論文内容の要旨

論文題目　Parallel Parsing and Robust Processing within Unification Based Grammar Formalism
(単一化に基づく文法枠組における並列構文解析と頑健処理)

氏名　　二宮　崇

本論文では、単一化文法の一つである主辞駆動句構造文法(Head-driven Phrase Structure Grammar)を用いる高効率かつ頑健な自然言語処理システムを実現するための効率化に関する手法および頑健化に関する手法について論じる。主辞駆動句構造文法は計算言語学の領域で最も広く使われている文法枠組の一つであり、その表現形式が数学的によく定義されていること、および詳細な言語学的記述を与えることが可能なことから柔軟かつ広範な自然言語処理を実現すると期待されている。しかしながら、その反面、解析時間における効率性の問題および被覆率における文法の脆弱性の問題が指摘されている。主辞駆動句構造文法の構文解析における文法規則の適用部は素性構造の単一化操作により実現されており、この単一化操作は高コストを要する計算である。また、脆弱性に関して、主辞駆動句構造文法には意味表現など詳細なレベルの記述と制約を与えることが可能であるが、そのような詳細な制約を与えつつ同時に多様な自然言語を被覆する文法を与えるのは容易ではなく、制約違反となり文法の被覆率が低下する。

本研究では共有メモリ型並列計算機上で稼動する並列構文解析器を実現することにより効率面の問題を解決し、文法の被覆率に対する問題は訓練コーパスを用いて新たに文法規則を獲得することにより解決する。実現した主辞駆動句構造文法のための並列構文解析器は非常に高効率であり、実世界に流通するテキストであっても十分実用に耐えうる速度で解析することが可能である。文法規則の獲得は、訓練コーパスに対して頑健な構文解析を行い、得られた構文解析結果から新たに文法規則を獲得することにより達成される。得られた文法規則には頑健処理が適用された結果が反映されているため、それらを用いた現実の応用先における実行時の構文解析でもより高い被覆率を達成することが可能になる。本論文では並列構文解析と頑健な構文解析による文法獲得について独立に議論するが、獲得された文法規則は並列構文解析器においても利用可能であり、その場合には高効率かつ頑健な構文解析の実現が可能になる。

最初に並列構文解析に関して述べる。本研究では、まず、素性構造を並列計算機上で高効率に処理するためのプログラミング環境を提供し、その上で高効率な並列構文解析器を実現する。

プログラミング環境はエージェントモデルに基づいており、素性構造を処理するための特殊なエージェントを提供することにより、素性構造処理のカプセル化、および高効率化

が同時に実現される。素性構造処理の効率化は、i) 共有メモリを通して素性構造を転送する、ii) オンデマンドで素性構造をコピーする、および、iii) 複数のエージェントが同時にメモリに対し読み書きできる機構を用意することにより実現される。すなわちエージェント間では、常に素性構造の実体が送受信されるわけではなく、通常は素性構造の ID のみが送受信されており、素性構造の実体に対する単一化操作が必要になったときに初めて上述の素性構造を処理するエージェント間で素性構造の実体が交換される。素性構造の実体の交換は共有メモリを通して行われるため、非常に低いオーヴァーヘッドで交換することが可能である。また、素性構造を処理する特殊なエージェントには各々書き込み可能な共有メモリ領域が割り当てられており、共有メモリ上に配置された素性構造の読み書きの際に他のエージェントと同期をとる必要はない。

以上の特徴を持つプログラミング環境上で開発された並列構文解析アルゴリズムは CKY アルゴリズムを基にしており、CKY 表と呼ばれるデータ構造の各要素に対しエージェントを割り当て、各エージェントは割り当てられた要素を計算する単純なタスクを処理することにより構文解析全体が行われる。CKY アルゴリズムはデータの分散性、並列化の粒度の点で非常に有利である。以上の並列化によって我々の構文解析器は非常に高速に稼動し、50 台用いた場合に、その解析時間は EDR 日本語コーパスを用いて一文あたり 78 ミリ秒であり、台数効果は 13.2 倍に達した。また、本論文では提案する並列構文解析アルゴリズムの計算量についても言及する。理論的な計算量は逐次計算機による解析も並列計算機による解析も共に指数オーダーであるが、経験的計算量においてはほぼ線形時間で解析されることを実験を通して確認した。

続いて、本研究では文法の被覆率を向上するための文法規則の獲得手法とそれを用いた高速な構文解析器について論じる。従来被覆率を向上する手法として様々な頑健な構文解析手法が提唱されているが、それらの中でも誤りパターンに応じた補足ルールの適用を行うルールベースの頑健処理手法が、意図した頑健処理が実現されるという点において健全かつ妥当な処理であり望ましい。しかしながらそれらのほとんどは「オンラインの頑健な構文解析」、すなわち、現実の応用先で実際に行われる構文解析実行時に補足ルールの適用を行う構文解析手法をとっており、解析時間の問題、および過剰に解析結果を出力するという問題があり、これらの手法は実用的ではない。本研究では、「オフラインの頑健な構文解析」を行い、その解析結果から被覆率を向上するための文法規則を新たに獲得するという手法をとる。「オフラインの頑健な構文解析」とは、「オンラインの頑健な構文解析」と同様、補足ルールを適用しつつ構文解析を行うことを指すが、現実の応用先で実際に解析を行う実行時ではなく、訓練コーパスを与え、訓練コーパスに対して構文解析を行う非実行時の静的な構文解析のことを指す。得られた文法規則には「オフラインの頑健な構文解析」で適用された補足ルールの (1) 適用条件、および (2) 適用結果が反映されているため、これらの文法規則を用いた構文解析では過剰な解析結果を出力することなく、補足ルールを適用した場合と同等の効果が得られる。また、「オフラインの頑健な構文解析」は次にあ

げる三つの利点をもつ。(a) 実行時解析ではないので多少解析時間がかかってもかまわない。(b) 括弧付きコーパスなどの正解コーパスを参照することにより誤った補足ルールの適用を抑制できる。すなわち、探索空間をより狭くすることを可能にし、正解コーパスと矛盾しないという意味において正しい解析結果を返すことを可能とする。(c)最終的には人手により解析結果を確認できる。「オフラインの頑健な構文解析」における補足ルールの適用方法は、(i) 元来与えられていた文法規則を汎化するか、(ii) 非主辞に相当する素性構造を汎化するかによって二つの方法に分かれる。

次に、「オフラインの頑健な構文解析」により得られた構文解析結果から新しく文法規則を獲得する手法について述べる。補足ルールの適用方法に応じて文法規則の獲得手法は異なり、(i)の方法による解析結果から文法規則を獲得する場合は、汎化された文法規則に適用条件を付加することにより新しい文法規則が得られる。この方法により得られた文法規則には、元の文法規則に記述されていた構造共有の情報も残されるため、構文木中の子ノードに記述された情報を親ノードへ伝播する能力が保たれる。(ii)の方法により得られた解析結果から学習する場合は、得られた構文木の断片そのものを文法規則とみなすことにより文法規則を獲得する。「オフラインの頑健な構文解析」により得られたある親ノードをあらわす素性構造をM、Mの子ノードをあらわす素性構造をL、Rとしたとき、得られる文法規則は$M \rightarrow L\,R$となる。一般に主辞駆動句構造文法における文法規則には親子間の構造共有の情報が記述されており、構文解析の過程で子ノードに含まれる情報が構造共有を通じて親ノードに伝播することにより親ノードが導出される。しかし、ここで得られる文法規則$M \rightarrow L\ R$には親子間の構造共有が存在せず、この文法規則は、ただ「LとRが出現したらMを親ノードとする」ということを意味する。このような文法規則だけを用いて一般的な自然言語現象を記述するのは困難であるが、これらの文法規則は例外的な親ノードの導出規則を容易に表現するため、元々用意されていた文法規則とあわせて用いることにより、元の文法規則だけでは捉えることが困難であった例外的自然言語現象の捕捉が可能になる。また、構文木中に出現する素性構造を字句どおりそのまま用いては、音韻素性などのそこに出現した単語および句に特有過ぎる情報まで含まれてしまうのでそのような素性の値は文法規則から削除する。以上二つの手法を用いて得られた文法規則を用い、テストコーパスに対する解の数および被覆率を各々実験を通して確認する。

主辞駆動句構造文法における文法規則は素性構造で与えられるが、新しく学習された文法規則は数千から数万のオーダーになるため、実行時構文解析においては、ある素性構造に対し単一化可能な文法規則を高速に検索する必要がある。本研究では、素性構造に存在するいくつかのパスの値をインデックスとみなすことにより高速に単一化可能な文法規則を求める手法も提案する。素性構造は数百のノードからなる根付有向グラフであるが、それらノードに付与された値の全てが検索のインデックスとして有効に働くわけではない。数千、数万の数からなる文法規則集合が与えられた際に、事前にどのパスの値がインデックスとして有効であるかを計算した表を生成しておき、実行時にはその表を参照すること

により求めるべき文法規則集合を高速に特定する。この手法の有効性は実験を通して確認する。