Домашнее задание №4

Томинин Ярослав, 778 группа

8 октября 2018 г.

1.

Поймем, что этот язык регулярный(так как с помощью модификации Ахо-Корасик можно построить ДКА, добавив преходы в стоковое состояние и заменив префиксные ссылки на переход в стоковое состояние). Но лемма о накачке- необходимое условие регулярности языка. Поэтому это условие выполняется.

В самой лемме можно выбрать p=L+1, где L - максимальная длина слова. Тогда для любого слова из алфавита длины большей L(а их пустое множество) будут выпольняться условия этой леммы.

2.

1)Найдем допустимые слова по алгоритму Евклида

(1,0)2017n+5=(0,1)503k+29

2017n-503k=24

Решим следующее

(1,0)2017n-(0,1)503k=1

(1,4)5 (0,1)-503

(1,4)5 (100,401)-3

(101,405)2 (100,401)-3

(101,405)2 (201,806)-1

(302,1211)1 (201,806)-1

1=302*2017-1211*503

Следовательно n=302*24+503*z, k=1211*24+2017*z, где z-натуральное число.

Тогда наш алфавит содержит слова

$L = \{a^n | n = 2017 * (302 * 24 + 503z) + 5, z \in N\}$

Построим ДКА для этого языка, тем самым докажем, что он регудярный.

ДКА будет над алфавитом a, и иметь 2017*302*24+5+1 состояний

Построим его следующим образом: из i в i+1 стрелка по a для всех кроме последнего(с ним разберемся позже)

На i состоянии будет прочитанно i букв a(это доказывается по индукции: база-для i=0 мы прочитали 0 букв, переход: если на i прочитанно i букв, то перейти в i+1 мы можем только из i по букве a)

Далее добавим из крайнего допускающего состояния переход по a в 2017*302*24+6-503*2017 состояние

Тогда из допускающего состояния мы еще сможем переходить в состояние с инвариантом 2017*302*24+5-503*2017 и идти, пока не дойдем до принимающего. Почему это дополнение не сломает наш инвариант: потому что пока мы не дойдем до принимающего состояния, эта стрелка не будет учавствовать и инвариант будет сохраняться, когда мы дойдем до конца, то мы можем пройти по стрелке и инвариант действительно сломается, но мы этого и хотим, ведь мы опять не сможем перейти по стрелке, пока не дойдем до конца(а до конца мы дойдем только через 503*2017-1 букву). Поэтому мы будем оказываться в допускающем состоянии тогда и только тогда, когда |w|=2017*(302*24+503z)+5

Так как мы постоили ДКА, то этот язык регулярный

2)

Предположим, что язык регулярный. Рассмотрим ДКА, который имеет p состояний и рассмотрим слово, длины большей, чем p. Тогда мы можем разбить это слово на $\omega = xyz = a^{200n^2+1}$, где y не пустое слово и $\forall \alpha xy^\alpha z \in L$

Рассмотрим слова $xy^{200n^2}z(|y| = k)$,тогда длина этого слова равна $(k + 1)200n^2$. Мы получили четное число, поэтому это слово не принадлежит L и мы пришли к противоречию.

3)

Будем решать эту задачу для алфавита а,б. Аналогично предыдущему пункту предположим, что это регулярный язык, посторим для него ДКА, который имеет p состояний и рассмотрим слово$\omega = \underbrace{b...b}_{p+1} a \underbrace{b...b}_{p+1}$, длины большей, чем p. Тогда мы можем разбить подслово $\underbrace{b...b}_{p+1}$этого слова на $\omega = xyz$, где y не пустое слово и xz попадает в то же состояние, что и xyz.(Действительно, ведь слово имеет больше ьукв, чем состояний и у нас будет два одинаковых состояния, подслово y будет приниматься между этими состояниями)

Но тогда слово $\omega = \underbrace{b...b}_{p+1-|y|} \ a \underbrace{b...b}_{p+1}$ должно принадлежать языку, но оно не явдяется палиндромом. Мы пришли к противоречию, поэтому язык не регулярный.

3.

Лемма о накачке

$\exists p \in N, \forall \omega \in L : |\omega| \geq p : \exists x, y, z; \omega = xyz, |xy| \leq p, y \neq \epsilon, \forall \alpha \in N : xy^\alpha z \in L$ Рассмотрим p=5, тогда любое слово, длина которого больше или равна 5, содержится в одном из трех множеств. Если это слово 1 типа, то выберем $x = \epsilon$,y=a, тогда полученные слова будут содержаться в 3 подмножестве. Если это второй тип, то x=b,y=b,тогда полученные слова будут содержаться во 2 подмножестве. Если это третий тип, то $x = \epsilon$,y=a,тогда полученные слова будут содержаться в 3 подмножестве. Поэтому лемма о накачке выполнена.

Докажем, что этот язык не регулярный. От противного, предположим, что это регулярный язык, посторим для него ДКА, который имеет p состояний и рассмотрим слово 1 типа, длины большей, чем 4p($|\omega| = 2^i$). Тогда есть состояние в которое мы попадаем 4 раза(назовеи крюком последовательность символов от этого состояния до этого состояния, таких крюков будет хотя бы 3), следовательно существует такое разбиение $\omega = xyz$, такое что y=bbb..bbb и $|y| < 2^i$(так как a может содержаться только в одном из крюков и крюки содержат хотя ьы один символ, если бы такого слова не нашлость, то длина всего слова была бы больше $2^i + 1$). Рассмотрим слово xy^2z: оно точно не содержится в 2 и 3 множествах(так как содержит одну букву a) и не содержится в 1 множестве, так как длина слова $1 + 2^i < |\omega'| < 1 + 2^{i+1}$. Поэтому это слово не содержится в нашем языке и мы пришли к противоречию. Следовательно, язык не регулярен.

4.

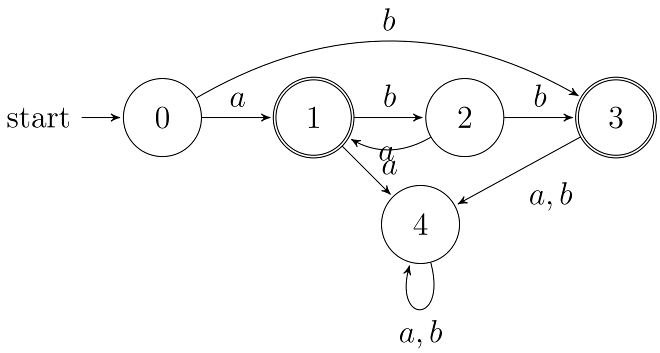
а) Нет, не верно. Мы уже знаем, что $F = \{a^n b^n | n \in N\}$ нерегулярен. Язык $L = a$ является регулярным. Их объединение равно регулярному пустому языку.

б) Построим полные ДКА для языков $F \cap R, F \cap \overline{R}$. Используем свое умения составления ДКА для объединения языков, имеющих полные ДКА. Но полученный ДКА и будет ДКА для языка F, поэтому язык F регулярен.

5.

а)

Продедаем алгоритм, который обсуждали на паре.

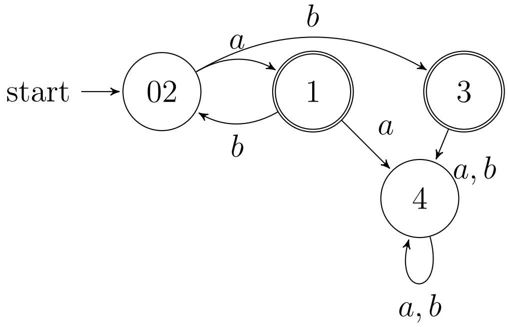


0,2,4|1,3 Так как из 4 идет переход по а в 4, а из 2 в 1, из 0 в 3

0,2|4|1,3 Так как переход по b из 3 в 4, а из 1 в 2

0,2|4|1|3 Сейчас если рассмотреть переходы по а, то из 0,2 в 1. Если рассмотреть переходы по б, то из 0,2 в 3.

Поэтому мы объединяем состояния 1 и 2.



b)
Рассмотрим теперь $\overline{L}$
Так как ДКА для $\overline{L}$ можно получить поменяв допускающие и недопускающие состояния в полном ДКА, то мы можем проделать алгоритм для того же ДКА, но с поменянными состояниями.
Заметим, что в алгоритме изначальное разделения на подмнодества останется таким же и переходы не изменятся. Следовательно все итеррации алгоритма останутся прежними и результат алгоритма, соответственно, будет таким же. Поэтому нам достаточно понять какие состояния будут допускающими в нашем автомате, подредактировав уже существующий(так как разделение на эквивалентности будет таким же)