ИНФОРМАТИКА, ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА И УПРАВЛЕНИЕ

DOI: 10.12737/22254

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

УМЕНЬШЕНИЕ КОЛИЧЕСТВА ЛИШНИХ ПЕРЕХОДОВ И СОСТОЯНИЙВ РАСПОЗНАВАТЕЛЯХ С МАГАЗИННОЙ ПАМЯТЬЮ

Ryazanov.iurij@yandex.ru

В статье рассматривается задача сокращения количества состояний распознавателя с магазинной памятью за счет исключения лишних переходов, которые никогда не срабатывают, и исключения лишних состояний, в которых не может оказаться распознаватель в процессе обработки допустимой цепочки. Предлагается алгоритм поиска лишних переходов и состояний, основанный на представлении распознавателя в виде графа. Приводится пример распознавателя, в котором предложенный алгоритм устраняет лишние переходы и состояния. Этот алгоритм не гарантирует исключения всех лишних переходов и состояний. Приводится пример распознавателя с лишними переходами и состояниями, которые не обнаруживаются алгоритмом. Предложенный алгоритм может быть использован при разработке программ обработки формальных языков.

Ключевые слова: *контекстно-свободный язык, распознаватель с магазинной памятью, состояние, переход, эквивалентные преобразования.*

Введение. Задача распознавания языка заключается в определении принадлежности заданной цепочки заданному языку. Для распознавания контекстно-свободных языков используются распознаватели с магазинной памятью (МП-распознаватели) [1–6]. В работе рассматривается один из классов МП-распознавателей с конечным множеством состояний. В процессе обработки входной цепочки МП-распознаватель переходит из одного состояния в другое и выполняет связанные с этим переходом действия. В множестве переходов МП-распознавателя можно выделить лишние переходы, которые никогда не срабатывают, а в множестве состояний – лишние состояния, в которых не может оказаться МП-распознаватель в процессе обработки допустимой цепочки. Такие переходы и состояния нужно исключать из МП-распознавателя. В работе предлагается алгоритм поиска и исключения лишних переходов и состояний, основанный на представлении МП-распознавателя в виде графа [6–9].

Решение задачи исключения лишних состояний в конечных распознавателях известно [10] и основано на применении алгоритмов поиска в графе распознавателя. Для МП-распознавателей такой подход также применим, но только в случае, когда в МП-распознавателе нет лишних переходов. Поэтому основное внимание в работе уделено поиску лишних переходов.

Определение распознавателя с магазинной памятью и конечным множеством состояний. В работе будем рассматривать класс МП-распознавателей [11, 12], которые можно построить по диаграммам Вирта [11] или синтаксическим диаграммам с многовходовыми компонентами [13]. МП-распознаватель из этого класса формально можно представить следующим образом:

$$M = (Q, \text{X}, \Gamma, I, S, P, W, \delta, \lambda, q_0, \boldsymbol{q}, \gamma_0),$$

где Q – конечное множество состояний, $Q = \{q_0, q_1,\ldots,q_m, \boldsymbol{q}\}$;

X – конечное множество входных символов, включающее концевой маркер ⊣, которым заканчивается входная цепочка;

$\Gamma = Q \cup \{\nabla\}$ – конечное множество магазинных символов (равно множеству состояний, дополненному маркером дна магазина ∇);

I – конечное множество операций над головкой, I = (*сдвиг*, *держать*). Операция *сдвиг* перемещает головку на одну позицию вправо, а *держать* – не изменяет положения головки;

S – конечное множество операций над состоянием, $S = \{сост(q_0), сост(q_1),\ldots,сост(q_m)\}$. Операция $сост(q_i)$ обозначает переход в состояние q_i;

P – множество операций над магазином, P = {*втолк*(q_0), *втолк* (q_1),…,*втолк* (q_m), *вытолк*, *не изменять*}.

W – конечное множество значений выхода, W = {*допустить*, *отвергнуть*};

q_0 – начальное состояние, $q_0 \in Q$;

$\boldsymbol{q}$ – допускающее состояние, $\boldsymbol{q} \in Q$;

γ_0 – начальное содержимое магазина, $\gamma_0 = \nabla \boldsymbol{q}$ (магазин содержит маркер дна и допускающее состояние);

$\delta : Q \times X \times \Gamma \to I \times S \times P$ – частичная функция переходов, которая состоянию, символу входной цепочки (находящемуся под головкой) и верхнему символу магазина ставит в соответствие операцию над головкой, состоянием и магазином, причем множество видов значений на тройке (q_m, x, q_s) равно {(*сдвиг*, *сост*(q_n), *не изменять*), (*держать*, *сост*(q_n), *втолк*(q_r)), (*держать*, *сост*(q_s), *вытолк*)}. Если значение функции на тройке (q_m, x, q_s) равно (*сдвиг*, *сост*(q_n), *не изменять*), то такой переход будем обозначать $(q_m, (x, \to), q_n)$, если значение функции на этой тройке равно (*держать*, *сост*(q_n), *втолк*(q_r)), то переход обозначим $(q_m, (x, \downarrow(q_r)), q_n)$, если же значение на этой тройке равно (*держать*, *сост*(q_s), *вытолк*), то переход обозначим $(q_m, (q_s, \uparrow), q_s)$.

$\lambda : Q \times X \times \Gamma \to W$ – частичная функция выходов, которая состоянию, символу входной цепочки (находящемуся под головкой) и верхнему символу магазина ставит в соответствие значение выхода – *допустить* или *отвергнуть*. Значение функции на тройке $(\boldsymbol{q}, ⊣, \nabla)$ равно *допустить*, а на всех остальных, на которых функция определена –*отвергнуть*.

Области определения функций δ и λ не пересекаются, а их объединение равно области отправления.

Тройка $(q_m, \alpha, \nabla\gamma)$, где q_m – состояние, α – часть входной цепочки, начиная с символа под головкой и заканчивая концевым маркером, γ – содержимое магазина, называется конфигурацией МП-распознавателя. Исходной конфигурацией является $(q_0, \alpha_0, \nabla \boldsymbol{q})$, где α_0 – вся входная цепочка (головка находится над первым символом).

Пусть конфигурацией МП-распознавателя является тройка $(q_m, x\alpha, \nabla\gamma q_s)$, где x – символ под головкой, q_s – верхний символ магазина. Если на тройке (q_m, x, q_s) определена функция переходов δ, то ее значение определяет операции над головкой, состоянием и магазином. При выполнении этих операций конфигурация изменяется. Если на тройке $(q_m x, q_s)$ определена функция выходов λ, то процесс распознавания заканчивается с результатом, равным значению функции λ. Такую конфигурацию назовем заключительной. Итак, работа МП-распознавателя заключается в изменении конфигураций. Последней является заключительная конфигурация, в которой определяется результат распознавания.

МП-распознаватель можно представить в виде взвешенного ориентированного мультиграфа, вершины которого соответствуют состояниям, а дуги – переходам. Начальное состояние будем показывать неотмеченной стрелкой, а допускающее состояние – жирным кружком. Переходу $(q_m, (x, \to), q_n)$ соответствует дуга из вершины q_m в вершину q_n, отмеченная меткой $(x, \to)$, переходу $(q_m, (x, \downarrow(q_r)), q_n)$ – дуга из вершиныq_m в вершину q_n, отмеченная меткой $(x, \downarrow(q_r))$, а переходу $(q_m, (q_s, \uparrow), q_s)$ – дуга из вершины q_m в вершину q_s, отмеченная меткой$(q_s, \uparrow)$. На рис. 1 приведен пример графа МП-распознавателя.

Лишние переходы и состояния. В МП-распознавателе могут существовать лишние переходы и состояния. Переход лишний, если не существует входной цепочки, при обработке которой этот переход срабатывает. В множестве лишних переходов можно выделить два класса:

1) переход $(q_m, (x, \to), q_n)$ или $(q_m, (x, \downarrow(q_s)), q_n)$ принадлежит первому классу, если не существует входной цепочки, при обработке которой МП-распознаватель будет находиться в состоянии q_m и анализировать символ x;

2) переход $(q_m, (q_n, \uparrow), q_n)$ принадлежит второму классу, если не существует входной цепочки, при обработке которой МП-распознаватель будет находиться в состоянии q_m и вверху магазина будет q_n.

В множестве лишних состояний можно выделить два подмножества: недостижимые и избыточные.

Состояние q_m принадлежит подмножеству недостижимых, если не существует входной цепочки, при обработке которой МП-распознаватель окажется в состоянии q_m .

Состояние q_m принадлежит подмножеству избыточных, если из состояния q_m МП-распознаватель не при каких условиях не может перейти в допускающее состояние.

Исключение лишних переходов может привести к появлению лишних состояний. В графе МП-распознавателя лишние состояния удаляются вместе с входящими и выходящими дугами.

Алгоритм уменьшения количества лишних переходов и состояний. Для решения задачи уменьшения количества лишних переходов и состояний определим переход как подозреваемый, если есть вероятность, что он лишний.

Любой переход вида $(q_m, (q_n, \uparrow), q_n)$ считаем подозреваемым, так как для его срабатывания в магазине должен быть символ q_n, но если символа q_n не может быть в магазине, когда МП-распознаватель находится в состоянии q_m, то этот переход лишний.

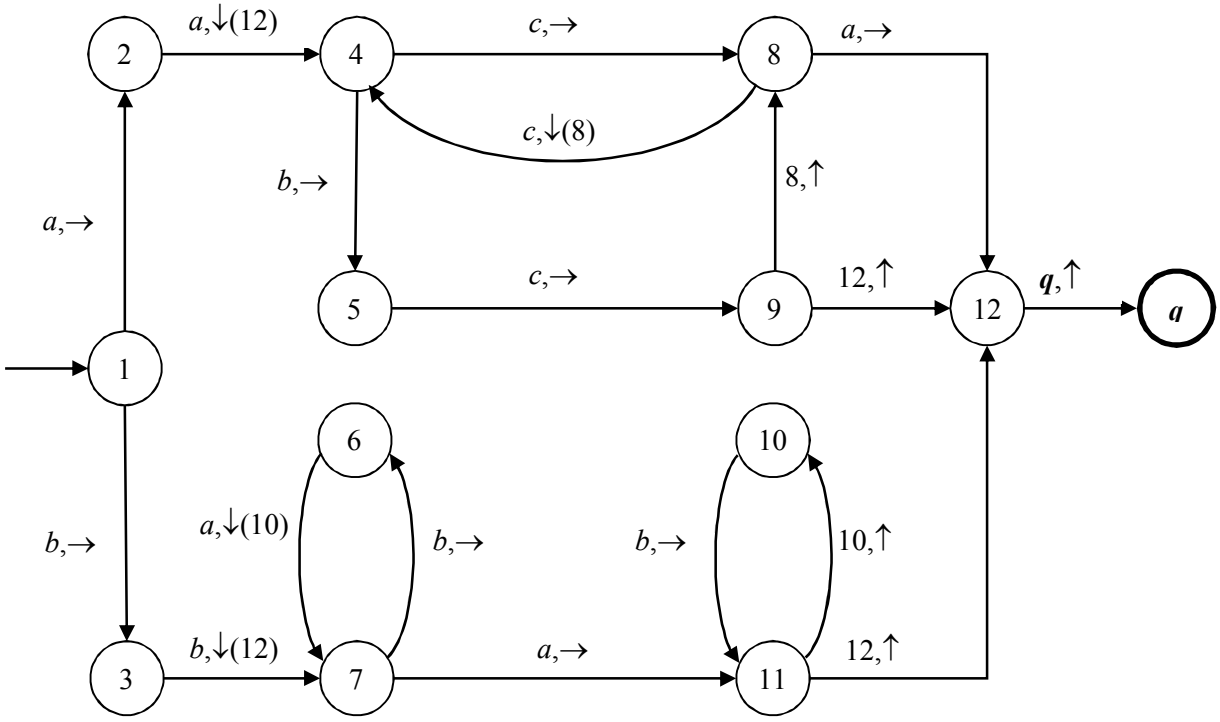

Рис. 1. Граф МП-распознавателя

Переход вида (q_m, (x, →), q_n) или (q_m, (x, ↓(q_s)), q_n) считаем подозреваемым, если не существует перехода в состояние q_m с выполнением операции *сдвиг*. Если на переходе в состояние q_m выполняется *сдвиг*, то следующим символом, анализируемым в состоянии q_m, может быть любой символ алфавита, в том числе и символ x, и переход сработает.

Идея алгоритма исключения лишних переходов заключается в следующем. Подозреваемые переходы исключаем из МП-распознавателя и сохраняем в некотором множестве. Затем, если подозрение для некоторого перехода опровергается, переход возвращается в МП-распознаватель.

Исключенный переход вида (q_m, (x, →), q_n) или (q_m, (x, ↓(q_s)), q_n) возвращается в МП-распознаватель, если в графе распознавателя существует путь от начальной вершины до q_m и существует цепочка α, при обработке которой срабатывают переходы, соответствующие этому пути, и, при достижении состояния q_m текущим символом в цепочке α будет символ x.

Исключенный переход вида (q_m, (q_n, ↑), q_n) возвращается в МП-распознаватель, если в графе распознавателя существует путь от начальной вершины до q_m и на этом пути есть переход (q_s (x, ↓(q_n)), q_r), который вталкивает в магазин символ q_n, необходимый для срабатывания перехода (q_m, (q_n, ↑), q_n).

Возвращение некоторых переходов может сделать истинными условия для возвращения других переходов, поэтому выполнение алгоритма продолжается, пока возвращаются удаленные переходы в МП-распознаватель.

В алгоритме будем использовать:

1) множество X — входной алфавит МП-распознавателя;

2) для каждого состояния q_m множество $M(q_m)$, в котором будут накапливаться символы входного алфавита, которые могут быть текущими в обрабатываемой цепочке, когда МП-распознаватель находится в состоянии q_m;

3) множество M_1 состояний, из которых существуют переходы с выталкиванием магазинного символа;

4) множество M_2 состояний, из которых существуют переходы с вталкиванием символа в магазин;

5) отношение E, элементы которого соответствуют переходам МП-распознавателя;

6) отношение E^T, представляющее собой транзитивное замыкание отношения E;

7) множество D, в котором сохраняются исключенные из МП-распознавателя переходы.

В п.1 алгоритма (см. ниже) выполняется инициализация множеств и исключение подозреваемых переходов, в п.2 – возвращение подозреваемых переходов, для которых подозрение опровержено, в п.3 – исключение лишних состояний.

Поиск лишних состояний выполняется по отношению E^T. Если $(q_0, q_m) \notin E^T$, то состояние q_m

недостижимо, а если $(q_m, \boldsymbol{q}) \notin E^T$, то состояние q_m избыточно.

Алгоритм исключения лишних состояний.

1. Для каждого состояния q_m МП-распознавателя выполнить: $M(q_m) := \varnothing$.

Для начального состояния q_0 выполнить: $M(q_0) := X$.

Для всех переходов вида $(q_m, (x, \rightarrow), q_n)$ выполнить: $M(q_n) := X$.

Сформировать множество $M_1 = \{q_m \mid$ существует переход $(q_m, (q_n, \uparrow), q_n)\}$.

Сформировать множество $M_2 = \{q_m \mid$ существует переход $(q_m, (x, \downarrow(q_s)), q_n)\}$.

Все переходы вида $(q_m, (q_n, \uparrow), q_n)$ исключить из МП-распознавателя и сохранить в множестве D.

Переход вида $(q_m, (x, \rightarrow), q_n)$ исключить из МП-распознавателя и сохранить в множестве D, если $x \notin M(q_m)$.

Переход вида $(q_m, (x, \downarrow(q_s)), q_n)$ исключить из МП-распознавателя и сохранить в множестве D, если $x \notin M(q_m)$.

Сформировать отношение $E = \{(q_m, q_n) \mid$ существует переход $(q_m, (x, \rightarrow), q_n)$ или $(q_m, (x, \downarrow(q_s)), q_n)\}$.

2. Вычислить E^T.

Для всех q_m и q_n выполнить:

если $(q_0, q_m) \in E^T$ и существует переход $(q_m, (x, \downarrow(q_s)), q_n)$,

то $M(q_n) := M(q_n) \cup \{x\}$.

Повторять

Для всех q_m и q_n выполнить:

если $(q_0, q_m) \in E^T$ и существует переход $(q_m, (q_n, \uparrow), q_n)$,

то $M(q_n) := M(q_n) \cup M(q_m)$

пока множества M изменяются.

Для всех q_m и q_n выполнить:

если $(q_0, q_m) \in E^T$ и $(q_m, (x, \rightarrow), q_n) \in D$ и $x \in M(q_m)$,

то переход $(q_m, (x, \rightarrow), q_n)$ исключить из D и включить в МП-распознаватель, пару (q_m, q_n) включить в отношение E.

Для всех q_r, $q_m \in M_1$, $q_s \in M_2$ выполнить:

если $(q_0, q_s) \in E^T$ и существует переход $(q_s, (x, \downarrow(q_n)), q_r)$ и $(q_r, q_m) \in E^T$ и $(q_m, (q_n, \uparrow), q_n) \in D$, то переход $(q_m, (q_n, \uparrow), q_n)$ исключить из D и включить в МП-распознаватель, пару (q_m, q_n) включить в отношение E.

Если E изменилось, выполнить п.2.

3. Для всех q_m и q_n выполнить:

если $(q_0, q_m) \notin E^T$, или $(q_m, \boldsymbol{q}) \notin E^T$, то состояние q_m удалить.

Пример. Рассмотрим выполнение алгоритма на примере МП-распознавателя, представленного графом на рис. 1.

После выполнения п.1 алгоритма получим МП-распознаватель, представленный графом на рис. 2, после выполнения п.2 – на рис. 3, а после выполнения п.3 – на рис. 4.

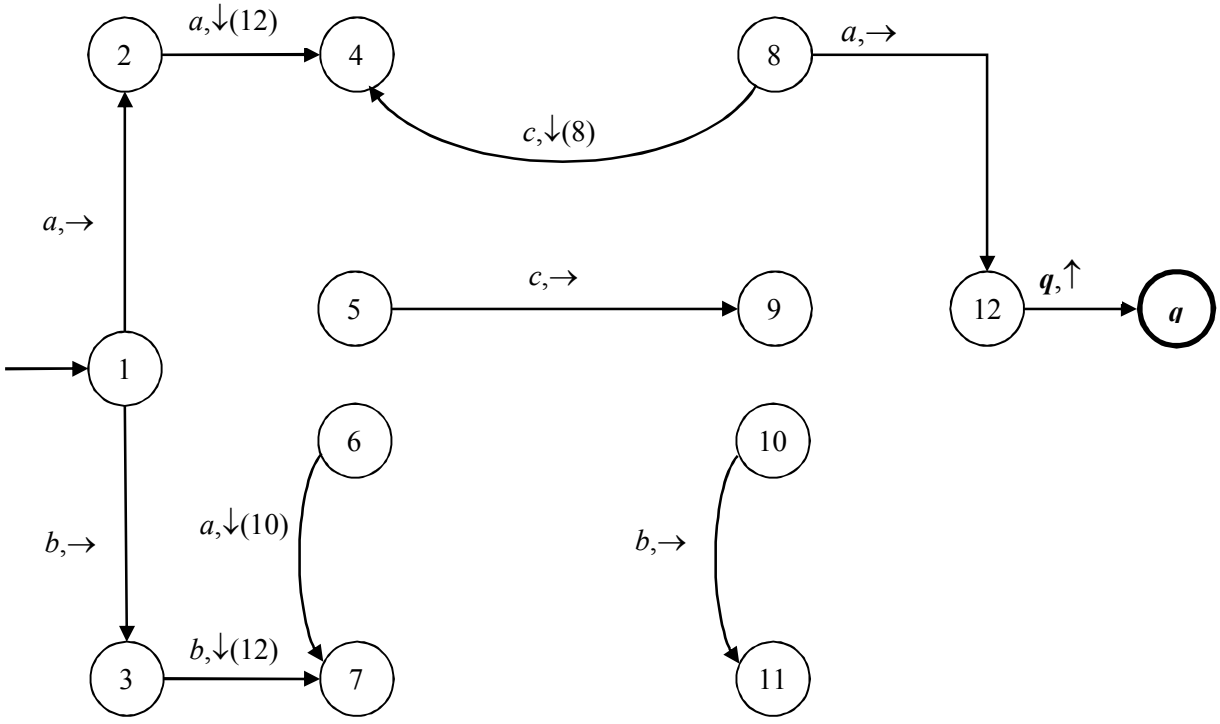

Рис. 2. Граф МП-распознавателя после выполнения п.1 алгоритма

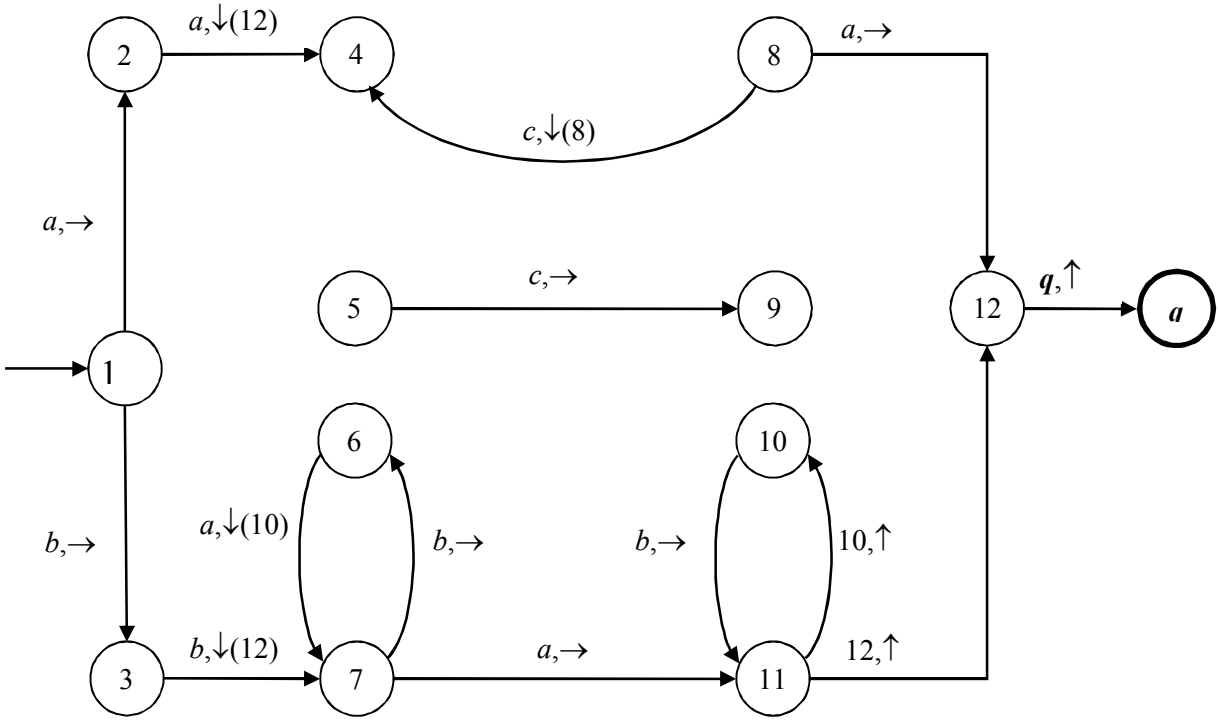

Рис. 3. Граф МП-распознавателя после выполнения п.2 алгоритма

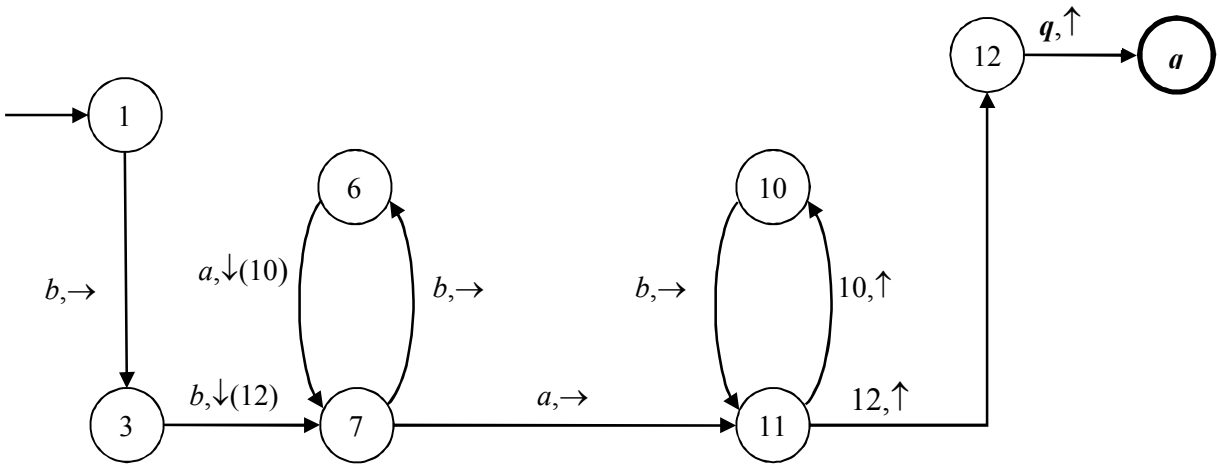

Рис. 4. Граф МП-распознавателя после выполнения п.3 алгоритма

В результате выполнения п.1 из МП-распознавателя исключены переходы из состояний 9 и 11 с операцией *вытолкнуть* и переходы из состояний 4 и 7, так как в эти состояния не входят дуги с операцией *сдвиг*.

При выполнении п.2 восстанавливается переход (7, (*b*, →), 6), так как состояние 3 достижимо из начального и есть переход (3, (*b*, ↓(12)), 7). Теперь состояние 6 достижимо и есть переход (6, (*a*, ↓(10)), 7), поэтому восстанавливается переход (7, (*b*, →), 11). Состояние 11 стало достижимым и в графе МП-распознавателя существует путь от начального состояния 1 до состояния 11 с переходами (3, (*b*, ↓(12)), 7) и (6, (*a*, ↓(10)), 7), поэтому восстанавливаются переходы (11, (10, ↑),10) и (11, (12, ↑), 12).

Переходы из состояния 4 не восстанавливаются, так как в него существует единственный переход (2, (*a*, ↓(12)), 4) из достижимого состояния и его недостаточно для восстановления переходов (4, (*b*, →), 5) и (4, (*c*, →), 8). Переходы из состояния 9 не восстанавливаются, так как оно недостижимо из начального.

После выполнения п.2 в МП-распознавателе состояния 5, 8 и 9 недостижимые, а состояния 2, 4, 5, 9 — избыточные (рис. 3), поэтому они удаляются при выполнении п.3 алгоритма.

Выводы. В статье представлен алгоритм, который позволяет уменьшить количество лиш-

них переходов и состояний, но не гарантирует получение МП-распознавателя без лишних переходов и состояний. Например, в МП-распознавателе (рис. 5) лишними являются переходы (1, ((*b*, ↓(4)), (3, (*b*, →), 4) и (4, (3, ↑), 3) и состояние 4, но они не обнаруживаются и не устраняются этим алгоритмом.

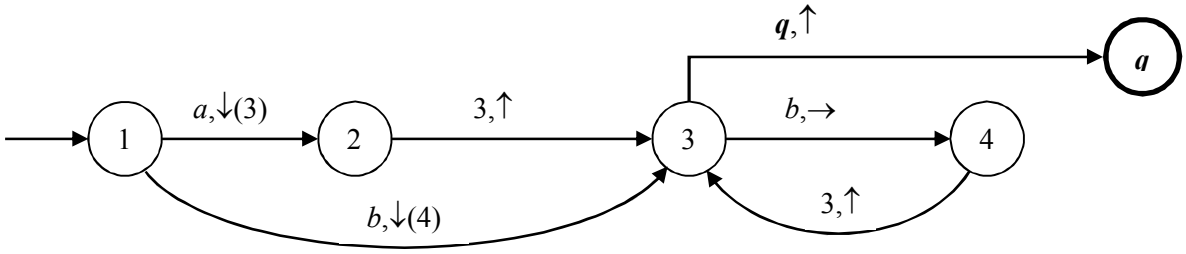

Рис. 5. Граф МП-распознавателя с лишними переходами и лишним состоянием

Дальнейшим развитием работы является поиск причин, по которым переходы могут быть лишними, и разработка дополнительных алгоритмов, позволяющих исключать лишние переходы, которые не обнаруживаются предложенным алгоритмом.

БИБЛИОГРАФИЧЕСКИЙСПИСОК

1. Schutzenberger M.P. “On context-free languages and pushdown automata”, Information and Control 6:3 (1963), pp. 246–264.

2. Ахо А., Ульман Дж. Теория синтаксического анализа, перевода и компиляции. М.: Мир, 1978. Т. 1. 612 с.

3. Льюис Ф., Розенкранц Д., Стирнз Р. Теоретические основы проектирования компиляторов. М.: Мир, 1979. 656 с.

4. Мартыненко Б.К. Синтаксически управляемая обработка данных. Изд. 2-е, дополн. СПб: Изд-во С.-Петербургского университета, 2004 г. 317 с.

5. Ахо А, Лам М., Сети Р., Ульман Дж. Компиляторы. Принципы, технологии и инструментарий. М: Издательский дом “Вильямс”, 2008. 1185 с.

6. Hopcroft J.E., Motwani R., J.D. Ullman J.D. Introduction to Automata Theory, Languages, and Computation (3rd ed.). Pearson., 2013. p. 496.

7. Белоусов А.И., Ткачев С.Б. Мультиграфовое представление автоматов с магазинной памятью // Наука и образование: научное издание МГТУ им. Н.Э. Баумана. 2012. № 9. С. 11.

8. Станевичене Л.И. Графы магазинных автоматов // Учредительная конф. Российской ассоциации "Женщины-математики". Тез. докл. М. 1993(а). С. 50

9. Вылиток А.А. О построении графа магазинного автомата // Вестн. Моск. ун-та. Сер. 15. 1996. № 3. С. 68–73.

10. Брауэр В. Введение в теорию конечных автоматов М.: Книга по Требованию, 2012. С. 272.

11. Рязанов Ю. Д. Синтез распознавателей с магазинной памятью по детерминированным синтаксическим диаграммам // Вестник ВГУ. Системный анализ и информационные технологии. 2014. №1. С. 138–145.

12. Рязанов Ю. Д., Савёлова И. Н. Преобразование распознавателя с магазинной памятью и одним состоянием в распознаватель с конечным множеством состояний // Моделирование, оптимизация и информационные технологии. 2015. № 4 (11). С. 13.

13. Polyakov V.M., Ryazanov Y.D. Virt Charts to Multiport Component Syntactic Charts Transformation // Global Journal of Pure and Applied Mathematics. 2015. Т. 11. № 5. С. 3939–3952.

Ryazanov Yu.D.

REDUCE THE NUMBER OF UNNECESSARY TRANSITIONS AND STATES IN THE PUSHDOWN RECOGNIZER

The article considers the problem of reducing the number of states of the pushdown recognizer by eliminating unnecessary transitions that never work, and eliminating unnecessary states in which cannot be recognizer in processing a valid chain. There are suggested a search algorithm of unnecessary transitions and states, based on the representation of recognizer as a graph. The article demonstrates the example of recognizer in which the proposed algorithm eliminates unnecessary transitions and states. This algorithm does not guarantee elimination of all unnecessary transitions and states. The article demonstrates the example of recognizer with unnecessary transitions and states that are not detected by the algorithm. The proposed algorithm can be used at software design the processing for formal languages.

Key words: *context-free language, pushdown recognizer, state, transition, equivalent transforming.*