される型階層が割り当てられているので，この手法が適用可能である．

3.2 選言的素性構造の Packing

選言的素性構造の Packing は，同じ語に対する語彙項目の等価な部分を 1 つにまとめることにより，語彙項目の数を削減する手法である．まとめられた部分の単一化は 1 回しか行なわれないため，この手法により冗長な単一化を避けることができる．2.2 節で第 2 の特徴として述べたように，XHPSG の文法は同じ語に対して大量の語彙項目を割り当てる．従って，この手法で語彙項目数が削減できれば，解析時間の短縮が期待される．

例えば，図 6 に示される 2 つの語彙項目が同じ語に割り当てられたとする．この 2 つの語彙項目は 3 つの素性を除いて等価であるから，それぞれにスキーマを適用す

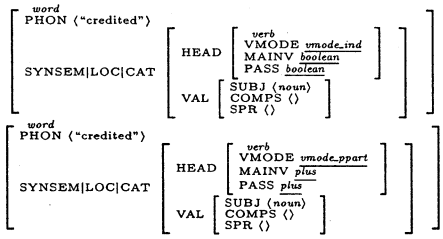

図 6: 同じ語に対する 2 つの語彙項目の一部分．下線部分が両者の相違点である．

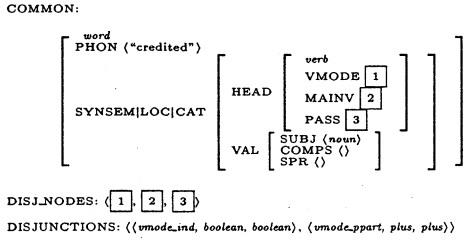

図 7: 図 6 の 2 つの語彙項目に Packing を適用して得られる語彙項目．

ると等価な単一化が繰り返されてしまう．従って，2 つの語彙項目を図 7 のように 1 つにまとめて，共通部分の処理を 1 回だけしか行なわないようにする．この図で，COMMON は共通部分，DISJ_NODES は異なっていたノードを示している．DISJUNCTIONS は，元々の素性構造の DISJ_NODES に対応する値のリストである．従って，元の素性構造を復元するには DISJUNCTIONS のリストの中の 1 つの要素を DISJ_NODES と単一化すれば良い．図 7 に示される構造を (COMMON, DISJ_NODES, DISJUNCTIONS をまとめて) *Packed Structure* と呼ぶ．Packed Structure は，通常の素性構造に選言的構造を導入したものに相当する．図 7 の例では，VMODE, MAINV, PASS 素性の値に選言的構造を導入したことになる．

本稿で提案する選言的素性構造の Packing では，(1) 選言的構造を導入する素性は予め与えておくこととし，また，(2) 選言的構造を導入する素性としない素性とが構造共有しないという条件を仮定した．これらの条件により，等価な部分が存在しても Packing が行なわれない場合があるという問題がある．しかしながら，第 1 の条件より，素性構造を共通部分とそうでない部分とに解析前に分離しておくことができ，さらに，その分離された部分は通常の素性構造であるため，LiLFeS システムに組み込まれた高速な単一化プログラムがそのまま利用可能である．また，選言的構造を導入することによる効果が大きいと期待される素性を，文法作成者が指定することができるという利点もある．さらに第 2 の条件より，解析前に分離された部分間で情報が伝播することがないので，共通部分とそうでない部分とを別々に処理することができる．

	平均解析時間 (秒)	
	入力 A	入力 B
旧システム	2.83	18.20
新システム	1.33	5.97
解析速度向上	2.13 倍	3.05 倍

表 1: 型の Combining と選言的素性構造の Packing の適用前 (旧システム) と適用後 (新システム) の解析時間．

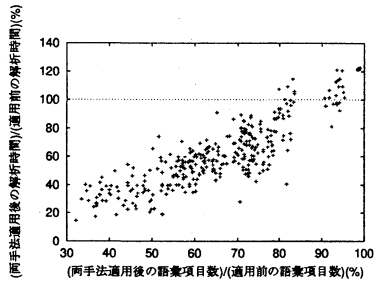

図 8: 2 つの手法の適用前と後での語彙項目数の比率と解析時間の比率の関係．

4 実験

型の Combining と選言的素性構造の Packing という 2 つの手法を XHPSG システムに実装し，これらの手法を使わない XHPSG システムとの解析時間比較実験を行なった．

現時点での両手法の実装には，次のような制限がある．

型の Combining Atom(素性を持たない型) の型階層しか合併できない．

選言的素性構造の Packing 再帰的な Packing を行なわない．

今回は，これらの制限の範囲内でのみ両手法を適用し，実験を行なった．これらの制限を取り除くのは，今後の研究課題として残されている．

実験は，以下のような環境で行なった．

実験機器 AlphaServer4100
(400MHz CPU, 4GB Memory)

入力 A: 337 文，平均 8.37 語 / 文
B: 16 文, 平均 11.88 語 / 文

出力 解析に要した実時間の平均

入力 B は，入力 A の中から解析に時間がかかるもの上位 16 文を取り出したものである．

実験結果は表 1 が示している．1 文の解析速度は 2.13 倍に上昇した．また，解析に時間がかかるものについては更に効果が大きく，解析速度は 3.05 倍となった．

各手法の効果についてさらに詳しい解析を行なった．まず，型の Combining では，入力 A 中の語に対する語

彙項目のノード数を9.64%削減することができた．型のCombiningを適用して得られた素性構造の処理は通常の素性構造と同じ，つまりオーバーヘッドがないため，そのまま解析時間の短縮に貢献したと考えられる．一方，選言的素性構造のPackingでは，入力A中の語に対する語彙項目の数を35.3%削減することができた．図8は，語彙項目数と解析時間の，両手法を適用する前と後との比率をグラフにしたものである．このグラフから，語彙項目数を30%にまで削減できれば，解析時間は5分の1にまで短縮できることがわかる．しかし，Packingの効果があまりない文に対しては，旧システムより解析時間が増加してしまっている．これは，Packed Structureの単一化をLiLFeS上のプログラムで処理しているため，述語呼び出しなどのオーバーヘッドが現れているものと考えられる．この問題に対しては，オーバーヘッドを減少させることと，より一般的なPackingを実装することにより語彙項目数を更に削減することで，更に解析時間を短縮できると考えられる．

5 関連研究

型のCombiningは，型階層で複雑な情報を記述することにより，素性構造のノード数を削減する手法である．型階層を用いて様々な情報を記述する手法は，Carpenter[1]によっていくつか紹介されている．3.1節で触れた型階層を合併する手法は，文献[1]で紹介されている，Conjunctionを型階層で記述する手法の1つの応用ととらえることができる．

選言的素性構造のPackingは，連言的に情報を記述する素性構造へ選言的構造を導入することに対応する．素性構造に選言的構造を導入する手法は現在までに数多く提案されており，最近ではDörre&Eisele[2]やNakano[4]の研究がある．これらはいずれも，素性構造の一部として一般的に選言的構造を導入する手法である．このことは，選言的構造を導入した素性構造をLiLFeS言語およびLiLFeSシステム中の素性構造処理プログラムに組み込むことに相当する．本稿で提案する手法は，LiLFeSシステム上のプログラムで選言的素性構造を処理するもので，既存の安定したLiLFeSシステムを変更することなく，また簡単に実装可能であるという特徴を持つ．

6 おわりに

大規模な英文解析システムXHPSGに，型のCombiningと選言的素性構造のPackingという2つの手法を実装した．その結果，解析速度を2.13倍上昇させることに成功した．

現時点では，両手法は制限された形でXHPSGシステムに実装されている．今後は，両手法の現時点での実装の制限を取り除き，さらに解析時間を短縮させる．具体的には，次の3点を考えている．

- 素性を持つ型を含む型階層の合併
- 再帰的なPacked Structureの導入
- 選言的構造を導入した素性構造の，LiLFeSシステムへの組み込み．

さらにその先の研究課題として，他の高速化手法と本稿で提案した手法とを統合し，さらなる高速化をはかる．具体的には，(1)コンパイラ版LiLFeSへの移行[10]や(2)HPSGのCFGへのコンパイル手法[9]を現在研究している．この2つの手法を適用することで，それぞれ3〜5倍の解析速度向上が見込まれている．これら全ての高速化手法を統合し，XHPSGシステムを，十分実用的な英文解析システムとして完成させたいと考えている．

参考文献

[1] Bob Carpenter. *The Logic of Typed Feature Structures.* Cambridge University Press, 1992.

[2] Jochen Dörre and Andreas Eisele. Feature logic with disjunctive unification. In *Proc. 10th COLING*, volume 2, pages 100–105, 1990.

[3] The XTAG Research Group. A lexicalized tree adjoining grammar for English. Technical report, IRCS Report 95-03, University of Pennsylvania, March 1995.

[4] Mikio Nakano. Constraint projection: An efficient treatment of disjunctive feature descriptions. In *Proc. 29th ACL*, pages 307–314, 1991.

[5] C. Pollard and I. A. Sag. *Head-Driven Phrase Structure Grammar.* University of Chicago Press, 1994.

[6] B. Srinivas, Christine Doran, Beth Ann Hockey, and Aravind Joshi. An approach to robust partial parsing and evaluation metrics. In *Proc. Workshop on Robust Parsing at European Summer School in Logic.* Language and Information, August 1996.

[7] Makino Takaki. *LiLFeS User's Guide*, 1997. Available in http://www.is.s.u-tokyo.ac.jp/~mak/lilfes/.

[8] Makino Takaki, Torisawa Kentaro, and Tsujii Jun'ichi. LiLFeS - practical programming language for typed feature structures. In *NLPRS '97*, 1997.

[9] Kentaro Torisawa and Jun'ichi Tsujii. Computing phrasal-signs in HPSG prior to parsing. In *Proc. 16th COLING*, pages 949–955, 1996.

[10] 吉田稔, 牧野貴樹, 鳥澤健太郎, and 辻井潤一. 素性構造処理言語LiLFeSの最適化技術. In 言語処理学会第4回年次大会発表論文集, March 1998.

[11] 建石由佳, 鳥澤健太郎, 牧野貴樹, 西田健二, 淵上正睦, and 辻井潤一. LTAG文法からの変換によるHPSG英文法の作成. In 情報処理学会研究報告 *NL-122*, pages 119–126, November 1997.