Теория формальных языков. Рубежный контроль №1

Вариант №11

Банников Арсений

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

Содержание

Задача 1

Язык всех завершающихся систем переписывания строк из одного правила. Алфавит: $\{f, g, \to\}$

Решение

Докажем нерегулярность при помощи отрицания леммы о накачке для регулярных языков. Рассмотрим слово $f^{n+1} \to f^n$, очевидно то что накачка не может содержать символ $\to$, так как при положительной накачке количество таких символов увеличивается, но оно должно оставаться равным одному. Следовательно накачку можно выбрать только либо слева, либо справа от $\to$. При выборе накачки из фрагмента справа положительная накачка выводит слово из языка, а при выборе накачки из левого фрагмента отрицательная накачка выводит слово из языка.

Примеры накачек:

$$f^{n_1} f^{i(n+1-n_1-n_2)} f^{n_2} \to f^n \tag{1}$$

В такой накачке $n + 1 - n_1 - n_2 \geq 1 \Rightarrow n_1 + n_2 < n + 1$. Следовательно при отрицательной накачке $(i = 0)$ мы получим слово $f^{n_1+n_2} \to f^n$. Полученное слово не пренадлежит языку так как $n_1 + n_2 \leq n$.

$$f^{n+1} \to f^{n_1} f^{i(n-n_1-n_2)} f^{n_2} \tag{2}$$

Данное слово эквивалентно $f^{n+1} \to f^{i(n-n_1-n_2)+n_1+n_2}$. Так как длина накачки ненулевая, то $n-n_1-n_2 \geq 1$. Взяв $i = n+1$ получим $(n+1)(n-n_1-n_2) \geq n \Rightarrow (n+1)(n-n_1-n_2)+n_1+n_2 \geq n$. А заначит полученное от положительной накачки слово не пренадлежит языку.

Мы рассмотрели все варианты накачки, а значит выбранное слово не накачивается, следовательно язык не регулярен.

Задача 2

Язык $\{a^{\frac{n}{\log_2 n}} b^n\}$

Решение

Заметим что область значений n не равна $\mathbb{N}$, так как например при $n = 8$ степень при a имеет нецелое значение. Найдем область допустимых значений n. Очевидно, что $n \in \mathbb{N} \cap \{x \mid \frac{x}{\log_2 x} \in \mathbb{N}\}$. Рассмотрим множество $\{x \mid \frac{x}{\log_2 x} \in \mathbb{N}\}$, в него входят только x такие, что $\log_2 x \in \mathbb{N}$. Для доказательства последнего утверждения допустим, что $\log_2 x \in \mathbb{R}$, возможны два случая: либо $\log_2 x \in \mathbb{I}$, либо $\log_2 x \in \mathbb{Q}$. Если $\log_2 x \in \mathbb{I}$, то $\frac{x}{\log_2 x} \in \mathbb{I}$, но такое невозможно в силу правила построения множества. Если $\log_2 x \in \mathbb{Q}$, то пусть $\log_2 x \notin \mathbb{N}$ что означает, что $\log_2 x = a + b, a \in \mathbb{N}, b \in (0; 1) \cap \mathbb{Q}$. Заметим что $x = 2^{\log_2 x} = 2^a 2^b$, но $2^a \in \mathbb{N}$, а $2^b \in \mathbb{I}$, из чего следует, что $2^a 2^b \in \mathbb{I}$, а значит и $x \in \mathbb{I}$, но так как $x \in \mathbb{N}$, получили противоречие. Следовательно $x = 2^p, p \in \mathbb{N}$.

Получаем, что множество $\{x \mid \frac{x}{\log_2 x} \in \mathbb{N}\}$ состоит из элементов вида $x = \frac{2^p}{p}, p \in \mathbb{N}$, следовательно $2^p = 0 \pmod p$. Это значит, что $p = 2^m, m \in \mathbb{N}$. Следовательно $n = 2^{2^m}, m \in \mathbb{N}$. Теперь заметим что при увеличении m количество символов a растет медленнее чем количество символов b (имеется в виду порядок роста). Но при накачке (если первый накачиваемый фрагмент состоит только из символов a, а второй только из символов b) рост количества букв у обеих частей линейный, следовательно в какой то момент нарушится отношение количества символов. Если же рассматривать вариант разбиения такой, что первый и второй фрагменты накачки состоят только из одинаковых букв, то очевидно что положительная накачка разрушает соотношение количества букв.

Итак, ниодно слово языка (при достаточно больших n) не накачивается, а значит язык не является КС

Задача 3

Грамматика

$$S \to SbbabSbaaabS \quad (1)$$
$$S \to SSbS \quad (2)$$
$$S \to a \quad (3)$$

Решение

Будем рассматривать только правила (2) и (3). Рассмотрим слово $a^{n+1}(ba)^n$. Оно получается раскрытием второго нетерминала по правилу (2), сентенциальные формы следующие:

$$S$$
$$SSbS$$
$$SSSbSbS$$
$$SSSSbSbSbS$$
$$...$$

Заметим, что если раскрывать только по второму и третьему правилу то количество букв то для слова ω верно, что $|\omega|_a = 2m + 1$ и $|\omega|_b = m$.

Рассмотрим пересечение данной граматики с регулярным языком $a^*(ba)^*$. Раскрытие по правилу (1) не лежит в пересечении из-за фрагмента $baaab$. Очевидно что мы должны выбирать такую накачку ω_2, что $|\omega_2|_a = 2m$ и $|\omega_2|_b = m$, но такой фрагмент может быть только на границе фрагментов звездочных групп. Мы очевидно мы не можем выбрать такой фрагмент, так как положительная накачка породит слово где справа от первой встретившейся b будет последовательность из a длиной хотя бы 2, что не соответствует пересекаемому регулярному языку. Следовательно исходный язык не регулярен.

Задача 4

Язык всех скобочных последовательностей, являющихся сдвигами слов из языка Дика (скобки только круглые).

Решение

Исходный язык можно переписать как $\{\omega \mid |\omega|_(= |\omega|_)\}$. Это верно так как мы можем преобразовать любое слово такого языка к слову из языка Дика, и для любого слова языка Дика мы можем получить перестановку с очевидно одинаковым числом открывающих и закрывающих скобок. Обсудим алгоритм преобразования перестановки слова к слову. Алгоритм состоит в том чтобы находить первую неверно пославленную закрывающую скобку и весь префикс, включая эту скобку, сдвигать в конец. Для доказательства корректности заметим полуинвариант. Введем счетчик, изначально равный нулю, и будем уменьшать его на один каждый раз если считываем закрывающую скобку и увеличивать если считываем открывающую. Полуинвариантом является минимальное значение данного счетчика за полный проход по слову, которое каждый раз будет увеличиваться. Очевидно что увеличиваться бесконечно оно не может, так как в конце разбора слова счетчик всегда равен нулю. Заметим что переместив начальный неправильный префикс в конец Мы увеличим полуинвариант на один. Это следует из того, что перемещая префикс без последней закрывающей скобки мы не изменяем полуинвариант, так как префикс без последней скобки является словом языка Дика. Теперь заметим, что если убрать закрывающую скобку, то мы увеличим полуинвариант. После того как мы убрали закрывающую скобку из начала счетчик после прохода по слову стал равным одному. Ну и очевидно, что добавив закрывающую скобку в конец полуинвариант не уменьшится. Следовательно полуинвариант всегда увеличивается. Значит в какой то момент он станет равным нулю, что означает что мы распознаем слово языка Дика.

Теперь построим распознаватель, который следит за равенством количества скобок...