Теория формальных языков. Рубежный контроль №2

Вариант №23

Киселев Кирилл

Теоретическая информатика и компьютерные технологии
МГТУ им. Н.Э. Баумана
декабрь 2023

Содержание

1 Задача 1

Язык SRS $a \to bab$, $a^3 \to a^2$, $ba \to ac$ над множеством базисных слов $b^n a^n$

1.1 Решение

Замечания:

1. Любое не пустое слово содержит хотя бы одну a
2. Буквы a могут только уменьшаться
3. $|w|_a \le |w|_b + |w|_c$
4. Можно бесконечно двигать влево самую первую букву a
5. Слова могут начинаться только с $b^i a^k$, где $k > 0$
6. Применение правила 3 ограничивает сдвиг тех a, которые находились правее буквы a, к которой было применено правило.

   Начальное количество букв a и b равно, далее количество исходных b может либо остаться таким же, либо с помощью правила 3 некоторые b, могут перейти в c, но суммарное количество b и c останется равным n. Буквы a могут быть переписаны с помощью правила 2, и их количество станет меньше n. Каждой буквой a с помощью первого и третьего правил могут порождаться некоторые блоки, в которых количество букв b и c сбалансированно, т.к. правило 1 порождает блоки вида $b^k a b^k$, в которых количество букв b сбалансированно. Поэтому при построении PDA стоит ввести два стековых символа, первый будет использовать для подсчета исходных b и букв c, полученных из этих букв b. Второй символ необходим для проверки баланса букв b и c, которые были пораждены с помощью повторения правил 2, 3. Также этот блок может содержать один внутренний блок такой же структуры, из-за наличия правила 3, тогда исходный блок должен заканчиваться на c^r, после чтения которых в стеке должны оставаться лишь символы первого вида.

Докажем, что данный язык не принадлежит классу `DCFL`. Рассмотрим следующие слова:

- $w_1 = b^{3n} a b^n b^{2n} c$
- $w_2 = b^n b^{2n} a b^{2n} a b^{2n} c^{2n} a^{n-2}$

Рассмотрим два случая:

1. Накачка общего префикса x. Пусть $x = x_0 x_1 x_2 x_3 x_4$.Рассмотрим слово w_2. Если $x_1 x_2 x_3 \in b^{3n}$, то отрицательная накачка выводит слово из языка, теряется баланс с буквами c. Если $x_1 x_2 x_3 \in b^n$, то любая$(i \neq 1)$ накачка выводит слово из языка, т.к. теряется баланс между буквами b, которые были пораждены с помощью первого правила. Если $x_1 \in b^{3n}$, $x_3 \in b^n$, то отрицательная накачка выводит слово из языка, т.к. нарушаются обе указанных выше причины. Пусть $x_1 = b^{k_1} a b^{k_2}$, $x_2 =$

b^{k_3} и $i = 2$. Получим слово: $b^n b^{2n} a[b^{k_2} b^{k_1} ab^{2n+k_3} ab^{2n} c^{2n}]a^{n-2}$. Подслово выделенное квадратными нарушает баланс между b и c, т.к. $k_1 + k_2 < n < 2n$. Аналогично для случая $x_2 = b^{k_1} ab^{k_2}$, $x_1 = b^{k_3}$

2. Синхронная накачка. Рассмотрим слово w_1. Пусть $x = x_0 x_1 x_2$, $y = y_0 y_1 y_2$. Т.к по условию леммы $|x_1 x_2| \leq n$, то $x_1 = b^{k_1}$ и $x_2 = b^{k_2}$, $k_1 + k_2 \leq n$, $k_1 > 0$. Если $y_1 = b^{k_3}, k_3 >= 0$, то накачка при $i > 1$ выводит слово из языка, т.к. теряется баланс между буквами b, порожденными первым правилом. Если $y_1 = c$, то накачка при $i > 2$ выводит слово из языка, т.к максимальная длина суффикса состоящего из букв c в данном случае может быть только единицей. Пусть $y_1 = b^{k_3} c, k_3 > 0$, тогда при $i = 0$ получим слово $w_3 = b^{3n} ab^{3n-k_1-k_2}$, которое не принадлежит языку, потому что оно могло быть получено только из базового слова ba, но т.к. $k_1 + k_2 \geq 2$, то слово из которого получено w_3 должно быть $b^{k_4} a, k_4 \geq 2$, а это слово не является базовым.

Таким образом, данный язык не принадлежит классу `DCFL`.

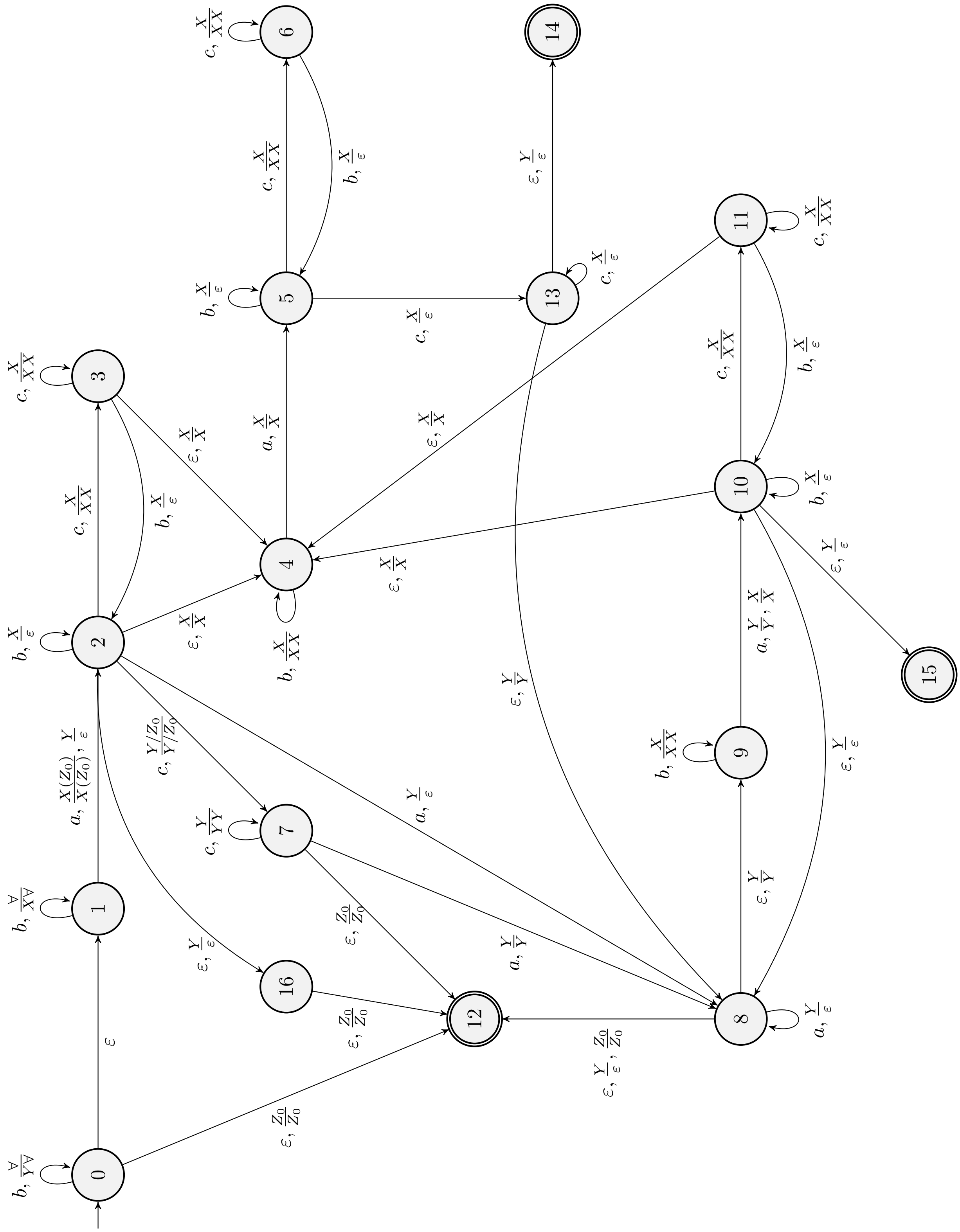

Рис. 1: Автомат для языка L

2 Задача 2

Язык $\left\{w \;\middle|\; |w|_{ab} = |w|_{baa} \;\&\; w = w^R\right\}$. Алфавит $\{a, b\}$

2.1 Решение

Пусть $L_1 = \{w \;\Big|\; |w|_{ab} = |w|_{baa}\}$, $L_2 = \{w \;\Big|w = w^R\}$. Язык L_1 регулярный, а язык L_2 контекстно-свободный. Значит исходный язык L является КС, как пересечение КС и регулярного языков.

Докажем недетерминированность L. Пусть n - длина накачки, положим $k = n + 1$. Тогда возьмем следующие слова:

$$w_1 = a^{2k}b^{2k}a^{2k}, w_2 = a^{2k}b^{2k}a^{3k}b^{2k}a^{3k}b^{2k}a^{2k}$$

Пусть $x = a^{2k}b^{2k}a^{2k-1}$, $y = a$, $z = a^{k+1}b^{2k}a^{3k}b^{2k}a^{2k}$. Необходимо рассмотреть 2 случая:

1. Рассмотрим общий перефикс x. Пусть $x = x_0x_1x_2x_3x_4$. В случаях: $x_1 = a^k$ и $x_3 = a^p$, $x_1 = a^k$ и $x_3 = b^p$, $x_1 = b^k$ и $x_3 = b^p$; отрицательная накачка в w_2 выводит слово из языка, т.к полученное слово уже не будет являться палиндромом. Если $x_1 = a^{k_1}b^{k_2}$, либо $x_2 = a^{k_1}b^{k_2}$, то отрицательная накачка выводит оба слова из языка

2. Пусть $x = x_0x_1x_2$, $y = y_0y_1y_2$, $z = z_0z_1z_2$. Т.к по условию леммы $|x_1x_2| \leq n$, то $x_1 = a^{k_1}$ и $x_2 = a^{k_2}$, $k_1 + k_2 \leq n$, $k_1 > 0$. Также y_1 равно либо пустому слову, либо a, тогда слово $x_0x_1^ix_2y_0y_1^iy_2$ при любом $i \neq 1$ не принадлежит L, т.к. не является палиндромом.

Следовательно, данный язык не является детерменированным КС языком.

3 Задача 3

Язык атрибутной грамматики для регулярок:

$$
\begin{array}{lll}
[S] \to [Regexp] & ; & \\
[Regexp] \to [Regexp][Regexp] & ; & Regexp_1.val \neq \varepsilon, Regexp_2.val \neq \varepsilon \\
 & & Regexp_0.val := Regexp_1.val + +Regexp_2.val \\
[Regexp] \to ([Regexp]|[Regexp]) & ; & Regexp_1.val \neq \varepsilon \vee Regexp_2.val \neq \varepsilon, \\
 & & Regexp_1.val \neq |, Regexp_0.val := | \\
[Regexp] \to ([Regexp])* & ; & Regexp_1.val \neq \varepsilon \\
 & & Regexp_1.val \neq *, Regexp_0.val := * \\
[Regexp] \to \varepsilon & ; & Regexp.val := \varepsilon \\
[Regexp] \to a & ; & Regexp.val := a \\
[Regexp] \to b & ; & Regexp.val := b
\end{array}
$$

3.1 Решение

Рассмотрим подвыражения, которые запрещены согласно ограничениям налагаемым условиями на аттрибут:

1. $(\varepsilon|\varepsilon)$
2. $((\cdot \mid \cdot) \mid \cdot)$
3. $(\varepsilon)*$
4. $((\cdot)*)*$

Для исключения подслов вида 1, 2 введем три новых нетерминала:

$$
\begin{array}{lll}
[Regexp'] & \to & (\varepsilon|[Regexp_{rhs}]) \\
[Regexp'] & \to & ([Regexp_{lhs}]|\varepsilon) \\
[Regexp_{rhs}] & \to & [Regexp'] \\
[Regexp_{rhs}] & \to & [Regexp_{iter}] \\
[Regexp_{rhs}] & \to & [Regexp_{rhs}][Regexp_{rhs}] \\
[Regexp_{rhs}] & \to & a \\
[Regexp_{rhs}] & \to & b
\end{array}
\qquad
\begin{array}{lll}
[Regexp_{lhs}] & \to & [Regexp_{lhs}][Regexp_{lhs}] \\
[Regexp_{lhs}] & \to & [Regexp_{iter}] \\
[Regexp_{lhs}] & \to & a \\
[Regexp_{lhs}] & \to & b
\end{array}
$$

Для того, чтобы исключить выражения вида 3, 4 введем новый нетерминал $Regexp_{iter}$

$$
\begin{array}{l}
[Regexp_{iter}] \to [Regexp'] \\
[Regexp_{iter}] \to [Regexp_{iter}][Regexp_{iter}] \\
[Regexp_{iter}] \to a \\
[Regexp_{iter}] \to b
\end{array}
$$

В итоге получим следующую грамматику описывающий язык данной атрибутной грамматики:

$$
\begin{array}{lcl}
[S] & \rightarrow & [Regexp] \\
[Regexp] & \rightarrow & [Regexp][Regexp] \\
[Regexp] & \rightarrow & [Regexp'] \\
[Regexp] & \rightarrow & [Regexp_{iter}] \\
[Regexp'] & \rightarrow & (\varepsilon|[Regexp_{rhs}]) \\
[Regexp'] & \rightarrow & ([Regexp_{lhs}]|\varepsilon) \\
[Regexp'] & \rightarrow & ([Regexp_{lhs}]|[Regexp_{rhs}) \\
[Regexp_{rhs}] & \rightarrow & [Regexp'] \\
[Regexp_{rhs}] & \rightarrow & [Regexp_{iter}] \\
[Regexp_{rhs}] & \rightarrow & [Regexp_{rhs}][Regexp_{rhs}] \\
[Regexp_{rhs}] & \rightarrow & a \\
[Regexp_{rhs}] & \rightarrow & b \\
[Regexp] & \rightarrow & \varepsilon \\
[Regexp] & \rightarrow & a \\
[Regexp] & \rightarrow & b
\end{array}
$$

$$
\begin{array}{lcl}
[Regexp_{iter}] & \rightarrow & [Regexp'] \\
[Regexp_{iter}] & \rightarrow & [Regexp_{iter}][Regexp_{iter}] \\
[Regexp_{iter}] & \rightarrow & a \\
[Regexp_{iter}] & \rightarrow & b \\
[Regexp_{lhs}] & \rightarrow & [Regexp_{lhs}][Regexp_{lhs}] \\
[Regexp_{lhs}] & \rightarrow & [Regexp_{iter}] \\
[Regexp_{lhs}] & \rightarrow & a \\
[Regexp_{lhs}] & \rightarrow & b
\end{array}
$$