МАТЕМАТИЧЕСКИЕ ОСНОВЫ ИНФОРМАТИКИ И ПРОГРАММИРОВАНИЯ

УДК 519.766.2

МИНИМИЗАЦИЯ СИНТАКСИЧЕСКИХ ДИАГРАММ С МНОГОВХОДОВЫМИ КОМПОНЕНТАМИ[1]

Ю. Д. Рязанов

Белгородский государственный технологический университет им. В. Г. Шухова, г. Белгород, Россия

Рассмотрена задача минимизации синтаксических диаграмм. Для её решения диаграммы Вирта (ДВ) преобразуются в синтаксические диаграммы с многовходовыми компонентами (СД), которые по структуре совпадают с ДВ, но отличаются тем, что нетерминалы в нетерминальных вершинах заменяются начальными узлами соответствующих компонент. На множестве узлов СД вводится отношение, обладающее свойством эквивалентности, которое разбивает множество узлов на классы эквивалентности. Доказано, что «стягивание» класса эквивалентности в один узел является эквивалентным преобразованием. Если классу эквивалентности принадлежат узлы различных компонент, то в результате «стягивания» происходит соединение компонент в одну, которая имеет несколько входов. Предложены алгоритмы разбиения множества узлов на классы эквивалентности и построения СД. Приводится пример, показывающий, что построенная по предложенным алгоритмам СД значительно меньше эквивалентной ей ДВ.

Ключевые слова: *формальный язык, синтаксическая диаграмма, отношение эквивалентности, минимизация.*

DOI 10.17223/20710410/41/9

MINIMIZATION OF SYNTAX DIAGRAMS WITH MULTIPORT COMPONENTS

Yu. D. Ryazanov

Belgorod State Technological University named after V. G. Shukhov, Belgorod, Russia

E-mail: razanov.yd@bstu.ru

The minimization problem for syntax diagrams is considered. For this purpose, we transform the Wirth diagrams (WD) into syntax diagrams with multiport components (SD), which are similar to WD in their structure, but differ in the fact that the non-terminals in nonterminal nodes are replaced with the starting nodes of the corresponding components. On the set of SD nodes, we introduce a relation which possesses the equivalence property, dividing the set of nodes into equivalence classes. We prove that uniting an equivalence class into one node is an equivalence transformation. If an equivalence class includes the nodes of various components, then,

[1]Работа поддержана грантом РФФИ № 16-07-00487.

as a result of uniting the class into one node, the components are united into one component, which has several inputs. The algorithms for dividing the set of nodes into equivalence classes and plotting a SD are suggested. The algorithm for dividing the set of nodes into equivalence classes is based on a serial partitioning of the set of nodes into subsets so that non-equivalent nodes fall into different subsets. After partitioning the set of nodes into equivalence classes, an SD is constructed. In the SD construction algorithm, for each equivalence class, the following actions are executed: only one node from the class is left in the SD, the remaining nodes of the class are deleted, and if some arc in the source diagram is going to the deleted node, then it is redirected to one of remaining nodes. We give an example which demonstrates that a SD plotted by the suggested algorithms is considerably smaller than the equivalent WD. The resulting SD, after minimization process, can be used to construct memory efficient programs for formal languages processing.

Keywords: *formal language, syntax diagram, equivalence relation, minimization.*

Введение

Синтаксические диаграммы Вирта представляют собой наглядный, интуитивно понятный графический способ задания синтаксиса языка, они используются для документирования языков программирования [1, 2] и в проектировании трансляторов [3 – 7]. Для построения трансляторов линейной сложности применяются детерминированные синтаксические диаграммы [8, 9], при этом размер транслятора зависит от размера синтаксической диаграммы. Поэтому с целью уменьшения размера транслятора целесообразно использовать «компактные» синтаксические диаграммы.

Характерной особенностью классических синтаксических диаграмм Вирта является то, что каждая компонента имеет ровно один вход и один выход. Эти ограничения позволяют легко строить наглядные и понятные диаграммы для задания контекстно-свободных языков.

В работе рассматриваются синтаксические диаграммы, в которых снято ограничение на количество входов в компонентах. Это позволяет сократить количество компонент и узлов диаграммы. Для сокращения количества компонент и узлов вводится отношение на множестве узлов, обладающее свойством эквивалентности, и предлагается алгоритм, позволяющий разбить множество узлов на классы эквивалентности и на основе этого преобразовать исходную диаграмму в более компактную.

Платой за компактность представления языка в виде синтаксической диаграммы с многовходовыми компонентами является некоторое снижение наглядности по сравнению с классическими диаграммами. Снижение наглядности представления языка в данном случае не является существенным, так как все преобразования могут быть выполнены автоматически по описанным алгоритмам.

В первой части работы поясняются понятия, связанные с диаграммами Вирта (ДВ), необходимые для дальнейшего изложения. Во второй части даётся определение синтаксической диаграммы с многовходовыми компонентами (СД) и способ преобразования ДВ в эквивалентную ей «вырожденную» СД. Далее определяются отношение на множестве узлов СД, названное отношением сильной эквивалентности; преобразование СД, сохраняющее сильную эквивалентность; алгоритм разбиения множества узлов на классы эквивалентности и сокращения СД. Все алгоритмы сопровождаются примерами их применения.

1. Синтаксические диаграммы Вирта

На рис. 1 представлен пример синтаксической диаграммы Вирта. Эта ДВ состоит

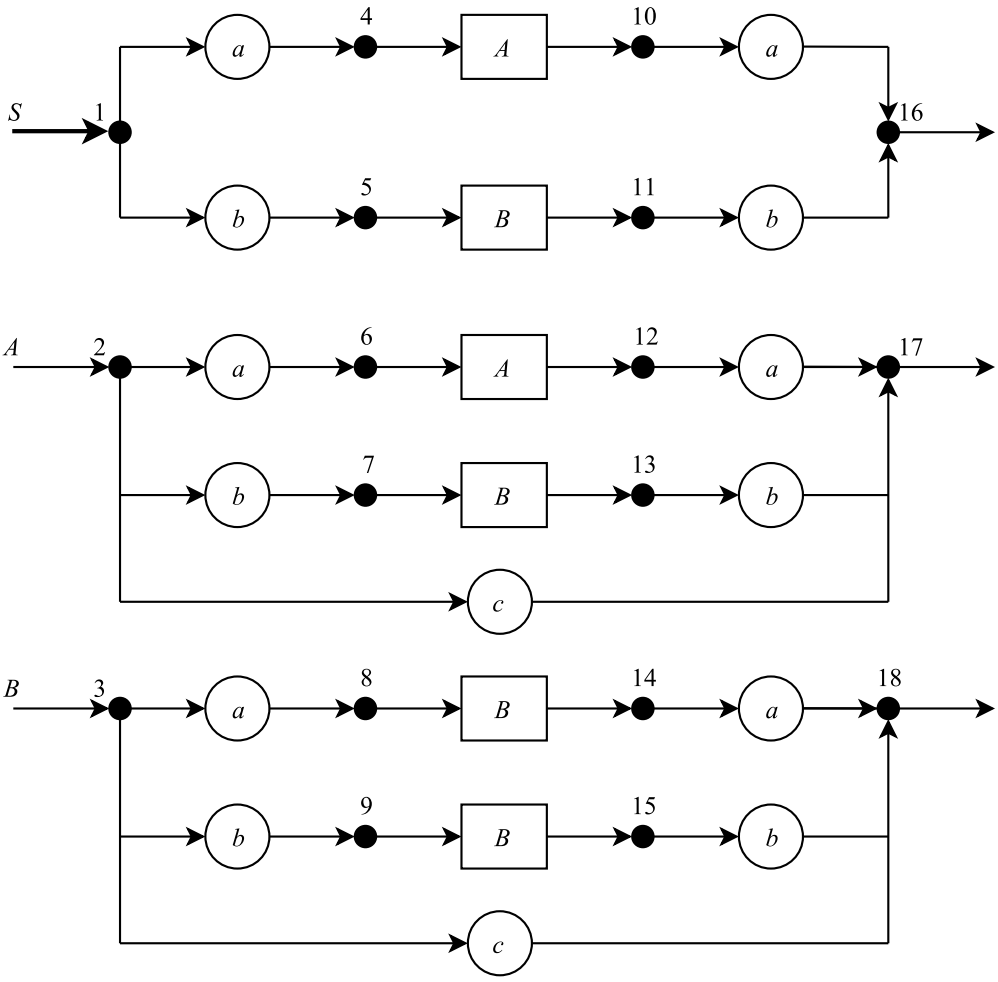

Рис. 1. Синтаксическая диаграмма Вирта

из трёх компонент, соответствующих нетерминалам S, A и B (S — начальный нетерминал). Прямоугольниками изображены нетерминальные вершины, в которые вписаны нетерминалы, кружками — терминальные вершины, в которые вписаны терминалы, жирными точками — узлы, обозначенные натуральными числами. Множество дуг разбивается на четыре подмножества:

1) входящие дуги, которые входят в узлы из одной точки входа;
2) выходящие дуги, которые выходят из узлов и входят в одну точку выхода;
3) внутренние дуги, которые связывают либо узлы с терминальными или нетерминальными вершинами, либо терминальные или нетерминальные вершины с узлами;
4) ε-дуги, связывающие между собой узлы (в примере на рис. 1 таких дуг нет).

Точки входа и выхода на диаграмме не показываются. Узлы, в которые входят входящие дуги, называются начальными (в примере на рис. 1 начальными являются узлы 1, 2 и 3), а узлы, из которых выходят выходящие дуги, — заключительными (в примере на рис. 1 заключительными являются узлы 16, 17 и 18).

С помощью ДВ можно получить (вывести) любую цепочку языка, заданного этой ДВ. Рассмотрим подробно способы получения цепочки языка.

Пусть ДВ G состоит из компонент $G_1, G_2, \ldots, G_{|N|}$, где $|N|$ — мощность множества нетерминалов. Рассмотрим некоторой путь в компоненте G_i от узла u до какого-либо заключительного узла в этой компоненте. Пройдём по нему и в процессе прохождения будем добавлять в изначально пустую цепочку α символы, записанные в терминальных и нетерминальных вершинах. Цепочку α будем называть цепочкой, связывающей узел u с заключительным узлом компоненты G_i. Множество всех цепочек, связывающих узел u с каким-либо заключительным узлом, образует язык $L(u)$. Очевидно, что язык $L(G_i)$ равен объединению всех $L(u_j)$, где u_j — начальный узел компоненты G_i. Вывод цепочки языка, заданного ДВ, можно выполнить по следующему правилу:

1. Записать цепочку, принадлежащую языку $L(G_1)$, где компонента G_1 соответствует начальному нетерминалу.
2. Если цепочка содержит некоторый нетерминал N_i, то заменить его цепочкой языка $L(G_i)$ и выполнить п. 2, иначе — вывод закончить.

Ниже представлен вывод цепочки в ДВ, изображенной на рис. 1:

$$S \Rightarrow aAa \Rightarrow abBba \Rightarrow abcba.$$

Процесс вывода терминальной цепочки в ДВ можно представить и по-другому — как «движение» по дугам от точки входа начальной компоненты к точке выхода. При этом если дуга идёт в терминальную вершину, то вписанный в неё символ добавляем в терминальную цепочку; если в нетерминальную, 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 терминальную цепочку добавляем концевой маркер ($\dashv$) и вывод заканчивается.

Символ x, который может быть добавлен в терминальную цепочку непосредственно после прохождения дуги e, принадлежит множеству выбора дуги e ($x \in$ ВЫБОР(e)).

В множестве ДВ можно выделить класс детерминированных ДВ. В детерминированной ДВ каждая компонента имеет только один начальный узел, не содержит ε-дуг и каждый её узел детерминированный. Узел является детерминированным, если множества выбора любых двух дуг, выходящих из него, не пересекаются. Детерминированные ДВ являются основой для построения программ-распознавателей линейной сложности [8, 9], поэтому в дальнейшем будем рассматривать только детерминированные ДВ.

2. Синтаксические диаграммы с многовходовыми компонентами

Синтаксическую диаграмму с многовходовыми компонентами (СД) определим восьмеркой $R = (T, U, U', U'', u_0, G, F_T, F_U)$, где

— T — конечное множество терминалов;
— U — конечное множество узлов;
— U' — конечное множество начальных узлов, $U' \subseteq U$;
— U'' — конечное множество заключительных узлов, $U'' \subseteq U$;
— u_0 — стартовый узел, $u_0 \in U'$;
— $G = (V, E)$ — ориентированный граф, $V = V_T \cup V_N \cup U$, V_T — множество терминальных вершин, V_N — множество нетерминальных вершин; $E = E_1 \cup E_2$, $E_1 \subseteq U \times (V_T \cup V_N)$ — множество дуг, выходящих из узлов и входящих в терминальные или нетерминальные вершины; $E_2 \subseteq (V_T \cup V_N) \times U$ — множество дуг, выходящих из терминальных или нетерминальных вершин и входящих в узлы;

— $F_T : V_T \to T$ — отображение множества терминальных вершин в множество терминалов;
— $F_U : V_N \to U'$ — отображение множества нетерминальных вершин в множество начальных узлов.

Терминальная вершина изображается на диаграмме кружком, в который вписан терминал в соответствии с отображением F_T. Нетерминальная вершина изображается прямоугольником, в который вписан узел из множества начальных узлов в соответствии с отображением F_U. Узел изображается жирной точкой, которая отмечается соответствующим номером. Начальные узлы отмечаются входящей стрелочкой, стартовый узел — жирной входящей стрелочкой, заключительные — выходящей стрелочкой. Дуга может выходить из терминальной или нетерминальной вершины и входить в узел, или выходить из узла и входить в терминальную или нетерминальную вершину. В каждую терминальную и нетерминальную вершину входит только одна дуга; из них выходит только одна дуга. На количество дуг, входящих в узлы и выходящих из узлов, ограничений нет.

Определим способ получения цепочки языка, заданного СД.

Будем говорить, что цепочка α, состоящая из терминалов и/или начальных узлов, связывает узел u СД с заключительным узлом u_k, если её можно получить, «двигаясь» в СД от узла u к узлу u_k и выписывая из вершин по пути символы (терминалы или начальные узлы) в изначально пустую цепочку. Множество всех цепочек, связывающих узел u с заключительными узлами, обозначим $L(u)$. Чтобы получить цепочку языка, заданного СД со стартовым узлом u_0, возьмём цепочку, принадлежащую $L(u_0)$. Если она содержит некоторый начальный узел u_i, заменим его на цепочку из множества $L(u_i)$. Если новая цепочка содержит начальный узел, то аналогичные действия повторяем. Полученная таким образом цепочка, не содержащая начальных узлов, принадлежит языку, заданному СД.

ДВ можно преобразовать в эквивалентную ей СД, заменив записанный в каждой нетерминальной вершине нетерминал на его начальный узел и определив начальный узел начальной компоненты (соответствующей начальному нетерминалу) стартовым. На рис. 2 представлена СД, полученная из ДВ (рис. 1).

Сравнивая способы получения цепочек языка по ДВ и СД, можно сделать вывод, что язык, заданный ДВ, равен языку, заданному СД, полученной из ДВ описанным способом.

Процесс получения одной из цепочек языка, заданного СД на рис. 2, можно представить следующим образом:

$$1 \Rightarrow a2a \Rightarrow ab3ba \Rightarrow abcba.$$

Такой способ преобразования позволяет получить «вырожденную» СД, в которой каждая компонента имеет один вход. Преобразования, описанные далее, позволят получить СД с многовходовыми компонентами и меньшим количеством компонент.

Для СД точно так же, как для ДВ, определяются понятия множество выбора дуги и детерминированность. В дальнейшем будем рассматривать СД, полученные из детерминированных ДВ, следовательно, СД тоже будут детерминированными.

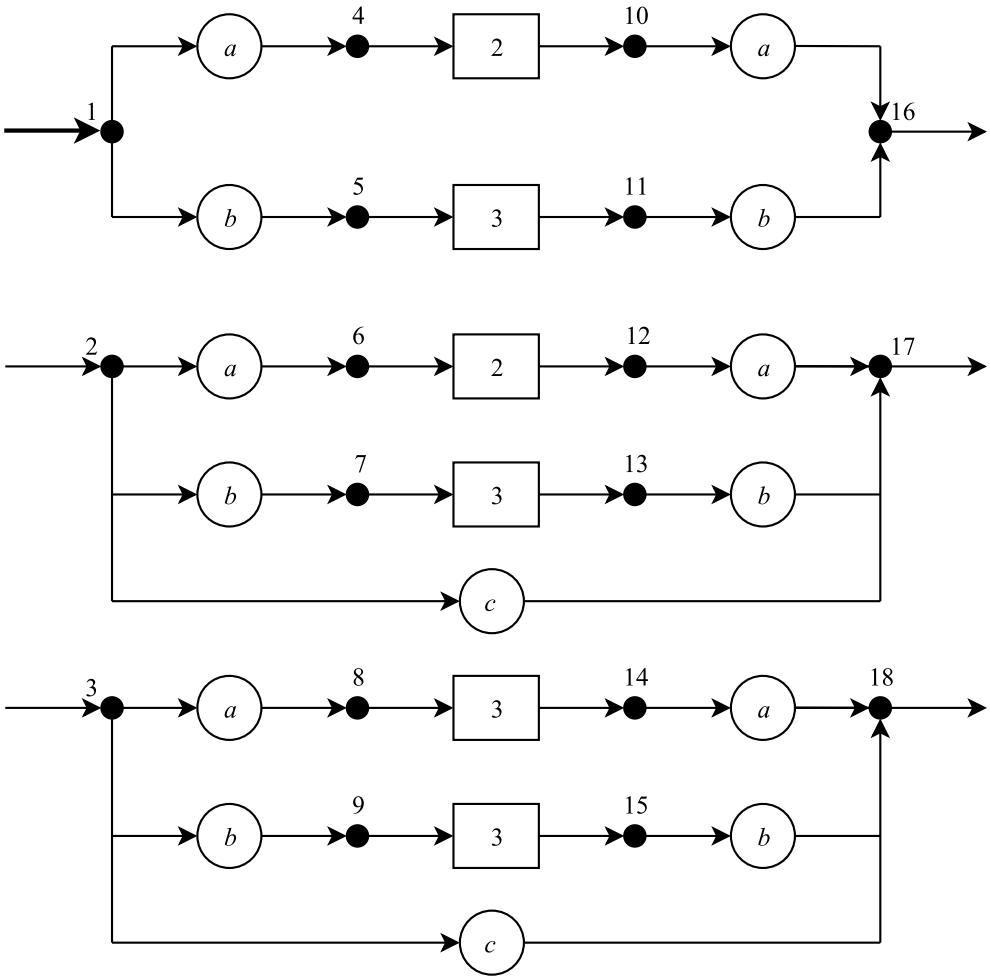

Рис. 2. Синтаксическая диаграмма с многовходовыми компонентами

3. Отношение сильной эквивалентности

Для определения отношения сильной эквивалентности на множестве узлов введём следующие обозначения:

1) (u_i, t, u_j) — указывает на то, что в СД существует путь длины два из узла u_i в узел u_j, дуга из узла u_i идёт в вершину с терминалом t, а из неё идёт дуга в узел u_j;
2) (u_i, M, u_k, u_j) — указывает на то, что в СД существует путь длины два из узла u_i в узел u_j, дуга из узла u_i с множеством выбора M идёт в вершину с узлом u_k, а из неё идёт дуга в узел u_j.

Пара узлов (u_i, u'_i) принадлежит отношению E°, если выполнены следующие условия:

1) путь (u_i, t, u_j) существует тогда и только тогда, когда существует путь (u'_i, t, u'_j) и $(u_i, u'_j) \in E^\circ$, где t принадлежит множеству терминалов;
2) путь (u_i, M, u_k, u_j) существует тогда и только тогда, когда существует путь (u'_i, M, u'_k, u'_j), $(u_j, u'_j) \in E^\circ$ и $(u_k, u'_k) \in E^\circ$, где u_k и u'_k принадлежат множеству начальных узлов;
3) $u_i \in U''$ тогда и только тогда, когда $u'_i \in U''$.

Из определения отношения E° следует, что цепочка α связывает узел u_i с заключительным узлом тогда и только тогда, когда цепочка α' связывает узел u'_i с заключительным узлом, где α и α' — цепочки одинаковой длины и i-й символ цепочки α либо равен i-му символу цепочки α' ($\alpha_i = \alpha'_i$), либо пара начальных узлов (α_i, α'_i) принадлежит отношению E°. Цепочки α и α' назовём сильно эквивалентными, а отношение E° — отношением сильной эквивалентности.

На множествах цепочек введём следующие операции сравнения:

1) $L_1 \subseteq^\circ L_2$ — истина, если для каждой цепочки множества L_1 существует сильно эквивалентная цепочка из множества L_2, иначе — ложь;
2) $L_1 =^\circ L_2$ тогда и только тогда, когда $L_1 \subseteq^\circ L_2$ и $L_2 \subseteq^\circ L_1$.

Операции $=^\circ$ и $\subseteq^\circ$ будем называть соответственно равенством и включением. Отношение E° можно теперь определить так:

$$E^\circ = \{(u_i, u'_i) : L(u_i) =^\circ L(u'_i)\}.$$

Очевидно, что

1) если $L_1 = L_2$, то $L_1 =^\circ L_2$;
2) если $L_1 \subseteq L_2$, то $L_1 \subseteq^\circ L_2$;
3) $L_1 \cup L_2 \subseteq^\circ L_3$ тогда и только тогда, когда $L_1 \subseteq^\circ L_3$ и $L_2 \subseteq^\circ L_3$.

Для множеств L_1, L_2, L_3 и L_4, таких, что $L_1 =^\circ L_3$ и $L_2 =^\circ L_4$, верно:

1) $L_1 \cup L_2 =^\circ L_3 \cup L_4$;
2) $L_1 L_2 =^\circ L_3 L_4$;
3) $L_1^* =^\circ L_3^*$.

Рассмотрим две СД $R_1 = (T, U^1, U'^1, U''^1, u_0^1, G^1, F_T^1, F_U^1)$ и $R_2 = (T, U^2, U'^2, U''^2, u_0^2, G^2, F_T^2, F_U^2)$, такие, что $(u_0^1, u_0^2) \in E^\circ$. Докажем, что $L(R_1) = L(R_2)$, т. е. эти СД определяют один и тот же язык.

Пусть на первом шаге вывода в СД R_1 получена промежуточная цепочка α, принадлежащая языку $L(u_0^1)$. Из $L(u_0^1) =^\circ L(u_0^2)$ следует, что на первом шаге вывода в СД R_2 можно получить промежуточную цепочку α', сильно эквивалентную цепочке α. Далее на каждом шаге вывода в СД R_1 и R_2 будем получать сильно эквивалентные цепочки вплоть до терминальной цепочки. Терминальные сильно эквивалентные цепочки равны.

Пусть на первом шаге вывода в СД R_2 получена промежуточная цепочка α, принадлежащая языку $L(u_0^2)$. Из $L(u_0^1) =^\circ L(u_0^2)$ следует, что на первом шаге вывода в СД R_1 можно получить промежуточную цепочку α', сильно эквивалентную цепочке α. Далее на каждом шаге вывода в СД R_1 и R_2 будем получать сильно эквивалентные цепочки вплоть до терминальной цепочки. Терминальные сильно эквивалентные цепочки равны.

В любом случае терминальные цепочки, которые можно получить при выводе в одной СД, можно получить и в другой СД.

4. Преобразование, сохраняющее сильную эквивалентность

Допустим, что над СД $R_1 = (T, U^1, U'^1, U''^1, u_0^1, G^1, F_T^1, F_U^1)$ выполнено преобразование и в результате получена СД $R_2 = (T, U^2, U'^2, U''^2, u_0^2, G^2, F_T^2, F_U^2)$, такая, что $(u_0^1, u_0^2) \in E^\circ$. В этом случае будем говорить, что над СД R_1 выполнено преобразование, сохраняющее сильную эквивалентность. Результат преобразования — СД R_2, которая эквивалентна СД R_1. Рассмотрим преобразование СД, сохраняющее сильную эквивалентность.

Пусть в СД $R_1 = (T, U, U'^1, U'', u_0^1, G^1, F_T, F_U)$: $U'^1 = \{u_0, u_1, \dots u_f, \dots, u_n\}$ — множество начальных узлов; $(u_i, u_i') \in E^\circ$; узлы u_i и u_i' не являются начальными. Преобразование заключается в том, что все дуги, входящие в узел u_i, перенаправляются в узел u_i'. В результате преобразования получается СД $R_2 = (T, U, U'^2, U'', u_0^2, G^2, F_T, F_U)$.

Докажем, что это преобразование сохраняет сильную эквивалентность. Из множества начальных узлов U'^1 выбираем произвольно узел u_f. Введём следующие обозначения:

1) $L_1(u_f, u_i)$ — множество цепочек, связывающих начальный узел u_f с узлом u_i в исходной СД;
2) $L_1(u_i, U'')$ — множество цепочек, связывающих узел u_i с заключительными узлами U'' в СД R_1;
3) $L_1(u_f, u_i, U'')$ — множество цепочек, связывающих начальный узел u_f с заключительными узлами U'', соответствующих путям, проходящим через узел u_i в СД R_1; $L_1(u_f, u_i, U'') = L_1(u_f, u_i)L_1(u_i, U'')$ — множество $L_1(u_f, u_i, U'')$ равно конкатенации множеств $L_1(u_f, u_i)$ и $L_1(u_i, U'')$;
4) $L_1(u_f, \overline{u_i}, U'')$ — множество цепочек, связывающих начальный узел u_f с заключительными узлами U'', соответствующих путям, не проходящим через узел u_i в СД R_1;
5) $L_1(u_f, U'')$ — множество цепочек, связывающих начальный узел u_f с заключительными узлами U'' в СД R_1; $L_1(u_f, U'') = L_1(u_f, u_i, U'') \cup L_1(u_f, \overline{u_i}, U'') = L_1(u_f, u_i)L_1(u_i, U'') \cup L_1(u_f, \overline{u_i}, U'')$;
6) $L_2(u_f, u_i')$ — множество цепочек, связывающих начальный узел u_f с узлом u_i' в СД R_2; $L_2(u_f, u_i') = L_1(u_f, u_i') \cup L_1(u_f, u_i)$;
7) $L_2(u_f, u_i', U'')$ — множество цепочек, связывающих начальный узел u_f с заключительными узлами U'', соответствующих путям, проходящим через узел u_i' в СД R_2;

$$L_2(u_f, u_i', U'') = L_2(u_f, u_i')L_1(u_i', U'') = (L_1(u_f, u_i') \cup L_1(u_f, u_i))L_1(u_i', U'') = \\ = L_1(u_f, u_i')L_1(u_i', U'') \cup L_1(u_f, u_i)L_1(u_i', U'');$$

8) $L_2(u_f, U'')$ — множество цепочек, связывающих начальный узел u_f с заключительными U'' в СД R_2;

$$L_2(u_f, U'') = L_2(u_f, u_i', U'') \cup L_1(u_f, \overline{u_i}, U'') = \\ = L_1(u_f, u_i')L_1(u_i', U'') \cup L_1(u_f, u_i)L_1(u_i', U'') \cup L_1(u_f, \overline{u_i}, U'').$$

Для доказательства того, что рассматриваемое преобразование сохраняет сильную эквивалентность, необходимо доказать, что $L_1(u_f, U'') =^\circ L_2(u_f, U'')$, т. е.

$$L_1(u_f, u_i)L_1(u_i, U'') \cup L_1(u_f, \overline{u_i}, U'') =^\circ \\ =^\circ L_1(u_f, u_i')L_1(u_i', U'') \cup L_1(u_f, u_i)L_1(u_i', U'') \cup L_1(u_f, \overline{u_i}, U'').$$

В свою очередь, $L_1(u_f, U'') =^\circ L_2(u_f, U'')$, если имеют место включения 1) $L_1(u_f, U'') \subseteq {}^\circ L_2(u_f, U'')$ и 2) $L_2(u_f, U'') \subseteq {}^\circ L_1(u_f, U'')$.

1. $L_1(u_f, U'') \subseteq^\circ L_2(u_f, U'')$, если

$$L_1(u_f, u_i)L_1(u_i, U'') \subseteq^\circ L_2(u_f, U'') \quad \text{и} \quad L_1(u_f, \overline{u_i}, U'') \subseteq {}^\circ L_2(uf, U'').$$

Истинность первого условия следует из того, что $L_1(u_f, u_i)L_1(u_i', U'') \subseteq^\circ L_2(u_f, U'')$ по определению и $L_1(u_i, U'') =^\circ L_1(u_i', U'')$, так как $(u_i, u_i') \in E^\circ$.

Истинность второго условия следует из того, что любая цепочка из множества $L_1(u_f, \overline{u_i}, U'')$ включена либо в $L_1(u_f, u'_i)L_1(u'_i, U'')$, либо в $L_1(u_f, \overline{u_i}, U'')$, т. е. она соответствует либо пути, проходящему через узел u'_i, либо пути, не проходящему через узел u'_i.

2. $L_2(u_f, U'') \subseteq^\circ L_1(u_f, U'')$, если

$$L_1(u_f, u'_i)L_1(u'_i, U'') \subseteq^\circ L_1(u_f, U''), \quad L_1(u_f, u_i)L_1(u'_i, U'') \subseteq^\circ L_1(u_f, U'')$$
$$\text{и} \quad L_1(u_f, \overline{u_i}, U'') \subseteq^\circ L_1(u_f, U'').$$

Истинность первого условия следует из того, что любая цепочка из множества $L_1(u_f, u'_i)L_1(u'_i, U'')$ включена либо в $L_1(u_f, u_i)L_1(u_i, U'')$, либо в $L_1(u_f, \overline{u_i}, U'')$, т. е. она соответствует либо пути, проходящему через узел u_i, либо пути, не проходящему через узел u_i.

Истинность второго условия следует из того, что $L_1(u_f, u_i)L_1(u_i, U'') \subseteq L_1(u_f, U'')$ по определению и $L_1(u_i, U'') =^\circ L_1(u'_i, U'')$, так как $(u_i, u'_i) \in E^\circ$.

Истинность третьего условия следует из того, что любая цепочка из множества $L_1(u_f, \overline{u_i}, U'')$ включена либо в $L_1(u_f, u_i)L_1(u_i, U'')$, либо в $L_1(u_f, \overline{u_i}, U'')$, т. е. она соответствует либо пути, проходящему через узел u_i, либо пути, не проходящему через узел u_i.

Таким образом, $L_1(u_f, U'') =^\circ L_2(u_f, U'')$, следовательно, рассматриваемое преобразование сохраняет сильную эквивалентность.

После выполнения преобразования в узел u_i не будет входить ни одна дуга. Он станет недостижимым из начальных узлов, поэтому его нужно удалить вместе с выходящими дугами, вершинами (терминальными и нетерминальными), в которые входят эти дуги, и дугами, выходящими из удаляемых терминальных и нетерминальных вершин. В результате выполнения описанных действий может получиться СД с узлами, в которые не входят дуги. Исключение недостижимых узлов, дуг, терминальных и нетерминальных вершин необходимо повторять, пока это возможно.

Если узел u_i начальный или стартовый, то узел u'_i нужно сделать соответственно начальным или стартовым. Узел u_i будет недостижимым, и его нужно удалить по описанным выше правилам. После этого некоторые нетерминальные вершины будут содержать несуществующий узел u_i. В таких вершинах его нужно заменить на u'_i.

Рассмотренное преобразование можно назвать «стягиванием» сильно эквивалентных узлов в один узел. Очевидно, что таким образом можно «стянуть» в один узел класс сильно эквивалентных узлов, что уменьшит размер СД. Если при этом «стягиваемые» узлы принадлежат различным компонентам, то происходит соединение компонент в одну, в которой начальными узлами будут начальные узлы соединяемых компонент. Таким образом появляются компоненты со многими входами.

5. Разбиение множества узлов на классы эквивалентности по отношению сильной эквивалентности

Отношение сильной эквивалентности определяет разбиение множества узлов СД на классы эквивалентности. Применим метод последовательного разбиения.

Сначала разобьём узлы СД на подмножества так, чтобы заключительные узлы попали в одно подмножество, а все остальные — в другое. Для СД на рис. 2 это будут подмножества $Q_1 = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15\}$ и $Q_2 = \{16, 17, 18\}$, которые образуют разбиение $R = \{Q_1, Q_2\}$.

Узлы, принадлежащие различным подмножествам, явно неэквивалентны (для них нарушено третье условие сильной эквивалентности); два узла из одного подмножества

могут быть неэквивалентными, так как при разбиении мы не учитывали первое и второе условия принадлежности пары узлов отношению сильной эквивалентности.

Для учёта этих условий построим таблицу T, в которой столбцы соответствуют узлам СД, а строки — терминалам. Клетку таблицы в строке t и столбце u будем обозначать $T[t, u]$.

Пути (u_i, t, u_j) соответствует клетка $T[t, u_i]$. Если $u_j \in Q_r$, то в клетку $T[t, u_i]$ запишем r. Пути (u_i, M, u_k, u_j) соответствуют клетки $T[t, u_i]$, такие, что $t \in M$. Если $u_j \in Q_r$ и $u_k \in Q_s$, то в клетку $T[t, u_i]$ запишем (r, s). Если узлы u_i и u'_i принадлежат одному подмножеству и $T[t, u_i] \neq T[t, u'_i]$, то это говорит о том, что пара узлов u_i и u'_i не принадлежит отношению сильной эквивалентности и узлы u_i и u'_i нужно включить в разные подмножества нового разбиения R'.

Анализируя таблицу T, формируем новое разбиение R'. Подмножество Q'_k разбиения R' формируется из элементов некоторого подмножества Q_k разбиения R: Q'_k является максимальным по мощности подмножеством множества Q_k, таким, что для каждой пары узлов u_i и u'_i, принадлежащих подмножеству Q'_k, для всех строк t таблицы верно, что $T[t, u_i] = T[t, u'_i]$. Если $R' \neq R$, то полагаем $R = R'$, заново строим таблицу T по описанным правилам и формируем новое разбиение. Если $R' = R$, то R' — множество классов эквивалентности.

Первая таблица T для СД на рис. 2 представлена в табл. 1.

Таблица 1

Терминалы	Q_1															Q_2		
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
a	1	1	1	1 1	1 1	1 1	1 1	1 1	1 1	2		2		2				
b	1	1	1	1 1	1 1	1 1	1 1	1 1	1 1		2		2		2			
c		2	2	1 1	1 1	1 1	1 1	1 1	1 1									

По табл. 1 видно, что подмножество Q_1 разбивается на подмножества $\{1\}$, $\{2, 3\}$, $\{4, 5, 6, 7, 8, 9\}$, $\{10, 12, 14\}$, $\{11, 13, 15\}$, а подмножество Q_2 не разбивается. В результате получаем разбиение $\{\{1\}, \{2, 3\}, \{4, 5, 6, 7, 8, 9\}, \{10, 12, 14\}, \{11, 13, 15\}, \{16, 17, 18\}\}$. По нему строим новую таблицу T (табл. 2).

Таблица 2

Терминалы	Q_1	Q_2		Q_3						Q_4			Q_5			Q_6		
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
a	3	3	3	4 2	5 2	4 2	5 2	4 2	5 2	6	6	6						
b	3	3	3	4 2	5 2	4 2	5 2	4 2	5 2				6	6	6			
c		6	6	4 2	5 2	4 2	5 2	4 2	5 2									

По табл. 2 видно, что подмножество $Q_3 = \{4, 5, 6, 7, 8, 9\}$ разбивается на подмножества $\{4, 6, 8\}$ и $\{5, 7, 9\}$, а остальные подмножества не разбиваются. В результате получаем разбиение $\{\{1\}, \{2, 3\}, \{4, 6, 8\}, \{5, 7, 9\}, \{10, 12, 14\}, \{11, 13, 15\}, \{16, 17, 18\}\}$. По нему строим табл. 3.

Таблица 3

Терминалы	Q_1	Q_2		Q_3			Q_4			Q_5			Q_6			Q_7		
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
a	3	3	3	5 2	5 2	5 2	6 2	6 2	6 2	7	7	7						
b	4	4	4	5 2	5 2	5 2	6 2	6 2	6 2				7	7	7			
c		7	7	5 2	5 2	5 2	6 2	6 2	6 2									

В табл. 3 нет различных столбцов, соответствующих узлам, принадлежащим одному подмножеству. Поэтому полученное разбиение является разбиением на классы эквивалентности.

6. Минимизация синтаксической диаграммы с многовходовыми компонентами на основе отношения сильной эквивалентности

Используя отношение сильной эквивалентности на множестве узлов СД R_1, можно построить СД R_2, эквивалентную R_1, которая будет иметь не больше узлов и компонент, чем R_1, а компоненты R_2 могут иметь большее количество входов, чем компоненты R_1. Сокращение количества узлов достигается за счёт «стягивания» всех узлов, принадлежащих одному классу сильной эквивалентности, в один узел.

Для преобразования СД R_1 в СД R_2 удобно R_1 представить в виде таблицы и преобразовать её в таблицу R_2 (СД R_2 будем называть сокращённой). Столбцы таблицы соответствуют узлам СД, а строки — терминалам и начальным узлам, которые встречаются в нетерминальных вершинах. Пути (u_i, x, u_j) соответствует клетка $T[x, u_i]$, в которой записан узел u_j. Начальные узлы отметим стрелкой, стартовый узел — жирной стрелкой, а заключительные — символом «1». Таблица СД (рис. 2) представлена в табл. 4.

Таблица 4

Таблица синтаксической диаграммы с многовходовыми компонентами

Терминалы и начальные узлы	↓	↓	↓													1	1	1
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
a	4	6	8							16		17		18				
b	5	7	9								16		17		18			
c		17	18															
2				10		12												
3					11		13	14	15									

Отношение сильной эквивалентности определяет разбиение множества узлов на классы $\{\{1\}, \{2, 3\}, \{4, 6, 8\}, \{5, 7, 9\}, \{10, 12, 14\}, \{11, 13, 15\}, \{16, 17, 18\}\}$. Возьмём по одному узлу из каждого класса и получим множество узлов $\{1, 2, 4, 5, 10, 11, 16\}$ сокращённой СД. В таблице СД (табл. 4) оставим только те столбцы, которые соответствуют узлам сокращённой СД. Если строка таблицы отмечена узлом, не принадлежащим множеству узлов сокращённой СД, то отметим её сильно эквивалентным ему узлом сокращённой СД. Строки, отмеченные одинаковыми узлами, объединим в одну. Если в клетке полученной таблицы встретится узел, который не принадлежит множеству узлов сокращённой СД, то его нужно заменить на сильно эквивалентный ему

узел сокращённой СД. Если узел сокращённой СД сильно эквивалентен начальному (стартовому) узлу исходной СД, то нужно сделать его начальным (стартовым).

Таблица сокращённой СД, эквивалентной СД на рис. 2, приведена в табл. 5, её графическое представление изображено на рис. 3. Эта СД содержит значительно меньше узлов, чем исходная (рис. 2), и одну компоненту с двумя входами.

Таблица 5

Таблица сокращённой СД

Терминалы и начальные узлы	↓	↓					1
	1	2	4	5	10	11	16
a	4	4			16		
b	5	5				16	
c		16					
2			10	11			

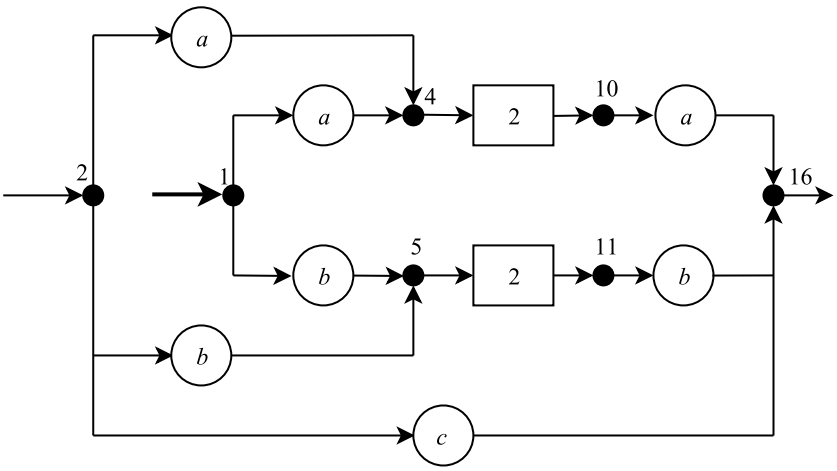

Рис. 3. Сокращённая СД

Заключение

Предложенный метод минимизации позволяет преобразовать диаграмму Вирта в более компактную синтаксическую диаграмму с многовходовыми компонентами. Такая диаграмма может быть использована для построения эффективных по памяти программ-распознавателей линейной сложности.

ЛИТЕРАТУРА

1. *Jensen K. and Wirth N.* Pascal User Manual and Report. N.Y.: Springer Verlag, 1975. 167 p.
2. *Jensen K. and Wirth N.* Pascal User Manual and Report. Berlin, Heidelberg: Springer Verlag, 1974. 170 p.
3. *Легалов А. И., Швец Д. А., Легалов И. А.* Формальные языки и трансляторы. Красноярск: Сибирский федеральный университет, 2007. 213 с.
4. *Карпов Ю. Г.* Теория и технология программирования. Основы построения трансляторов. СПб.: БХВ-Петербург, 2005. 272 с.
5. *Свердлов С. З.* Методы трансляции. Вологда: ВоГУ, 2016. 235 с.
6. *Свердлов С. З.* Конструирование компиляторов. Saarbruken: Lap Lambert, 2015. 571 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. 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.

REFERENCES

1. *Jensen K. and Wirth N.* Pascal User Manual and Report. New York, Springer Verlag, 1975. 167 p.
2. *Jensen K. and Wirth N.* Pascal User Manual and Report. Berlin, Heidelberg, Springer Verlag, 1974. 170 p.
3. *Legalov A. I., Shvets D. A. and Legalov I. A.* Formal'nyye yazyki i translyatory [Formal Languages and Translators]. Siberian Federal University, Krasnoyarsk, 2007. 213 p. (in Russian)
4. *Karpov Yu. G.* Teoriya i tekhnologiya programmirovaniya. Osnovy postroyeniya translyatorov [The Theory and Technology of Programming. Fundamentals of Translators]. Saint-Petersburg, BHV-Petersburg Publ., 2005. 272 p. (in Russian)
5. *Sverdlov S. Z.* Metody translyatsii [Methods of Translation]. Vologda, VSU Publ., 2016. 235 p. (in Russian)
6. *Sverdlov S. Z.* Konstruirovaniye kompilyatorov [Compiler Design]. Saarbruken, Lap Lambert, 2015. 571 p. (in Russian)
7. *Martynenko B. K.* Sintaksicheskie diagrammy N. Virta i graf-shemy v syntax-tehnologii [Syntactic charts and graph-schemes in the SYNTAX-Technology]. Computer Tools in Education, 2014, no. 2, pp. 3–19. (in Russian)
8. *Ryazanov Yu. D. and Seval'neva M. N.* Analiz sintaksicheskih diagramm i sintez programm-raspoznavatelej linejnoj slozhnosti [The analysis of syntax diagrams and automatic generation of linear-time programs recognizer]. Belgorod State University Scientific Bulletin. History. Political science. Economics. Information technologies, 2013, no. 8, pp. 128–136. (in Russian)
9. *Polyakov V. M. and Ryazanov Yu. D.* Algoritm postroeniya nerekursivnyh programm-raspoznavatelej linejnoj slozhnosti po determinirovannym sintaksicheskim diagrammam [Algorithm for not recursive linear-time programs recognizer design from deterministic syntax diagrams]. Bulletin of BSTU named after V. G. Shukhov, 2013, no. 6, pp. 194–199. (in Russian)