АЛГОРИТМ РАСПОЗНАВАНИЯ ЦЕПНЫХ И ЦИКЛИЧЕСКИХ ПРАВИЛ В КОНТЕКСТНО-СВОБОДНОЙ ГРАММАТИКЕ

С. Б. КАЛАЧЕВА

Санкт-Петербургский институт информатики и автоматизации РАН

СПИИРАН, 14-я линия ВО, д. 39, Санкт-Петербург, 199178

<svet@iias.spb.su>

УДК 681.3

Калачева С. Б. **Алгоритм распознавания цепных и циклических правил в контекстно-свободной грамматике** // Труды СПИИРАН. Вып. 5. — СПб.: Наука, 2007.

Аннотация. *Приводится и обосновывается алгоритм распознавания цепных и циклических правил для контекстно-свободной грамматики.* — Библ. 1 назв.

UDC 681.3

Kalacheva S. B. **Formal Algorithm of Recognition of the Chain and Cyclic Rules for the Context Free Grammars** // SPIIRAS Proceedings. Issue 5. — SPb.: Nauka, 2007.

Abstract. *Formal algorithm of recognition of the chain and cyclic rules for the context free grammars is presented and substantiated.* — Bibl. 1 items.

1. Введение

Контекстно-свободная грамматика (КС-грамматика) определяется как четверка множеств [1]

$$G = (N, T, P, S),$$

где N – множество нетерминальных символов, или нетерминалов, T – множество терминальных символов, или терминалов, P – множество правил грамматики вида $A \Rightarrow \alpha$, где $A \in N$, $\alpha \in (N \cup T)^*$, где "$*$" обозначает цепочки символов в алфавите N и T, S – начальный символ грамматики.

Цепным правилом называется правило вида: $A_i \Rightarrow A_j \Rightarrow ... \Rightarrow A_k$, где $A_i, A_j, ..., A_k \in N$.

Циклическим правилом называется цепное правило, для которого $i = k$.

КС-грамматика $G = (N, T, P, S)$ называется *грамматикой без циклов*, если в ней нет циклов.

Без нарушения общности (так как имеется операция дизъюнкции), будем считать, что каждому нетерминалу соответствует единственное правило.

2. Алгоритм нахождения циклических правил

Ниже предлагается простой алгоритм выявления циклических правил. Покажем, что алгебраическая структура $(G^*, *, <\textit{пробел}>, |)$ изоморфна алгебраической структуре $(B^*, \oplus, \oplus, \wedge)$, где $B^* = (B, \{1\}, P_b, S_b)$, в которой B — множество булевых переменных, сопоставляемых нетерминалам $n \in N$ и принимающих значение $\{0,1\}$, $S_b \in B$ — начальный символ, P_b — множество правил, которые

строятся по множеству правил P грамматики G^*. Введем функцию, отображающую

$G^* \to B^*$ (поэлементно),

операцию итерации:

$n_1 * n_2 \to b_1 \oplus b_2 \equiv 1, n_1, n_2 \in N \cup T, b_1, b_2 \in B \cup \{0,1\}$,

операцию конкатенации:

$n_1 < \textit{пробел} > n_2 \to b_1 \oplus b_2 \equiv 1, n_1, n_2 \in N \cup T, b_1, b_2 \in B \cup \{0,1\}$,

операцию дизъюнкции:

$n_1 \mid n_2 \to b_1 \wedge b_2, n_1, n_2 \in N \cup T, b_1, b_2 \in B \cup \{0,1\}$.

В грамматике G^* заменяются нетерминалы $n \in N$ на переменные $b \in B$, терминальные символы $t \in T$ заменяются на 1, затем P — множество правил КС-грамматики — заменяется множеством P_b, причем бинарные операции $\{*, < \textit{пробел} >, |\}$ заменяются операциями $\{\oplus, \oplus, \wedge\}$ соответственно.

Пример.

Исходная грамматика:

R1 : t | s<*пробел*>r*R2;

R2 : R1 | t<*пробел*>(p*R2 | R1);

где $R1, R2 \in N, t \in T$.

Замена:

$b1 : 1 \wedge (1 \oplus (1 \oplus b2))$,

$b2 : b1 \wedge (1 \oplus ((1 \oplus b2) \wedge b1))$.

После преобразования:

$b1 : 1$,

$b2 : b1$.

Тогда можно оставить в стороне рассмотрение алгоритмов на грамматиках и перейти к рассмотрению бинарной структуры, а конечный результат для булевской структуры отобразить обратно в грамматики.

Алгоритм нахождения циклических правил заключается в следующем:

1)Построить правила для изоморфных булевых множеств.

2)Присвоить $i = 1$.

3)Всем булевским переменным, кроме b_i, присвоить значение один: $\forall b_i \in B : b_j = 1,\ 0 \le j < n, j \ne i$, n - число правил; b_i присвоить значение 0.

4)Подставить значения b_i в правила, построенные по исходным правилам и посчитать по их правым частям новые значения (пересчитываются только правила для $b_j \ne 0$).

5)Проанализировать новые полученные значения. Если хотя бы одно значение изменилось, перейти к пункту 4).

6)Процесс заканчивается, когда все значения b_i остаются прежними.

7)Перейти к следующему нетерминалу: $i = i + 1$.

8)Если $i \le n$, перейти к пункту 3), иначе работа алгоритма закончена.

9)Если 2 или больше булевских значений равны 0, значит соответствующие правила цепные.

Труды СПИИРАН. 2007. Вып. 5. ISSN 2078-9181 (печ.), ISSN 2078-9599 (онлайн)
SPIIRAS Proceedings. 2007. Issue 5. ISSN 2078-9181 (print), ISSN 2078-9599 (online)
www.proceedings.spiiras.nw.ru

3. Пример применения алгоритма

Рассмотрим следующий фрагмент грамматики:

```
01  Program: Expression;
02  Expression: Decimal_number |
03       '-' Expression |
04       Expression '+' Expression |
05       Expression '-' Expression |
06       Expression '*' Expression |
07       Expression '/' Expression |
08       '(' Expression ')';
09  Decimal_number: Unsigned_integer |
10                  Unsigned_integer Decimal_fraction ;
11  Decimal_fraction: '.' Unsigned_integer;
12  Unsigned_integer: Digit|Digit Unsigned_integer;
14  Digit: '0'|'1'|'2'|'3'|'4'|'5'|'6'|'7'|'8'|'9';
```

Очевидно, что правила в строках 2-10 цепные. Рассмотрим работу алгоритма.

1)Правила для изоморфных булевых множеств:

$b1 : b2;$

$b2 : b3 \wedge (1 \oplus b2) \wedge (b2 \oplus 1 \oplus b2) \wedge (b2 \oplus 1 \oplus b2) \wedge (b2 \oplus 1 \oplus b2) \wedge (b2 \oplus 1 \oplus b2) \wedge (1 \oplus b2 \oplus 1);$

$b3 : b5 \wedge (b5 \oplus b4);$

$b4 : 1 \oplus b5;$

$b5 : b6 \wedge (b6 \oplus b5);$

$b6 : 1 \wedge 1 \wedge 1 \wedge 1 \wedge 1 \wedge 1 \wedge 1 \wedge 1 \wedge 1 \wedge 1;$

или после преобразования:

$b1 : b2;$

$b2 : b3;$

$b3 : b5;$

$b4 : 1;$

$b5 : b6;$

$b6 : 1;$

2)Присвоим всем булевым переменным, кроме $b1$, значение 1, $b1$ присвоим значение 0:

$b1 = 0, b2 = b3 = b4 = b5 = b6 = 1.$

Получаем:

$$b1 = 1 \quad b2 = 1 \quad b3 = 1 \quad b4 = 1 \quad b5 = 1 \quad b6 = 1.$$

Второй проход алгоритма дает:

$$b1 = 1 \quad b2 = 1 \quad b3 = 1 \quad b4 = 1 \quad b5 = 1 \quad b6 = 1.$$

3)Переходим к $b2$. Полагаем:

$b2 = 0, b1 = b3 = b4 = b5 = b6 = 1.$

Получаем:

$$b1 = 0 \quad b2 = 1 \quad b3 = 1 \quad b4 = 1 \quad b5 = 1 \quad b6 = 1.$$

Второй проход алгоритма дает:

Труды СПИИРАН. 2007. Вып. 5. ISSN 2078-9181 (печ.), ISSN 2078-9599 (онлайн)
SPIIRAS Proceedings. 2007. Issue 5. ISSN 2078-9181 (print), ISSN 2078-9599 (online)
www.proceedings.spiiras.nw.ru

$$b1=0 \quad b2=1 \quad b3=1 \quad b4=1 \quad b5=1 \quad b6=1.$$

4)Переходим к $b3$. Полагаем:

$b3=0, b1=b2=b4=b5=b6=1.$

Получаем:

$$b1=1 \quad b2=0 \quad b3=1 \quad b4=1 \quad b5=1 \quad b6=1.$$

Второй проход алгоритма дает:

$$b1=0 \quad b2=0 \quad b3=1 \quad b4=1 \quad b5=1 \quad b6=1.$$

Третий проход дает:

$$b1=0 \quad b2=0 \quad b3=1 \quad b4=1 \quad b5=1 \quad b6=1.$$

Значит, правила в строках 01-08 — цепные.

5)Переходим к $b4$. Полагаем:

$b4=0, b1=b2=b3=b5=b6=1.$

Получаем:

$$b1=1 \quad b2=1 \quad b3=1 \quad b4=1 \quad b5=1 \quad b6=1.$$

Второй проход алгоритма дает:

$$b1=1 \quad b2=1 \quad b3=1 \quad b4=1 \quad b5=1 \quad b6=1.$$

6)Переходим к. $b5$ Полагаем:

$b5=0, b1=b2=b3=b4=b6=1.$

Получаем:

$$b1=1 \quad b2=1 \quad b3=0 \quad b4=1 \quad b5=1 \quad b6=1.$$

Второй проход алгоритма дает:

$$b1=1 \quad b2=0 \quad b3=0 \quad b4=1 \quad b5=1 \quad b6=1.$$

Третий проход дает:

$$b1=0 \quad b2=0 \quad b3=0 \quad b4=1 \quad b5=1 \quad b6=1.$$

Четвертый проход дает:

$$b1=0 \quad b2=0 \quad b3=0 \quad b4=1 \quad b5=1 \quad b6=1.$$

Значит, правила в строках 1-10 — цепные.

7)Переходим к $b6$. Полагаем:

$b6=0, b1=b2=b3=b4=b5=1.$

Получаем:

$$b1=1 \quad b2=1 \quad b3=1 \quad b4=1 \quad b5=0 \quad b6=1.$$

Второй проход алгоритма дает:

$$b1=1 \quad b2=1 \quad b3=0 \quad b4=1 \quad b5=0 \quad b6=1.$$

Третий проход дает:

$$b1=1 \quad b2=0 \quad b3=0 \quad b4=1 \quad b5=0 \quad b6=1.$$

Четвертый проход дает:

$$b1=0 \quad b2=0 \quad b3=0 \quad b4=1 \quad b5=0 \quad b6=1.$$

Пятый проход дает:

$$b1=0 \quad b2=0 \quad b3=0 \quad b4=1 \quad b5=0 \quad b6=1.$$

Правила грамматики в строках 1-10 и 12 — цепные.

Исправленная грамматика:

```
01  Expression: Unsigned_integer |
02        Unsigned_integer '.' Unsigned_integer |
03        '-' Expression |
04        Expression '+' Expression |
05        Expression '-' Expression |
```


Труды СПИИРАН. 2007. Вып. 5. ISSN 2078-9181 (печ.), ISSN 2078-9599 (онлайн)
SPIIRAS Proceedings. 2007. Issue 5. ISSN 2078-9181 (print), ISSN 2078-9599 (online)
www.proceedings.spiiras.nw.ru

```
06       Expression '*' Expression |
07       Expression '/' Expression |
08       '(' Expression ')';
12  Unsigned_integer: Digit|Digit Unsigned_integer;
13  Digit: '0'|'1'|'2'|'3'|'4'|'5'|'6'|'7'|'8'|'9';
```

1)Правила для изоморфных булевых множеств:

$b1 : b2 \wedge (b2 \oplus 1 \oplus b2) \wedge (1 \oplus b1) \wedge (b1 \oplus 1 \oplus b1) \wedge (b1 \oplus 1 \oplus b1) \wedge (b1 \oplus 1 \oplus b1) \wedge (b1 \oplus 1 \oplus b1) \wedge$
$(1 \oplus b1 \oplus 1)$;

$b2 : b3 \wedge b3 \oplus b2$;

$b3 : 1 \wedge 1 \wedge 1 \wedge 1 \wedge 1 \wedge 1 \wedge 1 \wedge 1 \wedge 1 \wedge 1$;

или после преобразования:

$b1 : b2$;

$b2 : b3 \wedge b3 \oplus b2$;

$b3 : 1$;

2)Присвоим всем булевым переменным, кроме $b1,1$; $b1$ присвоим значение 0:
$b1 = 0, b2 = b3 = 1.$

Получаем:

$$b1 = 1 \quad b2 = 1 \quad b3 = 1.$$

Второй проход алгоритма дает:

$$b1 = 1 \quad b2 = 1 \quad b3 = 1.$$

Можно показать, что так же обстоит дело и с другими булевыми переменными.

Цепных правил нет.

Разумеется, такие выкладки громоздки и алгоритм предназначен для компьютерной обработки.

4. Обосновывающие теоремы

Теорема 1.
Правила КС-грамматики не являются цепными тогда и только тогда, если после завершения работы приведенного выше алгоритма всем правилам, кроме рассматриваемого, будет сопоставлено значение $b_i = 1$.

Доказательство.
Необходимость.
То, что правило не является цепным, означает, что в грамматике отсутствует вывод $A_i \Rightarrow A_j \Rightarrow ... \Rightarrow A_k$. Доказательство от противного. Пусть $b_j = 0$ для некоторого j, отличного от i. Тогда последняя операция – конъюнкция с параметром 0, то есть, $\exists j,...,k$ такие что $b_k : \beta_k \wedge b_j$, $..., b_i : \beta_i \wedge b_j$ Тогда правила для $A_k, A_m, ..., A_j, A_i \in N$: $A_k : \alpha_k \mid A_m$; $\alpha_k \in (N \cup T)^*$, $..., A_i : \alpha_i \mid A_j$; $\alpha_i \in (N \cup T)^*$. Но тогда существует вывод $A_i \Rightarrow A_j \Rightarrow ... \Rightarrow A_k$, $A_k, A_m, ..., A_j$, $A_i \in N$ и правила цепные, а это не так, что и требовалось доказать.

Труды СПИИРАН. 2007. Вып. 5. ISSN 2078-9181 (печ.), ISSN 2078-9599 (онлайн)
SPIIRAS Proceedings. 2007. Issue 5. ISSN 2078-9181 (print), ISSN 2078-9599 (online)
www.proceedings.spiiras.nw.ru

Достаточность.
Пусть $b_j = 1$ для всех $j \neq i$. То, что правило не является цепным, означает, что в грамматике отсутствует вывод $A_i \Rightarrow A_j \Rightarrow ... \Rightarrow A_k$. Доказательство от противного. Пусть в грамматике такой вывод присутствует. Тогда $\exists j,...,k$, такие что $A_i \Rightarrow A_j \Rightarrow ... \Rightarrow A_k$, A_k, A_m, …, A_j, $A_i \in N$. Тогда $A_k : \alpha_k \mid A_m$; $\alpha_k \in (N \cup T)^*$, …, $A_i : \alpha_i \mid A_j$; $\alpha_i \in (N \cup T)^*$ Тогда $b_k : \beta_k \wedge b_m$, …, $b_i : \beta_i \wedge b_j$ И тогда, положив $b_i = 0$, получим на первом проходе алгоритма для i $b_k = 0$, и так далее, потом $b_j = 0$, что противоречит условию. Следовательно, правило не цепное, что и требовалось доказать.
Теорема доказана.

Теорема 2.
Правило КС-грамматики не является цикличным тогда и только тогда, когда после завершения работы приведенного выше алгоритма для нахождения цепных правил ему будет сопоставлено значение $b_i = 1$.

Доказательство.
Необходимость.
То, что правило не является цикличным, означает, что в грамматике отсутствует вывод $A_i \Rightarrow A_j \Rightarrow ... \Rightarrow A_k$, где $i = k$. Доказательство от противного. Пусть $b_i = 0$. Тогда последняя операция – конъюнкция с параметром 0, то есть $\exists j,...,k$, такие что $b_k : \beta_k \wedge b_i$, …, $b_i : \beta_i \wedge b_j$ Тогда правила для A_k, …, A_j, $A_i \in N$: $A_k : \alpha_k \mid A_i; \alpha_k \in (N \cup T)^*$,…, $A_i : \alpha_i \mid A_j; \alpha_i \in (N \cup T)^*$. Но тогда существует вывод $A_i \Rightarrow A_j \Rightarrow ... \Rightarrow A_k \Rightarrow A_i$, $A_k,...A_j, A_i \in N$ и правило цикличное, а это не так, что и требовалось доказать.
Достаточность.
Пусть $b_j = 1$. То, что правило не является цикличным, означает, что в грамматике отсутствует вывод $A_i \Rightarrow A_j \Rightarrow ... \Rightarrow A_k \Rightarrow A_i$. Доказательство от противного. Пусть в грамматике такой вывод присутствует. Тогда $\exists j,...,k$, такие что $A_i \Rightarrow A_j \Rightarrow ... \Rightarrow A_k \Rightarrow A_i$, $A_k,...A_j, A_i \in N$. Тогда A_k, …, A_j, $A_i \in N$: $A_k : \alpha_k \mid A_i$; $\alpha_k \in (N \cup T)^*$, …, $A_i : \alpha_i \mid A_j$; $\alpha_i \in (N \cup T)^*$. Тогда $b_k : \beta_k \wedge b_i$, …, $b_i : \beta_i \wedge b_j$ И тогда, положив $b_i = 0$, получим на первом проходе алгоритма для i $b_k = 0$, и так далее, потом $b_j = 0$, а потом $b_i = 0$, что противоречит условию. Следовательно, правило не циклическое, что и требовалось доказать.
Теорема доказана.

5. Заключение

На базе данного алгоритма реализуется распознавание цепных и циклических правил в программном верификаторе грамматики. Это необходимо для преобразования грамматик перед разбором. К сожалению, алгоритм не исправляет цепные и циклические правила и здесь требуется участие человека.

Литература

1. *Hopcroft J. E., Ullman J. D.* Formal Languages and Their Relation to Automata. MA: Addison-Wesley Publishing Company, 1969. 242 p.