Рязанов Ю.Д., аспирант
Белгородский государственный технологический университет им. В.Г. Шухова

УСТРАНЕНИЕ ЛЕВОЙ РЕКУРСИИ В СИНТАКСИЧЕСКИХ ДИАГРАММАХ

Ryazanov.iurij@yandex.ru

В статье рассматривается задача распознавания контекстно-свободных языков. Для построения эффективных распознавателей используются детерминированные синтаксические диаграммы. В работе рассматривается класс леворекурсивных синтаксических диаграмм. Даны определения самолеворекурсивной и леворекурсивной компоненты синтаксической диаграммы. Показано, что леворекурсивная диаграмма не является детерминированной и не может быть использована для построения эффективных распознавателей. Предложен алгоритм преобразования леворекурсивной синтаксической диаграммы в эквивалентную ей диаграмму без левой рекурсии, которая может быть детерминированной. Применение этого алгоритма расширяет класс синтаксических диаграмм, которые можно использовать для построения эффективных распознавателей формальных языков.

Ключевые слова: *контекстно-свободный язык, распознаватель, синтаксическая диаграмма, левая рекурсия, эквивалентные преобразования.*

Введение. Одним из наглядных способов задания синтаксиса языка являются синтаксические диаграммы (СД). Они применяются как для документирования языков программирования [1, 2], так и в проектировании программ обработки языков [3 – 7]. Основой таких программ является распознаватель языка. Для построения распознавателя линейной сложности [8, 9] используются детерминированные СД [8 – 11]. В множестве СД можно выделить класс леворекурсивных СД, которые не являются детерминированными и не могут быть использованы для построения эффективных распознавателей. В статье предложен алгоритм преобразования леворекурсивной синтаксической диаграммы в эквивалентную ей диаграмму без левой рекурсии. Применение этого алгоритма расширяет класс СД, которые можно использовать для построения эффективных распознавателей языков.

Основные понятия. На рис. 1 приведен пример СД, которая представляет собой ориентированный несвязный граф. В ней заглавными буквами (*S*,*A*,*B*) обозначены нетерминалы, прописными (*a*,*b*,*c*) – терминалы.

Каждому нетерминалу соответствует связная компонента графа. Компонента именуется соответствующим нетерминалом, имеет только одну точку входа и одну точку выхода и конечное множество узлов, терминальных и нетерминальных вершин. Точки входа и выхода на диаграмме компоненты не изображаются. Терминальная вершина изображается кружочком, в который вписан терминальный символ, нетерминальная – прямоугольником, в который вписан нетерминальный символ. Узел изображается на диаграмме жирной точкой. В точку входа не входит ни одна дуга и выходит конечное множество дуг (входные дуги компоненты). Узлы, в которые входят входные дуги, называются начальными. Из точки выхода не выходит ни одна дуга и входит конечное множество дуг (выходные дуги компоненты). Узлы, из которых выходят выходные дуги, называются заключительными. Дуги, соединяющие узлы, называются ε-дугами.

СД называется псевдодетерминированной (ПСД) [12], если в ней только один начальный узел и для каждого узла СД верно, что любые две дуги, выходящие из узла, идут в вершины, содержащие различные символы. В ПСД нет ε-дуг. СД, изображенная на рис. 1, является ПСД. Любую СД можно преобразовать в эквивалентную ей псевдодетерминированную СД [12].

Пусть в компоненте A синтаксической диаграммы существует путь l от точки входа до точки выхода. Если начать движение по этому пути и, проходя через терминальную или нетерминальную вершину, переписывать символ из вершины в некоторую изначально пустую цепочку α, то по окончании движения, когда придем в точку выхода, будет сформирована цепочка α, соответствующая пути l. Если путь l от точки входа до точки выхода не проходит ни через одну терминальную или нетерминальную вершину, то цепочка α пустая.

Пусть L_A — множество всех путей от точки входа до точки выхода в компоненте A. Определим функцию $F : L_A \to \alpha_A$, которая каждому пути $l \in L_A$ ставит в соответствие цепочку α по описанному выше правилу.

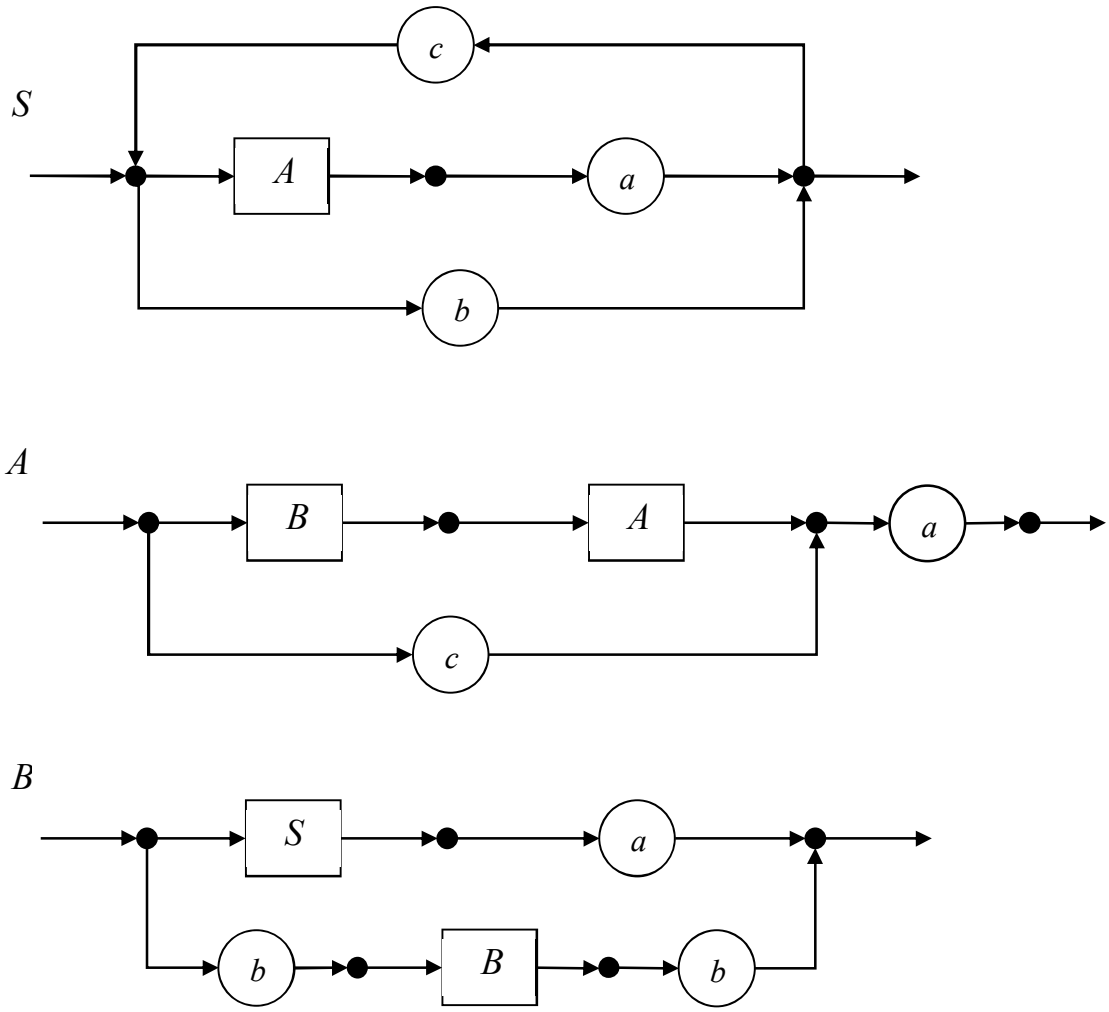

Рис. 1. Синтаксическая диаграмма

Выводимые из нетерминала S цепочки определим следующим образом:

1) цепочка S – выводимая;

2) если цепочка $\gamma A\beta$ выводимая, где γ и β – цепочки, состоящие из терминалов и нетерминалов, возможно пустые, а A – нетерминал, и некоторая цепочка $\alpha \in \alpha_A$, то цепочка $\gamma\alpha\beta$ выводимая. Цепочка $\gamma\alpha\beta$ получается из цепочки $\gamma A\beta$ путем замены нетерминала A на цепочку $\alpha \in \alpha_A$.

Цепочку языка можно получить, «двигаясь» по дугам СД от точки входа начальной компоненты к ее точке выхода. При этом если дуга идет в терминальную вершину, то вписанный в нее символ добавляем в цепочку, если дуга идет в нетерминальную вершину, то переходим в соответствующую компоненту и движемся по ней аналогичным образом до точки выхода, после чего возвращаемся в предыдущую компоненту и продолжаем движение. После прохождения выходной дуги начальной компоненты в цепочку добавляем концевой маркер.

Символ x, который может быть добавлен в цепочку после прохождения выходящей из узла дуги e, принадлежит множеству выбора дуги e (ВЫБОР(e)). Узел u называется детерминированным, если множества выбора любых двух дуг, выходящих из узла u, не пересекаются. СД является детерминированной, если в ней все узлы детерминированные. В работах [8, 11] описаны алгоритмы вычисления множеств выбора дуг, выходящих из узлов, и определения принадлежности СД классу детерминированных СД.

Самолеворекурсивные компоненты синтаксических диаграмм. Компоненту A ПСД, в которой из начального узла одна из дуг идет в нетерминальную вершину с нетерминалом A, назовем самолеворекурсивной. Нетерминальную вершину с нетерминалом A, в которую идет дуга из начального узла, назовем леворекурсивной. Если считать компоненту A начальной в СД и в процессе левого вывода использовать только эту компоненту, то в ней можно вывести цепочки терминалов и нетерминалов вида $\beta\alpha^*$, где β – цепочка терминалов и нетерминалов, соответствующая пути от начального узла до заключи-

тельного, не проходящая через леворекурсивную вершину, а α* – последовательность цепочек терминалов и нетерминалов, возможно пустая, соответствующих путям от узла, в который входит дуга из леворекурсивной вершины, до заключительного узла. Точно такие же цепочки будем получать, если из компоненты A исключим леворекурсивную вершину, а из заключительных узлов проведем ε-дуги в узел, в который шла дуга из леворекурсивной вершины. Таким образом рекурсия в самолеворекурсивной компоненте заменяется итерацией.

На рис. 2 изображена самолеворекурсивная компонента ПСД. Устраняя левую рекурсию в этой компоненте по описанному выше правилу, получим компоненту, представленную на рис. 3.

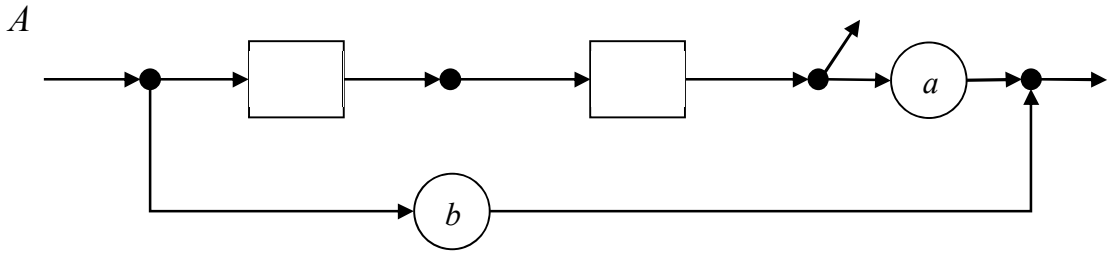

Рис. 2. Самолеворекурсивная компонента

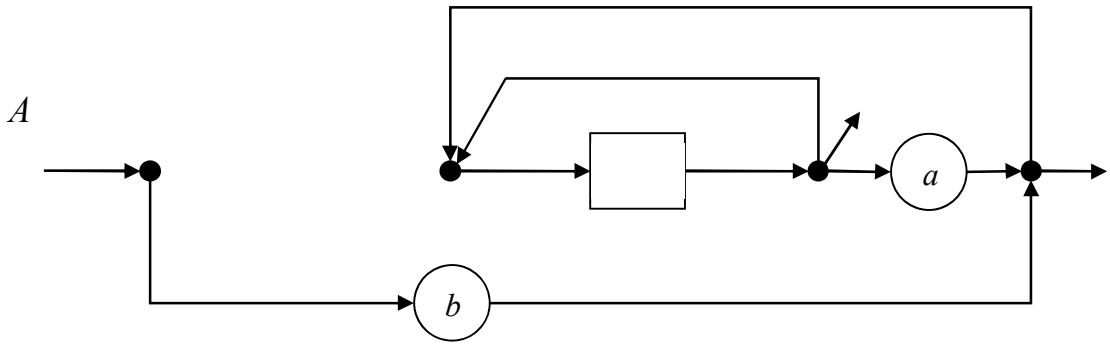

Рис. 3. Компонента без левой рекурсии

Компонента на рис. 3 не является псевдодетерминированной, так как содержит ε-дуги. Преобразуем ее в псевдодетерминированную (рис. 4), используя алгоритм, описанный в работе [12].

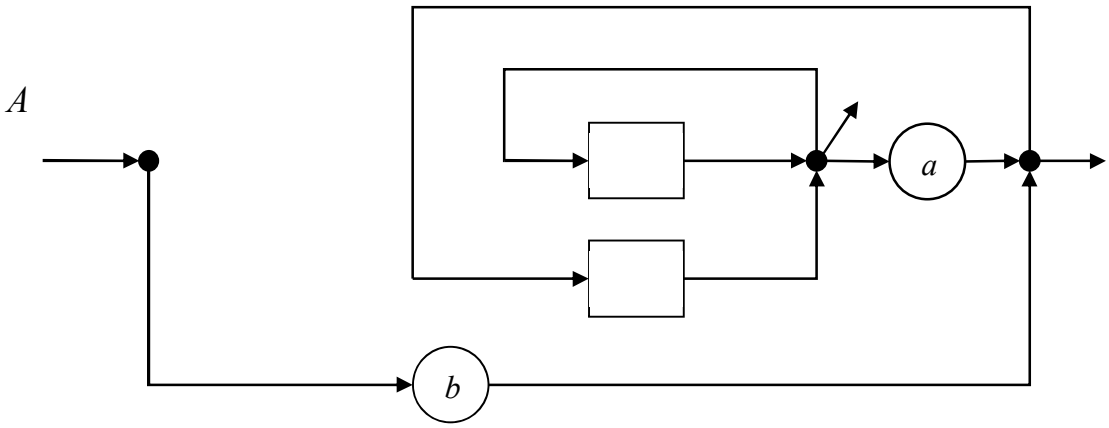

Рис. 4. Псевдодетерминированная компонента без левой рекурсии

Леворекурсивные компоненты синтаксических диаграмм. Компоненту A назовем леворекурсивной, если существует выводимая из A цепочка $A\alpha$, т. е. из A выводится цепочка, начинающаяся с нетерминала A. Очевидно, что самолеворекурсивная компонента является леворекурсивной. Для определения, является ли несамолеворекурсивная компонента A леворекурсивной, введем множество M^A нетерминалов, с которых начинается хотя бы одна цепочка, вы-

водимая из A. Такое множество можно получить следующим образом:

1. $M^A := \varnothing$.

2. Если из начального узла компоненты A дуга идет в нетерминальную вершину с нетерминалом X, то X добавить в множество M^A.

3. Если $X \in M^A$ и из начального узла компоненты X дуга идет в нетерминальную вершину с нетерминалом Y, то Y добавить в множество M^A.

4. Повторять п.3, пока множество M^A растет.

Если $A \in M^A$, то компонента A леворекурсивная.

СД, содержащую хотя бы одну леворекурсивную компоненту, назовем леворекурсивной.

Пусть из начального узла леворекурсивной компоненты A дуга идет в вершину с нетерминалом X. Если множество M^X нетерминалов, с которых начинается хотя бы одна цепочка, выводимая из X, содержит нетерминал A, то такую дугу назовем леворекурсивной. Дугу, выходящую из начального узла, не являющуюся леворекурсивной, назовем нелеворекурсивной.

Покажем, что множества выбора леворекурсивной дуги и нелеворекурсивной дуги пересекаются.

Очевидно, что множество выбора дуги, входящей в нетерминальную вершину с нетерминалом X равно объединению множеств выбора дуг, выходящих из начального узла компоненты X.

Рассмотрим обязательно существующую в леворекурсивной СД последовательность дуг e_1, e_2,…, e_k, такую, что

e_1 – леворекурсивная дуга, идущая из начального узла компоненты A, в вершину с нетерминалом X_1;

e_2 – дуга, идущая из начального узла компоненты X_1, в вершину с нетерминалом X_2;

e_k – дуга, идущая из начального узла компоненты X_{k-1}, в вершину с нетерминалом A.

Пусть e – нелеворекурсивная дуга компоненты A, тогда

ВЫБОР(e) $\subseteq$ ВЫБОР(e_k) $\subseteq$ … $\subseteq$ ВЫБОР(e_2) $\subseteq$ ВЫБОР(e_1), т. е.

ВЫБОР(e) $\cap$ ВЫБОР(e_1) $\neq \varnothing$.

Следовательно, начальный узел леворекурсивной компоненты не является детерминированным и леворекурсивная СД так же не является детерминированной.

Устранение левой рекурсии в синтаксических диаграммах. Выше показано, что леворекурсивная СД не является детерминированной, поэтому, для преобразования СД в детерминированную, нужно, как минимум, устранить левую рекурсию.

Алгоритм устранения левой рекурсии следующий.

1. $K_1 := \varnothing$ – множество обработанных компонент (нетерминалов);

$K_2 := N$ – множество необработанных компонент (в начале алгоритма все компоненты (нетерминалы) необработанны).

2. Пока $K_2 \neq \varnothing$, выполнять п. 3.

3. Обработать компоненту A, $A \in K_2$ по следующему алгоритму:

3.1. Если в компоненте A существует дуга, идущая из начального узла в вершину с нетерминалом X, $X \in K_1$ и $A \in M^X$, то заменить вершину с нетерминалом X компонентой X, иначе выполнить п. 3.3.

3.2. Полученную в п. 3.1 компоненту A преобразовать в псевдодетерминированную и выполнить п. 3.1.

3.3. Если компонента A самолеворекурсивная, то устранить в ней левую рекурсию.

3.4. $K_2 := K_2 \setminus \{A\}$, $K_1 := K_1 \cup \{A\}$, т. е. исключить компоненту A из множества необработанных компонент и включить ее в множество обработанных компонент.

4. Конец алгоритма.

Рассмотрим устранение левой рекурсии в СД, представленной на рис. 1.

В этой диаграмме все компоненты леворекурсивные, т. к. $M^S = \{A, B, S\}$ и $S \in M^S$, $M^A = \{B, S, A\}$ и $A \in M^A$, $M^B = \{S, A, B\}$ и $B \in M^B$.

В исходном состоянии $K_1 = \varnothing$, $K_2 = \{S, A, B\}$.

Множество K_2 не пусто, берем из него компоненту S. В компоненте S нет дуги, идущей из начального узла в вершину с нетерминалом, принадлежащим множеству K_1, т. к. K_1 — пусто. Компоненту S исключаем из множества K_2 и включаем в множество K_1: $K_1 = \{S\}$, $K_2 = \{A, B\}$.

Из множества K_2 берем компоненту A. В ней нет дуги, идущей из начального узла в вершину с нетерминалом, принадлежащим множеству K_1. Компоненту A исключаем из множества K_2 и включаем в множество K_1: $K_1 = \{S, A\}$, $K_2 = \{B\}$.

Из множества K_2 берем компоненту B. В ней есть дуга, идущая из начального узла в вершину с нетерминалом S, принадлежащим множеству K_1. Множество $M^S = \{A, B, S\}$, оно содержит компоненту B. Заменяем в компоненте B вершину с нетерминалом S компонентой S (рис. 5). Полученную компоненту преобразуем в псевдодетерминированную (рис. 6).

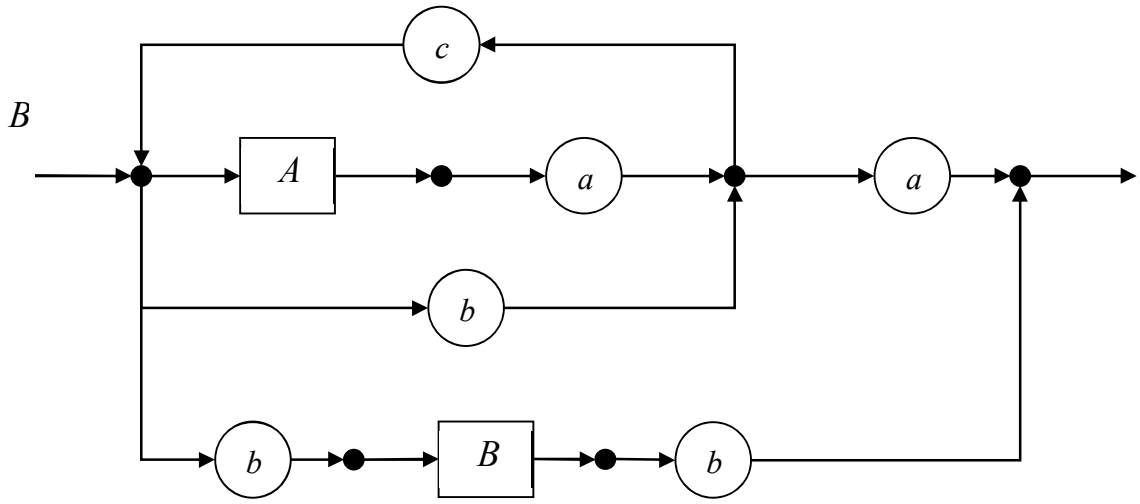

Рис. 5. Компонента *B*

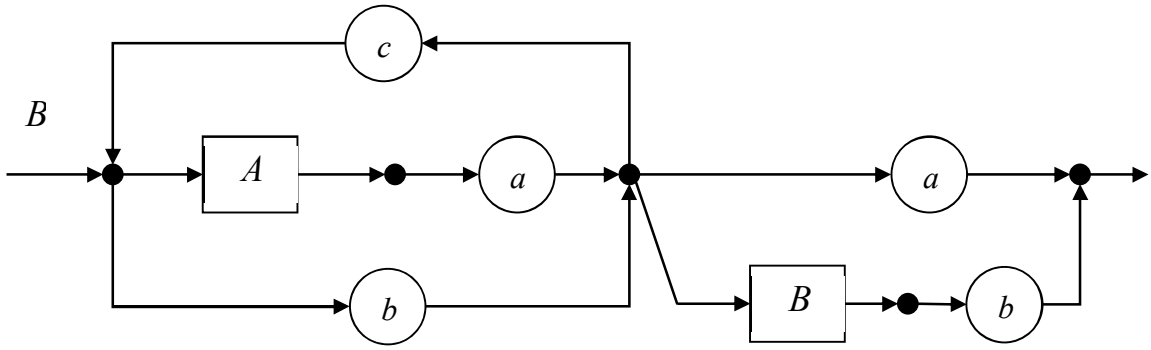

Рис. 6. Псевдодетерминированная компонента *B*

В этой компоненте (рис. 6) есть дуга, идущая из начального узла в вершину с нетерминалом *A*, принадлежащим множеству K_1. Множество $M^A = \{B, S, A\}$, оно содержит компоненту *B*. Заменяем в компоненте *B* (рис. 6) вершину с нетерминалом *A* компонентой *A* (рис. 7).

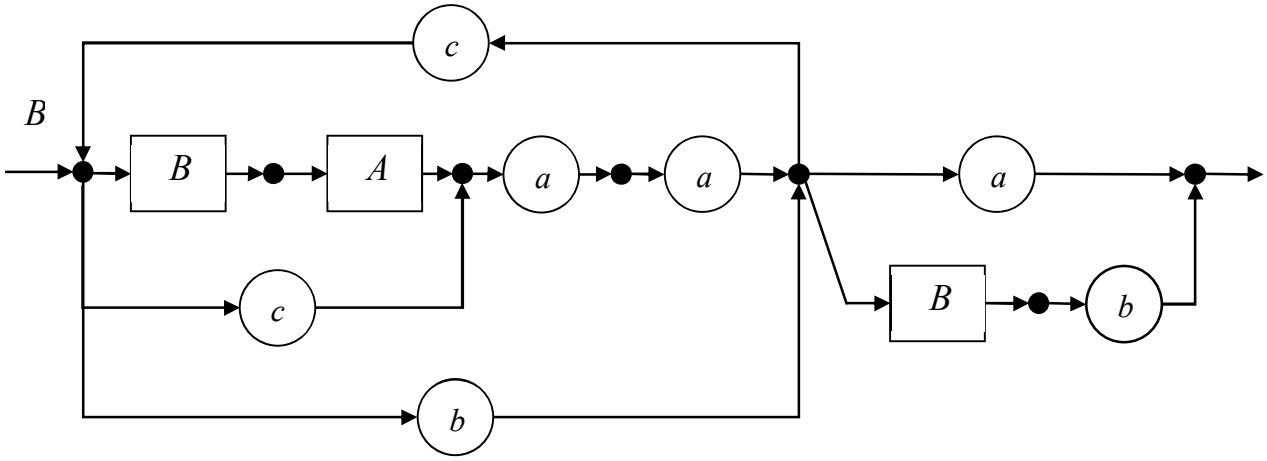

Рис. 7. Самолеворекурсивная компонента *B*

Компонента *B* (рис. 7) самолеворекурсивная. Устраняя левую рекурсию, получаем компоненту *B* (рис. 8).

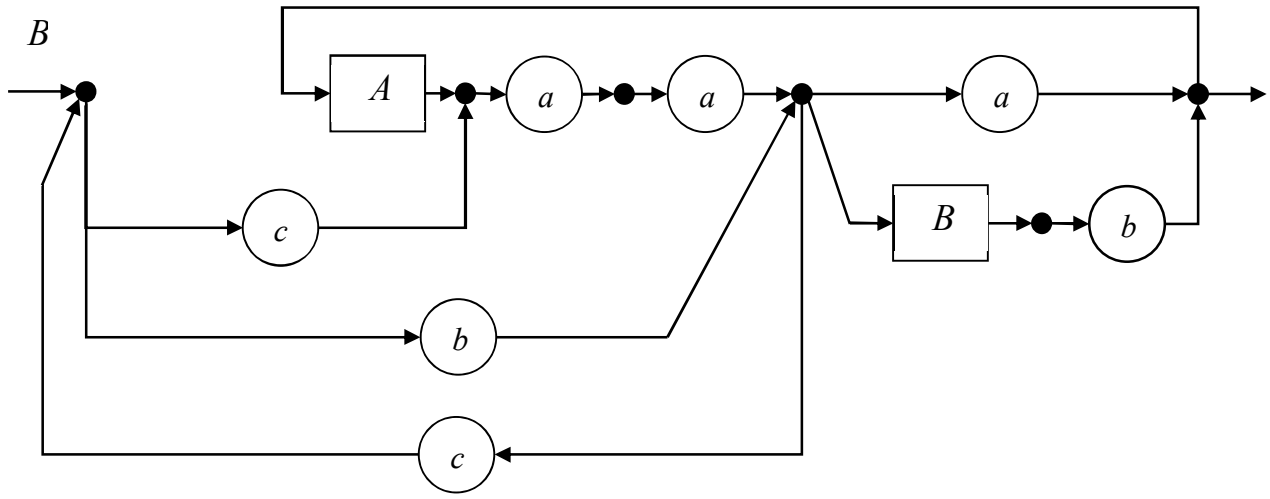

Рис. 8. Компонента *B* без левой рекурсии

В этой компоненте нет дуги, идущей из начального узла в вершину с нетерминалом, поэтому компоненту *B* считаем обработанной, исключаем ее из множества K_2 и включаем в множество K_1: $K_1 = \{S, A, B\}$, $K_2 = \varnothing$.

Множество K_2 пусто, СД без левой рекурсии получена. В этой СД действительно нет ни самолеворекурсивных компонент, ни леворекурсивных, т. к. $M^S = \{A, B\}$ и $S \notin M^S$, $M^A = \{B\}$ и $A \notin M^A$, $M^B = \varnothing$.

Выводы. В статье дано определение самолеворекурсивной и леворекурсивной компоненты СД, предложен алгоритм, определяющий принадлежность СД классу леворекурсивных СД, показано, что леворекурсивные СД не являются детерминированными и не могут быть использованы для построения эффективных распознавателей. Предложен алгоритм преобразования леворекурсивной синтаксической диаграммы в эквивалентную ей диаграмму без левой рекурсии. Это преобразование является обязательным в процессе преобразования недетерминированной СД в детерминированную и построения эффективных распознавателей языков.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Jensen K., Wirth N. Pascal User Manual and Report, Springer-Verlag, New York, 1975. p. 167.

2. Jensen K., Wirth N. Pascal Standard Iso – Jackson Libri, 1996. p.290.

3. Легалов А.И. Основы разработки трансляторов. URL: http://www.softcraft.ru/translat (дата обращения 22.01.2016).

4. Легалов А.И., Швец Д.А., Легалов И.А. Формальные языки и трансляторы. Красноярск: Сибирский федеральный университет. 2007. 213 с.

5. Карпов Ю.Г. Теория и технология программирования. Основы построения трансляторов. СПб.: БХВ-Петербург, 2005. 272 с.

6. Свердлов С.З. Языки программирования и методы трансляции. СПб.: Питер, 2007. 638 с.

7. Мартыненко Б.К. Синтаксические диаграммы Н. Вирта и граф-схемы в Syntax-технологии // Компьютерные инструменты в образовании. 2014. № 2. С. 3–19.

8. Рязанов Ю.Д., Севальнева М.Н. Анализ синтаксических диаграмм и синтез программ-распознавателей линейной сложности // Научные ведомости БелГУ. Сер. История. Политология. Экономика. Информатика. 2013, № 8 (151). Вып. 26/1. С. 128–136.

9. Поляков В.М., Рязанов Ю.Д. Алгоритм построения нерекурсивных программ-распознавателей линейной сложности по детерминированным синтаксическим диаграммам // Вестник БГТУ им. В.Г. Шухова. 2013. № 6, С. 194–199.

10. Рязанов Ю.Д. Преобразование недетерминированных синтаксических диаграмм в детерминированные // Вестник ВГУ, серия: системный анализ и информационные технологии, 2015. № 1. С. 139–147.

11. Рязанов Ю.Д., Крамаренко П.В. Графовый способ анализа синтаксических диаграмм // Научный электронный архив. URL: http://econf.rae.ru/article/8214 (дата обращения: 20.01.2016)

12. Рязанов Ю.Д., Севальнева М.Н. Псевдодетерминированные синтаксические диаграммы // Прикладная математика, управление и информатика: сборник трудов Междунар. молодеж. конф., Белгород, 3–5 октября 2012 г. : в 2 т., Белгород : ИД «Белгород». 2012, Т2. С. 546–553.

Ryazanov Yu. D.

ELIMINATION OF LEFT RECURSION IN SYNTAX DIAGRAMMS

In this article the problem of the recognition of context-free languages is considered. The deterministic syntax diagrams are used for constructing effective recognizers. The class of left-recursive syntax diagrams is being considered in this article. The components of syntax diagrams with direct and indirect left recursion are defines in this paper. It is shown that left-recursive diagram is not deterministic and can not be used to construction effective recognizers. The author suggest the algorithm for converting left-recursive syntax diagram into equivalent diagram without left recursion, which can be deterministic. This algorithm expanded the class of syntax diagrams, which can be used for constructing effective recognizers.

***Key words**: context-free language, recognizer, syntax diagram , left recursion , equivalent transforming.*

Рязанов Юрий Дмитриевич, аспирант кафедра программного обеспечения вычислительной техники и автоматизированных систем
Белгородский государственный технологический университет им. В.Г. Шухова
Адрес: Россия, 308012, Белгород, ул. Костюкова, д. 46
E-mail: Ryazanov.iurij@yandex.ru