

4X-6

文法型計算モデルへの
否定の導入

山下義行 中田育男 (筑波大学電子情報工学系)

1. はじめに

Coupled Context Free Grammar(CCFG)に基づく計算モデル(文法型モデル)を提案した[1][2][3]。この計算モデルでは、複数の文脈自由文法の対応する生成規則を集合としてCouplingし、さらに書き換えの制御を行なうメタ記号 $\approx$ を付加したものをプログラム(CCFGプログラム)と考える。

この計算モデルの特徴のひとつとして、「任意のpurePrologプログラムはCCFGプログラムへ機械的に等価変換可能である[2]」ことがあげられる。このプログラム変換はひとつのHorn節をひとつの規則集合へ変換するもので、これは論理型計算モデルと文法型計算モデルの背後の数学的構造が類似していることを示すにとどまらず、論理型モデルの研究成果を文法型モデルへ容易に導入、展開できることを示唆している。本発表では、そのような展開のひとつとして、この文法型モデルへ『否定表現』を導入、定義することを試みる。

基本となるアイディアは、purePrologからCCFGへのプログラム変換を拡張し、否定記号 $\sim$ を持つ文法型モデルを導くことである(下図参照)。

pureProlog $\rightarrow$ CCFG
↓(拡張)

否定を持つProlog $\rightarrow$ 否定を持つCCFG
否定の意味論としては閉世界仮説に基づく論理型モデルのnegation as failureを用いる。

2. 構文則

CCFGプログラムの生成規則の右辺は否定を表すメタ記号 $\sim$ を含んでもよい。その場合 $\sim$ の係る部分文字列の範囲を明示するためにメタ記号()を用いることができる。たとえば

$\sim(a b) c, \sim a \sim b \sim c, a \sim(b \sim c)$
などは生成規則の右辺に用いてもよい。

【例1】以下は、CCFGプログラムである。

$\{X \rightarrow \sim Y\} \{Y \rightarrow a\} \{Y \rightarrow a Y\}$

3. 否定の意味の基本的な考え方

論理型モデルの閉世界仮説に基づく操作的意味論はnegation as failure[4]として知られている。論理型モデルの証明過程は文法型モデルの構文解析過程に対応しており、証明できないことは構文解析できないことに対応する。

文法型モデルにおける構文解析とは、書き換えの対象である文形式が $\alpha \approx \beta$ のようにメタ記号 $\approx$ を含む場合、 α と β を等しい終端記号列に書き換えていく操作をいう[2]。もし $\alpha \approx \beta \Rightarrow x \approx x \quad x \in T^*$ と、 α と β が共に同じ終端記号列を導出したならば、この構文解析は成功である。

【例2】例1のプログラムを用いて文形式 $a a \approx Y$ の書き換えを行なうと、

$a a \approx Y \Rightarrow a a \approx a Y \Rightarrow a a \approx a a$

となり、Yによる $a a$ の構文解析は成功する。

この構文解析を、文形式が $\alpha \approx \sim \beta$ という形で否定記号を含む場合に拡張する。このとき、もし $\alpha \approx \sim \beta \Rightarrow x \approx \sim x$ という導出が可能ならば、構文解析は失敗と解釈する。もし $\alpha \approx \sim \beta \Rightarrow x \approx \sim x$ という導出が存在しないならば、構文解析は成功、 $\beta \Rightarrow x$ と解釈する。

【例3】例1のプログラムを用いて文形式 $a a \approx X$ の書き換えを行なうと、

$a a \approx X \Rightarrow a a \approx \sim Y \Rightarrow a a \approx \sim a a$

という導出が存在するから、Xによる $a a$ の構文解析は失敗である。 $a b \approx X$ の書き換えを行なうと、

$a b \approx X \Rightarrow a b \approx \sim a b$

という導出は存在しないから、Xによる $a b$ の構文解析は成功し、 $X \Rightarrow a b$ である。

上の議論に整合するように否定の表示的意味を与える。メタ記号 $\approx$ 、否定記号 $\sim$ を含まない任意の文形式 α に対し α の表示的意味 $[\alpha]$ を

$[\alpha] = \{x \mid \alpha \Rightarrow x\} \quad \dots (1)$

Introduction of Negation to Coupled Grammars
Yoshiyuki YAMASHITA and Ikuo NAKATA
University of Tsukuba.

であるとするとき、 α の否定形 $\sim\alpha$ に対し

$$[\sim\alpha] = T^* - [\alpha] \quad \dots (2)$$

ここに T は全ての終端記号の集合である。

と定義する。

【例4】例1のプログラムについて

$$[Y] = \{a, aa, aaaa, \dots\}$$

$$[X] = [\sim Y] = T^* - [Y]$$

$$= \{\epsilon, b, ab, bb, \dots\}$$

となる。 $[X]$ は $\{x \mid x \approx X \text{ が構文解析可能}\}$ に等しい(表示的操作の意味は完全である)。

4. 否定の表示の意味論

規則集合の右辺または文形式中に $\sim\sim a$ のような2重否定が現れる場合、この表示の意味を

$$[\sim\sim a] = [a]$$

であるをしたい。そこで3章の考え方を基として、 $\alpha \approx \sim\beta$ という特殊な場合のみでなく、文形式中の任意の部分に任意個の否定記号が現れる一般的な場合に対する拡張を行なう。

文形式中に多重に係る否定記号を一重にするための公理を与える。 α_i ($i \geq 1$) を T^+ の要素として、

$$\sim(\alpha_1 : \sim\alpha_2 : \alpha_3 : \dots) = \sim\alpha_1 : \alpha_2 : \sim\alpha_3 : \dots$$

$$\sim(\sim\alpha_1 : \alpha_2 : \sim\alpha_3 : \dots) = \alpha_1 : \sim\alpha_2 : \alpha_3 : \dots$$

ここに $:$ は連接の演算子を表す。

である。すなわち否定記号は、連接された各 α_i または $\sim\alpha_i$ に対し分配則を満たすとする。否定記号が1重にのみ係るような文形式を『否定を含む文形式の標準形』という。公理から任意の文形式は標準形へ変形できる。

標準形の表示の意味は、 β_i ($i \geq 1$) を α_i または $\sim\alpha_i$ として、

$$[\beta_1 : \beta_2 : \dots : \beta_n] = [\beta_1] : [\beta_2] : [\dots] : [\beta_n]$$

とする。各 $[\beta_i]$ の意味は(1)(2)式に従う。

【例5】たとえば、 $abc \approx \sim((\sim a)d(\sim c))$ が導出されたとき、公理より

$$\sim((\sim a)d(\sim c)) = a(\sim d)c$$

となり、

$$abc \in [a(\sim d)c] = \{a\} : (T^* - \{d\}) : \{c\}$$

であるから、導出(構文解析)は成功である。

5. 導出の戦略

否定に関する操作的表示の意味論の完全性を要求することは實際上多くの困難が伴うが、操作の意味を表示の意味に近付けるための重要な導出の戦略が

考えられる。

論理型モデルにおける否定を含む導出法の最も重要な戦略は、「基礎例(ground instance)を優先して評価する」ことである。否定記号の係った変数を含む原子式の評価は一般には正確な答えを与えないから、そのような場合には変数を何らかの方法で定項に束縛した後に評価を行なう[4]。

文法型モデルでも同様に、否定を含む文形式は、その文形式と $\approx$ で結ばれた別の否定を含まない文形式が終端記号列に書き換えられた後に評価する(書き換える)。たとえば、もし $X \approx \sim Y$ という文形式があるならば、 X を適当な終端記号列へ書き換えた後に $\sim Y$ の書き換えを行なう。もし $\sim X \approx \sim Y$ のように両辺に否定記号が現れた場合には、この文形式を $* \approx \sim X \approx \sim Y$ と変形し、 $*$ を適当な終端記号列に書き換えた後に $\sim X$ と $\sim Y$ の書き換えを行なう。ここに $*$ は

$$[*] = T^*$$

と定義された非終端記号である。この書き換え戦略は、もし構文木が有限ならば、正しい答えを与える。

6. おわりに

文法型計算モデルは論理型モデルと数学的に密接に関係しており、文法型モデルへの否定の導入も論理型モデルのnegation as failureを利用して議論を始め、基本的考え方は論理型モデルをそのまま受け継いで展開した。否定の意味論に関する本質的な問題は両モデルに共通であるが、構文や基本データ型(項 vs. 列)の性質の違いから、形式化は異なる様相を呈することがわかった。今後、実際の言語処理系作成と合わせてさらに詳しく検討して行く。

参考文献

- [1] 山下義行、中田育男：文脈自由文法の拡張とそれに基づく計算モデル、情報処理学会ソフトウェア基礎論研究会資料SF15-5(1985)
- [2] Yamashita, Y. and Nakata, I.: Coupled Context Free Grammars and..., 京大数理解析研究所SSE資料(1986)
- [3] 中田育男、山下義行：CCFGにおけるプログラム変換、情報処理学会ソフトウェア基礎論研究会資料SF17-3(1986)
- [4] Clark, K.L.: Negation as failure, "Logic and Data Bases.", Plenum Press