構文解析システムＳＡＸのＣＩＬによる実現

3K-1

杉村領一（第二研究室），松本裕治（第一研究室）
（財）新世代コンピュータ技術開発機構研究所

０．はじめに

構文解析システムＳＡＸ[1],[2],[3] は、ＤＣＧ[4]で記述された文法をProlog のプログラムに変換するトランスレータより成る。変換手法は、ＤＣＧで記述された文法を並列論理型言語へ変換する事を目的として考えられたものであるが、逐次言語による実現をＳＡＸと呼んでいる。ＣＩＬ[5] はPrologのデータ構造を部分項により拡張すると共に、Prolog-II[6],ESP[7]などで実現されている遅延実行制御を可能にした論理型言語である。本報告ではＳＡＸとＣＩＬを組み合わせる事により高速で、意味記述力の高い言語処理系が構築できる事を示す。

１．ＳＡＸの概要

ＳＡＸはその動作が決定的になっており副作用を用いず、完全にコンパイルされた形で高速に動作する。また、トップダウン予測を行っており更に高速な解析を実現している。ところがＳＡＸを逐次処理環境で使用するには二つの問題点があった。

まず、並列処理を前提として開発されたため、逐次（または疑似並列）型言語の上で実現する場合にはPatr-II[9]などと同様にＤＣＧの論理変数による環境複写の問題を有した。

また、ＳＡＸは構文的な曖昧性については対処しているがそれ以外の部分は決定的に動作するため、ＤＣＧの補強項に曖昧性がある場合、この中の一つだけの解しか引き出さなかった。

これらの問題点は文法または補強項に曖昧性が出た時点で環境の複写を行えば解消されるが、解析時に多くの曖昧性が出る場合に、並列型言語と同様、最終的な解釈の成否を問わず環境を曖昧さの数だけ複写する必要があり、解析時間を遅らせる。ところが、ＣＩＬの遅延実行機能を用いて補強項の解釈を構文的な処理が終了するまで遅らせる事により、高速にＤＣＧのほぼフルセットの解析が可能になる見通しを得た。本稿では遅延実行制御の説明の後、其れ其れの問題点に触れると共に解決法を示し、今後の展望に付いて触れる。

２．遅延実行制御

ＣＩＬによる遅延実行制御のプログラムは例えば図１の(1)のようになる。freeze(F,...)は論理変数Ｆがグランドになるまで式の実行が遅延される。 freeze(F,...)そのものはサクセスする。

```
(1)  top(X) :- freeze(F,(X = a;X = b)),
               freeze(F,(X = b;X = a)),
               write('konnichiwa'),nl,
               F = atom.
(2)  | ?- top(X).
     konnichiwa
     X = a ;
     X = b ;
     no.
     | ?-
```

- 図１ -

このプログラムの実行例を(2)に示す。まず(1) の第一,第二,式は実行が遅延されwrite文が実行される。論理変数Ｆにグランドのatomがバインドされた後、二つの式が評価される。本例の場合遅延された式の間でバックトラックが起きている。以上のように、遅延実行は遅延条件となっている変数がグランドになった位置に遅延された式が挿入されたかのように実行される。

３．問題点とその解決策

３．１．環境複写

構文的な曖昧さがある場合には環境の複写を行う必要がある。具体的には複数の解釈で共通の論理変数を共有している場合に、各々の解釈が異なるオブジェクトをこの論理変数に単一化する可能性があり、其れ其れの解釈を成功させるために、この論理変数を解釈毎に複写する必要が出るのである。構文的な曖昧さが出る度に環境の複写を行うと、最終的には使われないような解釈についてまで環境を複写する必要があり無駄が多い。

３．２．意味記述部の曖昧さ

また、ＳＡＸは決定的に解析を行ない、補強項の曖昧さのうち一つだけを処理する。つまり、補強項を処理した後にバックトラックは起こらず他の曖昧さは評価されない。

Inplementation of SAX parsing system in CIL

Ryoichi SUGIMURA, Yuji MATSUMOTO
Institute for New Generation Computer Technology (ICOT)

3．3．解決策

我々はこれらの問題に対処するためＣＩＬで実現されている遅延実行制御を用いた。つまり、構文処理だけをまず行い、補強項に記述された処理を全て遅延実行制御で遅らせる。構文処理を終了した段階で、補強項の処理を行えばよい。この方法により、構文的な曖昧さの数だけ環境の複写を行えば良くなり処理効率を上げる事が可能になった。

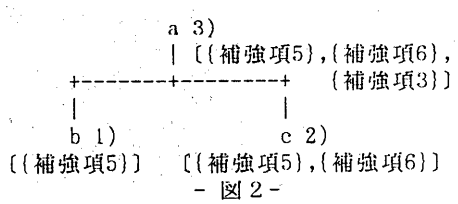
しかし、本方式では例えば構文解析中に補強項に動的に環境を評価して構文解釈数を減少させるような式が記述されている場合には、これを動的に評価できない。これを解決する方法として、ユーザに動的に評価すべき補強項と評価を構文解析終了時点まで遅らせる補強項を明示的に指示させる方法を検討中である。方法としては、例えば図２のような記法を考えている。

```
a --〉 b,{補強項1},c,{補強項2}#{補強項3}.
b --〉〔終端記号1〕,{補強項4}#{補強項5}.
c --〉〔終端記号2〕,{補強項5}#{補強項6}.
```

- 図２-

つまり、解釈を動的に行う必要のあるものについては、ＤＣＧの中に補強項の1,2,4,5，のように示し、評価を構文解析終了後に行うものは、＃の後に補強項の3,5,6,のように示す。

本記法によれば補強項の遅延実行は、ＣＩＬの遅延実行制御を用いなくても行う事が可能になる。つまり、遅延実行すべき補強項は構文処理の結果と共にボトムアップにこれを積み上げ、構文処理が終わった段階で、積み上げられた補強項を順次評価する。図１の文法の場合には例えば図３の1)から3)ような順に積み上げ、補強項の，5,6,3,の順に評価する。



- 図２-

上記補強項３，５，６の解釈の順は、ＳＡＸコンパイラで種々の制御が可能であり、遅延実行制御を用いて補強項の評価をする場合に較べてユーザは明示的に補強項の処理を掴む事が可能になる。

補強項の曖昧さも遅延実行制御で解決が可能である。つまり、補強項の処理を最後まで遅延させて構文処理が終了した後に、まとめて補強項の処理を行えば、遅延実行された補強項の間でバックトラックが起き、補強項の解釈を全て評価できる。

4．実験結果

ＳＡＸでは終端記号を一文字毎のリストにして文法を書けば、全解探索による形態素解析の機能を盛り込める。我々は、ＤＣＧで記述された日本語の文法をＳＡＸプログラムに変換し、形態素、構文、意味解析の実験を行ない以下の結果を得た。解析はＰＳＩマシン〔10〕で行った。また当該文法は談話理解実験システムＤＵＡＬＳの構文意味解析部として用いられている。

１）文法数　　２４７　ＤＣＧ規則数
　　４７０　ＳＡＸ規則数
２）意味記述　ＣＩＬ
３）平均処理語数　　１２語／秒

5．おわりに

ＳＡＸは決定的な処理により十分な構文解析速度を持ち、遅延実行制御によりＤＣＧのほぼフルセットを扱う事ができる。前述したように、ユーザが明示的に補強項の評価順序を指定出来るようにコンパイラを改良し、逐次実行型の構文解析システムとして十分に使えるようにしたいと考えている。

［謝辞］

当研究にあたり、ＩＣＯＴ横井第二研究室室長、古川第一研究室長には暖かい御示唆を頂きました。また、ＩＣＯＴ各研究員の方々、及び電総研の方々にも種々の議論を頂きました。本紙面をおかりして御礼申し上げます。

［参考文献］

〔1〕松本裕治,並列構文解析,自然言語53-2,情報処理学会,1986.
〔2〕松本裕治,並列構文解析法とその評価,冬のLAシンポジウム,1986.
〔3〕松本裕治,杉村領一,論理型言語に基づく構文解析システムＳＡＸ,ソフトウェア科学会論文誌,1986.
〔4〕Pereifa,F.C.N. and Warren,D,H,D., "Definite Clause Grammars--A survey of the Formalism and a comparison with Augmented Tramsition Networks," Artifical Intelligence, 13,pp.231-278, 1980.
〔5〕Mukai Kuniaki,Unification over Complex Indeterminates in Prolog,ICOT Technical Report, No.TG113, 1985.
〔6〕Prolog II version 2.1 VAX-11,Manual
〔7〕Chikayama Takashi,ESP Reference Manual, ICOT Technical Report, No..TR044
〔8〕Ueda Kazunori,Guarded Horn Clauses,ICOT Technical Report, No.103,1985.
〔9〕Pereira,F.C.N.,A Structure-Sharing Representation for Unification-Based Grammar Formalisms,23rd Annual Meeting of the Association for Computational Linguistics,1985,pp137-144.
〔10〕Nishikawa,H. and Yokota,M. and Yamamoto,A. and Taki,K. and Uchida,S., The Personal Inference Machine (PSI): Its Design Philosophy and Machine Architecture,ICOT Technical Report TR-013,1983.