

1. Язык, описывающийся следующей атрибутивной грамматикой:

$$\begin{aligned} S &\rightarrow AT && ; \quad T.rng > A.iter \\ A &\rightarrow aA && ; \quad A_0.iter := A_1.iter + 1 \\ A &\rightarrow \varepsilon && ; \quad A.iter := 0 \\ T &\rightarrow TcT && ; \quad T_0.rng := \max(T_1.rng, T_2.rng) \\ T &\rightarrow K && ; \quad T.rng := K.rng \\ K &\rightarrow aK && ; \quad K_0.rng := K_1.rng + 1 \\ K &\rightarrow bK && ; \quad K_0.rng := 0 \\ K &\rightarrow \varepsilon && ; \quad K.rng := 0 \end{aligned}$$

2. Язык $\{wcvw_{pref}zw_{suff} \mid w, z \in \{a, b\}^* \text{ \& } v \in \{a, b, c\}^*\}$. Здесь w_{pref} — непустой префикс слова w ; w_{suff} — непустой суффикс слова w .

3. Язык, описывающийся следующей атрибутивной грамматикой:

$$\begin{aligned} S &\rightarrow SbS && ; \quad S_0.iter := 2 \cdot S_1.iter, S_1.iter == S_2.iter \\ S &\rightarrow a && ; \quad S.iter := 1 \end{aligned}$$

4. Язык SRS с правилами $a \rightarrow bab, ba \rightarrow ab$ над базисным словом aa (единственным).

Решение задачи IV

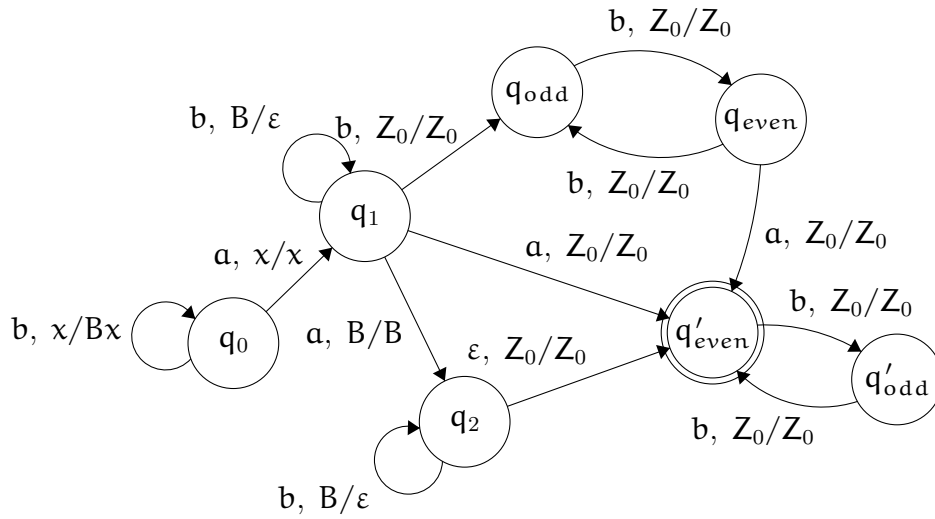
\mathcal{L} — язык SRS с правилами $a \rightarrow bab, ba \rightarrow ab$ над базисным словом aa (единственным).

Правила переписывания показывают, что во-первых, буквы a в словах языка всегда ровно две, а во-вторых, относительно слов, порождаемых по первому правилу $b^n ab^{n+m} ab^m$ они могут перемещаться только влево, причём как угодно далеко. При этом, если мы сначала применим первое правило, а потом второе, то получим композицию $a \rightarrow abb$, а если мы сначала применим второе, а потом первое — то $ba \rightarrow babb$, то есть правила перестановочны, коль скоро есть хотя бы одна буква b слева от a . Значит, чтобы получить произвольное слово языка, можно сначала максимально применить первое правило, а потом второе.

Обратим также внимание, что если какое-то слово получилось из второй слева буквы a применением k правил 1, то мы могли бы получить его, применив те же k правил 1 к самой левой букве a , а затем к ней же — к правил 2, и ко второй букве a — к правил 2. Пусть область, порождаемая правилами

$a \rightarrow bab$, обозначена красным для первой буквы a и синим — для второй. Тогда $ab^k ab^k$ — то же, что $b^k ab^k a \rightarrow ab^{2k} a \rightarrow ab^k ab^k$. Поэтому можно принять допущение, что правила $a \rightarrow bab$ применяются только к самой левой букве a .

Таким образом, можно установить, что слова языка \mathcal{L} имеют вид $b^{n_1} ab^{n_2} ab^{n_3}$, где $n_2 + n_3 \geq n_1$ и сумма $n_1 + n_2 + n_3$ чётна. Это детерминированный контекстно-свободный (но не регулярный) язык, DPDA для которого выглядит так:



Язык не является $LL(k)$, в чём можно убедиться, рассмотрев слова $b^{n+k} ab^{n+k} a$ и $b^{n+k} a ab^{n+k}$ с предполагаемым lookahead для грамматики в ГНФ, равным k , и наблюдаемым префиксом b^n . В силу бесконечности классов по Майхиллу–Нероуду в языке b^* относительно \mathcal{L} , стек на этом префиксе разрастается как угодно сильно, в частности, можно взять такое n , чтобы он уж точно содержал $k + 3$ элемента. Тогда в слове $b^{n+k} a ab^{n+k}$ последний элемент стека заведомо разворачивается в $C_1 = b^s$ (т.к. префикс $b^k a^2$ точно прочитан раньше: каждый из $k + 2$ первых элементов стека считал как минимум по одной букве). В слове $b^{n+k} ab^{n+k} a$ последний элемент стека точно разворачивается в $C_2 = b^t a$. Этот последний элемент один и тот же, и lookahead у него в обоих случаях равен концу строки. Подменой C_1 на C_2 (или наоборот) получаем слово не из \mathcal{L} .

Язык не является VPL-языком. По условию, в VPL-языке все терминалы строго делятся на непересекающиеся категории: вызывающие, возвращающие и внутренние. Если назначить b вызывающим терминалом, то стек (по-

что) всегда будет только наполняться; если назначить b читающим — стек (почти) всегда будет пуст. И тот, и другой случай приводят к тому, что количество букв b слева и справа от первой буквы a сравнить невозможно. «Почти» здесь — указание, что в принципе ничего не мешает назначить букву a читающей или вызывающей, однако поскольку их только две в слове, они изменяют поведение стека лишь конечным образом.

Решение задачи III

Посмотрим на несколько первых итераций, порождающих слова языка \mathcal{L} :

$$\begin{aligned} S &\rightarrow SbS & ; & \quad S_0.\text{iter} := 2 \cdot S_1.\text{iter}, S_1.\text{iter} == S_2.\text{iter} \\ S &\rightarrow a & ; & \quad S.\text{iter} := 1 \end{aligned}$$

Слово aba семантическому критерию удовлетворяет, т.к. оно получается разбором двух нетерминалов с атрибутом $\text{iter} = 1$. При этом для aba соответствующий атрибут равен 2.

Из этих базовых слов мы не можем получить слово $ababa$, т.к. атрибуты его левых и правых частей не равны. То же самое верно и про $ababab$. Слово $abababab$ построить можно, и его атрибут iter будет равен 4.

Мы видим, что на каждой новой итерации мы получаем возможность скомбинировать только два слова с предыдущей итерации, при этом атрибут iter полученного слова будет не равен атрибутам iter никаких ранее построенных слов. Если на i -й итерации было построено слово $(ab)^k a$, то на $i + 1$ -й итерации будет построено слово $(ab)^k ab(ab)^k a = (ab)^{2k+1} a$. Отсюда следует, что множество слов языка в свёрточной форме выглядит так: $\{(ab)^{2^i-1} a\}$.

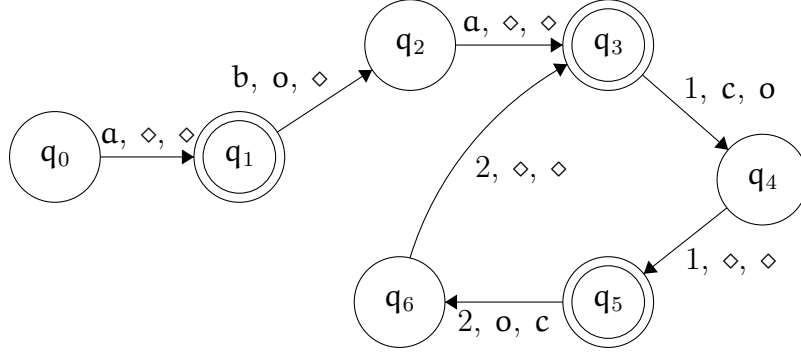
Проще всего обосновать не КС-свойство для данного языка с помощью теоремы Париха. Действительно, кратности букв b (впрочем, как и a) в словах языка описываются экспоненциально растущей функцией, которая не может быть представлена как объединение линейных. Доказательство не КС-свойства языка \mathcal{L} с помощью леммы о накачке предоставляется читателю.

Построим backref-регулярку для данного языка, пользуясь наблюдением, что $2^n - 1 = 1 + 2 + 2^2 + \dots + 2^{n-1}$.

$$a([{}_1ba]_1([{}_2x_1x_1]_2[{}_1x_2x_2]_1)^*[{}_2x_1x_1]_2?)?$$

Заметим, что эту backref-регулярку можно преобразовать в детерминированный MFA. Единственной проблемой, которая в ней может возникнуть, является переход от итерации Клини к последнему чтению x_1x_1 , который ре-

шается тем, что промежуточное состояние внутри итерации перед чтением $[1x_2x_2]_1$ нужно сделать финальным.



Решение задачи II

$$\mathcal{L} = \{wcvw_{\text{pref}}zw_{\text{suff}} \mid w, z \in \{a, b\}^* \text{ \& } v \in \{a, b, c\}^*\}$$

Можно заметить, что для любого слова в \mathcal{L} при рассмотрении $w_{\text{pref}} = s_{\text{fst}}w'_{\text{pref}}$ и $w_{\text{suff}} = w'_{\text{suff}}s_{\text{last}}$, где s_{fst} и s_{last} — первая и последняя буквы соответственно, получается слово вида $wcv s_{\text{fst}}w'_{\text{pref}}zw'_{\text{suff}}s_{\text{last}}$, в котором можно положить $z' = w'_{\text{pref}}zw'_{\text{suff}}$, а новыми префиксом и суффиксом — только первую и последнюю буквы. И это слово также принадлежит \mathcal{L} , однако его структура подходит под большее число случаев w , чем исходная. Поэтому язык \mathcal{L} по факту представляет собой следующий:

$$\mathcal{L} = \{s_{\text{fst}}ws_{\text{last}}cv s_{\text{fst}}zs_{\text{last}} \mid s_{\text{fst}}ws_{\text{last}}, z \in \{a, b\}^* \text{ \& } v \in \{a, b, c\}^*\}$$

Поскольку w, v, z здесь можно заменить регулярками $(a|b)^*$ и $(a|b|c)^*$, то данный язык регулярен и является объединением следующих языков:

- $ac(a|b|c)^*a(a|b)^*a$
- $bc(a|b|c)^*b(a|b)^*b$
- $a(a|b)^*ac(a|b|c)^*a(a|b)^*a$
- $a(a|b)^*bc(a|b|c)^*a(a|b)^*b$
- $b(a|b)^*ac(a|b|c)^*b(a|b)^*a$

- $b(a|b)^*bc(a|b|c)^*b(a|b)^*b$

Префикс-свойство для него не выполняется, потому что внутри подвыражения $(a|b|c)^*$ или последнего подвыражения $(a|b)^*$ может встретиться повтор последующего суффикса. В качестве контрпримера, подтверждающего это, достаточно взять слова $aasaa$ и слово $aasaaa$, принадлежащие \mathcal{L} .

Всякий регулярный язык является LL(1). Соответствующую грамматику можно построить, тупо взяв ДКА для него и преобразовав его в праволинейную грамматику с эpsilon-правилами для самой последней буквы (иначе будет LL(1)-конфликт; нигде, кроме как с финальными состояниями, конфликта быть не может в силу того, что автомат детерминирован).

Решение задачи I

Сначала посмотрим на атрибутивную грамматику.

$$\begin{aligned}
 S &\rightarrow AT && ; && T.rng > A.iter \\
 A &\rightarrow aA && ; && A_0.iter := A_1.iter + 1 \\
 A &\rightarrow \varepsilon && ; && A.iter := 0 \\
 T &\rightarrow TcT && ; && T_0.rng := \max(T_1.rng, T_2.rng) \\
 T &\rightarrow K && ; && T.rng := K.rng \\
 K &\rightarrow aK && ; && K_0.rng := K_1.rng + 1 \\
 K &\rightarrow bK && ; && K_0.rng := 0 \\
 K &\rightarrow \varepsilon && ; && K.rng := 0
 \end{aligned}$$

Все атрибуты синтетические, кроме того, язык K явно регулярен, и атрибут rng в нём изменяется, только пока в префиксе встречаются исключительно буквы a . То есть если $K \rightarrow a^n(b(a|b)^*)^?$, то $K.rng = n$.

Теперь посмотрим на язык T . Заметим, что его можно переписать в форме $(Kc)^*K$, устранив левую рекурсию (поскольку язык K не содержит букв c , а разделители между K -развёртками не содержат a и b , то можно K представить единственным токеном и далее, например, применить метод Блума–Коха). При этом семантическое свойство будет выглядеть так: $T.rng = \max(K.rng)$.

Осталось разобраться с атрибутом A . Здесь, очевидно, если $A \rightarrow a^m$, то $A.iter = m$. Но именно в случае A есть неоднозначность границы разбора: если $S \rightarrow AKcT$, причём $K \rightarrow a^n(b(a|b)^*)^?$, $A \rightarrow a^m$, то мы можем сделать какие угодно разборы префикса a^{n+m} начиная от $K \rightarrow a^{n+m}(b(a|b)^*)^?$,

$A \rightarrow \varepsilon$ и кончая $K \rightarrow (b(a|b)^*)^?$, $A \rightarrow a^{n+m}$. С точки зрения языка, наиболее выгодно положить $K \rightarrow a^{n+m}(b(a|b)^*)^?$, $A \rightarrow \varepsilon$ — это даст возможность породить наиболее широкий класс слов.

Таким образом, условие на атрибуты вырождается в $T.rng > 0$, а грамматика для данного языка может быть преобразована в следующую форму:

$$\begin{aligned} S &\rightarrow T && ; \quad T.rng > 0 \\ T &\rightarrow KcT && ; \quad T_0.rng := \max(K.rng, T_1.rng) \\ T &\rightarrow K && ; \quad T.rng := K.rng \\ K &\rightarrow aK && ; \quad K_0.rng := K_1.rng + 1 \\ K &\rightarrow bK && ; \quad K_0.rng := 0 \\ K &\rightarrow \varepsilon && ; \quad K.rng := 0 \end{aligned}$$

Эта грамматика определяет регулярный язык $((a|b)^*c)^+a^+(a|b)^*(c(a|b)^*)^?$.

Если бы условие на атрибуты в корне было $T.rng < A.iter$, тогда наиболее выгодное с точки зрения языка решение по неоднозначности было бы $K \rightarrow (b(a|b)^*)^?$, $A \rightarrow a^{n+m}$. То есть требовалось бы сравнить по длине максимальные подстроки из букв a в префиксах, идущих сразу после буквы c , с префиксом a^{n+m} (и если ни одной буквы c в слове нет, то это условие тривиально выполняется, если только в начале слова есть хотя бы одна буква a). Будем рассматривать только нетривиальную ситуацию — буквы c присутствуют. Упрощенная атрибутная грамматика станет такая:

$$\begin{aligned} S &\rightarrow ABcT && ; \quad T.rng < A.iter \\ A &\rightarrow aA && ; \quad A_0.iter := A_1.iter + 1 \\ A &\rightarrow \varepsilon && ; \quad A.iter := 0 \\ T &\rightarrow KcT && ; \quad T_0.rng := \max(T_1.rng, T_2.rng) \\ T &\rightarrow K && ; \quad T.rng := K.rng \\ K &\rightarrow aK && ; \quad K_0.rng := K_1.rng + 1 \\ K &\rightarrow B && ; \quad K.rng := 0 \\ B &\rightarrow b(a|b)^* | \varepsilon && ; \quad \text{атрибутов нет} \end{aligned}$$

В свёрточной форме язык переписывается как $\{a^n(b(a|b)^*)^?(ca^i(b(a|b)^*)^?)^+ \mid \forall i(n > i)\}$.

Похоже, что есть более чем двойственная взаимосвязь между блоками a^i (т.к. каждый из них нужно сравнивать с самым первым, что означало бы, что из первого блока нужно читать в стек несколько раз). Попробуем опровергнуть КС-свойство для полученного языка путём лёммы о накачке. Для этого сначала пересечём наш язык с $a^+ca^+ca^+$, чтобы избавиться от лишних подстрок, определяемых конечноавтоматным поведением, и от накачек, наращающих число K -фрагментов.

Пусть p — длина накачки. Рассмотрим слово $a^{p+1}ca^pca^p$. Отдельно

фрагмент a^{p+1} накачивать мы не можем — отрицательная накачка сразу же выведет из языка. Накачивать его вместе с фрагментом a^p тоже не получится — отрицательная накачка даст слово, в котором первый фрагмент не больше по длине, чем фрагмент a^p . Однако накачивать только $a^p c a^p$ (в любых вариациях) тоже нельзя — это сразу же приведёт к выходу из языка при повторной положительной накачке.

Можно заметить, что язык конъюнктивен. Действительно, для крайней пары нетерминалов A и K можно КС-свободно описать слова вида

$$a^{n+m+1}(b(a|b)^*|\varepsilon)c(a|b|c)^*ca^n(b(a|b)^*|\varepsilon)$$

Например, такой грамматикой:

$$S' \rightarrow S'a | S'b | Cb | C$$

$$C \rightarrow aCa | aAc | aAcWc$$

$$A \rightarrow aA | bW | \varepsilon$$

$$W \rightarrow (a|b|c)W | \varepsilon$$

Конъюнктивная грамматика готова:

$$S \rightarrow S' \ \& \ ScK$$

$$S \rightarrow aK$$

$$K \rightarrow (a|b)K | \varepsilon$$

Заметим, что этот язык — не DMFA. Используем замкнутость DMFA относительно пересечения с регулярными языками и пересечём наш язык с a^+ca^+ . Получим язык слов вида $a^{n+m}ca^n$. Его префиксы a^* определяют бесконечное число классов эквивалентности по Майхиллу–Нероуду, поэтому для них можно применить JL (то есть положить v_n подсловом префикса a^+). Положим $v_n = a^n$. Для p_n есть два варианта: $p_n = a^{n+k}$ и $p_n = a^{n+k}ca^i$, где $i < 2n + k$. Существуют такие z_i , что слова $p_nv_nz_i$ входят в язык, а именно: $z_i = c$ в первом случае и $z_i = a$ во втором случае. Однако слова $a^{n+k}c$ и $a^{n+k}ca^i$ также входят в язык, а их суффиксы (после p_n) не начинаются с v_n . Поэтому ни с какой позиции слова запомненный префикс v_n детерминированно прочитано быть не может.