# Задание 2
## НКА и алгоритмы поиска подстрок

**Литература:**

1. *Хопкрофт Д., Мотвани Р., Ульман Д.*
Введение в теорию автоматов, языков и вычислений.
М.: Вильямс, 2002.

2. *Ахо А., Ульман Д.*
Теория синтаксического анализа, перевода и компиляции
М.: Мир, 1978. Гл. 0, 2.

3. *Серебряков В.А., Галочкин М.П., Гончар Д.Р., Фуругян М.Г.*
Теория и реализация языков программирования.
М.: МЗ-пресс, 2006.

4. *Шень. А. Х.*
Программирование: теоремы и задачи
М.: МЦНМО, 2004.

5. *Журавлёв Ю.И., Флёров Ю.А, Вялый М.Н.*
Дискретный Анализ. Формальные системы и алгоритмы.
М.: МЗ-пресс, 2010.

**Ключевые слова**[1]: *язык, регулярные выражения, конкатенация, объединение, итерация, конечные автоматы (КА), детерминированные и недетерминированные КА, регулярные языки.*

Упражнения из этого задания не обязательно техать и присылать: они не оцениваются. Их стоит техать, если вы хотите получить обратную связь. Задачи, помеченные звёздочкой, являются необязательными, но их решение сильно поощряется бонусными очками.

# 1 Построение по регулярному выражению конечного автомата

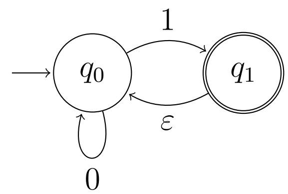
На семинаре мы разобрали алгоритм построения детерминированного конечного автомата по регулярному выражению. Однако, его нельзя на-

[1] минимальный необходимый объём понятий и навыков по этому разделу)

звать простым. Построить недетерминированный автомат по регулярному выражению гораздо проще. В каком-то смысле, если вы имеете дело с регулярным выражением, вы имеете дело с НКА.

Напомним, что помимо обычных переходов недетерминированные автоматы, имеют также $\varepsilon$-переходы, т.е. переходы вида $\delta(q_i, \varepsilon) = q_j$. Наличие таких переходов означает, что попав в состояние $q_i$, автомат может перейти в состояние $q_j$ не обрабатывая следующий символ слова.

**Пример 1.**

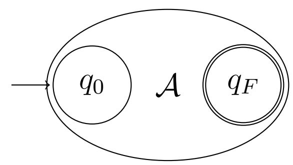



Легко видеть, что данный автомат принимает язык, состоящий из слов, оканчивающихся на 1. При прочтении 1, автомат переходит из состояния $q_0$ в $q_1$, дальше, если во входном слове ещё остались необработанные символы, автомат делает $\varepsilon$-переход из состояния $q_1$ в $q_0$ и продолжает обработку слова.

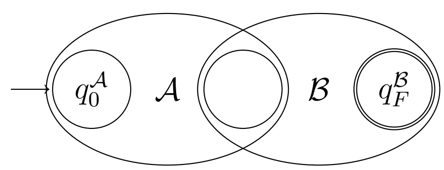
Для построения НКА по РВ будем использовать определение регулярного языка. Напомним определение класса регулярных языков $\mathsf{REG}$.

- $\varnothing \in \mathsf{REG}$.
- $\forall \sigma \in \Sigma : \ \{\sigma\} \in \mathsf{REG}$.
- $\forall X, Y \in \mathsf{REG} \ : \ X \cdot Y, \ X|Y, \ X^* \in \mathsf{REG}$.
- Больше нет регулярных языков.

Мы будем строить НКА по РВ из каждого пункта данного определения. С первыми двумя пунктами проблем нет – их я оставляю как лёгкое упражнение. Перейдём сразу к третьему пункту. Допустим уже построены автоматы $\mathcal{A}$ и $\mathcal{B}$ для регулярных языков $X$ и $Y$ соответственно. Мы будем предполагать, что оба автомата имеют всего одно принимающее состояние. Если в автомате несколько принимающих состояний, то можно построить эквивалентный ему автомат с единственным принимающим состоянием, добавив к множеству состояний состояние $q_F$, которое будет единственным принимающим, и добавив $\varepsilon$-переходы в $q_F$ из старого множества $F$: $\forall q \in F : \delta(q, \varepsilon) = q_F$. Будем схематично обозначать автоматы эллипсами, и помечать в них только начальное и принимающее состояние. Таким образом, автомат $\mathcal{A}$ имеет вид



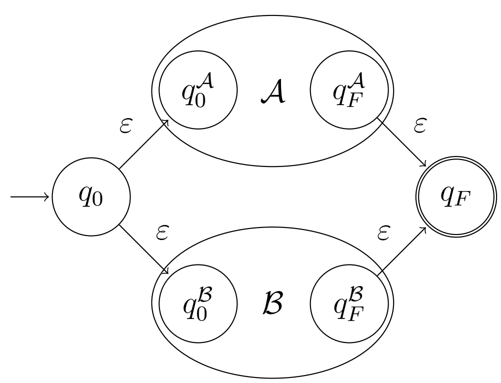

В дальнейшем, мы будем предполагать, что начальное состояние на схеме находится слева, а принимающее справа. Построим явно автомат распознающий $X \cdot Y$, $L(\mathcal{A}) = X, L(\mathcal{B}) = Y$.




Для этого по автомату $\mathcal{A}$ распознающему язык $X$ и автомату $\mathcal{B}$, распознающему язык $Y$ мы строим автомат, распознающий $X \cdot Y$ объединяя множества состояний $\mathcal{A}$ и $\mathcal{B}$ так, что $q_0 = q_0^{\mathcal{A}}$, $q_F^{\mathcal{A}} = q_0^{\mathcal{B}}$, $F = \{q_F^B\}$. Опять получили автомат с единственным принимающим состоянием.

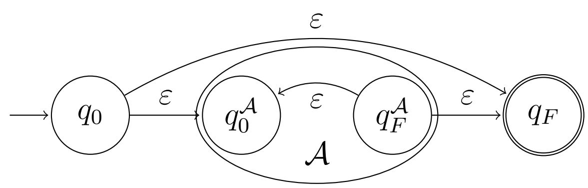
**Упражнение 1.** Доказать, что построенный автомат распознаёт язык $X \cdot Y$.

Для построения языка $X|Y$ используем следующую конструкцию:



**Упражнение 2.** Доказать, что построенный автомат распознаёт язык $X|Y$.

И наконец перейдём к построению автомата для языка $X^*$:

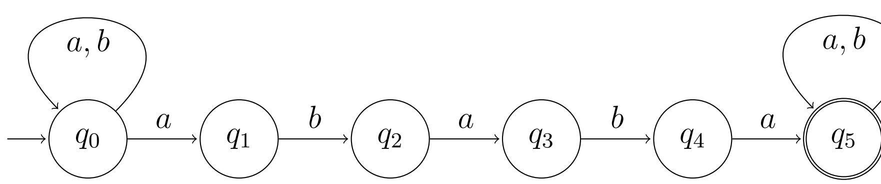



**Упражнение 3.** Доказать, что построенный автомат распознаёт язык $X^*$.

**Задача 1.** Постройте НКА по регулярному выражению $(a(a|b))^*b$.

## 2 Распознавание текстов

С помощью НКА можно легко описать язык, в который входят все слова, содержащие в качестве подслова некоторое слово. Например, следующий автомат распознаёт слова, содержащие подслово $ababa$.




**Задача 2*.** Докажите, что по НКА данного вида ($\Sigma^* w \Sigma^*$) можно построить ДКА, число состояний которого не превосходит $|w|+1$.

**Задача 3.** Постройте НКА, распознающий слова, в которых есть хотя бы одно из подслов $abba, abab, baa$.

# 3 Детерминированный конечный автомат

В этом разделе представлены задачи на построение автоматов и изучение свойств ДКА. В случае, когда речь идёт об автомате $\mathcal{A}$, для сокращения записи мы будем подразумевать, что данный автомат задан набором

$$\mathcal{A} = (Q_\mathcal{A},\ \Sigma,\ q_0^\mathcal{A},\ \delta_\mathcal{A},\ F_\mathcal{A}).$$

**Задача 4.** Постройте ДКА, который

1) распознаёт язык, все слова которого содержат чётное число нулей;

2) распознаёт язык, все слова которого содержат нечётное число единиц;

3) распознаёт язык, все слова которого содержат чётное число нулей и нечётное число единиц.

Из определения регулярных языков следует, что они замкнуты относительно операции объединения. Следующий алгоритм показывает, что регулярные языки замкнуты относительно пересечения. Пусть заданы ДКА $\mathcal{A}$ и $\mathcal{B}$. Построим ДКА $\mathcal{C}$, распознающий язык $L(\mathcal{A}) \cap L(\mathcal{B})$, следующим образом:

- $Q_\mathcal{C} = Q_\mathcal{A} \times Q_\mathcal{B}$;
- $q_0^\mathcal{C} = (q_0^\mathcal{A}, q_0^\mathcal{B})$;
- $\forall \sigma \in \Sigma : \delta_\mathcal{C}((q_\mathcal{A}, q_\mathcal{B}), \sigma) = (\delta_\mathcal{A}(q_\mathcal{A}, \sigma), \delta_\mathcal{B}(q_\mathcal{B}, \sigma))$;
- $F_\mathcal{C} = F_\mathcal{A} \times F_\mathcal{B}$

Конструкции такого типа – в которых состояния итогового объекта являются парами исходных, а функция переходов согласована с исходными объектами, – называются *конструкциями произведения.*

**Упражнение 4.** Докажите корректность данного алгоритма.

**Задача 5.** Заменим в конструкции произведения для пересечения регулярных языков последний пункт на

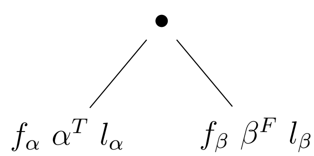
- $F_\mathcal{C} = F_\mathcal{A} \times Q_\mathcal{B} \cup Q_\mathcal{A} \times F_\mathcal{B}$.

Верно ли, что тогда автомат $\mathcal{C}$ распознаёт объединение языков $L(\mathcal{A}) \cup L(\mathcal{B})$?

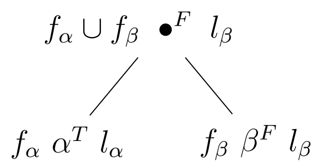
**Задача 6.** Постройте автомат из задачи **4(3)** используя автоматы, построенные в первых двух пунктах и конструкцию произведения.

# 4 О построении ДКА по РВ

На семинаре мы изучили алгоритм построения ДКА по РВ. На первом этапе мы строили дерево и вычисляли значения атрибутов *firstpos*, *lastpos* и *nullable*. Принцип вычисления этих атрибутов состоит в том, что имея значение атрибутов в дочерних узлах, мы можем вычислить атрибуты самого узла. Например для узла




атрибуты узла $\bullet$ будут




потому что регулярное выражение $\alpha \cdot \beta$ порождает слово $w = uv$, такое что РВ $\alpha$ порождает $u$, РВ $\beta$ порождает $v$, при этом слово $u$ может быть пустым, а слово $v$ – нет. Отсюда, если $u \neq \varepsilon$, то $w$ начинается с тех же символов, что и $u$, а значит $f_\alpha \subseteq f_\bullet$, а если $u = \varepsilon$, то $w$ начинается с тех же символов, что и слово $v$, поэтому $f_\beta \subseteq f_\bullet$, отсюда $f_\bullet = f_\alpha \cup f_\beta$. Поскольку слово $w$ имеет непустой суффикс $v$, порождённый $f_\beta$, то $l_\bullet = l_\beta$.

**Упражнение 5.** Найти алгоритм вычисления атрибутов во всех остальных случаев для узлов с операциями $\bullet$ и $|$. Если не получается придумать алгоритм, его можно найти в книге Серебрякова. Доказать корректность вычисления атрибутов.

**Задача 7*.** Мы рассматривали алгоритм построения ДКА по РВ, в котором не встречается пустое слово. В случае когда оно встречается, исходное РВ $R$ может быть преобразовано либо в выражение $R'$, либо в выражение $R' \,|\, \varepsilon$, где в выражение $R'$ пустое слово уже не входит. Предложите алгоритм, который осуществляет такое преобразование.

**Задача 8*.** Мы рассматривали алгоритм построения ДКА по РВ, в котором не встречается пустое множество. В случае когда оно встречается, исходное РВ $R$ может быть преобразовано либо в выражение $R'$, либо в выражение $R' \mid \varepsilon$, причём пустое множество и постое слово в выражение $R'$ не входят. Предложите алгоритм, который осуществляет такое преобразование.

**Указание:** Введите новый атрибут в дерево.