

Содержание

Программа курса	5
Список литературы	7
Задание	9
Регулярные языки	10
Контрольные вопросы	15
Приложение конечных автоматов для поиска под- строк	15
Алгоритм Кнута–Морриса–Пратта	15
Структура данных «Словарь»	17
Алгоритм Ахо–Корасик	18
Контекстно-свободные языки	21
Контрольные вопросы	24
Приложение КС-грамматик для сжатия данных	25
Элементы синтаксического анализа	30
LL-анализ	30
Контрольные вопросы	31
LR-анализ	32
Контрольные вопросы	34

Атрибутные грамматики	35
Дополнительные задачи	39
Регулярные языки	39
КС-языки	40
Элементы синтаксического анализа	41

Программа курса

«Теория и реализация языков программирования»

1. Известные и перспективные направления эффективного применения теории формальных языков как математической дисциплины. Алфавиты, цепочки, языки и их представление. Формальное определение грамматики. Типы грамматик по Хомскому и их свойства. Связь машин Тьюринга и грамма-тик типа 0. Линейно-ограниченные автоматы и их связь с КЗ-грамматиками.
2. Лексический анализ. Регулярные языки (РЯ) и регулярные выражения (РВ). Конечные автоматы (КА). Детерминированные и недетерминированные КА (ДКА и НКА). Эквивалентность классов языков, определяемых КА, РВ и автоматными грамматиками (грамматики типа 3: леволinéйные – ЛЛ, праволинейные – ПЛ). Свойства замкнутости РЯ. Лемма о накачке для РЯ. Теорема Майхилла–Нероуда и минимальные автоматы. Алгоритмы поиска подстрок и КА. Алгоритм Кнута–Морриса–Пратта (КМП-алгоритм), реализация структуры данных «словарь», алгоритм Ахо–Корасик.

Алгоритмы по теме КА

- Построение ДКА по НКА.
- Построение НКА по РВ.
- Построение ДКА по РВ.

- Построение РВ по НКА.
 - По РВ R проверить, что слово принадлежит $L(R)$.
 - Построить по языку L язык L^R .
 - Построение произведения (конкатенации) РЯ, дополнение РЯ, пересечение РЯ.
 - Построение минимального автомата по ДКА.
 - КМП-алгоритм.
 - Построение по НКА эквивалентных ЛЛ- и ПЛ-грамматик.
 - Построение эквивалентного НКА по ЛЛ- и ПЛ-грамматике.
 - Решение уравнений с регулярными коэффициентами.
3. Синтаксический анализ: КС-грамматики (КСГ). Преобразования КС-грамматик, приведённые грамматики. Канонические формы КС-грамматик (нормальная форма Хомского). Свойства замкнутости КС-языков (КСЯ), незамкнутость КСЯ относительно пересечения. Дерево вывода КСГ. Однозначность КС-грамматик. Однозначность праволинейной грамматики, построенной по ДКА. Лемма о накачке для КСЯ. Проверка принадлежности слова КСЯ, заданному КСГ (алгоритм Кока–Янгера–Касами). Сжатие строк на основе КС-грамматик (straight-line grammars): алгоритм Лемпеля–Зива–Велча.
4. Синтаксический анализ: автоматы с магазинной памятью (МА). Детерминированные и недетерминированные МА. Обобщённые МА и их эквивалентность стандартным МА. Эквивалентность МА, распознающих по конечному состоянию (F-МА) и по опустошению магазина (N-МА). Эквивалентность КСГ и МА. Однозначность КСГ, построенной по детерминированному N-МА (без доказательства).

Алгоритмы по теме КСГ и МА

- Удаление недостижимых и бесполезных символов в КСГ. Удаление циклов.
 - Удаление левой рекурсии в КСГ.
 - Приведение КСГ к нормальной форме Хомского и нормальной форме Грейбах.
 - Проверка принадлежности слова КСЯ, заданному КСГ (алгоритм Кока–Янгера–Касами).
 - Преобразование N-МА \rightarrow F-МА.
 - Преобразование F-МА \rightarrow N-МА.
 - Преобразование КСГ в эквивалентный N-МА.
 - Преобразование N-МА в эквивалентную КСГ (с доказательством корректности для N-МА с одним состоянием).
5. Дополнительные сведения из теории конечных автоматов. Минимизация числа состояний и эквивалентность детерминированного конечного автомата (ДКА).
6. Предсказывающий разбор сверху вниз. Алгоритм разбора сверху вниз. Функции FIRST и FOLLOW. Конструирование таблицы предсказывающего анализатора. LL(1)-грамматики. Удаление левой рекурсии. Левая факторизация. Рекурсивный спуск. LL(k)-грамматики. Разбор снизу вверх типа сдвиг-свёртка. Основа. LR(k)-анализаторы. Конструирование LR(1)-таблицы. LR(1)-грамматики. Варианты LR-анализаторов. LR(k)-грамматики.
7. Элементы теории перевода. Синтаксически управляемый перевод. Атрибутные грамматики.

Список литературы

- [1] Ахо А., Сети Р., Ульман Дж. Компиляторы. Принципы, технологии, инструменты. М.-СПб.-Киев: Вильямс, 2001.
- [2] Мартыненко Б.К. Языки и трансляции. СПб.: СПбГУ, 2004. Доступно по ссылке http://trpl7.ru/t-books/Martin/Martinenko_FLT_Cont.htm.
- [3] Серебряков В. А., Галочкин М. П., Гончар Д. Р., Фуругян М. Г. Теория и реализация языков программирования: учеб. пос. М.: МЗ-Пресс, 2006. 352 с.
- [4] Хопкрофт Дж., Мотвани Р., Ульман Дж. Введение в теорию автоматов, языков и вычислений. М.: Вильямс, 2002.
- [5] Ахо А., Лам М., Сети Р., Ульман Дж. Компиляторы. Принципы, технологии и инструментарий. М.-СПб.-Киев: Вильямс, 2011. 1184 с.
- [6] Шень А. Х. Программирование: теоремы и задачи. М.: МЦ-НМО, 2004. Доступно по ссылке <https://www.mccme.ru/free-books/shen/shen-progbook.pdf>.

Задание

Задачи, выделенные в дополнительный раздел, а также задачи, помеченные звёздочкой, являются дополнительными и необязательными. Контрольные вопросы являются полноценными задачами, хотя и выделены в отдельные блоки. Решение всех задач должно быть обосновано. Отдельные указания по необходимости обоснования в некоторых задачах являются акцентированием и вовсе не означают, что в других задачах обоснование не требуется. Использование алгоритмов из курса (см. программу) считается обоснованием. При использовании алгоритма проверяющий должен иметь возможность проверить корректность протокола: решения в духе «автомат построен по алгоритму, но вот только ответ» не оцениваются.

Если в формулировке вопроса задачи используются обороты «верно ли, что» и «может ли быть», то в случае положительного ответа приведите доказательство, а в случае отрицательного – контрпример. Верное рассуждение без контрпримера оценивается в половину задачи.

Всё вышесказанное относится ко всем письменным работам курса.

Ссылки на литературу указаны по библиографии в программе курса.

Регулярные языки

Задача 1. Определим язык $L \subseteq \{a, b\}^*$ индуктивными правилами:

1. $\varepsilon, b, bb \in L$;
2. вместе с любым словом $x \in L$ в L также входят слова $ax, bax, bba x$;
3. никаких других слов в L нет.

Язык $T \subseteq \{a, b\}^*$ состоит из всех слов, в которых нет трёх букв b подряд.

1. Докажите или опровергните, что $L = T$.¹
2. Запишите язык T в виде регулярного выражения.
3. Постройте конечный автомат, принимающий T . Докажите (по индукции), что построенный автомат принимает язык T .

¹Если равенство неверно, то нужно явно указать слово, принадлежащее одному языку и не принадлежащее другому. Если равенство верно, то нужно провести доказательство по индукции в обе стороны: $L \subseteq T$ и $T \subseteq L$.

Задача 2. Верно ли, что

1) $\varepsilon \in \{a, aab, aba\}$?

2) $\emptyset \in \{a, aab, aba\}$?

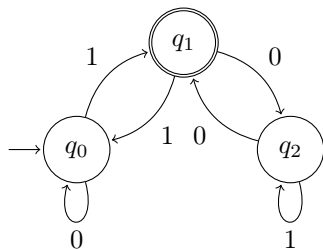
Задача 3.

1. Задайте множество $\{a^n \mid n > 0\} \times \{b^n \mid n \geq 0\}$ формулой, которая не использует символ \times .

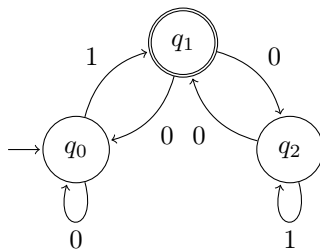
2. Опишите язык $\{a^{3n} \mid n > 0\} \cap \{a^{5n+1} \mid n \geq 0\}^*$ регулярным выражением.

Задача 4. Автоматы \mathcal{A} и \mathcal{B} заданы диаграммами. Выполните следующие задания.

Автомат \mathcal{A} :



Автомат \mathcal{B} :



Для каждого автомата ответьте на следующие вопросы (1–2).

1. Автомат задан через граф переходов. Запишите определение автомата в виде $(Q, \Sigma, \delta, q_0, F)$. Опишите элементы каждого множества.

2. Является ли автомат детерминированным?

Ответьте на вопросы.

3. Опишите последовательность конфигураций автомата \mathcal{A} при обработке слова $w = 011001$. Верно ли, что $w \in L(\mathcal{A})$?
4. Принимает ли автомат \mathcal{B} слово $v = 0101001$?
5. Укажите по одному слову, принадлежащему $L(\mathcal{A})$, $L(\mathcal{B})$, и по одному слову, не принадлежащему $L(\mathcal{A})$, $L(\mathcal{B})$. Все четыре слова должны быть различными.

Задача 5. Выполните следующие задания.

1. Построить ДКА, принимающий язык L , состоящий из всех слов в алфавите $\{0, 1\}$, которые содержат чётное число нулей и нечётное число единиц.
2. Построить эквивалентную левoliniейную грамматику. Будет ли она однозначной?
3. Построить регулярное выражение для языка L^R .

Задача 6. Будут ли регулярными следующие языки?

1. $L = \{a^{2016n+5} \mid n = 0, 1, \dots\} \cap \{a^{503k+29} \mid k = 401, 402, \dots\} \subseteq \{a^*\}$.
2. $L_2 = \{a^{200n^2+1} \mid n = 1000, 1001, \dots\} \subseteq \{a^*\}$.
3. Язык L_3 всех слов в алфавите $\{0, 1\}$, которые представляют числа в двоичной записи, дающие остаток два при делении на три (слово читается со старших разрядов). Например, $001010 \notin L_3$ ($1010_2 = 10_{10} = 3 \times 3 + 1$), а $10001 \in L_3$ ($10001_2 = 17_{10} = 5 \times 3 + 2$).

Задача 7. Постройте НКА, принимающий язык $L_3 = \{ \text{Множество слов в алфавите } \{a, b\}, \text{ у которых третий от конца}^2 \text{ символ}$

²Последний символ слова равен первому символу с конца слова.

равен « a ». Затем, используя алгоритм, постройте соответствующий полный ДКА.

Задача 8. Порождает ли регулярное выражение $(ab)^*(ba)^*$ тот же язык, что распознаёт ДКА $M = (\{A, B, C, D\}, \{a, b\}, \delta, A, \{A, D, E\})$, где функция переходов задана следующим образом:

$$\delta(A, a) = B, \delta(A, b) = C, \delta(B, b) = D, \delta(C, a) = E,$$

$$\delta(D, a) = B, \delta(D, b) = C, \delta(E, b) = C.$$

Задача 9. Покажите, что следующий язык удовлетворяет лемме о разрастании для регулярных языков, но сам регулярным не является:

$$L = \{ab^{2^i} \mid i \geq 0\} \cup \{b^j \mid j \geq 0\} \cup \{a^m b^n \mid m > 1, n \geq 0\}.$$

Задача 10. Решите уравнения с регулярными коэффициентами. В каждом пункте нужно выполнить три задания:

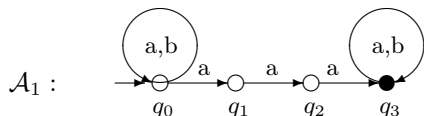
- 1) найти частное решение;
- 2) найти решение, минимальное по включению;
- 3) найти все решения.

$$1. X = ((110)^* + 111^*)X.$$

$$2. X = (00 + 01 + 10 + 11)X + (0 + 1 + \varepsilon).$$

$$3. \begin{cases} Q_0 &= 0Q_0 + 1Q_1 + \varepsilon, \\ Q_1 &= 1Q_0 + 0Q_2, \\ Q_2 &= 0Q_1 + 1Q_2. \end{cases}$$

Задача 11. Автомат \mathcal{A}_1 задан диаграммой. Выполните следующие задания. Достаточно выполнить хотя бы один из пунктов 2 или 3.



1. По диаграмме \mathcal{A}_1 постройте праволинейную грамматику G .
2. Запишите определяющую систему уравнений для G . Найдите её наименьшую неподвижную точку и вычислите регулярное выражение α_1 для $L(\mathcal{A}_1)$.
3. Определите регулярное выражение α_2 для $L(\mathcal{A}_1)$ с помощью индуктивного вычисления множеств R_{ij}^k .
4. Выберите какое-нибудь регулярное выражение α_1 или α_2 и постройте НКА \mathcal{A}_2 по регулярному выражению.
5. Выберите какой-нибудь НКА \mathcal{A}_1 или \mathcal{A}_2 и постройте ДКА D_1 .
6. Выберите какое-нибудь регулярное выражение α_1 или α_2 и постройте ДКА D_2 .
7. Выберите какой-нибудь ДКА D_1 или D_2 , дополните его, если нужно, до полного и постройте минимальный полный ДКА $\min \mathcal{A}$ для L . Для каждой пары состояний укажите соответствующие различающие их цепочки.
- 8*. По алгоритму КМП (Кнута–Мориса–Пратта) постройте автомат для L и сравните его с $\min \mathcal{A}$.

Контрольные вопросы

Несмотря на название раздела, все решения задач должны быть строго обоснованы.

Задача 12. Верно ли, что если пересечение языков $L_1, L_2 \subseteq \{a, b\}^*$ содержит язык $F = \{a^n b^n \mid n \geq 1\}$: $F \subseteq L_1 \cap L_2$, то хотя бы один из языков L_1 и L_2 является нерегулярным?

Задача 13. Пусть $X_1, X_2, \dots, X_n, \dots$ – бесконечное семейство регулярных языков.

1. Верно ли, что язык $X = \bigcup_{n=1}^{\infty} X_n$ является регулярным языком?
2. Верно ли, что язык $X = \bigcap_{n=1}^{\infty} X_n$ является регулярным языком?

Задача 14. Язык L_1 объединили с конечным языком R и получили язык L ($L = L_1 \cup R$). Язык L оказался регулярным. Верно ли, что язык L_1 мог быть нерегулярным?

Задача 15. Язык задан контекстно-зависимой грамматикой, которая не является контекстно-свободной. Может ли он быть регулярным?

Приложение конечных автоматов для поиска подстрок

Алгоритм Кнута–Морриса–Пратта

Одно из простых и естественных применений регулярных выражений – поиск в тексте (слове) t вхождения некоторого слова

w . В терминах регулярных выражений задача состоит в проверке принадлежности слова t языку, порождаемому РВ $\Sigma^*w\Sigma^*$. Однако сходу придумать алгоритм, который выполняет эту проверку за линейное время, не очень просто. Таким алгоритмом является алгоритм Кнута–Морриса–Пратта (КМП), и здесь мы опишем его на языке автоматов. С самим алгоритмом можно познакомиться, например, в 10-й главе книги [6].

Для того чтобы описать КМП-алгоритм, нам потребуется сначала ввести понятие префикс-функции.

Определение 1. Назовём *префикс-функцией* функцию $l : \Sigma^* \rightarrow \Sigma^*$, которая возвращает самый длинный собственный³ префикс слова w , являющийся одновременно его суффиксом.

Пример 1. Приведём пример вычисления префикс-функции.

$$\begin{aligned} l(a^{n+1}) &= a^n & l(aabaaba) &= aabaa \\ l(ababa) &= aba & l(aabaa) &= aa \\ l(abb) &= \varepsilon \end{aligned}$$

У префикс-функции есть важное свойство – все собственные префиксы слова w , которые являются его суффиксами, лежат в последовательности $l(w), l(l(w)), \dots$

Зафиксируем слово w . Обозначим подслово $w_i w_{i+1} \dots w_j$ слова $w = w_1 w_2 \dots w_n$ через $w[i, j]$, здесь всюду $w_k \in \Sigma$. Для удобства будем считать, что $w[0, 0] = \varepsilon$, а $w[0, k] = w[1, k]$ при $k > 0$. Определим *автомат Кнута–Морриса–Пратта* (КМП-автомат) для слова w .

Определение 2. КМП-автомат \mathcal{A}_w для слова $w \in \Sigma^*$ длины n – это автомат, заданный набором $(Q, \Sigma, q_0, \delta, F)$, где

- $Q = \{ w[0, 0], w[0, 1], w[0, 2], \dots, w[0, n] \}$;

³То есть префикс, не совпадающий со всем словом w .

- $q_0 = w[0, 0]$;
- $\delta(w[0, k], a) = \begin{cases} w[0, k + 1] & \text{при } w[0, k + 1] = w[0, k]a \text{ и } k < n; \\ l(w[0, k]a) & \text{при } w[0, k + 1] \neq w[0, k]a \text{ и } k < n; \\ w[0, n] & \text{при } k = n. \end{cases}$
- $F = \{w[0, n]\}$.

Замечание 1. В качестве множества состояний КМП-автомата выступает множество слов, поэтому применение для состояний операций со словами, например « $w[0, k+1] = w[0, k]a$ », при определении функции переходов корректно и осмысленно.

Задача 16. Постройте КМП-автомат для слова *abababb* и продемонстрируйте его работу на слове *ababababb*.

Структура данных «Словарь»

Под словарём понимают структуру данных, с помощью которой можно проверять вхождение слова в множество. У этой структуры данных есть следующие операции:

- **in**: добавить слово x в словарь;
- **test**: проверить, входит ли слово x в словарь;
- **out**: удалить слово x из словаря.

Эту структуру данных можно реализовать с помощью автомата. На рис. 1 показан пример словаря с содержимым $S = \{a, ba, ab, abb, abab\}$, который реализован через ДКА. ДКА A принимает слово x тогда и только тогда, когда $x \in S$.

Задача 17. Постройте алгоритмы, реализующие операции **in** и **out** для ДКА, реализующих словарь. В результате этих операций должен получиться ДКА, который реализует словарь с соответственно изменившимся содержимым.

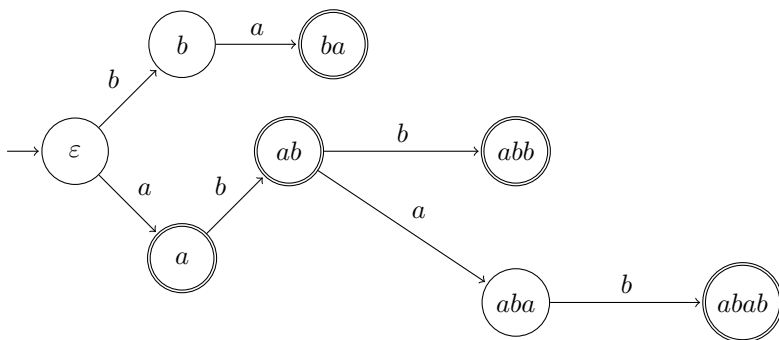


Рис. 1. ДКА \mathcal{A} реализует словарь

Задача 18. Постройте алгоритм, который получает на вход конечное множество слов S и возвращает ДКА, реализующий словарь с содержимым S . Используйте при построении только известные из курса алгоритмы для регулярных множеств (выражений) и конечных автоматов.

Замечание. В задачах, в которых требуется построить алгоритм, необходимо доказательство его корректности.

Задача 19. Постройте ДКА для словаря $\{ac, acb, b, ba, c, cbb\}$. Добавьте в полученный словарь слово ab и удалите слово ac .

Алгоритм Ахо-Корасик

Алгоритм Ахо-Корасик проверяет вхождение в текст t слов из заранее построенного словаря. Причём алгоритм находит все вхождения за линейное время!

Автомат для алгоритма Ахо-Корасик (рис. 2) построен следующим образом. Он содержит все состояния и переходы, что и соответствующий автомат для словаря (рис. 1), но помимо этого к принимающим состояниям относятся те состояния, некоторый

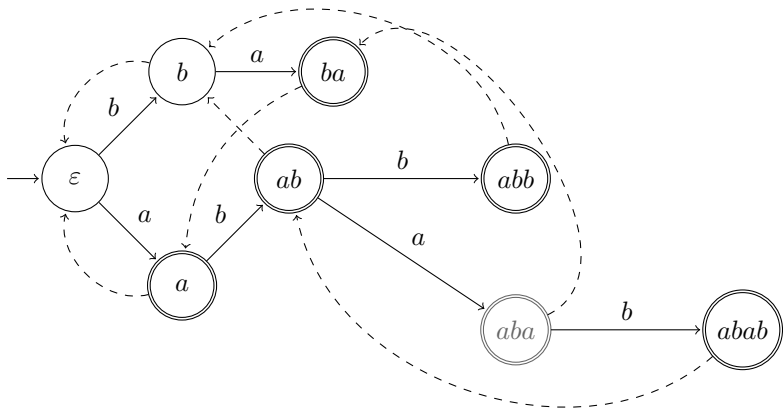


Рис. 2. Автомат Ахо-Корасик

суффикс которых является принимающим: состояние aba выделено серым, поскольку слово aba не лежит в словаре, однако в словаре лежит слово ba . Пунктирные переходы используются в случае сбоя: переход из состояния u (состояние = некоторое слово) по пунктирной линии осуществляется в случае сбоя: если из u нет перехода по a , то автомат переходит в состояние s , в которое ведёт пунктирная стрелка, и пытается перейти по a из s и т.д. Переход из u в s добавляется согласно следующему правилу. Слово s должно быть самым длинным суффиксом слова u ($u = ps$), который является состоянием автомата Ахо-Корасик. Такие переходы называют *суффиксными ссылками*.

Так, автомат Ахо-Корасик (рис. 2), по сути, является ДКА (рис. 3), граф которого получается из автомата Ахо-Корасик добавлением явных переходов, которые были заданы суффикс-

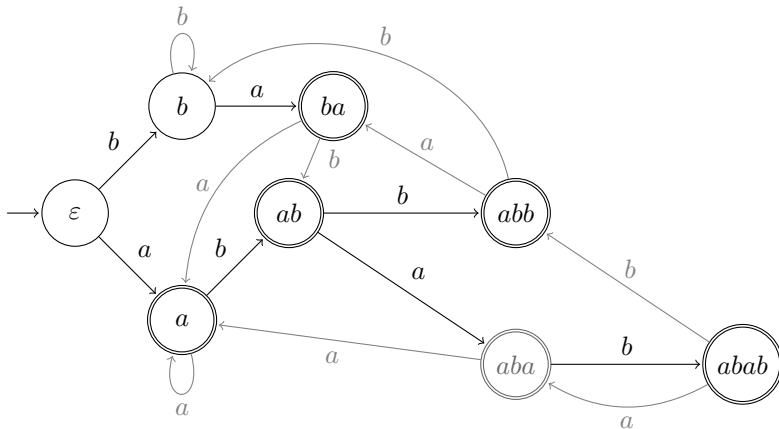


Рис. 3. ДКА Ахо-Корасик

ными ссылками. ДКА (и соответственно автомат Ахо-Корасик) оказывается в принимающем состоянии тогда и только тогда, когда в тексте на входе автомата встретилось слово из словаря (возможно несколько одновременно). Заметим, что количество одновременно встретившихся подслов текста из словаря зависит только от состояния автомата.

Непосредственное построение ДКА Ахо-Корасик, как правило, неэффективно для приложений – суффиксные ссылки позволяют существенно сократить таблицу переходов и уменьшить тем самым длину описания автомата.

Задача 20. Постройте для словаря $S = \{ac, acb, b, ba, c, cbb\}$ (который вы строили в предыдущем разделе) автомат Ахо-Корасик. Посчитайте с его помощью количество различных вхождений слов из словаря S в подслово $acbacbb$.

Контекстно-свободные языки

Задача 21. Язык $L^=$ является языком всех слов с равным числом символов a и b .

1. Покажите индукцией¹ по длине слова, что КС-грамматика с правилами $S \rightarrow SS \mid aSb \mid bSa \mid \varepsilon$ порождает язык $L^=$.

2*. Грамматика называется линейной, если в правые части правил вывода входит не более одного нетерминала. Покажите, что язык $L^=$ не порождается никакой линейной КСГ.

Задача 22. Палиндромами называют слова, которые одинаково читаются слева направо и справа налево, например, «ротор».

1. Покажите, что язык палиндромов в произвольном алфавите является КС-языком.

2. Покажите, что дополнительный язык (язык всех непалиндромов) также является КС-языком.

¹Другие доказательства, кроме индукции, не принимаются.

3. Покажите, что дополнительный язык к языку

$$U = \{a^n b^n c^n \mid n \geq 0\}$$

является КС-языком.²

Задача 23. Являются ли следующие языки КС-языками:

- 1) $SQ = \{ww \mid w \in \{a, b\}^*\}$,
- 2) $\Sigma^* \setminus SQ$,
- 3) $\{a^{3^n} \mid n > 0\}$?

Задача 24. Выполните следующие задания.

1. Постройте магазинный автомат (МА), распознающий язык L из задачи 16.

2*. Постройте детерминированный МА, распознающий тот же язык, и приведите доказательство его корректности по индукции.

Задача 25. Язык Дика с двумя типами скобок D_2 порождается грамматикой

$$S \rightarrow SS \mid (S) \mid [S] \mid \varepsilon.$$

1. Постройте недетерминированный МП-автомат, распознающий язык D_2 .

2. Постройте детерминированный МП-автомат, распознающий язык D_2 , и приведите доказательство его корректности по индукции.

²Так как сам язык U не является КС-языком, то это означает, что в отличие от регулярных языков множество КС-языков не замкнуто относительно дополнения.

Задача 26. Для языка

$$L = \{w \mid w = xcy; x, y \in \{a, b\}^*; |x| = |y|\}$$

- 1) постройте КС-грамматику G , порождающую язык L ;
- 2) постройте недетерминированный МА, эквивалентный грамматике G ;
- 3) продемонстрируйте работу построенного МА на слове $acab$ (проанализируйте все варианты поведения).

Задача 27. Заданы грамматика $G = \{ \{ A, B, C, D, E, F, S \}, \{a, b\}, \{S \rightarrow AB \mid C, A \rightarrow aE \mid a, E \rightarrow aE \mid \varepsilon, B \rightarrow bB \mid Bb \mid b, C \rightarrow CD, F \rightarrow ab, D \rightarrow aba\}, S \}$ и магазинный автомат $M = (\{q_0\}, \{a, b\}, \{S, a, b, A, B\}, \{\delta(q_0, \varepsilon, S) = \{(q_0, AB)\}, \delta(q_0, \varepsilon, A) = \{(q_0, aA), (q_0, a)\}, \delta(q_0, \varepsilon, B) = \{(q_0, bB), (q_0, b)\}, \delta(q_0, a, a) = \{(q_0, \varepsilon)\}, \delta(q_0, b, b) = \{(q_0, \varepsilon)\}, q_0, S\}$, принимающий слова опустошением магазина.

1. Эквивалентны ли грамматика G и N -автомат³ M ?
2. Однозначна ли грамматика G ? Если нет, то постройте эквивалентную ей однозначную грамматику.
3. Является ли автомат M детерминированным? Если нет, постройте эквивалентный ему детерминированный МА.

Задача 28. Определим языки $L_1 = \Sigma^*aab\Sigma^*$, где $\Sigma = \{a, b\}$, и

$$L = \{w \mid w \in \overline{L_1}, |w|_a \geq |w|_b\}.$$

1. Является ли дополнение языка L КС-языком?

³Мы называем N -автоматом МП-автомат, допускающий по пустому стеклу, а F -автоматом — МП-автомат, допускающий по принимающему состоянию.

2. Является ли дополнение языка L регулярным языком?

Задача 29. Язык L задан КС-грамматикой с правилами:

$$S \rightarrow aSa \mid aSb \mid bSa \mid bSb \mid a.$$

1. Является ли L регулярным языком?
2. Является ли дополнение L регулярным языком?
3. Является ли L КС-языком?
4. Является ли дополнение L КС-языком?

Задача 30. Язык L задан КС-грамматикой с правилами:

$$S \rightarrow aSb \mid A \mid B \mid \varepsilon, \quad A \rightarrow aAa \mid \varepsilon, \quad B \rightarrow bBb \mid \varepsilon.$$

1. Является ли L регулярным языком?
2. Является ли дополнение L регулярным языком?
3. Является ли L КС-языком?
4. Является ли дополнение L КС-языком?

Контрольные вопросы

Задача 31. КС-грамматика называется *левооднозначной*, если каждое слово порождаемого ею языка имеет единственный левый вывод. Аналогично определяется *правооднозначная грамматика*. Можно ли построить пример левооднозначной, но не правооднозначной КС-грамматики?

Задача 32. Пусть L_1 – КС-язык, не являющийся регулярным, а L_2 – не КС-язык. Может ли язык L_2L_1 быть регулярным языком? При положительном ответе привести пример.

Приложение КС-грамматик для сжатия данных

Некоторые алгоритмы сжатия строк можно описать в терминах КС-грамматик. Мы рассмотрим два таких алгоритма. Первый из них носит название «Straight-line program» (SLP) и состоит в следующем. Слово w описывают с помощью КС-грамматики G_w , которая порождает единственное слово: $L(G_w) = w$. Грамматику G_w называют «Straight-line grammar» (SLG); этим же термином иногда называют и описываемый нами частный случай метода сжатия SLP: в роли программ выступают КС-грамматики.

Пример 2. Грамматика, описываемая правилами

$$S \rightarrow A_1 A_1, A_1 \rightarrow A_2 A_2, A_2 \rightarrow A_3 A_3, \dots, A_{n-1} \rightarrow A_n A_n, A_n \rightarrow a,$$

порождает единственное слово a^{2^n} . Длина описания грамматики не превосходит cn , для некоторой константы $c > 0$, то есть имеет длину порядка логарифма от длины порождаемого слова, что является хорошим коэффициентом сжатия.

Задача 33. Постройте SLG G_n , порождающую слово

$$a^n b a^{n-1} b a^{n-2} b \dots a b a b a^2 b a^3 b \dots a^n b.$$

Длина описания G_n должна быть cn , $c > 0$. В качестве решения можно построить SLG G_5 .

Замечание 2. Преимуществом описанного метода сжатия является возможность эффективной проверки сжатого слова на регулярные события без разархивации. То есть существует алгоритм, получающий на вход описание НКА \mathcal{A} и SLP G_w и проверяющий непустоту пересечения $L(\mathcal{A}) \cap L(G_w)$ за полиномиальное время от длин описаний \mathcal{A} и G_w , но не w .

Задача 34*: Постройте описанный выше алгоритм и докажите его корректность.

Мы описали общий метод сжатия SLP, но не описали пока алгоритма сжатия строк в грамматики. Таких алгоритмов существует несколько, одним из популярных алгоритмов сжатия такого типа является алгоритм Лемпеля–Зива–Велча (Lempel–Ziv–Welch, LZW). Опишем работу этого алгоритма на примере сжатия конкретной строки: *aababbbbaabaabab*.

Таблица 1

Разбиение строки алгоритмом LZW

<i>a</i>	<i>ab</i>	<i>abb</i>	<i>b</i>	<i>ba</i>	<i>aba</i>	<i>abab</i>
A_1	A_2	A_3	A_4	A_5	A_6	A_7

Таблица 1 представляет собой словарь. Она устанавливает взаимно однозначное соответствие между нетерминалами и словами: слово w_i в построенной в итоге грамматике будет выводимо из A_i и только из A_i (но не обязательно за один шаг вывода). Опишем алгоритм заполнения таблицы-словаря.

1. В начале работы словарь пуст, слово u – необработанный суффикс слова w – совпадает с w , $i = 1$.
2. Алгоритм ищет максимальный префикс x необработанной части входа u , который был добавлен в словарь.
3. Если $u = xav$, $a \in \Sigma$, то алгоритм добавляет в словарь слово $w_i = xa$, удаляет префикс xa из u , увеличивает i на 1 и переходит к предыдущему шагу, если $u \neq \varepsilon$. Если же $u = \varepsilon$, алгоритм заканчивает работу.

4. Если $u = x$ и x уже соответствует некоторому нетерминалу A_j , то алгоритм добавляет в грамматику правило $A_i \rightarrow A_j$ и завершает работу. Обратим внимание, что x на этом шаге является суффиксом w .

Так, первая буква слова w всегда будет приписана нетерминалу A_1 ; далее в нашем примере за первой a идёт подслово ab , которое приписывается нетерминалу A_2 , поскольку первая буква подслова a уже была приписана A_1 ; далее идёт подслово abb – подслово ab уже было приписано A_2 и т.д.

Нетрудно заметить, что искомая SLG имеет вид

$$S \rightarrow A_1 A_2 \dots A_7, \quad A_1 \rightarrow a, \quad A_2 \rightarrow A_1 b, \quad A_3 \rightarrow A_2 b,$$

$$A_4 \rightarrow b, \quad A_5 \rightarrow A_4 a, \quad A_6 \rightarrow A_2 a, \quad A_7 \rightarrow A_6 b.$$

Но как её эффективно построить алгоритмически, равно как и таблицу 1? Для этого воспользуемся техникой, базирующейся на конечных автоматах.

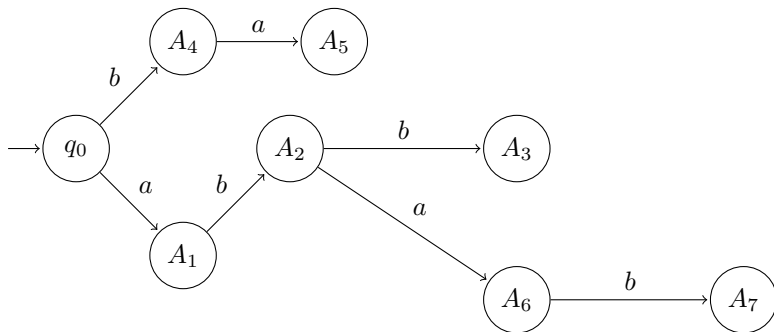


Рис. 2. LZW-автомат

В процессе построения SLG по алгоритму LZW мы строим LZW-автомат (рис. 2), который, по сути, реализует словарь. Од-

нако, помимо стандартных функций словаря, LZW-автомат пометает каждую вершину, кроме начальной, нетерминалом A_i , устанавливая тем самым соответствие между словом w_i и состоянием автомата: $q_0 \xrightarrow{w_i} A_i$.

Итак, опишем алгоритм LZW построения SLG. В начале работы алгоритма словарь (реализуемый LZW-автоматом) пуст; обозначим через u необработанную часть входа – в начале работы $u = w$.

Алгоритм находит кратчайший префикс $x \neq \varepsilon$ слова $u = xy$, которого ещё нет в словаре, и добавляет его в словарь, пометая вершину, соответствующую этому слову новым нетерминалом A_i . Алгоритм повторяет этот процесс, удалив из u префикс x ($u = y$), до тех пор, пока либо слово u не окажется пустым, либо u не будет содержаться в словаре. При этом если префикс x добавляется в словарь, то $x = va$, $a \in \Sigma$, а слово v уже было добавлено в словарь и ему соответствует некоторый нетерминал A_j . Тогда алгоритм добавляет в грамматику переход $A_i \rightarrow A_j a$. Если же u целиком содержится в словаре, то ему уже соответствует нетерминал A_j – в этом случае алгоритм добавляет правило $A_i \rightarrow A_j$. После окончания построения LZW-автомата алгоритм добавляет к грамматике правило $S \rightarrow A_1 \dots A_n$, где n – номер последнего добавленного нетерминала.

Приведённый алгоритм, очевидно, работает за линейное время (от длины w). Строку, сжатую алгоритмом LZW, легко декодировать, как и строку, заданную произвольной SLG: нужно вывести единственную строку из грамматики, при этом каждый нетерминал раскрывается единственным образом. Также для алгоритма LZW справедливо замечание 2.

Задача 35. Постройте LZW-автомат и SLG G_w по описанному выше алгоритму для слова w :

- 1) $w = a^8$; 2) $w = \text{tobeornottobeortobeornot}$.

Задача 36. Постройте для слова $w = \text{tobeornottobeortobeornot}$ SLG, которая оптимальнее, чем построенная по алгоритму LZW.

Численным показателем оптимальности является сумма длин правых частей всех правил SLG.

Элементы синтаксического анализа

LL-анализ

Задача 37. Определить, являются ли $LL(k)$ -грамматиками следующие грамматики (заданные правилами). Если да, указать точное значение k :

- а) $S \rightarrow Ab, \quad A \rightarrow Aa \mid a;$
б) $S \rightarrow Ab, \quad A \rightarrow aA \mid a;$
в) $S \rightarrow aAb, \quad A \rightarrow BB, \quad B \rightarrow ab \mid A \mid \varepsilon;$
г) $S \rightarrow aAb, \quad A \rightarrow AaAb \mid \varepsilon;$
д) $S \rightarrow aB, \quad B \rightarrow aBB \mid b.$

Задача 38. Построить $LL(1)$ -грамматику, эквивалентную грамматике из задачи **26(б)**, и управляющую таблицу для неё.

Задача 39. Написать для грамматики эквивалентную LL(1)-грамматику, построить LL(1)-анализатор и продемонстрировать его работу на слове $baab$.

$$S \rightarrow baaA \mid babA \quad A \rightarrow \varepsilon \mid Aa \mid Ab$$

Задача 40*. Докажите, что язык $a^* \cup a^n b^n$ не является LL(1)-языком, то есть не существует LL(1)-грамматики, порождающей этот язык.

Задача 41. Язык L задан неоднозначной КС-грамматикой

$$G = \{\{S\}, \{(,)\}, \{S \rightarrow (S) \mid SS \mid ()\}, S\}.$$

Написать LL(1)-грамматику для языка L .

Контрольные вопросы

Задача 42. Существует ли такая праволинейная (не обязательно регулярная праволинейная) грамматика, которая не является LL(1)-грамматикой?

Задача 43. В приведённой грамматике¹ G есть правило $S \rightarrow AB$ и при этом $\text{FIRST}(A) \cap \text{FIRST}(B) = \varepsilon$. Верно ли, что грамматика G может быть LL(1)-грамматикой?

Задача 44. Пусть для некоторых двух нетерминалов A и B приведённой КСГ G пересечение $\text{FOLLOW}(A) \cap \text{FOLLOW}(B)$ оказалось непустым. Верно ли, что грамматика G не является LL(1)-грамматикой?

¹Грамматика называется приведённой, если в ней нет недостижимых и бесплодных символов. В литературе также встречаются неэквивалентные определения этого термина.

LR-анализ

Задача 45. Дана грамматика

$$G = \{ \{A, S\}, \{a, b, c\}, \{ S \rightarrow Aa \mid b \mid \varepsilon; A \rightarrow Ab \mid c \}, S \}.$$

Является ли грамматика G LR(k)-грамматикой? При положительном ответе на вопрос найти минимальное k и построить соответствующий анализатор. Построить дерево разбора для цепочки $cbba$.

Задача 46. Дана грамматика

$$G = \{ \{A, S\}, \{a\}, \{ S \rightarrow A; A \rightarrow aAa \mid a \}, S \}.$$

Является ли грамматика G LR(k)-грамматикой? При положительном ответе на вопрос найти минимальное k и построить соответствующий анализатор. Построить дерево разбора для цепочки $aaaaa$.

Задача 47. Дана грамматика

$$G = \{ \{A, S\}, \{a, b, c\}, \{ S \rightarrow Aa \mid b; A \rightarrow Ab \mid c \}, S \}.$$

Является ли грамматика G LR(k)-грамматикой? При положительном ответе на вопрос найти минимальное k и построить соответствующий анализатор. Продемонстрировать работу анализатора на цепочке $cbbab$.

Задача 48. Зафиксируем КС-грамматику G и рассмотрим множество её LR(0)-ситуаций. Будем говорить, что между двумя ситуациями $\alpha.X\beta$ и $\alpha X.\beta$ определён переход по $X \in N \cup T$; также между ситуациями $A \rightarrow \alpha.B\beta$ и $B \rightarrow \cdot\gamma$ определён ε -переход.

Конечный автомат, в качестве состояний которого выступают LR(0)-ситуации, а переходы определены по правилам, указанным выше, называют (недетерминированным) LR(0)-автоматом или (недетерминированным) автоматом Кнута. Автомат, полученный в результате детерминизации описанного автомата, называют детерминированным LR(0)-автоматом или детерминированным автоматом Кнута.

1. Выпишите все LR(0)-ситуации для грамматики G , заданной правилами $S \rightarrow aS \mid b$.
2. Постройте недетерминированный автомат Кнута для грамматики G .
3. Постройте детерминированный автомат Кнута для грамматики G .
4. Постройте LR(0)-анализатор для грамматики G . Сравните автомат Кнута с таблицей переходов LR(0)-анализатора для грамматики G .

Задача 49. Грамматика G задана правилами:

$$S \rightarrow Ab, \quad A \rightarrow aAa, \quad A \rightarrow B, \quad B \rightarrow b.$$

1. Построить LR(1) и LR(0)-анализаторы для грамматики G по алгоритму из курса.
2. Постройте LR(0)-анализатор по LR(1)-анализатору из пункта 1 следующим образом. Сотрите все аванпечки и постройте управляющую таблицу LR(0)-анализатора по получившемуся автомату Кнута. Верно ли, что полученный LR(0)-анализатор является анализатором для грамматики G ? То есть для любого слова, порождаемого G , анализатор строит корректный правый разбор, а слова, не порождаемые G , анализатор отвергает.
3. Покажите, что LR(0)-анализатор для грамматики G из пункта 1 можно построить путём применения следующей процедуры,

схожей с процедурой минимизации ДКА, к LR(0)-автомату, полученному из LR(1)-анализатора в пункте 2.

В случае минимизации LR(0)-автомата все состояния с операциями свёртки оказываются на первом шаге в разных группах (разных ящиках), если свёртки происходят по разным правилам; состояния с операциями сдвига находятся в одном ящике. Далее процедура минимизации LR(0)-автомата не отличается от процедуры минимизации ДКА.

Контрольные вопросы

Задача 50. При построении LR(1)-анализатора для грамматики G в одном множестве оказались ситуации $[A \rightarrow .aA\alpha, b]$ и $[B \rightarrow \beta.a, a]$, где α, β – некоторые цепочки из $(N \cup T)^*$. Может ли грамматика G оказаться LR(0)-грамматикой?

Атрибутные грамматики

Следующим за синтаксическим анализом этапом в процессе компиляции является генерация кода. В основе этого этапа лежат вычисления по дереву разбора, которые описывают с помощью атрибутных грамматик. Мы не будем детально изучать эту тему, а изучим лишь частный случай атрибутных грамматик (с синтезируемыми атрибутами).

Определение 3. КС-грамматика G называется *атрибутной с синтезируемыми атрибутами*, если каждому нетерминалу поставлен в соответствие набор переменных (атрибутов), и при этом для каждого правила грамматики

$$X_0 \rightarrow u_0 X_1 u_1 X_2 \dots u_{n-1} X_n u_n, X_i \in N, u_i \in \Sigma^*$$

задан набор правил вычисления некоторых атрибутов

$$X_0[\text{attr}] = f(X_1[\text{attr}_1], X_2[\text{attr}_2], \dots, X_n[\text{attr}_n]);$$

здесь $X_i[\text{attr}_i]$ – значение атрибута attr_i для нетерминала X_i , а f – некоторая функция. Набор правил вычислений атрибутов называют *атрибутной схемой*.

Замечание 3. В книге [3] используются другие обозначения для атрибутных грамматик (традиционные для этой области).

Вместо $X_i[\text{attr}]$ пишут $\text{attr}(X_i)$. Кроме того, там рассмотрен более общий случай атрибутивных грамматик – с синтезируемыми и наследуемыми атрибутами. Выполняя задание, можно придерживаться одного из двух типов обозначений (заметим, что они непротиворечивы). Однако, пожалуйста, не смешивайте эти два набора.

Пример 3. Грамматика G задана правилами

$$S \rightarrow 1D \mid 0, D \rightarrow 1D \mid 0D \mid 1 \mid 0$$

и порождает язык двоичных записей натуральных чисел. Определим атрибутивную схему для этой грамматики:

$$\begin{array}{llll} S \rightarrow 0 & D \rightarrow 1 & D \rightarrow 0 & S \rightarrow 1D \\ S[\text{val}] = 0 & D[\text{val}] = 1 & D[\text{val}] = 0 & S[\text{val}] = D[\text{ord}] + D[\text{val}] \\ & D[\text{ord}] = 2 & D[\text{ord}] = 2 & \end{array}$$

$$\begin{array}{ll} D_0 \rightarrow 1D_1 & D_0 \rightarrow 0D_1 \\ D_0[\text{val}] = D_1[\text{ord}] + D_1[\text{val}] & D_0[\text{val}] = D_1[\text{val}] \\ D_0[\text{ord}] = 2 \times D_1[\text{ord}] & D_0[\text{ord}] = 2 \times D_1[\text{ord}] \end{array}$$

В случае, если правило содержит несколько одинаковых нетерминалов, мы нумеруем их вхождение и различаем атрибуты, как в случае двух последних правил. Нетерминал S имеет единственный атрибут val , а нетерминал D – атрибуты val и ord . Приведённая атрибутивная схема вычисляет значение числа по его двоичной записи. Атрибут ord равен 2^l , где l – длина слова, выведенного из D , атрибут val равен значению числа, двоичная запись которого выведена из нетерминала.

Приведём пример вычисления атрибутов для слова 1101. Атрибуты вычисляются снизу вверх.

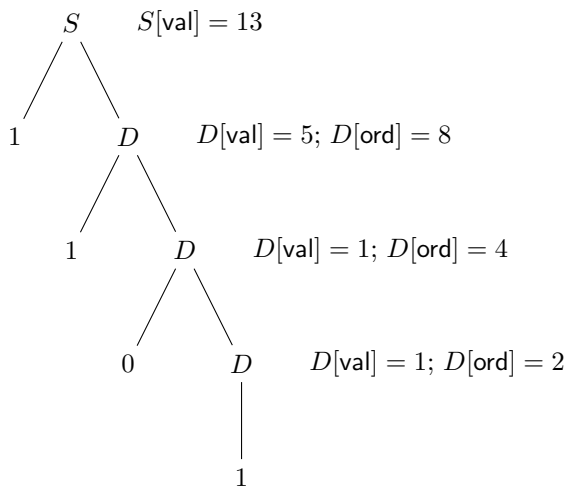


Рис. 1. Вычисление атрибутов

Задача 51. Грамматика $\text{RExpr} = \langle \{E, T\}, \{a, b, +, *, (,)\}, P, E \rangle$ имеет множество правил P :

$$E \rightarrow (E)^* \mid T + T \mid TT \quad T \rightarrow (E) \mid (E)^* \mid a \mid b$$

и порождает регулярные выражения над алфавитом $\{a, b\}$.

1. Постройте для грамматики RExpr LR(1)-анализатор.
2. Дополните грамматику RExpr до атрибутной так, чтобы она вычисляла атрибуты `firstpos`, `lastpos` и `nullable` согласно алгоритму их вычисления при построении ДКА по РВ.

Указание. Вычисление атрибута нужно определять через описание функции, которую можно описать на языке программирования или псевдокоде. Пример вычисления атрибута `nullable` для правила $E \rightarrow T_1 + T_2$:

```

E[nullable] = function( T1[nullable], T2[nullable] ){
    if( T1[nullable] or T2[nullable] ){
        return True;
    } else{ return False; }
}

```

- 3*. Добавьте в атрибутную схему вычисление атрибута `followpos`.
4. Воспользовавшись анализатором, постройте дерево разбора для РВ $(a + ab)^* + a(b)^*$ и вычислите атрибуты `firstpos`, `lastpos`, `nullable` (и `*followpos`) согласно атрибутной схеме.

Дополнительные задачи

В этот раздел входят задачи для подготовки к контрольным работам и экзаменам, а также задачи повышенной сложности для студентов, претендующих на высокие оценки. Задачи данного раздела не являются обязательными для прохождения процедуры сдачи задания, если только не входят в требования семинариста. Во всех письменных общекурсовых работах значение k в задачах на построение LR(k)-анализаторов не превосходит единицу.

Регулярные языки

Задача 52. Пусть X – регулярный язык над алфавитом $\Sigma = \{a, b\}$. Верно ли, что язык $\bigcap_{n=1}^{\infty} (\Sigma^* \setminus X)^n$ является регулярным?

Задача 53. Приведите пример бесконечного регулярного языка X над алфавитом $\Sigma = \{a, b\}$, такого что $X \cap (\Sigma^* \setminus X)^R = X$ и $X \neq \Sigma^*$.

Задача 54. Найдите разбиения на минимальное число классов

правоинвариантной¹ (И/ИЛИ левоинвариантной) эквивалентности, которые индуцируют следующие языки.

1. Язык, порождаемый выражением $00(10 + 01)^*$.
2. Язык $\{a^{n^2} \mid n \geq 0\}$ в однобуквенном алфавите.

КС-языки

Задача 55. Язык L задан грамматикой G с правилами:

$$S \rightarrow bSa \mid AB \mid \varepsilon, \quad A \rightarrow bAb \mid b, \quad B \rightarrow aBa \mid \varepsilon.$$

Является ли язык L и его дополнение регулярным языком, КС-языком?

Задача 56. Являются ли следующие языки КС-языками:

- 1) $\{x \mid x \in \{c, b\}^*, |x|_c = |x|_b, \forall u, v : x = uv, |u| \neq 0, |v| \neq 0, |u|_c > |u|_b\}$,
- 2) $\{a^{3^n} \mid n > 0\}$?

Задача 57*. Пусть A – МА. Постройте МА B , принимающий все префиксы языка $L(A)$, т.е.

$$L(B) = \{x \mid \exists y \in \Sigma^* : xy \in L(A)\} \subseteq \Sigma^*.$$

Задача 58. Для языка

$$L = \{w \mid w = xc^{3k}y; x, y \in \{a, b\}^*; |xy|_a = 2n; n, k \geq 0\}$$

¹Правоинвариантная эквивалентность – другое название эквивалентности Майхилла–Нероуда. Левоинвариантная эквивалентность определяется симметрично правоинвариантной.

($|xy|_a$ – число символов a в слове xy)

- 1) постройте КС-грамматику G , порождающую язык L ;
- 2) постройте недетерминированный МА, эквивалентный КСГ G ;
- 3) продемонстрируйте работу построенного МА на слове $accab$ (проанализируйте все варианты поведения).

Задача 59. Заданы языки $L_1 = \{a^n b^n c^m : n \geq 1, m \geq 0\}$, $L = \{f^n a^m b^m : n \geq 0, m \geq 0\}$. Для языка $L_1 \cup L_2$ построить однозначную КС-грамматику и детерминированный МП-автомат. Решение обосновать.

Элементы синтаксического анализа

Задача 60. Язык L задан неоднозначной КС-грамматикой $G = \{ \{S\}, \{a, \cdot, \wedge, [,], (,)\}, \{S \rightarrow a \mid S.S \mid S[S] \mid S^\wedge \mid S(S)\}, S \}$.

Написать LL(1)-грамматику для языка L .

Задача 61. Дана грамматика $G = \{ \{A, B, C, D, E, S\}, \{a, b\}, \{S \rightarrow AB, A \rightarrow a, B \rightarrow CD \mid aE, C \rightarrow ab, D \rightarrow bb, E \rightarrow bba\}, S \}$. Является ли грамматика G LR(k)-грамматикой? При положительном ответе на вопрос найти минимальное k и построить соответствующий анализатор. Построить дерево разбора для слова $aabbb$ с помощью анализатора.

Задание содержит авторские задачи педагогического коллектива кафедры МОУ и классические задачи теории формальных языков.

С методическими материалами по курсам кафедры МОУ можно ознакомиться на страницах:

<http://www.mou.mipt.ru>, <http://trpl7.ru>,
<http://lrk.umeta.ru>, <http://rubtsov.su>.