# Домашнее задание от 28.09.
# Теоретическая информатика. 2 курс.
# Решения.

Глеб Минаев @ 204 (20.Б04-мкн)

7 октября 2021 г.

## Содержание



**Задача 2.** Пусть ДКА $A = (\Sigma, Q, q_0, \delta, F)$ реализует язык $L$. Рассмотрим ДКА

$$B := (\Sigma, Q^Q \times \{0; \dots; Q + Q!\}, q_0' := (\mathrm{Id}_Q, 0), \eta, \{(f, r) \in Q^Q \times \{0; \dots; Q + Q!\} \mid f^r(q_0) \in F\}),$$

где

$$\eta((f, r), a) := (\delta_a \circ f, r'), \qquad \text{где} \qquad r' = \begin{cases} r + 1 & \text{если } r + 1 \leqslant Q + Q! \\ r + 1 - Q! & \text{если } r + 1 > Q + Q!. \end{cases}$$

Тогда несложно видеть, что $\eta^*(q_0', w) = (\delta_w, n)$, где $n$ — ближайшее к $|w|$ число, что $n \leqslant |w|$, $n \equiv |w| \pmod{Q!}$ и $0 \leqslant n \leqslant Q + Q!$. Заметим, что $w^{|w|} \in L$ тогда и только тогда, когда $\delta_w^{|w|}(q_0) \in F$. Но $\delta_w$ — есть отображение из $Q$ в себя, а $Q$ конечно. Значит если мы будем следить за орбитой $q_0$, то рано или поздно она зациклится, т.е.

$$(\delta_w^0(q_0), \delta_w^1(q_0), \delta_w^2(q_0), \dots) = (p_0, \dots, p_k, c_1, \dots, c_m, c_1, \dots, c_m, \dots).$$

Отсюда понятно, что если $|w| \geqslant k + m + 1$, то $\delta_w^{|w|}(q_0) = \delta_w^{|w|-m}(q_0)$. Следовательно, если $|w| > Q + Q!$, то $f^{|w|}(q_0) = f^{|w|-Q!}(q_0)$. Поэтому $\delta_w^{|w|}(q_0) = \delta_w^n(q_0)$. Значит $w^{|w|} \in L$ тогда и только тогда, когда $w$ принимается машиной $B$. Таким образом

$$L(B) = f(L).$$

---

**Задача 3.** Уточним условие задачи: $|\Sigma| \geqslant 2$; покажем, что иначе задача неверна. Пусть $\Sigma = \{a\}$. Пусть также дан НКА $A$ с $n$ состояниями, реализующий такой язык, который не реализует ни один ДКА с $< 2^n$ состояниями. Рассмотрим ДКА $A'$, построенный по $A$ по алгоритму с лекции.

- В $A'$ ровно $2^n$ состояний. Значит он минимальный, а поэтому у него все состояния достижимы, т.е. для $A$ верно, что достижимо всякое его подмножество состояний.
- Понятно, что в $A$ все состояния достижимы, так как иначе их можно убрать и ещё раз сгенерировать $A'$, и тогда у последнего будет не более $2^{n-1} < 2^n$ состояний.

- Заметим, что если $A$ содержит цикл (в том числе длины 1), то (так как у нас имеется только один единственный символ в алфавите, а цикл достижим) считывая буквы $a$, мы будем сначала идти к циклу, а потом будем ходить по нему кругами. А тогда пустое множество состояний $A$ недостижимо. Значит в $A$ нет циклов.

- Значит множество состояний $A$ можно линейно упорядочить, что все переходы по $a$ будут только направо. Но тогда через ровно $n$ считываний мы не будем находится ни в одном состоянии: сначала мы точно уйдём из первого (самого левого) состояния и к нему не вернёмся, затем из второго, затем из третьего, и т.д. за $n$ шагов мы уйдём из всех состояний. Т.е. в $A'$ достижимо не более $n+1$ состояний.

Таким образом имеем противоречие.

Значит будем решать задачу для $\Sigma = \{a; b\}$ (если алфавит больше, то это не ничего не испортит). Тогда построим НКА

$$A = \{\Sigma, Q := \{1; \ldots; n\}, q_0 := \{1\}, \delta, F := \{n\}\},$$

где

$$\delta(k, s) := \begin{cases} \{1; k+1\} & \text{если } k < n \wedge s = a, \\ \{k+1\} & \text{если } k < n \wedge s = b, \\ \{1\} & \text{если } k = n \wedge s = a, \\ \varnothing & \text{иначе.} \end{cases}$$

Пусть $A'$ — ДКА, построенный по $A$ по алгоритму с лекции. У $A'$ ровно $2^n$ состояний, поэтому мы хотим, чтобы он был минимальным. Для этого нам надо, чтобы всякое состояние $A$ принималось, а из всех состояний $A$ принимались попарно разные языки. Это равносильно тому, что в $A$ всякое подмножество $Q$ принимается, и языки, принимаемые из всех подмножеств $Q$, были попарно различными. Обозначим язык, принимаемый из подмножества $S \subseteq Q$ за $L(S)$.

Чтобы показать, что языки, принимаемые из всех подмножеств, попарно различны достаточно показать (а на деле равносильно), что у всякого $L(\{k\})$ есть уникальное слово $w_k$, не лежащее в других $L(\{l\})$. Действительно, поскольку $L(S_1 \cup S_2) = L(S_1) \cup L(S_2)$. А тогда $w_k \in L(S) \Leftrightarrow k \in S$. А в таком случае $L(S_1) = L(S_2) \Leftrightarrow S_1 = S_2$.

Ещё раз: нам нужно показать, что всякое подмножество состояний достижимо, а из каждого состояния принимается некоторое слово, которое не принимается из других состояний.

Покажем как принять любое подмножество. Давайте рассмотрим функцию

$$\varphi : 2^Q \to \{0; 1; \ldots; 2^n - 1\}, S \mapsto \sum_{k \in S} 2^{k-1}.$$

Очевидно, $\varphi$ — биекция. При этом для всякого $M \in \{1; \ldots; 2^n - 1\}$.

$$\varphi(\delta^*(\varphi^{-1}(M), a)) = (2M + 1) \,\%\, 2^n, \qquad \varphi(\delta^*(\varphi^{-1}(M), b)) = (2M) \,\%\, 2^n,$$

где $\%$ — операция взятия остатка по модулю (как в программировании); если $n = 0$, то и результат в обоих случаях тоже будет 0. (Если $M < 2^{n-1}$, т.е. $n \notin \varphi^{-1}(M)$, то операция $\%$ ничего не делает.) Следовательно, задачу достижимости всякого подмножества состояний можно переформулировать так.

> У нас есть две операции: $M \to 2M$ и $M \to 2M + 1$. Как из 1 такими операциями получить любое число от 1 до $2^n - 1$.

(0 получаем считыванием $b^n$. Также о операции % можно забыть, так как если без неё применить к 1 любые $n$ операций, то получится число не меньшее $2^n$, что не решает задачу. Т.е. всякое решение у задачи (последовательность операций для получения того или иного числа) состоит из не более $n-1$ операции, а тогда получить число более $2^n-1$ нельзя, что значит ненужность операции %.) При этом эти операции в реальности есть просто приписывание нуля и единицы соответственно в конец числа в двоичной системе счисления. Следовательно, чтобы (последними двумя операциями) получить любое положительное число, нужно лишь уметь писать последовательность цифр, идущих после первой (которая есть единица). Следовательно, несложно видеть, что всякое подмножество состояний $A$ достижимо.

Теперь у каждого состояния найдём слово, которое принимается только из этого состояния. Ответ неожиданно простой: для состояния $k$ подойдёт слово $b^{n-k}$. Действительно,

$$\delta(k,b) := \begin{cases} \{k+1\} & \text{если } k < n, \\ \varnothing & \text{иначе.} \end{cases}$$

Таким образом понятно, что из языка $b^*$ из состояния $k$ принимается только слово $b^{n-k}$.

---

**Задача 4.** Пусть дан синхронизируемый ДКА $A$ на $n$ состояниях. Покажем, что для любых двух состояний $q_1$ и $q_2$ есть слово длины $\leqslant n^2$, синхронизирующее их. Рассмотрим синхронизирующее слово всего автомата $w$. Рассмотрим вместо автомата $A$ автомат $A^2$, т.е. автомат, где множество состояний — $Q^2$,

$$\delta((p_1,p_2),q) := (\delta(p_1,a),\delta(p_2,a)),$$

начальное состояние — $(q_1,q_2)$, а множество принимающих не важно. Тогда по определению $w$ при прочтении $w$ мы приходим на диагональ. Если $|w| > |Q|^2 = |Q^2|$, то можем применить лемму о накачке для сдувания $w$: она утверждает, что есть разбиение $w = xyz$, где $|y| > 0$ и ещё несколько условий, что для всякого $l \geqslant 0$ верно $\delta(q_0, xy^lz) = \delta(q_0,w)$, а тогда $xz = xy^0z \in L(A^2)$, $|xz| < |w|$, $\delta(q_0,xz) = \delta(q_0,w)$. Это значит, что минимальное слово, синхронизирующее $q_1$ и $q_2$, имеет длину не более $|Q|^2 = n^2$.

Теперь покажем, что есть слово длины $\leqslant n^2(n-1)$, синхронизирующее весь автомат. Построим его следующим образом. Поставим в каждое состояние фишку. Каждая фишка будет передвигаться согласно $\delta$ и называемому символу (слову). Наша цель: сказать такое слово, что все фишки встанут в одно состояние. Пронумеруем фишки числами от 1 до $n$. Сначала скажем слово, синхронизирующее состояния, на которых стоят фишки 1 и 2. Затем скажем слово, синхронизирующее состояния, на которых в данный момент (после произнесения предыдущего слова) стоят фишки 1 и 3. Затем — 1 и 4, затем — 1 и 5, и т.д. до $n$. Итого после называния слова $k$ фишка $k+1$ встала на то же состояние, что и 1, а значит и будет стоять в самом конце (после произнесения последнего слова). Значит все фишки встали в одно состояние — состояние, на котором стоит фишка 1. При этом было произнесено $n-1$ слово длины $\leqslant n^2$. Значит итоговая конкатенация — синхронизирующее слово длины $n^2(n-1)$.

---

**Задача 5.** Давайте рассмотрим грамматику, где терминальные символы — $\{a;b;c\}$, нетерминальные — $\{S;P;R;T;A;B;C\}$, начальный символ — $S$, а правила:

$s)\ S \to PR$, $f_{Ab})\ Ab \to bA$, $f_{Ca})\ Ca \to aC$, $r_C)\ CR \to cR$, $e)\ TR \to \varepsilon$.

$p_a)\ P \to aPA$, $f_{Ac})\ Ac \to cA$, $f_{Cb})\ Cb \to bC$, $t)\ P \to T$,

$p_b)\ P \to bPB$, $f_{Ba})\ Ba \to aB$, $f_{Cc})\ Cc \to cC$, $t_a)\ Ta \to aT$,

$p_c)\ P \to cPC$, $f_{Bb})\ Bb \to bB$, $r_A)\ AR \to aR$, $t_b)\ Tb \to bT$,

$f_{Aa})\ Aa \to aA$, $f_{Bc})\ Bc \to cB$, $r_B)\ BR \to bR$, $t_c)\ Tc \to cT$,

Покажем, что мы получили строку вида $ww$. Пусть мы получили какую-то строку. Рассмотрим путь преобразований от $S$ до неё.

- Первое преобразование в любом случае $s$, т.е. было получено слово $PR$.
- Больше букв $S$ не было, так как нет правил, которые могут создать $S$.
- Больше букв $P$ и $R$ тоже не появлялось, так как единственное правило, их создающее, — $s$ — больше не может быть применено.
- $P$ будет преобразовано в $T$ правилом $t$, так как нет других правил, убирающих $P$.
- Больше символов $T$ не появиться, так как нет других правил, добавляющих $T$.
- $T$ и $R$ будут удалены вместе правилом $e$.
- $R$ всегда находится справа на краю, так как нет правил, которые что-то приписывают справа от неё.
- Пока ещё присутствует $P$, слева от $P$ нет нетерминальных символов (несложно видеть по индукции). Все терминальные символы находятся между $P$ и $R$.
- Пока присутствует $T$ все терминальные символы находятся между $T$ и $R$.
- Правило $e$ является последним применённым.
- Терминальные символы появляются только перед буквой $R$ или перед буквой $P$ и не исчезают.
- Порядок терминальных символов в строке не меняется, но они появляются либо “где-то в середине строки”, либо в самом конце.
- Порядок нетерминальных символов $A$, $B$ и $C$ не меняется. Они появляются с самого лева, а исчезают с самого права.
- Рассмотрим действия до действия $t$, но после $s$. У нас есть единственный символ $P$ и единственный символ $R$. $P$ справа скраю. Слева от $P$ только терминальные символы. Между $P$ и $R$ — только символы $a$, $b$, $c$, $A$, $B$, $C$. Пусть $w_1$ — подстрока из нетерминальных симолов слева от $P$ (т.е. подстрока всех нетерминальных символов, написанных слева от $P$; в разные моменты, после разных действий значение $w_1$ может оказаться разным), $w_2$ — подстрока нетерминальных символов между $P$ и $R$, а $v$ — подстрока символов $A$, $B$ и $C$ с применением замены на $a$, $b$ и $c$ соответственно.

  Например, если в какой-то момент была написана строка $abbcPCabBR$, то в этот момент $w_1 = abbc$, $w_2 = ab$, $v = cb$ (так как получена из $CB$).

  Тогда несложно показать по индукции, что в любой момент $w_1 = w_2 v^R$.

- Пусть в момент действия $t$ значение $w_1$ стало равно $W$ (т.е. просто запомнили значение $w_1$ в данный момент и обозначили запомненное за $W$). Пусть $w$ — подстрока нетерминальных символов, а $v$ — всё ещё подстрока символов $A$, $B$ и $C$, где применили замену. Тогда несложно показать по индукции, что в любой момент $wv^R = WW$.
- Следовательно перед выполнением $e$ $wv^R = WW$, но $v = \varepsilon$ (так как все нетерминальные символы находятся между $T$ и $R$, но между ними ничего нет). Следовательно, после применения $t$ осталось слово $w = WW$.
- Следовательно принимаются только слова вида $WW$.

Покажем, что всякую строку вида $ww$ можно принять.

- Для этого сначала выполним $s$, получим строку $PR$.
- Теперь будем читать $w$ слева направо и для каждого символа $l$ применим правило $p_l$. Получим строку $wPW^RR$, где $W$ — строка $w$, где $a$, $b$ и $c$ заменили на $A$, $B$ и $C$ соответственно, а $W^R$ — ещё и развернули.
- Теперь будем по одному брать самый правый символ у $W^R$ — пусть $L$, применять к нему правило $r_L$, получая $l$, а затем пропихивать $l$ через весь остаток $W^R$ влево до конца с помощью правил $f_{...}$. Тогда получим строку $wPwR$.
- Теперь применим $t$ и получим $wTwR$.
- Теперь применяя $t_{...}$, получим строку $wwTR$.
- Наконец применяя $e$, получаем строку $ww$.

---

**Задача 5.** Давайте рассмотрим грамматику, где терминальные символы — $\{a; b; c\}$, нетерминальные — $\{S; P; P_{\neq}; P_{<}; P_{>}; R; T; A; B\}$, начальный символ — $S$, а правила:

$s)$ $S \to PR$, $p_{b-})$ $P \to bP_{>}$, $p_{\neq:B})$ $P_{\neq} \to P_{\neq}B$, $f_{Ab})$ $Ab \to bA$, $t_{\neq})$ $P_{\neq} \to cT$,

$p_{aA})$ $P \to aPA$, $p_{-A})$ $P \to P_{<}A$, $p_{>:a})$ $P_{>} \to aP_{>}$, $f_{Ba})$ $Ba \to aB$, $t_{>})$ $P_{>} \to cT$,

$p_{aB})$ $P \to aP_{\neq}B$, $p_{-B})$ $P \to P_{<}B$, $p_{>:b})$ $P_{>} \to bP_{>}$, $f_{Bb})$ $Bb \to bB$, $t_{<})$ $P_{<} \to cT$,

$p_{bA})$ $P \to bP_{\neq}A$, $p_{\neq:a})$ $P_{\neq} \to aP_{\neq}$, $p_{<:A})$ $P_{<} \to P_{<}A$, $r_A)$ $AR \to aR$, $t_a)$ $Ta \to aT$,

$p_{bB})$ $P \to bPB$, $p_{\neq:b})$ $P_{\neq} \to bP_{\neq}$, $p_{<:B})$ $P_{<} \to P_{<}B$, $r_B)$ $BR \to bR$, $t_b)$ $Tb \to bT$,

$p_{a-})$ $P \to aP_{>}$, $p_{\neq:A})$ $P_{\neq} \to P_{\neq}A$, $f_{Aa})$ $Aa \to aA$, $r_C)$ $CR \to cR$, $e)$ $TR \to \varepsilon$.

Покажем, что мы получили строку вида $ucv$, где $u, v \in \{a; b\}^*$, $u \neq v$. Пусть мы получили какую-то строку. Рассмотрим путь преобразований от $S$ до неё.

- Первое преобразование в любом случае $s$, т.е. было получено слово $PR$.
- Больше букв $S$ не было, так как нет правил, которые могут создать $S$.
- Больше букв $P$ и $R$ тоже не появлялось, так как единственное правило, их создающее, — $s$ — больше не может быть применено.

- $P$ будет преобразовано либо в $P_{\neq}$, либо в $P_{>}$, либо в $P_{<}$ правилами $p_{...}$ кроме $p_{aA}$ и $p_{bB}$, так как нет других правил, убирающих $P$.
- Больше букв $P_{\neq}$, $P_{>}$ и $P_{<}$ не появиться, а из этих трёх появиться ровно одна.
- Та единственная, которая появилась, будет преобразована в $T$ правилом $t_{\neq}$, $t_{>}$ или $t_{<}$ (в зависимости от того, что появилось), так как нет других правил, удаляющих данные буквы.
- Больше символов $T$ не появиться, так как нет других правил, добавляющих $T$.
- $T$ и $R$ будут удалены вместе правилом $e$.
- $R$ всегда находится справа на краю, так как нет правил, которые что-то приписывают справа от неё.
- Пока ещё присутствует $P$, слева от $P$ нет нетерминальных символов (несложно видеть по индукции). Все терминальные символы находятся между $P$ и $R$.
- Аналогичное верно для $P_{\neq}$, $P_{>}$ и $P_{<}$.
- Аналогичное верно для $T$.
- Правило $e$ является последним применённым.
- Терминальные символы появляются только перед буквой $R$, перед буквой $P$, перед буквой $P_{\neq}$, перед буквой $P_{>}$ или перед буквой $P_{<}$ и не исчезают.
- Порядок терминальных символов в строке не меняется, но они появляются либо “где-то в середине строки”, либо в самом конце.
- Порядок нетерминальных символов $A$, $B$ и $C$ не меняется. Они появляются с самого лева, а исчезают с самого права.
- Рассмотрим действия до действий, порождающих $P_{\neq}$, $P_{>}$ или $P_{<}$, но после $s$. У нас есть единственный символ $P$ и единственный символ $R$. $P$ справа скраю. Слева от $P$ только терминальные символы. Между $P$ и $R$ — только символы $a$, $b$, $A$, $B$. Пусть $w_1$ — подстрока из нетерминальных симолов слева от $P$ (т.е. подстрока всех нетерминальных символов, написанных слева от $P$; в разные моменты, после разных действий значение $w_1$ может оказаться разным), $w_2$ — подстрока нетерминальных символов между $P$ и $R$, а $v$ — подстрока символов $A$ и $B$ с применением замены на $a$ и $b$ соответственно.

  Например, если в какой-то момент была написана строка $abbabPBaAbBR$, то в этот момент $w_1 = abbab$, $w_2 = ab$, $v = bab$ (так как получена из $BAB$).

  Тогда несложно показать по индукции, что в любой момент $w_1 = w_2 v^R$.
- Пусть применено было правило, порождающее $P_{\neq}$ (из $P$). Пусть $w_1$, $w_2$ и $v$ определены всё также. Тогда после создания $P_{\neq}$ уже верно, что $w_1 \neq w_2 v^R$, так как у $w_1$ и $w_2 v^R$ на концы были добавлены разные символы (а до этого $w_1 = w_2 v^R$). Тогда несложно показать по индукции, что и в любой момент до применения $t_{\neq}$ будет верно $w_1 \neq w_2 v^R$ (так как эти строки только и будут продолжаться справа).
- Аналогично после $t_{>}$ мы имеем $|w_1| > |w_2 v|$, а тогда до $t_{>}$ верно, что $|w_2 v|$ не изменяется, а $|w_1|$ не уменьшается. Для $P_{<}$ верно утверждение, аналогичное $P_{>}$, но с неравенствами в обратную сторону.

- Пусть в момент перед действием $t_{\neq}/t >/t_<$ значение $w_1$ стало равно $U$ (т.е. просто запомнили значение $w_1$ в данный момент и обозначили запомненное за $U$), а $w_2 v^R - V$. Рассмотрим период между действиями $t_{\neq}/t >/t_<$ и $e$ Пусть $w$ — подстрока нетерминальных символов, а $v$ — всё ещё подстрока символов $A$ и $B$, где применили замену. Тогда сразу после действия $t_{\neq}/t >/t_<$ имеем, что $wv^R = UcV$. Тогда несложно показать по индукции, что в любой момент $wv^R = UcV$.

- Следовательно перед выполнением $e$ $wv^R = UcV$, но $v = \varepsilon$ (так как все нетерминальные символы находятся между $T$ и $R$, но между ними ничего нет). Следовательно, после применения $t$ осталось слово $w = UcV$.

- Следовательно принимаются только слова вида $UcV$. При этом мы показали, что $U \neq V$ (так как попадали в случай либо $P_{\neq}$, либо $P_>$, либо $P_<$). Значит язык, получаемый данной грамматикой является подмножеством требуемого языка.

Покажем, что всякую строку вида $ucv$ ($u, v \in \{a; b\}^*$, $u \neq v$) можно принять.

- Для этого сначала выполним $s$, получим строку $PR$.

- Теперь будем читать $u$ и $v$ одновременно слева направо. Пока в обоих словах читаются одинаковые символы, для каждого читаемого символа $l$ (в порядке чтения) применим правило $p_{lL}$ (где $L$ — нетерминальная альтернатива для $l$: $A$ для $a$ и $B$ для $b$). Получим строку $wPW^R R$, где $w$ — самый длинный общий префикс $u$ и $v$, $W$ — строка $w$, где $a$ и $b$ заменили на $A$ и $B$ соответственно, а $W^R$ — ещё и развернули.

- Поскольку $u \neq v$, то после $w$ начинается различие $u$ и $v$; их бывает три вида: 1) $u$ и $v$ содержат следующими разные символы, 2) $u$ закончилась, а $v$ — нет, и 3) $v$ закончилась, а $u$ — нет.

  - В первом случае применим правило $p_{kL}$, где $k$ — следующий по порядку чтения символ $u$, а $l$ — следующий символ $v$. Получим строку $wkP_{\neq}LW^R R$. Сначала дочитаем строку $u$ и на каждый символ $j$ применим операцию $p_{\neq:j}$. Получим строку $uP_{\neq}LW^R R$. Далее дочитаем $v$ и по аналогии будем применять $p_{\neq:J}$. Получим строку $uP_{\neq}V^R R$.
  - Во втором случае применим правило $p_{-L}$, где $l$ — следующий символ $v$. Получим строку $uP_<LU^R R$. Теперь только лишь дочитаем строку $v$ и на каждый символ $j$ применим операцию $p_{<:J}$. Получим строку $uP_<V^R R$.
  - Во третьем случае применим правило $p_{k-}$, где $k$ — следующий символ $u$. Получим строку $vkP_<V^R R$. Теперь только лишь дочитаем строку $u$ и на каждый символ $j$ применим операцию $p_{>:j}$. Получим строку $uP_>V^R R$.

- Теперь будем по одному брать самый правый символ у $W^R$ — пусть $L$, применять к нему правило $r_L$, получая $l$, а затем пропихивать $l$ через весь остаток $W^R$ влево до конца с помощью правил $f_{\ldots}$. Тогда получим строку $uP_{\neq}vR/uP_<vR/uP_>vR$.

- Теперь применим $t$ и получим $ucTvR$.

- Теперь применяя $t_a$ и $t_b$, получим строку $ucvTR$.

- Наконец применяя $e$, получаем строку $ucv$.

---