⑪ cu−Prolog for Constraint−Based Grammar

H.Tsuda(ICOT,日本)

発表要旨

cu−Prologは，制約ベースの自然言語文法理論の実現に適した，記号的・組合せ的制約を対象とする制約論理型言語である。ユーザ定義のProlog述語を制約として記述でき，制約解消系にはプログラム変換の技法であるunfold/fold変換をダイナミックに用いている。

cu−Prologの最新版ではデータ構造としてPST（Partially Specified Term：部分項）を採用し，それにより制約ベースの文法理論でしばしば用いられる選言的素性構造にも自然に対応することができた。

cu−Prologの特徴を生かした応用として，JPSG（日本語句構造文法）のパーザがある。多義語の辞書や，文法の構造原理が制約により自然に記述され，構文解析における曖昧性解消のプロセスが制約変換により自動的に表現されている。

質疑応答

質問：Ait−Kaciの研究との関連性は。

回答：PSTについては，Ait−Kaciのϕ−termとの関連性が既に指摘されている。cu−PrologでもPSTの単一化で参考にしている。

コメント：Ait−Kaciはその後LIFE（Ait−Kaci and Lincoln,1988）という自然言語処理の枠組みを提唱している。LIFEでも素性構造の選言について言及しているので，関連性を述べた方が良いだろう。

（その後，調査したところ，同論文によればLIFEは，ラティス，素性構造，集合などを含む自然言語文法の記述体系である。ただし，実装はデモ程度のものしか行なわれていない。実現に当たってのアルゴリズムやヒューリスティックスについては，やはりcu−Prologと同様の考察が必要であると思われる。）

質問：制約変換のcompletenessは。

回答：unfold/fold変換はホーン節に対してsoundかつcompleteである。ただし，cu−Prologの制約変換は，探索規則が固定しており，すべての制約に対して完全であるとは言えない。finiteな述語や一部のモジュラー述語に関しては完全であることが証明されている。

質問：unfold/fold変換は処理が重くなるのでは。

回答：確かにunfold/fold変換はプログラム変換の技法なので，それをダイナミックに使っては処理が重くなる場合もあるのも確かである。unfold変換はともかく，fold変換のチェックには指数オーダーの計算量がかかる場合もある。

制約の前処理を行ない，ダイナミックな計算を減らすことが考えられる。