オブジェクト指向パーザ POWER における構文的曖昧さの漸進的解消機構

高橋博之　　柴山威　　宮崎正弘

新潟大学大学院自然科学研究科

1 はじめに

自然言語処理では解析文の持つ曖昧性をどのようにして解消するかが重要な問題である。曖昧性の解消を文末まで行なわない方式では解析の途中で可能性が組合せ的に爆発してしまうことがしばしばある。したがって曖昧性の解消は文の途中、必要な情報が揃った時点で漸進的に行なわれることが望ましい [1] [2]。

また、形態素、構文、意味の各レベルの解析を分離して行なう方式では、形態素解析では構文や意味の情報を、構文解析では意味の情報を曖昧性解消に利用できないため、しばしば初期の段階で可能性の組合せが爆発してしまう。したがって漸進的な曖昧性解消のためはこれらの処理を統合して行なうことが望ましい。

本稿ではオブジェクト指向パーザ POWER[3] において漸進的な曖昧性解消を行なうための手法について述べる。POWER はオブジェクト指向の考えに基づいたパーザである。解析の流れを手続きで柔軟に制御できるため、拡張性が高いのが特長である。柴山ら [4] は POWER と形態素解析との融合法を提案している。また、POWER は意味解析の中心となる概念=単語を中心に記述するため、意味解析との融和性が高い。

漸進的な曖昧性解消では全ての可能性の解析を並行して行ない、随時その結果を比較する。このような並列処理を行なうための拡張を POWER に対して行なった。また、効率を向上させるために途中結果の照合により処理の統合を行なう。さらに曖昧さを表現するデータ構造を導入することで、より柔軟な統合を可能とした。

2 オブジェクト指向パーザ POWER

オブジェクト指向パーザ POWER はオブジェクト指向の考え方を用いたパーザである。POWER では単語は自律的に動作するオブジェクトであるとして、各オブジェクトの相互メッセージ通信によって解析が進められる。

オブジェクトはメッセージを受け取るとそれをメソッドと呼ばれる手続きで処理し、必要に応じてさらにメッセージを送信する。多くのオブジェクト指向言語がそうであるように、メッセージの送信は制御の移動を意味する。メッセージは往復で一組となっていて、メッセージを受け取った単語がメッセージを処理して結果を返すと元のオブジェクトに制御が戻る。つまり、メッセージ送信は手続きの呼び出しと同じように働く。このように、POWER では解析の流れが手続きで柔軟に制御されるため、拡張性が高い。

オブジェクトはその語順に従って一列に並べられ、各オブジェクトは基本的に隣接の単語にのみメッセージを送ることができる。遠くの単語と通信するためには途中の単語に中継してもらう必要がある。このことにより遠くの単語より近くの単語に係りやすいという制約を実現している。また、オブジェクトは自分の識別子をメッセージに乗せて送ることができ、これによって得た識別子を使うことで直接目的の単語とメッセージ通信ができる。POWER では相手の必要な情報がわからない場合、とりあえず自分の識別子を送り、欲しい情報

Incremental disambiguuation with POWER, the object-oriented parser.

Hiroyuki Takahashi (hiro@tinlp.info.eng.niigata-u.ac.jp), Takeshi Shibayama, Masahiro Miyazaki (miyazaki@info.eng.niigata-u.ac.jp)

Niigata University

は相手から問い合わせをしてもらうという手法を用いて単語間を流れる情報量を減らしている。

各オブジェクトには最低一回実行の機会を与えるためにシステムからrunメッセージと呼ばれる特殊なメッセージが送られる。runメッセージの送信は文頭の単語=オブジェクトから順に行なわれる。このrunメッセージの流れをrunシーケンスと呼ぶ。このrunシーケンスが解析の主要な流れを表す。つまり解析はrunシーケンスが文頭から出発して文末に達した時点で終る。このように解析は全体として文頭から文末へと漸進的に進む。

3 統合的な解析

漸進的な曖昧性解消には、形態素、構文、意味の各解析レベルの統合が不可欠である。形態素解析についてはすでにPOWERとの融合法が提案されている。また、単語中心に記述するPOWERは意味解析との融和性も高い。

3.1 形態素解析との融合

形態素解析を構文解析と分離して行なう場合、形態素解析の曖昧性の全パターンについてそれぞれ独立に構文解析を行なう必要がある。しかしこの方法では形態素解析で構文、意味情報を利用できないため曖昧性が十分にしぼり込めず、それ加えて構文解析でも曖昧性が発生するため、その組合せで全パターン数が爆発してしまう。

この問題を解決するために、柴山ら[4]は形態素解析の結果を展開せずにネットワーク構造のままで入力し、枝分かれで処理を分岐していく方法を提案した。もちろんこの方法でも後方へ向かって処理が次々と分岐して可能性の総数が爆発してしまう。そこで、枝の合流点でセレクタと呼ばれる曖昧性解消手続きを適用することにより解析途中で曖昧性を解消していた。本稿ではこの機構を全てのレベルの曖昧性解消に一般化する。

3.2 意味解析との融合

意味の解析では概念を主に取り扱う。概念とは通常単語に対応するため、単語中心に記述するPOWERでは意味解析との融合が容易である。各単語はその概念情報を蓄積し、その情報を交換することで意味処理を随時行なうことができる。

例えば、試験的に実装された日本語文法においては動詞の格パターンや名詞のカテゴリなどの意味情報が使用されている。これらの情報は名詞と動詞の格関係の解析に使用される。

4 並列処理のための機構

曖昧性の漸進的解消のためには、全ての可能性を並列的に試す必要がある。POWERでは複数の解析処理を並列に実行するために、スレッドと呼ばれる処理の単位を導入する。各スレッドはオブジェクトネットワークを共有するが、各オブジェクトの実行コンテキスト、すなわちオブジェクト固有の変数の値は別々のものを持つ。そのため同じ単語に相当するオブジェクトがスレッド毎に異なる動きをすることができる。

4.1 スレッドのライフサイクル

スレッドのライフサイクルを図1に示す。POWERにおけるスレッドのライフサイクルはOSにおけるそれと類似している。すなわちスレッドは待ち行列に蓄えられ、その先頭のスレッドが実行される。スレッドでの解析が成功終了、あるいは途中で失敗、あるいは後述するようにゲートによって曖昧性解消のためにブロックされると、そのスレッドは実行状態から外され、キューの次のスレッドが実行状態になる。スレッドが成功終了した場合そのスレッドは終了状態になり、最終的に終了状態にあるスレッド（群）が解となる。解析に失敗した場合はそのスレッドは破棄される。ブロックされスレッドは一定の条件を満たすまで、待機状態に置かれ、その後スレッドキューの最後に戻されるか、場合によっては破棄され、消滅する。

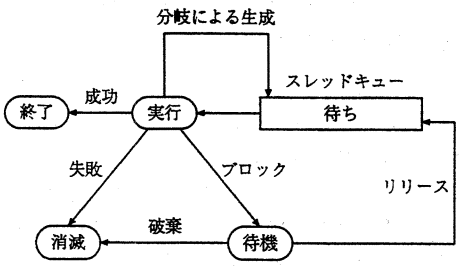

図 1: スレッドのライフサイクル

4.2 スレッドキュー

システムはスレッドをスレッドキューと呼ばれる待ち行列で管理する。システムは常にキューの先頭のスレッドを実行する。そのスレッドが終了、破棄、あるいは待機状態に移ると、システムはスレッドの次のスレッドを実行状態に移す。現在のシステムはシングルプロセッサで動作するため同時に実行状態に置かれるのは一つのスレッドのみであるが、マルチプロセッサシステムでは複数のスレッドを同時に実行できる（基本的にスレッド間のデータ交換、共有は行なわれない）。

4.3 スレッドの生成

解析開始時にはただ一つのスレッドが存在する。解析中に何らかの曖昧性が発生した場合、スレッドは二つに分岐する。スレッドが分岐するのは以下の二つの場合である。

1. メッセージの送信先が複数ある場合。
2. オブジェクトのメソッドが明示的に曖昧性の発生を指定した場合。

前者は形態素解析での曖昧性に、後者は構文解析での曖昧性に相当する。スレッドの分岐時にはシステムは新しいスレッドを生成し、その新しいスレッドにカレントスレッドの状態（全てのオブジェクトの持つ変数の内容）をコピーし、スレッドキューの先頭に入れる。つまりカレントスレッドの動作が終ると、その次に実行される。

5 曖昧性解消のための機構

曖昧性の解消は同じルートを流れる複数のスレッドの解析内容を比較することで行なわれる。そのための機構として、ゲートとセレクタというものを導入する。ゲートはルートの合流点などに置かれ、そこを通過するスレッドをブロックする。ブロックされたスレッド群にはセレクタと呼ばれる手続きが適用され、（可能なら）曖昧性の解消が行なわれる。

セレクタはゲートがブロックしたスレッドに対して曖昧性の解消処理を適用する。実際にどのような判定を加えるかは本稿では詳しくは触れないが、柴山ら [4] は（主に形態素解析の曖昧性解消を目的として）動詞の格パターンによる判定を提案している。この他にも名詞句の構造や連用修飾先の曖昧性の解消などを行なうことができる。これらの曖昧性の解消には各単語の持つ構文的、意味的情報が最大限利用される。

6 処理の効率化

漸進的に曖昧性を解消する場合、各解析スレッドを並列に実行するが、それぞれのスレッドで部分的に同じ解析を行なうことがしばしばある。このような共通の処理はスレッドを統合して一回だけ実行することで、解析の効率を上げることができる。本機構では解析の途中結果を比較し、それが一致したスレッドの統合を行なう。さらに、曖昧さを表現するデータ構造を導入することで、類似した途中結果を持つスレッドの統合も行なう。

6.1 単純な統合

各ゲートはそれを通過して後方に流れたメッセージを記録する。もし run シーケンスがそのゲートに到達した時、複数のスレッドから同じメッセージ列が流れていたなら、後方の解析に及ぼす影響は基本的に同じであると考えられる。そういう場合にはスレッドを統合して一本にする。もちろんその後の解析でメッセージが合流点を越えて戻ることがあり得るが、その場合は再びスレッドが分岐する。

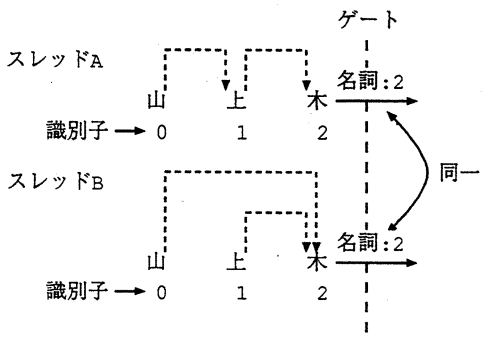

図 2: スレッドの統合の例 (1)

例として、「山の上の木を...」という文の解析における解析経過の一部を図 2 に示す。これは「木」の処理が終った時の状況である。ここで、実線が

メッセージ送信を示す。破線は「山」「上」「木」の各オブジェクトによって判断された連体修飾関係である。図に示した二つのパターンでは連体修飾関係の判断は異なっているが、後方に送られるメッセージは同じである。従って後方の解析に対する影響は同じなので、スレッドを統合できる。

ここでメッセージの「名詞:2」というのはメッセージ「名詞」に引数「2」を付けて送信したという意味である。この引数の「2」は単語「木」の識別子である。後でこの単語「木」に対する詳細な情報が必要になった時にはこの識別子を使って問い合わせが行なわれる。そしてその場合には処理は再び分岐する。ここで、もし「木」の全ての情報を（連体修飾関係の情報を含めて）送ってしまうと、このような統合はできないし、また情報が多くなるためそれらの比較にかかる計算量が増大してしまう。

6.2 曖昧データ構造による高度な統合

曖昧さを表現するデータ構造を導入することで、スレッドの統合をより多くの場合、つまりメッセージ列が完全に一致しなくても行なうことができる。オブジェクトの識別子はその使用法が限られている（別のオブジェクトに渡すか、メッセージの宛先として指定するかのどちらか）ため、文法記述の変更なしに、曖昧データ構造を導入できる。そこで、本機構ではオブジェクトの識別子のみを曖昧なデータ構造で表現できることにした。

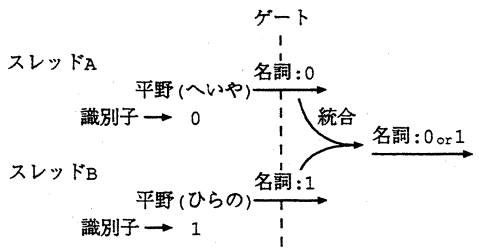

図 3: スレッドの統合の例 (2)

例として「平野が来た。」という文の解析経過を図に示す。「平野」には普通名詞（へいや）、固有名詞（ひらの）の曖昧さがあり、これは形態素解析時に発生する。そのため POWER には普通名詞と固有名詞が並列になったネットワーク構造が入力され、その分岐点でスレッドが分岐する。

それぞれの「平野」の処理が終った時の状況を図 3 に示す。それぞれのスレッドでのゲートを通過したメッセージは引数の識別子のみが異なっている（この時点では普通名詞と固有名詞のふるまいは同じ。後の問い合わせで異なる返答をする）。そこで、この識別子を曖昧な識別子（0 または 1）に置き換えることで、スレッドの統合が行なえる。このメッセージは動詞「来た」が受け取り、動詞の格パターンチェックのためにこの曖昧識別子を使って詳細なデータ（この場合名詞のカテゴリコード）の問い合わせを行なう。するとまたスレッドが二つに分かれ、それぞれが別の返答をする。

この場合、スレッドは統合されたあとすぐ分離してしまうが、「平野が」と「来た」の間に大きな名詞句が挿入されたような場合には、その解析処理が二回行なわれるのを防ぐことができる。

7 おわりに

本稿ではオブジェクト指向パーザ POWER で漸進的な曖昧性解消を行なうために、並列処理機構を導入し、曖昧性を効率的に解消する手法を提案した。

各種の曖昧性解消法の実装と評価が今後の課題である。

参考文献

[1] C.S.Mellish : Computer Interpretation of Natural Language Descriptions, Ellis Horwood, 1985 (邦訳: 田中穂積 : コンピュータのための自然言語処理理解の基礎, サイエンス社, 1987)

[2] 秋葉友良, 伊藤克亘, 奥村学, 田中穂積:増進的曖昧性解消モデルに基づいた統合的日本語解析, 「自然言語処理における統合」シンポジウム論文集, pp.93-100 (1991)

[3] 高橋博之, 宮崎正弘:オブジェクト指向パーザ POWER, 自然言語処理学会第 2 回年次大会, B1-5(1996)

[4] 柴山威, 宮崎正弘:構文解析との部分融合による日本語形態素解析の曖昧性解消法, 情報処理学会第 53 回全国大会, 1L-6(1996)