УДК 004.05

Н.Н. БЕНИДЗЕ

ЗАДАЧА ОЦЕНКИ ВЕРИЦИКАЦИИ (КОРРЕКТНОСТИ) АЛГОРИТМОВ И КОМПЬЮТЕРНЫХ ПРОГРАММ

Сухумский государственный университет,
ул. А. Политковская,12, Тбилиси, 380000,
Грузия, тел.: (99593) 24-67-43,
E-mail: **n_benidze@yahoo.com**

Аннотация. В статье исследуются вопросы, связанные с задачей установления правильности алгоритмов и соответствующих им компьютерных программ.В работе особое место занимает анализ связи между конечными автоматами, порождающими грамматиками и языками, который позволяет сформулировать принцип корректности алгоритмов и программ синтаксического анализа.

Abstract. In this article various questions are observed related with the problems regarding evaluation of correctness for algorithms and corresponding computer programs.
The research contains analysis of the relations between the final automats which are derived from grammars and languages. This analysis helps to formalize the principals of verification for the algorithms and the syntax analyzing programs.

Ключевые слова: конечный автомат, формальные языки, формальная грамматика, регулярные множество, принцип корректности алгоритмов, синтаксический анализ.

ВВЕДЕНИЕ

Одной из наиболее плодотворных по значению и влиянию на технологию программирования областей исследования является задача оценки корректности (верификации) алгоритмов и соответствующих им компьютерных программ.

Являясь альтернативой тестированию компьютерных программ, возможность их верификации существенным образом опирается на математическую спецификацию последних, которая часто не соответствует требованиям в точности, предъявляемым к программам.

Такая специфическая формализация открывает возможность строгого доказательства соответствия программы и математической спецификации. Однако главная трудность здесь заключается, в формулировке математически корректной версии неформальных требований к программе.

В настоящей работе сделана попытка формализации таких специфических требований к компьютерным программам, соответствующим алгоритмам синтаксического анализа.

ОСНОВНЫЕ ПОНЯТИЯ И ОПРЕДЕЛЕНИЯ. КОНЕЧНЫЕ АВТОМАТЫ

Мы будем рассматривать конечные автоматы, имеющие конечное число состоянии “внутренней” памяти и конечное число входных символов. Как только входной символ вводится в автомат, автомат переходит из одного состояния в другое, причем этот переход является детерминированным. Кроме того, во множестве состояний выделяется начальное состояние и множество заключительных состояний. Цепочка (конечная последовательность входных символов) допускается конечным автоматом, если начав работать в начальном состояний и прочтя всю цепочку входных символов, автомат переходить в одно из заключительных состояний. Таким образом, каждый конечный автомат функционирует как устройство, распознающее т.н. регулярное множество цепочек (оно допускает те и только те цепочки, которые принадлежать этому множеству).

Приведем теперь формальные определения понятий, о которых только что шла речь.

Определение1.1. Конечным автоматом называется упорядоченная пятерка:

$$A = (Q,\ \Sigma,\ \delta,\ q_0,\ F),$$

где Q – непустое конечное множество неструктурированных состоянии; $q_0 \in Q$ – начальное

состояние; $F \subseteq Q$ – непустое множество заключительных состоянии; Σ – непустое конечное множество входных символов; δ – функция, называемая функцией перехода:

$$\delta \ : \ Q \times \Sigma \to \Im(Q).$$

Определение1.2. Множество цепочек, допустимых конечным автоматом A, называется регулярным множеством, оно обозначается как $T(A)$ и задается следующим образом:

$$T(A) = \{ \ X \mid \ X \in \Sigma^{+} \ \wedge \ \hat{\delta}(q_0, X) \bigcap F = \varnothing ,$$

где $\hat{\delta}$ функция определяется рекурсивно как:

$$\hat{\delta}(q, a_1 a_2 ... a_n) = \bigcup_p \delta(p, a_n), \quad n \geq 2,$$

$$P \in \hat{\delta}(q, a_1 a_2 ... a_{n-1}) \ \text{ и } \ \hat{\delta}(q, a_1) = \delta(q, a_1) .$$

ГРАММАТИКИ И ФОРМАЛЬНЫЕ ЯЗЫКИ

Предположим, что дано конечное непустое множество Σ терминальных символов (терминалов). Предположим также, что N - непустое, конечное множество всех нетерминальных символов (нетерминалов), такое что $N \bigcap \Sigma = \varnothing$.

Обозначим через V множество всех терминальных и нетерминальных символов т. е. $V = \Sigma \bigcup N$ (V называется словарем формального языка), а через $V^{*} = (\Sigma \bigcup N)^{*}$ - множество всех возможных комбинации символов из множества V, включая цепочку нулевой длины ε. С помощью этих обозначении определим $\Im$ как множество пар

$$\Im = \{ \ (\xi, \beta) \ \mid \ \xi \in V^{*} - \varepsilon \times N \times V^{*} \wedge \ \beta \in V^{*} - \varepsilon \ \}.$$

Каждая упорядоченная пара (ξ, β) из множества $\Im$ называется порождающим правилом и обычно она записывается как $\xi \to \beta$.

Определение2.1. Формальная грамматика G - это четверка

$$G = (N, \ \Sigma, \ P, \ S), \tag{1}$$

где P непустое конечное подмножество $\Im$ (множество синтаксических правил), а $S \in N$ специальный символ, называемый начальным символом грамматики или аксиома языка.

Для определения языка $L(G)$, порожденным грамматикой G, введем обозначение Σ^{*} и следующие отношения:

- Σ^{*} - множество всех предложений конечной длины, полученных из терминалов.
- Отношение $\underset{G}{\Rightarrow}$ на множестве V^{*} определяется так:

  $\gamma_1 \underset{G}{\Rightarrow} \gamma_2$ (где $\gamma_1, \gamma_2 \in V^{+}$), если существуют $\delta_1, \delta_2 \in V^{*}$ и порождающее правило $\xi \to \beta \in P$ такие, что

$$\gamma_1 = \delta_1 \ \xi \ \delta_2 \quad \wedge \quad \gamma_2 = \delta_1 \ \beta \ \delta_2 . \tag{2}$$

- Отношение $\overset{*}{\underset{G}{\Rightarrow}}$ на множестве V^{*} определяется так:

$$\gamma_0 \underset{G}{\overset{*}{\Rightarrow}} \gamma_n, \text{ если } \gamma_0,\gamma_1,...,\gamma_n \in V^* \quad (n \geq 0) \text{ и } \gamma_{i-1} \underset{G}{\Rightarrow} \gamma_i \qquad (i = 1,2,...,n). \tag{3}$$

Упорядоченная последовательность $\gamma_0,\gamma_1,...,\gamma_n$ называется выводом длины n.

Определение2.2. Языком $L(G)$, порожденным грамматикой G, называется следующее подмножество Σ^*:

$$L(G) = \left\{ \quad x \mid S \underset{G}{\overset{*}{\Rightarrow}} x \ \wedge \ x \in \Sigma^* \ \wedge \ S \in N \ \right\}. \tag{4}$$

СВЯЗЬ МЕЖДУ КОНЕЧНЫМИ АВТОМАТАМИ, ФОРМАЛЬНЫМИ ГРАММАТИКАМИ И ЯЗЫКАМИ

В [1] приведены доказательства следующих двух теорем :

Теорема 1. Если $A = (Q,\Sigma,\delta,q_0,F)$ конечный автомат, то существует праволинейнная грамматика $G = (N,\Sigma',P,S)$ такая, что

$$L(G) = T(A).$$

Теорема 2. Если $G = (N,\Sigma,P,S)$ праволинейнная грамматика, то существует конечный автомат $A = (Q,\Sigma',\delta,q_0,F)$ такой, что

$$T(A) = L(G).$$

Опираясь на эти теоремы, можно доказать следующую теорему:

Теорема 3. Для любого алгоритма синтаксического анализа грамматики $G = (N,\Sigma,P,S)$ существует конечный автомат $A = (Q,\Sigma',\delta,q_0,F)$.

Доказательство: Учитывая, что $P \subset \Im$ алгоритм синтаксического анализа грамматики G является программной реализацией синтаксиса формального языка $L(G)$, т.е. множества

$$\Im = \left\{ \ (\xi,\beta) \ \mid \ \xi \in V^+ \times N \times V^* \wedge \ \beta \in V^+ \ \right\}.$$

В силу теоремы 2, для праволинейной грамматики $G = (N,\Sigma,P,S)$, существует конечный автомат $A = (Q,\Sigma',\delta,q_0,F)$, такой что

$$N = Q, \quad \Sigma = \Sigma', \quad S = q_0 \text{ и } L(G) = T(A).$$

Учитывая определения (2) - (4), множество правил вывода языка $L(G)$ совпадает с множеством P, т. е.

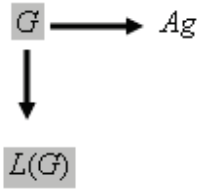
$$P = T(A),$$

ч. т. д.

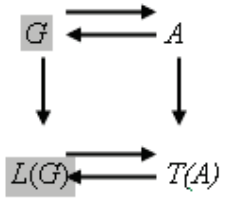
ПРИНЦИП КОРРЕКТНОСТИ АЛГОРИТМОВ СИНТАКСИЧЕСКОГО АНАЛИЗА

Обозначим через Ag алгоритм синтаксического анализа грамматики G, порождающая язык $L(G)$. Таким образом $G \to L(G)$ и $G \to Ag$.

Сложившуюся ситуацию удобно представить диаграммой отношении.



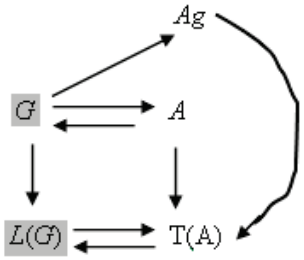
Аналогичным образом можно изобразить результаты теорем 1, 2 и 3



теоремы 1 и 2

теорема 3

Формальное объединение всех этих диаграмм отношений приводить к диаграмме



Объединенная диаграмма отношении позволяет сформировать принцип корректности алгоритмов синтаксического анализа следующим образом:

Принцип корректности алгоритмов синтаксического анализа заключается в следующем: алгоритм Ag синтаксического анализа грамматики G является корректным, если он записан на языке регулярных выражении.

СПИСОК ЛИТЕРАТУРЫ

1. Оллонгрен А. Определение языков программирования интерпретирующими автоматами / А. Оллонгрен. – М.: Мир, 1977.
2. Ахо А. Теория синтаксического анализа, перевода и компиляции / А. Ахо, Дж. Ульман. – Том 1. – М.: Мир, 1978.
3. Бенидзе Н. К вопросу о корректности программ / Н. Бенидзе // Internet Education Science (IES) -2008. New Informational and Computer Technologies in Education and Science. – Ukraine, Vinnytsia, VNTU, octomber 7-11 ,2008. – vol. 2 . – section H. – p.p. 545-550.

Надійшла до редакції 21.11.2010р.

БЕНИДЗЕ Н.Н. – докторант Грузинского университета им. Андрея Первозванного, ассистент – профессор Сухумского государственного университета, Тбилиси, Грузия.