属性値主導型 拡張 LL(1) 文法の提案

7G－1

大木康幸　平見知久　大下敬治　山下義行　中田育男
筑波大学

1 はじめに

構文解析手法の研究の1つに、先読み記号以外にも属性文法の属性の値を用いる属性値主導構文解析法の研究がある。属性値主導構文解析を用いれば、従来先読み記号だけでは構文解析が不可能な文法も構文解析可能になり、より広い文法クラスが扱える。

[1][2]で提案された拡張1パス型属性文法は、拡張LL(1)文法(LL(1)文法に構文の反復などの記述を許した文法)に、属性値主導構文解析のための記法を導入した文法である。しかし[1][2]では、終端記号にしか属性条件(属性値の条件)を記述できないという制限があるため、柔軟な文法記述ができない。例えば、複数の属性条件を1つの終端記号に記述するのではなく、幾つかの構文規則に分散して記述したい場合があるが、[1][2]ではできない。また[1][2]では文法の定式化が十分に行なわれていないという問題もあった。

本論文では、[1][2]よりも柔軟な属性値主導構文解析のための記述ができる文法を提案し、その定式化を行う。この文法を属性値主導型　拡張LL(1)文法(Attribute-directed Extended LL(1) grammar、以下AELL(1)文法と略す)と呼ぶ。

2 属性値主導構文解析

手続き呼び出し文と代入文の文法の例を次に示す。

$$\begin{aligned} &statement \rightarrow \text{IDENT } ':=' \ expression. && (1)\\ &\quad \mid \text{IDENT } '(' \ parameter \ ')' && (2)\end{aligned}$$

この文法は先頭記号IDENTが同じであるため、LL(1)文法ではない。左括り出しの技法を用いればLL(1)文法に変更可能であるが、文法は複雑になる。

拡張1パス型属性文法[1][2]では、終端記号にそれを導出した時の属性条件 $\langle ... \rangle$ を次のように記述できる。

$$\begin{aligned} &statement \rightarrow \text{IDENT}_{(\uparrow kind=var)} \ ':=' \ expression && (1')\\ &\quad \mid \text{IDENT}_{(\uparrow kind=proc)} \ '(' \ parameter \ ')' && (2')\end{aligned}$$

上矢印の付いた名前$\uparrow kind$は終端記号IDENTの合成属性を表す（同様に下矢印の付いたものは非終端記号の継承属性を表す）。この文法の場合、IDENTを読み込んだ時点で合成属性$kind$の値がvarなら(1')、$proc$なら(2')の構文規則を選択できる。

しかし[1][2]では、終端記号にしか属性条件を記述できないため、文法記述が複雑になることがある。例として関数呼び出しの実引数の文法を示す。各非終端記号の継承属性$\downarrow param$は、$value$なら値渡し、$reference$なら参照渡しの実引数であることを表す。

$$\begin{aligned} &actual\_parameter(\downarrow param) \rightarrow \\ &\qquad expression(\downarrow param) \\ &\quad \mid \text{IDENT}_{(\uparrow kind=var \ \wedge \ \downarrow param=reference)} \\ &expression(\downarrow param) \rightarrow \\ &\qquad \text{IDENT}_{(\uparrow kind=var \ \wedge \ \downarrow param=value)} \\ &\quad \mid \text{NUMBER}\end{aligned}$$

値渡しの条件$\downarrow param = value$が$expression$の規則内に書かれている。本来この条件は$actual_parameter$の中に記述すべきである。これを解決するため、終端記号以外にも記述可能な属性条件の記法を考案した。$\langle \downarrow param = ... \rangle$が新しい属性条件の記法である。

$$\begin{aligned} &actual\_parameter(\downarrow param) \rightarrow \\ &\qquad \langle \downarrow param = value \rangle \ expression \\ &\quad \mid \ \langle \downarrow param = reference \rangle \ \text{IDENT}_{(\uparrow kind=var)} \\ &expression \rightarrow \text{IDENT}_{(\uparrow kind=var)} \ \mid \text{NUMBER}\end{aligned}$$

このような記法を導入した文法のうち、属性値主導構文解析が可能な文法をAELL(1)文法と呼ぶ。

3 属性値主導型 拡張 LL(1) 文法

3.1 条件つき終端記号

属性値主導構文解析では先読み記号と属性条件を同時に扱う。本論文ではこれらを（終端記号, 属性条件）の組で表し、条件つき終端記号と呼ぶ。次に、条件つき終端記号について(長さ1の)連接$conc_1$を定義する。

[定義] 条件つき終端記号t_1, t_2の連接$conc_1(t_1, t_2)$:

$$conc_1(\ (x,p),\ (y,q)\ ) = \begin{cases} (x,p) & x \neq \varepsilon \text{の時} \\ (y, p \wedge q) & x = \varepsilon \text{の時} \end{cases}$$

3.2 *CFirst*, *CFollow*, *CDirector* 集合

次に(拡張)LL(1)文法で用いる構文規則eの先頭記号集合$First(e)$を条件つき終端記号を扱うものへと拡張し、これを$CFirst(e)$と呼ぶことにする。$CFirst(e)$はeを導出した時に先頭に来る終端記号と、その時点で満足する属性条件を、条件つき終端記号の集合として表している。例として次の構文規則eを考える。

$$e(\downarrow a, \downarrow b) \rightarrow \langle a = 1 \rangle \ x \ y \ \langle b = 2 \rangle \ z$$

この場合、$CFirst(e) = \{\ (x,\ a = 1)\ \}$となる。条件$\langle b = 2 \rangle$は先頭記号$x$を導出する際には必要ない条件であるため、$CFirst(e)$に含めない。

次の例を考える。構文規則中の§PLUS$(\uparrow c, \downarrow b, \downarrow v)$は、$c = b + v$という属性評価を行なう意味規則である。

A proposal for Attribute-directed Extended LL(1) grammar
Yasuyuki OHKI, Tomohisa HIRAMI, Keiji OHSHITA, Yoshiyuki YAMASHITA, Ikuo NAKATA
(Univ. of Tsukuba)

$$S(\downarrow a) \rightarrow A\ (\ e_1(\downarrow a) \mid e_2(\downarrow a)\ )\ x$$
$$e_1(\downarrow b) \rightarrow \S\mathrm{PLUS}(\uparrow c, \downarrow b, \downarrow 1)\ \langle c = 3\rangle\ x$$
$$e_2(\downarrow b) \rightarrow \S\mathrm{PLUS}(\uparrow c, \downarrow b, \downarrow 2)\ \langle c = 3\rangle$$

この場合、$First(e_1) = \{x\}$である。e_1からxを導出する際に必要な属性条件は、一見$\langle c = 3\rangle$に思える。しかし、非終端記号Aの直後において値が決定している属性はaだけであり、条件$\langle c = 3\rangle$を評価できない。このため、$CFirst(e_1)$では、パラメタの受渡し$a = b$、意味規則§PLUS$(\uparrow c, \downarrow b, \downarrow 1)$も属性条件に含める必要がある。この文法の$CFirst(e_1)$, $CFirst(e_2)$を次に示す。また、表1に$CFirst(e)$の計算方法を示す。

$$\begin{aligned} CFirst(e_1) &= \{\ (x,\ a = b \wedge PLUS(\uparrow c, \downarrow b, \downarrow 1) \wedge c = 3)\ \} \\ &= \{\ (x,\ a = b \wedge c = b + 1 \wedge c = 3)\ \} \\ &= \{\ (x,\ 3 = a + 1)\ \} \\ &= \{\ (x,\ a = 2)\ \} \\ CFirst(e_2) &= \ldots\text{同様}\ldots = \{\ (\varepsilon, a = 1)\ \} \end{aligned}$$

$Follow$集合を条件つき終端記号に拡張した$CFollow$集合も、$CFirst$集合と同様に定義できる。$CFollow$集合の計算方法を表2に示す。先の文法の$CFollow$集合は以下の通りである。

$$CFollow(e_1) = CFollow(e_2) = \{\ (x, true)\ \}$$

表1, 2に示した計算方法は$First$, $Follow$集合の計算方法をそのまま拡張したものとなっている。このため、$CFirst$, $CFollow$集合は$First$, $Follow$集合と同時に計算することができる。

同様に$Director$集合を拡張した$CDirector$集合を定義する。また先の文法の例を示す。

[定義] 条件つき$Director$集合$CDirector$:

$$CDirector(e) = conc_1(\ CFirst(e),\ CFollow(e)\ )$$
$$\begin{aligned} CDirector(e_1) &= conc_1(\{\ (x, a = 2)\ \},\ \{\ (x, true)\ \}) \\ &= \{\ (x, a = 2)\ \} \\ CDirector(e_2) &= \ldots\text{同様}\ldots = \{\ (x, a = 1)\ \} \end{aligned}$$

3.3 AELL(1) 条件

与えられた文法がAELL(1)文法である条件、AELL(1)条件の定義を次に示す。定義中の$AL(e)$は構文規則eの左側で値が決定する属性の集合、$AS(e)$はeで決定する属性の集合を表す。

[定義]AELL(1) 条件:

文法中の各構文規則eにおいて、次が成立する。

1. $e = e_1|e_2$の場合、
   $(u,p) \in CDirector(e_1)\ \wedge\ (u,q) \in CDirector(e_2)$
   $\Rightarrow$ $a \in AL(e) \cup AS(u)$、aの定義域の値全てに対し、$p \wedge q = false$が成り立つ。
2. $e = \{e_1\}$の場合、
   $(u,p) \in CDirector(e_1)\ \wedge\ (u,q) \in CFollow(e)$
   $\Rightarrow$ $a \in AL(e) \cup AS(u)$、aの定義域の値全てに対し、$p \wedge q = false$が成り立つ。

先述の文法は、1.で$u = x$, $p = (a = 2)$, $q = (a = 1)$となるが、aの定義域の値全てに対し$p \wedge q = false$となり、AELL(1)条件を満足する。

表1: $CFirst$集合の計算方法

構文規則e	$CFirst(e)$
ε	$\{\ (\varepsilon, true)\ \}$
$\langle p\rangle$	$\{\ (\varepsilon, p)\ \}$
x	$\{\ (x, true)\ \}$, ただし$x \in V_T$
$x_{\langle p\rangle}$	$\{\ (x, p)\ \}$, ただし$x \in V_T$
§R(式1, ..., 式n)	$\{\ (\varepsilon, R(\text{式}1, \ldots, \text{式}n))\ \}$, Rは意味関数
A(式1, ..., 式n) ただし$A \in V_N$	$CFirst(\ \langle p\rangle\ e_1\ \langle q\rangle\ )$ $(\ A(exp_1, \ldots, exp_n) \rightarrow e_1\ ) \in P$ $p = \bigwedge_{exp_i = \downarrow \ldots}(exp_i = \text{式}_i)$ $q = \bigwedge_{exp_i = \uparrow \ldots}(exp_i = \text{式}_i)$
e_1e_2	$conc_1(CFirst(e_1), CFirst(e_2))$
$e_1 \mid e_2$	$CFirst(e_1) \cup CFirst(e_2)$
$\{e_1\}$	$\{\ (\varepsilon, true)\ \} \cup CFirst(e_1)$

表2: $CFollow$集合の計算方法

構文規則e	$CFollow$集合の計算方法
e_1e_2	$CFollow(e_2) = CFollow(e)$ $CFollow(e_1) =$ $conc_1(CFirst(e_2), CFollow(e_2))$
$e_1 \mid e_2$	$CFollow(e_1) = CFollow(e)$ $CFollow(e_2) = CFollow(e)$
$\{e_1\}$	$CFollow(e_1) = CFollow(e)$
$A(\ldots.)$	生成規則右辺に現れる全てのAに対し、部分的に$CFollow$集合を求め、その集合和を取ったものが、$CFollow(A)$となる。 ($A = S$の時、更に$\{\ (\$, true)\ \}$と和を取る)

4 おわりに

拡張LL(1)文法に属性値主導構文解析のための記法を採り入れた、属性値主導型 拡張LL(1)文法を提案した。これにより、以前提案された拡張1パス型属性文法では不可能であった、構文中に埋め込まれた意味規則の選択が可能となった。

今後はこの文法を我々の研究室で開発したコンパイラ生成系EAGLEに採り入れ、記述の簡潔さを評価したい。

参考文献

[1] 金谷英信, 中川裕之, 中田育男: 拡張LL(1)パーサ生成系の提案, 情報処理学会 全国大会予稿集 Vol.44, No.5(1992), pp135-136

[2] 中川裕之, 金谷英信, 中田育男: 拡張1パス型属性文法の提案, 情報処理学会 全国大会予稿集 Vol.45, No.5(1992), pp65-66